batman-adv - Bug #370

batadv_mcast_mla_update causes WARNING (+panic reboot)

12/31/2018 07:48 PM - Sven Eckelmann

Status:	New	**Start date:**	12/31/2018
Priority:	Immediate	**Due date:**	
Assignee:	Linus Lüssing	**% Done:**	0%
Category:		**Estimated time:**	0.00 hour
Target version:			

Description

https://lists.open-mesh.org/pipermail/b.a.t.m.a.n/2018-December/018421.html

syzbot found the following crash on:

HEAD commit: eed9688f8513 Merge branch 'ras-core-for-linus' of git://gi..
git tree: upstream
console output: https://syzkaller.appspot.com/x/log.txt?x=10785d57400000
kernel config: https://syzkaller.appspot.com/x/.config?x=fa5c63e12fd85b25
dashboard link: https://syzkaller.appspot.com/bug?extid=050927a651272b145a5d
compiler: gcc (GCC) 8.0.1 20180413 (experimental)

Unfortunately, I don't have any reproducer for this crash yet.

IMPORTANT: if you fix the bug, please add the following tag to the commit:
Reported-by: syzbot+050927a651272b145a5d@syzkaller.appspotmail.com
Reported-by: syzbot+83f2d54ec6b7e417e13f@syzkaller.appspotmail.com

```
QAT: Invalid ioctl
bond0 (unregistering): Releasing backup interface bond_slave_0
bond0 (unregistering): Released all slaves
WARNING: CPU: 0 PID: 28 at net/batman-adv/multicast.c:371
batadv_mcast_mla_tt_add net/batman-adv/multicast.c:371 [inline]
WARNING: CPU: 0 PID: 28 at net/batman-adv/multicast.c:371
__batadv_mcast_mla_update net/batman-adv/multicast.c:636 [inline]
WARNING: CPU: 0 PID: 28 at net/batman-adv/multicast.c:371
batadv_mcast_mla_update+0x248f/0x2da0 net/batman-adv/multicast.c:661
Kernel panic - not syncing: panic_on_warn set ...
CPU: 0 PID: 28 Comm: kworker/u4:2 Not tainted 4.20.0+ #390
Hardware name: Google Google Compute Engine/Google Compute Engine, BIOS
Google 01/01/2011
Workqueue: bat_events batadv_mcast_mla_update
Call Trace:
  __dump_stack lib/dump_stack.c:77 [inline]
  dump_stack+0x1d3/0x2c6 lib/dump_stack.c:113
  panic+0x2ad/0x55c kernel/panic.c:188
  __warn.cold.8+0x20/0x45 kernel/panic.c:540
  report_bug+0x254/0x2d0 lib/bug.c:186
  fixup_bug arch/x86/kernel/traps.c:178 [inline]
  do_error_trap+0x11b/0x200 arch/x86/kernel/traps.c:271
  do_invalid_op+0x36/0x40 arch/x86/kernel/traps.c:290
  invalid_op+0x14/0x20 arch/x86/entry/entry_64.S:973
RIP: 0010:batadv_mcast_mla_tt_add net/batman-adv/multicast.c:371 [inline]
RIP: 0010:__batadv_mcast_mla_update net/batman-adv/multicast.c:636 [inline]
RIP: 0010:batadv_mcast_mla_update+0x248f/0x2da0
net/batman-adv/multicast.c:661
Code: 49 c1 ee 03 48 b8 00 00 00 00 00 fc ff df be f8 f8 ff ff 66 41 89 34
06 48 c1 ea 03 c6 04 02 f8 e9 f7 e7 ff ff e8 41 f3 bf f9 <0f> 0b e9 b8 e3
ff ff 48 8b bd 48 fc ff ff e8 1e fe 02 fa e9 99 df
RSP: 0018:ffff8880a9fe7338 EFLAGS: 00010293
RAX: ffff8880a9422000 RBX: 0000000000000001 RCX: ffffffff87bf0125
RDX: 0000000000000000 RSI: ffffffff87bf1d6f RDI: 0000000000000007
RBP: ffff8880a9fe7738 R08: ffff8880a9422000 R09: 0000000000000006
```

```
R10: 0000000000000000 R11: ffff8880a9422000 R12: ffff8880a9fe7490
R13: 0000000000000000 R14: 1ffff110153fce92 R15: ffff8880a9fe7710
  process_one_work+0xc90/0x1c40 kernel/workqueue.c:2153
  worker_thread+0x17f/0x1390 kernel/workqueue.c:2296
  kthread+0x35a/0x440 kernel/kthread.c:246
  ret_from_fork+0x3a/0x50 arch/x86/entry/entry_64.S:352
Kernel Offset: disabled
Rebooting in 86400 seconds..
```

Related issues:

Related to batman-adv - Bug #369: batadv_mcast_mla_tt_retract causes WARNING ...	**New**	**12/31/2018**

History

#1 - 12/31/2018 10:45 PM - Sven Eckelmann

- Description updated

#2 - 01/02/2019 09:17 PM - Linus Lüssing

Hm, this is basically the same as #369.

I'm wondering... could it be that these two lines:

https://git.open-mesh.org/batman-adv.git/blob/983e498130cc07fd383edafcd0425e377df3dfb1:/net/batman-adv/multicast.c#l661
https://git.open-mesh.org/batman-adv.git/blob/983e498130cc07fd383edafcd0425e377df3dfb1:/net/batman-adv/multicast.c#l662

were swapped by the compiler or CPU?

Or the delayed_work_pending(&bat_priv->mcast.work) might not be cleared yet when that WARN_ON line is reached.

#3 - 01/03/2019 09:21 AM - Sven Eckelmann

- Related to Bug #369: batadv_mcast_mla_tt_retract causes WARNING (+panic reboot) added

#4 - 01/12/2019 09:08 PM - Sven Eckelmann

- Description updated

More information (+reproducer... which looks not really like a reproducer for batman-adv) was posted at
https://lists.open-mesh.org/pipermail/b.a.t.m.a.n/2019-January/018456.html

```
syzbot has found a reproducer for the following crash on:

HEAD commit:    4b3c31c8d4dd Merge branch 'i2c/for-current' of git://git.k..
git tree:       upstream
console output: https://syzkaller.appspot.com/x/log.txt?x=12fecc7f400000
kernel config:  https://syzkaller.appspot.com/x/.config?x=b05cfdb4ee8ab9b2
dashboard link: https://syzkaller.appspot.com/bug?extid=83f2d54ec6b7e417e13f
compiler:       gcc (GCC) 9.0.0 20181231 (experimental)
syz repro:      https://syzkaller.appspot.com/x/repro.syz?x=139c6408c00000

IMPORTANT: if you fix the bug, please add the following tag to the commit:
Reported-by: syzbot+83f2d54ec6b7e417e13f@syzkaller.appspotmail.com

IPv6: ADDRCONF(NETDEV_CHANGE): team0: link becomes ready
bridge0: port 2(bridge_slave_1) entered blocking state
bridge0: port 2(bridge_slave_1) entered disabled state
device bridge_slave_1 entered promiscuous mode
IPv6: ADDRCONF(NETDEV_UP): veth1_to_team: link is not ready
WARNING: CPU: 0 PID: 26 at net/batman-adv/multicast.c:337
batadv_mcast_mla_tt_retract+0x3d7/0x4b0 net/batman-adv/multicast.c:337
```

```
kobject: 'lo' (000000007d40203a): kobject_uevent_env
Kernel panic - not syncing: panic_on_warn set ...
CPU: 0 PID: 26 Comm: kworker/u4:2 Not tainted 5.0.0-rc1+ #21
Hardware name: Google Google Compute Engine/Google Compute Engine, BIOS
Google 01/01/2011
Workqueue: bat_events batadv_mcast_mla_update
Call Trace:
  __dump_stack lib/dump_stack.c:77 [inline]
  dump_stack+0x1db/0x2d0 lib/dump_stack.c:113
  panic+0x2cb/0x65c kernel/panic.c:214
kobject: 'lo' (000000007d40203a): fill_kobj_path: path
= '/devices/virtual/net/lo'
kobject: 'queues' (0000000029ca31b3): kobject_add_internal: parent: 'lo',
set: '<NULL>'
  __warn.cold+0x20/0x48 kernel/panic.c:571
kobject: 'queues' (0000000029ca31b3): kobject_uevent_env
  report_bug+0x263/0x2b0 lib/bug.c:186
kobject: 'queues' (0000000029ca31b3): kobject_uevent_env: filter function
caused the event to drop!
  fixup_bug arch/x86/kernel/traps.c:178 [inline]
  fixup_bug arch/x86/kernel/traps.c:173 [inline]
  do_error_trap+0x11b/0x200 arch/x86/kernel/traps.c:271
  do_invalid_op+0x37/0x50 arch/x86/kernel/traps.c:290
kobject: 'rx-0' (00000000a8654327): kobject_add_internal: parent: 'queues',
set: 'queues'
  invalid_op+0x14/0x20 arch/x86/entry/entry_64.S:973
RIP: 0010:batadv_mcast_mla_tt_retract+0x3d7/0x4b0
net/batman-adv/multicast.c:337
kobject: 'rx-0' (00000000a8654327): kobject_uevent_env
Code: 48 8b 45 d0 65 48 33 04 25 28 00 00 00 0f 85 c8 00 00 00 48 81 c4 a0
00 00 00 5b 41 5c 41 5d 41 5e 41 5f 5d c3 e8 29 07 b0 f9 <0f> 0b e9 de fc
ff ff e8 9d f2 f3 f9 e9 e3 fd ff ff 48 89 df e8 b0
RSP: 0018:ffff8880a96676b0 EFLAGS: 00010293
RAX: ffff8880a96586c0 RBX: 0000000000000001 RCX: ffffffff87d1ed13
RDX: 0000000000000000 RSI: ffffffff87d1f037 RDI: 0000000000000007
kobject: 'rx-0' (00000000a8654327): fill_kobj_path: path
= '/devices/virtual/net/lo/queues/rx-0'
RBP: ffff8880a9667778 R08: ffff8880a96586c0 R09: 0000000000000003
R10: 0000000000000000 R11: ffff8880ae62dc7b R12: ffff8880a9667868
R13: 0000000000000000 R14: 1ffff110152ccf0d R15: ffff88808820dc40
kobject: 'tx-0' (00000000e6698f16): kobject_add_internal: parent: 'queues',
set: 'queues'
kobject: 'tx-0' (00000000e6698f16): kobject_uevent_env
kobject: 'tx-0' (00000000e6698f16): fill_kobj_path: path
= '/devices/virtual/net/lo/queues/tx-0'
  __batadv_mcast_mla_update net/batman-adv/multicast.c:635 [inline]
  batadv_mcast_mla_update+0x78e/0x2980 net/batman-adv/multicast.c:661
kobject: 'veth1_to_bond' (00000000a8d46f44): kobject_add_internal:
parent: 'net', set: 'devices'
kobject: 'veth1_to_bond' (00000000a8d46f44): kobject_uevent_env
kobject: 'veth1_to_bond' (00000000a8d46f44): fill_kobj_path: path
= '/devices/virtual/net/veth1_to_bond'
kobject: 'queues' (000000006228a66d): kobject_add_internal:
parent: 'veth1_to_bond', set: '<NULL>'
kobject: 'queues' (000000006228a66d): kobject_uevent_env
kobject: 'queues' (000000006228a66d): kobject_uevent_env: filter function
caused the event to drop!
kobject: 'rx-0' (000000008bc09a3a): kobject_add_internal: parent: 'queues',
set: 'queues'
  process_one_work+0xd0c/0x1ce0 kernel/workqueue.c:2153
kobject: 'rx-0' (000000008bc09a3a): kobject_uevent_env
kobject: 'rx-0' (000000008bc09a3a): fill_kobj_path: path
= '/devices/virtual/net/veth1_to_bond/queues/rx-0'
kobject: 'tx-0' (000000007f0c3f6d): kobject_add_internal: parent: 'queues',
set: 'queues'
kobject: 'tx-0' (000000007f0c3f6d): kobject_uevent_env
kobject: 'tx-0' (000000007f0c3f6d): fill_kobj_path: path
= '/devices/virtual/net/veth1_to_bond/queues/tx-0'
kobject: 'batman_adv' (00000000f9be0f83): kobject_add_internal:
parent: 'veth1_to_bond', set: '<NULL>'
kobject: 'bond_slave_1' (000000003c754cbc): kobject_add_internal:
parent: 'net', set: 'devices'
kobject: 'bond_slave_1' (000000003c754cbc): kobject_uevent_env
  worker_thread+0x143/0x14a0 kernel/workqueue.c:2296
kobject: 'bond_slave_1' (000000003c754cbc): fill_kobj_path: path
```

```
= '/devices/virtual/net/bond_slave_1'
kobject: 'queues' (000000002a7c877f): kobject_add_internal:
parent: 'bond_slave_1', set: '<NULL>'
kobject: 'queues' (000000002a7c877f): kobject_uevent_env
kobject: 'queues' (000000002a7c877f): kobject_uevent_env: filter function
caused the event to drop!
kobject: 'rx-0' (00000000bd1d2cc7): kobject_add_internal: parent: 'queues',
set: 'queues'
kobject: 'rx-0' (00000000bd1d2cc7): kobject_uevent_env
kobject: 'rx-0' (00000000bd1d2cc7): fill_kobj_path: path
= '/devices/virtual/net/bond_slave_1/queues/rx-0'
kobject: 'tx-0' (00000000ccbdecac): kobject_add_internal: parent: 'queues',
set: 'queues'
kobject: 'tx-0' (00000000ccbdecac): kobject_uevent_env
kobject: 'tx-0' (00000000ccbdecac): fill_kobj_path: path
= '/devices/virtual/net/bond_slave_1/queues/tx-0'
kobject: 'batman_adv' (0000000021ba1ff8): kobject_add_internal:
parent: 'bond_slave_1', set: '<NULL>'
  kthread+0x357/0x430 kernel/kthread.c:246
kobject: 'ip6gretap0' (0000000011d834fd): kobject_add_internal:
parent: 'net', set: 'devices'
  ret_from_fork+0x3a/0x50 arch/x86/entry/entry_64.S:352
kobject: 'ip6gretap0' (0000000011d834fd): kobject_uevent_env
Kernel Offset: disabled
Rebooting in 86400 seconds..
```

#5 - 03/24/2019 09:56 AM - Sven Eckelmann

Here are two new reports:

- https://syzkaller.appspot.com/bug?extid=979ffc89b87309b1b94b
- https://syzkaller.appspot.com/bug?extid=83f2d54ec6b7e417e13f

#6 - 03/29/2019 08:27 AM - Sven Eckelmann

Here is another one: https://lists.open-mesh.org/pipermail/b.a.t.m.a.n/2019-March/018681.html

#7 - 03/29/2019 09:51 AM - Sven Eckelmann

Btw. I am unsure whether there is a chance that the two lines mentioned in #370#note-2 are actually reordered. And set_work_pool_and_clear_pending (please read the comments) should make sure that the called worked function sees the correct pending bits (and thus is allowed to requeue himself).

Adding a barrier() between these two lines just causes following change:

```
--- test1       2019-03-29 09:35:31.227958491 +0100
+++ test2       2019-03-29 09:36:11.819255852 +0100
@@ -2585,9 +2585,9 @@
     2c4f:       48 8b 95 68 fc ff ff    mov    -0x398(%rbp),%rdx
     2c56:       48 b8 00 00 00 00 00    movabs $0xdfffffc0000000000,%rax
     2c5d:       fc ff df
-    2c60:       48 c7 c7 00 00 00 00    mov    $0x0,%rdi
-    2c67:       48 c1 ea 03             shr    $0x3,%rdx
-    2c6b:       c6 04 02 f8             movb   $0xf8,(%rdx,%rax,1)
+    2c60:       48 c1 ea 03             shr    $0x3,%rdx
+    2c64:       c6 04 02 f8             movb   $0xf8,(%rdx,%rax,1)
+    2c68:       48 c7 c7 00 00 00 00    mov    $0x0,%rdi
     2c6f:       48 89 fa                mov    %rdi,%rdx
     2c72:       48 c1 ea 03             shr    $0x3,%rdx
     2c76:       80 3c 02 00             cmpb   $0x0,(%rdx,%rax,1)
```

in this function. On first glance, it only affects how the batadv_mcast_mla_list_free and batadv_mcast_start_timer (jiffies calculation) overlap. There is most likely a good reason why this makes sense for our target CPU but it shouldn't affect the WARN_ON at the beginning of the function batadv_mcast_mla_tt_retract/batadv_mcast_mla_tt_add. At least this CPU core is already finished with it before it schedules the next iteration of batadv_mcast_mla_update.

#8 - 04/22/2019 02:21 AM - Linus Lüssing

Hm, set_work_pool_and_clear_pending() is interesting indeed. I'm wondering whether it makes a difference that we are in a loop, a circle.

set_work_pool_clear_pending() says for the smp_mb() call in the end:

```
     * The following mb guarantees that previous clear of a PENDING bit
     * will not be reordered with any speculative LOADS or STORES from
     * work->current_func, which is executed afterwards.
```

So yes, the next work->current_func call with our WARN_ON(delayed_work_pending()) call should not reorder backwards, over the smp_mb() to in front of the set_work_data(!PENDING) call in set_work_pool_and_clear_pending().

But what about an execution of work->current_func before the set_work_pool_and_clear_pending()? Could our WARN_ON(delayed_work_pending()) be moved forward over the next smp_wmb() to after set_work_data(!PENDING)? The smp_wmb() only prevents movemens of STOREs, right?

The delayed_work_pending() calls just a test_bit() which is part of non-atomic.h. And there only contains some bitwise operations with no additional barriers. Notably no READ_ONCE() or other smp_* calls for instance.

(One question, an smp_mb() does not need to be paired with READ_ONCE() / WRITE_ONCE(), does it? smp_mb() prevents the movement of any instruction over this barrier?)

#9 - 04/22/2019 02:24 AM - Linus Lüssing

PS: I'm working on a patch to remove the ordering assumptions with explicit spinlocks. Would still be interesting to know the cause of this to be able to phrase the commit message as accurate as possible and to determine whether something needs to be submitted to net/stable.

#10 - 04/22/2019 02:35 AM - Linus Lüssing

> "Adding a barrier() between these two lines just causes following change:"

Which gcc optimization level did you use?

#11 - 04/22/2019 09:08 AM - Sven Eckelmann

Linus Lüssing wrote:

> > "Adding a barrier() between these two lines just causes following change:"
>
> Which gcc optimization level did you use?

Whatever was used by the "reproducer" syzcall stuff. Which should be -O2 with a lot of flags.