syzbot: automated kernel testing

Linux Plumbers Conference 2018
Nov 13, 2018, Vancouver
Dmitry Vyukov, dvyukov@
Agenda

- Motivation
- syzbot workflow
- Pain points/wishes
- Future work
"Stable" releases

Number of backports in "stable" branches

Backports/month

- v3.4
- v3.18
- v4.4
- v4.9
- v4.14
"Stable" releases

- >95% of backports are fixes
- + not backported fixes (700+)
- + not fixed upstream bugs (300+)
- + not found bugs (XXXX+)
- + not detectable yet bugs (XXXX+) (info leaks, races)

Every "looks good and stable" release contains >20'000 bugs.

No, not getting better over time.
syzkaller/syzbot

**syzkaller**: kernel fuzzer
- grammar-based
- coverage-guided
- open-source: [github.com/google/syzkaller](https://github.com/google/syzkaller)

**syzbot**: automation on top of syzkaller
- continuous kernel/syzkaller build
- automatic reporting
- dashboard: [syzkaller.appspot.com](http://syzkaller.appspot.com)
syzbot report

SUBJECT: BUG: corrupted list in locks_delete_block
TO: linux-fsdevel@, linux-kernel@, jlayton@, viro@

HEAD commit: 442b8cea2477 Add linux-next specific files for 20181109
git tree: linux-next
console output: https://syzkaller.appspot.com/x/log.txt?x=12b1262b400000
kernel config: https://syzkaller.appspot.com/x/.config?x=2f72bdb11df9fbe8
dashboard link: https://syzkaller.appspot.com/bug?extid=13eb7470890c56ce3f37
C reproducer: https://syzkaller.appspot.com/x/repro.c?x=11b5fa2b400000

------------[ cut here ]--------------
kernel BUG at lib/list_debug.c:53!
Call Trace:
__list_del_entry include/linux/list.h:117 [inline]
locks_delete_block+0xce/0x3d0 fs/locks.c:716
locks_mandatory_area+0x48b/0x6a0 fs/locks.c:1398
rw_verify_area+0x2f2/0x360 fs/read_write.c:386
vfs_writev+0x1f1/0x360 fs/read_write.c:1004
...
## Bug stats

<table>
<thead>
<tr>
<th></th>
<th>Reported</th>
<th>Fixed</th>
<th>Fixed, %</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Upstream (syzbot)</strong></td>
<td>1400</td>
<td>960</td>
<td>69</td>
</tr>
<tr>
<td><strong>Upstream (manual)</strong></td>
<td>560</td>
<td>?</td>
<td>?</td>
</tr>
<tr>
<td>Internal (syzbot)</td>
<td>1740</td>
<td>388</td>
<td>22</td>
</tr>
<tr>
<td>Internal (manual)</td>
<td>470</td>
<td>?</td>
<td>?</td>
</tr>
<tr>
<td>Fuchsia</td>
<td>70</td>
<td>7</td>
<td>10</td>
</tr>
<tr>
<td><strong>OpenBSD</strong></td>
<td>20</td>
<td>10</td>
<td>50</td>
</tr>
<tr>
<td>gVisor</td>
<td>120</td>
<td>80</td>
<td>67</td>
</tr>
<tr>
<td><strong>Akaros</strong></td>
<td>35</td>
<td>3</td>
<td>9</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td>4415</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Manual bug reporting

Discover => Assess => Report => Ping => Support => Test => Fixed
[automated] dup? non-actionable? symbolize find maintainers find commit find config compose report answer questions

Works fine only you have reported one-two bugs.
Automated bug reporting [syzbot]

Discover => Report => Triage => Debug => Write test => Fix => Test => Mail => Fixed

[automated] [automated] [------------------ still on the developer ------------------] [aided] [automated]
Bug life-cycle

- Discovered
- Reported to mailing lists
  - Has Fix
    - Duplicate
    - Invalid
    - #syz dup: other bug
    - #syz invalid
    - #syz fix: Reported-by:
  - #syz upstream
- Reported to moderation
  - [automatic]
Patch testing

For a bug with a reproducer, reply with:

```
#syz test: git://repo/address.git branch
#syz test: git://repo/address.git commit-hash
[optionally attach patch]
```

Can be used for:

- fix patch testing
- retesting on latest HEAD (is it still happening?)
- observing other failure modes
- debugging (add additional checks, logging)
Reproducers

- Not all bugs have reproducers
  - Races/non-determinism
  - Accumulated state
  - Interactions between concurrent tests
  - ...

- Sometimes reproducers don't work for developers
  - The crash was triggered by the reproducer on fresh machine
  - Wrong source
  - Wrong config
  - No debugging configs
  - Different hardware
  - ...

- Fix ratio:
  - with repro: ~73%
  - w/o repro: 66%
Future automation

- bisection
- committed fix testing
- retesting on latest tree
- fix bisection
- pings
- auto-closing stale bugs
<table>
<thead>
<tr>
<th>Title</th>
<th>Repro</th>
<th>Count</th>
<th>Last</th>
<th>Reported</th>
</tr>
</thead>
<tbody>
<tr>
<td>WARNING: locking bug in loop_control_ioctl</td>
<td>C</td>
<td>552</td>
<td>now</td>
<td>2d06h</td>
</tr>
<tr>
<td>unregister_netdevice: waiting for DEV to become free (2)</td>
<td>C</td>
<td>187656</td>
<td>now</td>
<td>88d</td>
</tr>
<tr>
<td><strong>kernel BUG at net/pv4/p_output.c:LINE!</strong></td>
<td>C</td>
<td>25880</td>
<td>now</td>
<td>121d</td>
</tr>
<tr>
<td>WARNING in xfrm6_tunnel_net_exit (2)</td>
<td>C</td>
<td>19073</td>
<td>now</td>
<td>176d</td>
</tr>
<tr>
<td>WARNING in bpfjit_free</td>
<td>syz</td>
<td>6591</td>
<td>9m</td>
<td>118d</td>
</tr>
<tr>
<td>WARNING in compat_copy_entries (2)</td>
<td>syz</td>
<td>7905</td>
<td>13m</td>
<td>250d</td>
</tr>
<tr>
<td><strong>KASAN: use-after-free Read in cma_cancel_operation</strong></td>
<td>C</td>
<td>338</td>
<td>15m</td>
<td>222d</td>
</tr>
<tr>
<td>general protection fault in perf_tp_event</td>
<td></td>
<td>216</td>
<td>18m</td>
<td>191d</td>
</tr>
<tr>
<td>KMSAN: uninit-value in ip_tunnel_xmit (2)</td>
<td>C</td>
<td>1916</td>
<td>35m</td>
<td>93d</td>
</tr>
<tr>
<td>possible deadlock in console_unlock</td>
<td>C</td>
<td>3961</td>
<td>41m</td>
<td>158d</td>
</tr>
<tr>
<td><strong>KASAN: slab-out-of-bounds Read in ip6_tnl_parse_tlv_enc_lim</strong></td>
<td>C</td>
<td>159</td>
<td>44m</td>
<td>54d</td>
</tr>
<tr>
<td>possible deadlock in ario_poll</td>
<td>C</td>
<td>2519</td>
<td>1h10m</td>
<td>62d</td>
</tr>
<tr>
<td>WARNING in clear_standby</td>
<td>C</td>
<td>1639</td>
<td>1h12m</td>
<td>51d</td>
</tr>
<tr>
<td>INFO: task hung in flush_work</td>
<td>C</td>
<td>398</td>
<td>1h16m</td>
<td>188d</td>
</tr>
<tr>
<td>possible deadlock in mon_bin_vma_fault</td>
<td>C</td>
<td>5822</td>
<td>1h20m</td>
<td>68d</td>
</tr>
<tr>
<td>possible deadlock in down_trylock (2)</td>
<td>C</td>
<td>2</td>
<td>1h21m</td>
<td>16d</td>
</tr>
<tr>
<td><strong>kernel panic: corrupted stack end detected inside scheduler (3)</strong></td>
<td>C</td>
<td>1596</td>
<td>1h22m</td>
<td>101d</td>
</tr>
<tr>
<td>WARNING in xfrm_state_fini (2)</td>
<td>C</td>
<td>26726</td>
<td>1h28m</td>
<td>286d</td>
</tr>
<tr>
<td><strong>KASAN: use-after-free Write in_free_event</strong></td>
<td>C</td>
<td>105</td>
<td>1h29m</td>
<td>125d</td>
</tr>
<tr>
<td>KASAN: null.ptr.deref Write in kthread_stop</td>
<td>C</td>
<td>671</td>
<td>1h31m</td>
<td>13d</td>
</tr>
<tr>
<td><strong>BUG: MAX_LOCKDEF CHAINS too low!</strong></td>
<td>C</td>
<td>142</td>
<td>1h33m</td>
<td>44d</td>
</tr>
<tr>
<td><strong>KASAN: slab-out-of-bounds Read in rds_cong_queue_updates (2)</strong></td>
<td>C</td>
<td>124</td>
<td>1h38m</td>
<td>122d</td>
</tr>
<tr>
<td>possible deadlock in free_ioctl_users</td>
<td>C</td>
<td>184</td>
<td>2h02m</td>
<td>63d</td>
</tr>
<tr>
<td><strong>KASAN: use-after-free Read in rds_cong_queue_updates (2)</strong></td>
<td>C</td>
<td>77</td>
<td>2h07m</td>
<td>111d</td>
</tr>
<tr>
<td><strong>kernel BUG at net/core/skbuff.c:LINE! (3)</strong></td>
<td>C</td>
<td>1136</td>
<td>2h21m</td>
<td>283d</td>
</tr>
<tr>
<td>WARNING in ext4_set_page_dirty</td>
<td>C</td>
<td>5590</td>
<td>2h34m</td>
<td>224d</td>
</tr>
<tr>
<td><strong>BUG: please report to <a href="mailto:deep@vger.kernel.org">deep@vger.kernel.org</a> =&gt; prev = 0, last = 0 at n</strong></td>
<td>C</td>
<td>7595</td>
<td>2h40m</td>
<td>371d</td>
</tr>
<tr>
<td>INFO: task hung in aead_recvmsg</td>
<td></td>
<td>13417</td>
<td>2h44m</td>
<td>336d</td>
</tr>
<tr>
<td>WARNING: kernel stack frame pointer has bad value (2)</td>
<td>C</td>
<td>322</td>
<td>2h56m</td>
<td>118d</td>
</tr>
<tr>
<td>WARNING: refcount bug in kobject_get</td>
<td>C</td>
<td>117</td>
<td>3h26m</td>
<td>62d</td>
</tr>
</tbody>
</table>
Unfixed bugs

Hundreds of bugs are unfixed:

- Some are bad vulnerabilities
- Some are "just bugs"
- All harm syzkaller’s ability to uncover new bugs

Need help:

- Fixing
- Routing
- Duping
- Invalidating
Syscall Descriptions

syzkaller is based on **declarative descriptions** of system calls:

```c
open(file filename, flags flags[open_flags],
    mode flags[open_mode]) fd
read(fd fd, buf buffer[out], size len[buf])
close(fd fd)
```

Tests **only** what's described.
FUSE example

resource fd_fuse[fd]

open(file ptr[in, string"/dev/fuse"],
     flags const[O_RDWR], mode const[0]) fd_fuse

write(fd fd_fuse, arg ptr[in, fuse_out[fuse_open_out]],
      len bytesize[arg])

fuse_open_out {
     fh const[0, int64]
     open_flags flags[fuse_open_flags, int32]
     padding const[0, int32]
}
Syscall descriptions

- Check if your subsystem has descriptions
- Check if necessary configs are enabled
- Check if it needs cmdline args, sysctls, setup
- Check how well it is tested
- Add descriptions
static void fuse_file_put(struct fuse_file *ff, bool sync)
{
    if (refcount_dec_and_test(&ff->count)) { /*covered*/
        struct fuse_req *req = ff->reserved_req; /*covered*/
        if (ff->fc->no_open) {
            /*
             * Drop the release request when client does not
             * implement 'open'
             */
            __clear_bit(FR_BACKGROUND, &req->flags);
            iput(req->misc.release.inode);
            fuse_put_request(ff->fc, req);
        } else if (sync) { /*covered*/
            __set_bit(FR_FORCE, &req->flags);
            __clear_bit(FR_BACKGROUND, &req->flags);
            fuse_request_send(ff->fc, req);
            iput(req->misc.release.inode);
            fuse_put_request(ff->fc, req);
        } else {
            req->end = fuse_release_end; /*covered*/
            __set_bit(FR_BACKGROUND, &req->flags);
            fuse_request_send_background(ff->fc, req);
        }
        kfree(ff); /*covered*/
    }
} /*covered*/
Stub/test devices

Examples:
- CONFIG_TUN (/dev/net/tun)
- CONFIG_VIDEO_VIVID (/dev/video0)
- CONFIG_MAC80211_HWSIM
- USB!

Allow to:
- write unit-tests for kernel (KernelCI, 0-day)
- test user-space code without hardware
- fuzz kernel in VMs

Need more of them!
Stub/test devices (contd)

Allow to reach:
- common code not reachable without a device
- external input paths
  - NFC
  - CAN
  - Bluetooth

Don'ts:
- single global device
- fixed number of devices
- only init_net namespace
- asynchronous processing
How you think kernel crashes look
WARNING: CPU: 0 PID: 4274 at drivers/dma-buf/dma-buf.c:992
CPU: 0 PID: 4274 Comm: syz-executor4 Not tainted 4.20.0-rc2
Hardware name: Google Compute Engine, BIOS Google 01/01/2011
Call Trace:
  vb2_vmalloc_detach_dmabuf+0x5a/0x80
  __vb2_plane_dmabuf_put.isra.5+0x122/0x310
  vb2_core_queue_release+0x62/0x80
  vb2_fop_release+0x77/0xc0
  vivid_fop_release+0x18e/0x440
  v4l2_release+0x224/0x3a0
  __fput+0x385/0xa30
  ____fput+0x15/0x20
  task_work_run+0x1e8/0x2a0
  exit_to_usermode_loop+0x318/0x380
  do_syscall_64+0x6be/0x820
How kernel crashes actually look
** 2158 printk messages dropped ** [ 50.671305] Call Trace:
** 2378 printk messages dropped ** [ 50.676929] [<ffffffff81b0ce6d>] ? security_file_permission+0x13d/0x190
** 4635 printk messages dropped ** [ 50.697826] 0000000000000000 3fe20028167234bc ffff8800b43179b0 fffffff81cc9b0f
** 4555 printk messages dropped ** [ 50.708497] Object ffff8801d3701170: 00 00 00 00 00 00 00 00 00 67 b4 b5 00 88 ff
  ........g......
** 5357 printk messages dropped ** [ 50.721064] ffff8801d3701080: fc fc fc fc fc fc fc fc fc fc fc fc fc fc fc fc
** 4598 printk messages dropped ** [ 50.731610] __slab_alloc.isra.74.constprop.77+0x50/0xa0
** 3637 printk messages dropped ** [ 50.740170]  ffff8801d3701280: fc fc fc fc fc fc fb fb fb fb fb fb fb fb fb fb
** 4491 printk messages dropped ** [ 50.750742] INFO: Allocated in fasync_helper+0x29/0x90 age=1 cpu=1 pid=6024
** 4370 printk messages dropped ** [ 50.761001]  [<ffffffff8123648d>] native_queued_spin_lock_slowpath+0x5ad/0x660
** 4510 printk messages dropped ** [ 50.771609]  
** 2979 printk messages dropped ** [ 50.778606] 0000000000000000 3fe20028167234bc ffff8800b43179b0 ffffffff81cc9b0f
** 3833 printk messages dropped ** [ 50.794205] SyS_fcntl+0x5be/0xc70
** 3490 printk messages dropped ** [ 50.803930] 0000000000000000 3fe20028167234bc ffff8800b43179b0 fffffff81cc9b0f
** 4495 printk messages dropped ** [ 50.811647] [ffffff814d3af4] print_trailer+0x114/0x120
** 3718 printk messages dropped ** [ 50.820379]  
** 4499 printk messages dropped ** [ 50.830930]  
** 3857 printk messages dropped ** [ 50.848107] 0000000000000000 3fe20028167234bc ffff8800b43179b0 fffffff81cc9b0f
** 3921 printk messages dropped ** [ 50.857615] run_ksoftirqd+0x20/0x60
** 3490 printk messages dropped ** [ 50.872518] [ffffff815bee10] ? fsnotify+0xe40/0xe40
** 3860 printk messages dropped ** [ 50.880974]  
** 4253 printk messages dropped ** [ 50.906245] [ffffff8123ab47] do_raw_write_lock+0x7/0x1d0
** 3636 printk messages dropped ** [ 50.914820] [ffffff81003044] ? lockdep_sys_exit_thunk+0x12/0x14
** 4057 printk messages dropped ** [ 50.924057] run_ksoftirqd+0x20/0x60
** 3600 printk messages dropped ** [ 50.932518] [ffffff815bee10] ? fsnotify+0xe40/0xe40
** 3600 printk messages dropped ** [ 50.940974]  
** 4253 printk messages dropped ** [ 50.956245] [ffffff8123ab47] do_raw_write_lock+0x7/0x1d0
** 3636 printk messages dropped ** [ 50.964820] [ffffff81003044] ? lockdep_sys_exit_thunk+0x12/0x14
** 3921 printk messages dropped ** [ 50.974057] run_ksoftirqd+0x20/0x60
** 2782 printk messages dropped ** [ 50.980621] [xffffff815bee10] ? fsnotify+0xe40/0xe40
WARNING: suspicious RCU usage
4.15.0-rc6-mm1+ #52 Not tainted
-----------------------------
net/netfilter/ipset/ip_set_core.c:2057 suspicious rcu_dereference_protected() usage!
other info that might help us debug this:
rcu_scheduler_active = 2, debug_locks = 1
3 locks held by kworker/u4:5/3913:
#0: ((wq_comp
Kernel crashes

- Is there a crash at all?
- When it starts/ends?
- What's its "identity"?
- Intermixed/split lines
Crash parsing

- 14 top level rules
  - INFO:
  - Booting the kernel.
  - UBSAN:
  - unregister_netdevice: waiting for
  - kernel BUG
  - Kernel BUG
  - invalid opcode:
- 74 sub-rules
- 400+ hardcoded function/file names, pieces of output, etc
- 350+ tests
WARN_ON: please use only for bugs
KASAN (KernelAddressSANitizer)

- Detects:
  - use-after-free
  - out-of-bounds on heap/stack/globals
- Detects bugs at the point of occurrence
- Outputs informative reports
- Based on compiler instrumentation (gcc4.9+ or clang)
- Fast: ~2x slowdown, ~2x memory overhead
- Upstream in 4.3 kernel
- Easy to use (CONFIG_KASAN=y)
KASAN: future work

- Print global var names
- Print stack frame description
- Collect and print `call_rcu()` stacks
- Instrument bitops
- Instrument DMA transfers
- Instrument skb linear buffer (?)
KMSAN (KernelMemorySANitizer)

KMSAN detects uses of uninitialized values.

Working version on [github](https://github).

So far found 110 bugs.
requires clang
KTSAN (KernelThreadSANitizer)

KTSAN detects data races.

Frozen prototype on [github](https://github).
Say NO to "benign" data races
Thanks!

Q&A

syzkaller@googlegroups.com
Dmitry Vyukov, dvyukov@
Backup
Sample of release backports

5b6717c6a3c0c USB: **handle_NULL** config in usb_find_alt_setting()
4253abe6a3aac USB: **fix_error** handling in usb_driver_claim_interface()
5eaaa5e9bd568 regulator: **fix_crash** caused by null driver data
b6adc1f24bb35 spi: rspi: **Fix** interrupted DMA transfers
082e34f367a54 spi: rspi: **Fix** invalid SPI use during system suspend
6074b71d617dd spi: sh-msiof: **Fix** handling of write value for SISTR register
d120858fca5f6 spi: sh-msiof: **Fix** invalid SPI use during system suspend
429773341c34c spi: tegra20-slink: explicitly enable/disable clock
dc89d37f9098c intel_th: **Fix** device removal logic
247cc73cd8f5e serial: cpm_uart: return immediately from console poll
2b7ba104769b4 tty: serial: **avoid_leaking** struct tty_struct
4fe780c1b00e x86/mm: Expand static page table for fixmap space
04bc4dd86d0f2 flopwy: **Do not copy a kernel pointer to user memory** in FDGETPRM ioctl
f88e50ea03000 ARM: dts: dra7: **fix** DCAN node addresses
99795ed0c62d9 iio: 104-quad-8: **Fix off-by-one_error** in register selection
a82a772da7508 Input: xen-kbdfront - fix multi-touch XenStore node's locations
91e30cae8903a fs/lock: skip lock owner pid translation in case we are in init_pid_ns
0c4439c444160 EDAC: **Fix memleak** in module init error path
a4f7bea878871 nfsd: **fix corrupted reply** to badly ordered compound
d06cc6bcf77345 gpio: **Fix wrong rounding** in gpio-menz127
5bcbbad1fac54 module: exclude SHN_UNDEF symbols from kallsyms api
05f7ab7a0e0c7 ASoC: dapm: **Fix** potential DAI widget pointer deref when linking DAI
3fd534a5480ec EDAC, i7core: **Fix memleaks** and use-after-free on probe and remove
c96c2f2b1b6a scsi: megaraid_sas: Update controller info during resume
a56b97a2fc2d6 iomap: complete partial direct I/O writes synchronously
13ab355240a9d scsi: bnx2i: **add_error_handling** for ioremap_nocache