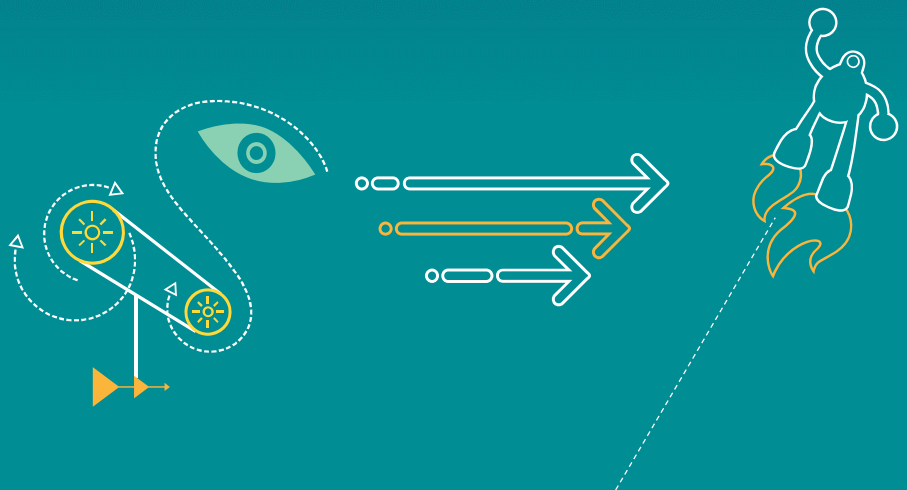
Android Crash Dump Examples

Qualcomm Technologies, Inc.



80-P7139-8 A

Qualcomm Technologies, Inc.
5775 Morehouse Drive
San Diego, CA 92121
U.S.A.

Revision History

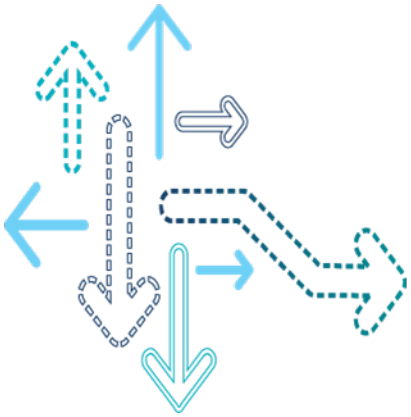
Revision	Date	Description
A	July 2016	Initial release

Contents

Introduction

- This document describes and shows examples of different types of crash issues, and is intended for engineers who are debugging Linux kernel/system crash issues.

Analysis of Data Abort Issues



Dmesg Get Panic Info

- NULL pointer issue

```
53143.231300:   <6> Unable to handle kernel NULL pointer dereference at virtual address 000001f8
53143.238498:   <6> pgd = c0004000
53143.241117:   <2> [000001f8] *pgd=00000000
53143.244673:   <6> Internal error: Oops: 805 [#1] PREEMPT SMP ARM
53143.250096:   <6> CPU: 7 PID: 25476 Comm: kworker/u16:4 Tainted: G        W    3.10.84-g7dff2d8 #1
53143.258526:   <2> Workqueue: DIAG_CNTL_SOCKET cntl_socket_read_work_fn
53143.264499:   <6> task: d3aed140 ti: cdbda000 task.ti: cdbda000
53143.269893:   <2> PC is at diagfwd_channel_close+0x14/0xcc
53143.274833:   <2> LR is at socket_close_channel+0x40/0x11c
53143.279784:   <2> pc : [<c0484ef4>]    lr : [<c0487358>]    psr: 200f0013
sp : cdbdbe60  ip : 00000000  fp : 00000000
53143.291239:   <2> r10: c15b4084  r9 : 00000003  r8 : c15b4374
53143.296448:   <2> r7 : c1896e48  r6 : c18966a0  r5 : cdbdbe90  r4 : 000001f4
53143.302957:   <2> r3 : 00000000  r2 : 00000000  r1 : ffffffd4  r0 : 000001f4
53143.309466:   <2> Flags: nzCv  IRQs on  FIQs on  Mode SVC_32  ISA ARM  Segment kernel
53143.316761:   <2> Control: 10c0383d  Table: 4692006a  DAC: 00000015

53143.995938:   <2> [<c0484ef4>] (diagfwd_channel_close+0x14/0xcc) from [<c0487358>] (socket_close_channel+0x40/0x11c)
53144.005919:   <2> [<c0487358>] (socket_close_channel+0x40/0x11c) from [<c04877a8>] (cntl_socket_read_work_fn+0x374/0x3c4)
53144.016330:   <2> [<c04877a8>] (cntl_socket_read_work_fn+0x374/0x3c4) from [<c013e134>] (process_one_work+0x280/0x41c)
53144.026497:   <2> [<c013e134>] (process_one_work+0x280/0x41c) from [<c013e4e8>] (worker_thread+0x218/0x36c)
53144.035702:   <2> [<c013e4e8>] (worker_thread+0x218/0x36c) from [<c0143ba8>] (kthread+0xa0/0xac)
53144.043947:   <2> [<c0143ba8>] (kthread+0xa0/0xac) from [<c01062e0>] (ret_from_fork+0x14/0x34)
53144.052012:   <6> Code: e2504000 03e00004 0a000028 e3a03000 (e5c43004)
53144.058858:   <6> ---[ end trace 704a28e5980ad825 ]---
```

Check Code

```
811 int diagfwd_channel_close(struct diagfwd_info *fwd_info)
812 {
813     if (!fwd_info)
814             return -EIO;
815
816     fwd_info->ch_open = 0;
817     if (fwd_info && fwd_info->c_ops && fwd_info->c_ops->close)
818             fwd_info->c_ops->close(fwd_info);
819
820     if (fwd_info->buf_1 && fwd_info->buf_1->data)
821             atomic_set(&fwd_info->buf_1->in_busy, 0);
822     if (fwd_info->buf_2 && fwd_info->buf_2->data)
823             atomic_set(&fwd_info->buf_2->in_busy, 0);
824
825     DIAG_LOG(DIAG_DEBUG_PERIPHERALS, "p: %d t: %d considered closed\n",
826              fwd_info->peripheral, fwd_info->type);
827
828     return 0;
829 }
```

Exception Flow

fwd_info->ch_open = 0; //crash at here fwd_info 0x1f4

- Fwd_info=info->fwd_ctxt
  - Info=&socket_dci[peripheral]
  - Info=&socket_dci[4] the socket_dci over flow

Check Assembler Code

- Found pointer fwd_info not correct

```
B::v.f
Up  Down  Args  Locals  Caller  Task:
-000 diagfwd_channel_close(
       [R4] fwd_info = 0x01F4)
     [D:0xC0E45BC8] __func__ = (100, 105, 97, 103, 102, 119

-001 __socket_close_channel(inline)
-001 socket_close_channel(
       [R0] info = 0x01F4)
     [D:0xC0E45E88] __func__ = (115, 111, 99, 107, 101, 116
     [D:0xC0E45E70] __func__ = (95, 95, 115, 111, 99, 107,

-002 cntl_socket_process_msg_client(inline)
-002 cntl_socket_read_work_fn(
       ?)
     [NUD:0xCDBDBE90] msg = ([NUD:0xCDBDBE90] srv = ([NUD:0
     [NUD:0xCDBDBE88] iov = ([NUD:0xCDBDBE88] iov_base = 0x
     [NUD:0xCDBDBEA4] read_msg = ([NUD:0xCDBDBEA4] msg_name
     [D:0xC0E45EA0] __func__ = (99, 110, 116, 108, 95, 115,
     [D:0xC16E8EB0] descriptor = ([D:0xC16E8EB0] modname =
     [R8] info = 0xC15B4374
     [D:0xC0E45EBC] __func__ = (99, 110, 116, 108, 95, 115,
     [R8] info = 0xC15B4374
     [D:0xC0E45EDC] __func__ = (99, 110, 116, 108, 95, 115,

-003 static_key_false(inline)
-003 trace_workqueue_execute_end(inline)
-003 process_one_work(
       [R4] worker = 0xE889AD00,
       [R5] work = 0xECDCE750)
     [R9] pwq = 0xECF6D000
     [R6] pool = 0xEE012800
     [R8] cpu_intensive = FALSE
     [D:0xC16D9F19] __warned = FALSE
     [D:0xC16D9F1A] __warned = FALSE
     [D:0xC16D9F1B] __warned = FALSE

-004 worker_thread(
```

```
B::Register.view
N _ R0   01F4   R8   C15B4374   S  Stack
Z _ R1   FFFFFFD4   R9   3
C _ R2   0   R10   C15B4084
V _ R3   0   R11   0
Q _ R4   01F4   R12   0
    R5   CDBDBE90   R13   CDBDBE60
0 _ R6   C18966A0   R14   C0487358
1 _ R7   C1896E48   PC   C0484EF4
2 _ SPSR            CPSR   0
```

```
B::d.l
Step  Over  Diverge  Return  Up  Go  Break
addr/line   code     label   mnemonic        comment
819
NUR:C0484EEC  0A000028        beq     0xC0484F94
822
NUR:C0484EF0  E3A03000        mov     r3,#0x0          ; r3,#0
NUR:C0484EF4  E5C43004        strb    r3,[r4,#0x4]
823
NUR:C0484EF8  E594305C        ldr     r3,[r4,#0x5C]
NUR:C0484EFC  E3530000        cmp     r3,#0x0          ; r3,#0
NUR:C0484F00  0A000003        beq     0xC0484F14
NUR:C0484F04  E5933004        ldr     r3,[r3,#0x4]
NUR:C0484F08  E3530000        cmp     r3,#0x0          ; r3,#0
NUR:C0484F0C  0A000000        beq     0xC0484F14
824
NUR:C0484F10  E12FFF33        blx     r3
826
NUR:C0484F14  E5943050        ldr     r3,[r4,#0x50]
NUR:C0484F18  E3530000        cmp     r3,#0x0          ; r3,#0
NUR:C0484F1C  0A000003        beq     0xC0484F30
NUR:C0484F20  E5932000        ldr     r2,[r3]
NUR:C0484F24  E3520000        cmp     r2,#0x0          ; r2,#0
827
NUR:C0484F28  13A02000        movne   r2,#0x0          ; r2,#0
NUR:C0484F2C  15832010        strne   r2,[r3,#0x10]
```

Check Assembler Code (cont.)

- Info pointer not correct so fwd_ctxt.

```
-002 cntl_socket_process_msg_client(inline)
-002 cntl_socket_read_work_fn(
       ?)
     ⊞ [NUD:0xCDBDBE90] msg = ([NUD:0xCDBDBE90] srv = ([NUD:0xCDBDBE90] cmd
     ⊞ [NUD:0xCDBDBE88] iov = ([NUD:0xCDBDBE88] iov_base = 0xCDBDBE90, [NUD
     ⊞ [NUD:0xCDBDBEA4] read_msg = ([NUD:0xCDBDBEA4] msg_name = 0x0, [NUD:0
     ⊞ [D:0xC0E45EA0] __func__ = "cntl_socket_read_work_fn"
     ⊞ [D:0xC16E8EB0] descriptor = ([D:0xC16E8EB0] modname = 0xC1162A8E, [D
     ⊞ [R8] info = 0xC15B4374
     ⊞ [D:0xC0E45EBC] __func__ = (99, 110, 116, 108, 95, 115, 111, 99, 107,
     ⊟ [R8] info = 0xC15B4374 → (
       ▪ [NUD:0xC15B4374] peripheral = 0,
       ▪ [NUD:0xC15B4375] type = 0,
       ▪ [NUD:0xC15B4376] port_type = 0,
       ▪ [NUD:0xC15B4377] inited = 0,
       ⊞ [NUD:0xC15B4378] opened = ([NUD:0xC15B4378] counter = -559067475),
       ⊞ [NUD:0xC15B437C] diag_state = ([NUD:0xC15B437C] counter = -1),
       ▪ [NUD:0xC15B4380] pkt_len = 4294967295,
       ▪ [NUD:0xC15B4384] pkt_read = 500,
       ▪ [NUD:0xC15B4388] svc_id = 10,
       ▪ [NUD:0xC15B438C] ins_id = 0,
       ▪ [NUD:0xC15B4390] data_ready = 0,
       ⊞ [NUD:0xC15B4394] flow_cnt = ([NUD:0xC15B4394] counter = 0),
       ⊞ [NUD:0xC15B4398] name = (0, 0, 0, 0, 173, 78, 173, 222, 255, 255,
       ⊞ [NUD:0xC15B43B0] lock = ([NUD:0xC15B43B0] rlock = ([NUD:0xC15B43B0
       ⊞ [NUD:0xC15B43C0] wait_q = ([NUD:0xC15B43C0] lock = ([NUD:0xC15B43C
       ⊞ [NUD:0xC15B43D8] remote_addr = ([NUD:0xC15B43D8] family = 0, [NUD:
       ⊞ [NUD:0xC15B43EC] hdl = 0xFE89FE89,
       ⊞ [NUD:0xC15B43F0] wq = 0xDEAD4EAD,
       ⊞ [NUD:0xC15B43F4] init_work = ([NUD:0xC15B43F4] data = ([NUD:0xC15B
       ⊞ [NUD:0xC15B4404] read_work = ([NUD:0xC15B4404] data = ([NUD:0xC15B
       ⊞ [NUD:0xC15B4414] fwd_ctxt = 0x01F4,
       ⊞ [NUD:0xC15B4418] read_wait_q = ([NUD:0xC15B4418] lock = ([NUD:0xC1
     ⊞ [D:0xC0E45EDC] __func__ = (99, 110, 116, 108, 95, 115, 111, 99, 107,

-003 static_key_false(inline)
-003 trace_workqueue_execute_end(inline)
```

```
0 _ R6    C18966A0   R14   C0487358
1 _ R7    C1896E48   PC    C0484F0C
2 _ SPSR             CPSR        0
```

```
B::Var.View socket_dci
⊟ (static struct diag_socket_info [4]) [D:0xC15B4084] socket_
  ⊞ ((uint8_t) [D:0xC15B4084] peripheral = 0, (uint8_t) [D:0x
  ⊞ ((uint8_t) [D:0xC15B4140] peripheral = 1, (uint8_t) [D:0x
  ⊞ ((uint8_t) [D:0xC15B41FC] peripheral = 2, (uint8_t) [D:0x
  ⊞ ((uint8_t) [D:0xC15B42B8] peripheral = 3, (uint8_t) [D:0x
```

```
B::v.v (struct diag_socket_info *)socket_dci[4]
⊞ [D:0xC15B4374] (struct diag_socket_info *)socket_dci[4] = 0x0
```

```
NUR:C0484F24 E3520000   cmp    r2,#0x0      ; r2,#0
NUR:C0484F28 13A02000   movne  r2,#0x0      ; r2,#0
NUR:C0484F2C 15832010   strne  r2,[r3,#0x10]
```

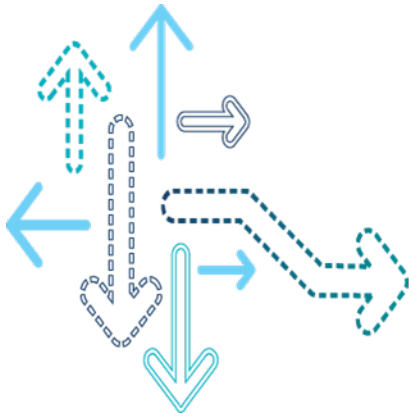
Check Code

- Confirm root cause

  Info= socket_dci[4] but in fact socket_dci array max is socket_dci[3] so over write.

  - Final fix https://us.codeaurora.org/cgit/quic/la/kernel/msm-3.18/commit/?h=LA.HB.1.1.1.c2&id=b6d359a6b7c31f48393b174542d2c717656b0e75

Analysis of Deadlock Issues



Deadlock Example 1

0237969

- Phenomenon: phone screen freeze, but adb shell still works
  - Use echo c> /proc/sysrq-trigger get a dump
  - Or use echo w>/proc/sysrq-trigger to get block task list
- Need to check all task list in disk sleep or running status
  - Check task call stack in mutex_lock
  - Check task call stack in spin_lock

Deadlock Example 1 (cont.)

```
**   50      2     50     0  DSleep  edab6400   100        0        0 kworker/u17:0
    [<c0d32474>] __schedule+0x594
    [<c0d31838>] schedule_timeout+0x24
    [<c0d32d4c>] wait_for_common+0x100
    [<c04016e4>] msm_rpm_smd_work+0x38
    [<c013e134>] process_one_work+0x280
    [<c013e4e8>] worker_thread+0x218
    [<c0143ba8>] kthread+0xa0
    [<c01062e0>] ret_from_fork+0x14
**   87      2     87     6  DSleep  ed329f40  RT50        0        0 irq/143-msm_iom
    [<c0d32474>] __schedule+0x594
    [<c0d327a4>] schedule_preempt_disabled+0x14
    [<c0d33f68>] __mutex_lock_slowpath+0x228
    [<c0d34168>] mutex_lock+0x20
    [<c06b6690>] cam_smmu_iommu_fault_handler+0xa4
    [<c0a513e8>] msm_iommu_secure_fault_handler_v2+0x33c
    [<c0176308>] irq_thread+0xcc
    [<c0143ba8>] kthread+0xa0
    [<c01062e0>] ret_from_fork+0x14
**  136      2    136     5  DSleep  ecd0d780  RT16        0        0 mdss_dsi_event
    [<c0d32474>] __schedule+0x594
    [<c03cbfd0>] dsi_event_thread+0xd0
    [<c0143ba8>] kthread+0xa0
    [<c01062e0>] ret_from_fork+0x14
**  381      1    381     5  DSleep  ed6cea40   112   129392    22212 surfaceflinger
    [<c0d32474>] __schedule+0x594
    [<c0d327a4>] schedule_preempt_disabled+0x14
    [<c0d33f68>] __mutex_lock_slowpath+0x228
    [<c0d34168>] mutex_lock+0x20
    [<c04a9f2c>] kgsl_iommu_flush_tlb_pt_current+0x30
    [<c04aa088>] kgsl_iommu_unmap+0x104
    [<c04a8240>] kgsl_mmu_unmap+0xa0
```

Deadlock Example 1 (cont.)

- Call stack

```
13637     1   2661    4  DSleep  deacb200   120   272512    44472 CAM_MctServ
   [<c0d32474>] __schedule+0x594
   [<c0d327a4>] schedule_preempt_disabled+0x14
   [<c0d33f68>] __mutex_lock_slowpath+0x228
   [<c0d34168>] mutex_lock+0x20
   [<c0a4d61c>] msm_iommu_detach_dev+0x1c
   [<c0113218>] arm_iommu_detach_device+0x30
   [<c06b6f40>] cam_smmu_detach+0x5c
   [<c06b7590>] cam_smmu_ops+0xd4
   [<c06dded0>] msm_cpp_subdev_ioctl+0xf10
   [<c06d896c>] msm_cpp_subdev_do_ioctl+0x1c0
   [<c0662a4c>] video_usercopy+0x200
   [<c065e158>] v4l2_ioctl+0x60
   [<c0216c58>] do_vfs_ioctl+0x4ac
   [<c0216da8>] sys_ioctl+0x50
   [<c0106240>] ret_fast_syscall+0x0

Code msm_iommu_detach_dev
   mutex_lock(&msm_iommu_lock);
   priv = domain->priv;
   if (!priv)
         goto unlock;
```

Deadlock Example 1 (cont.)

```
Task
**   87     2     87    6  DSleep  ed329f40  RT50       0       0 irq/143-msm_iom
    [<c0d32474>] __schedule+0x594
    [<c0d327a4>] schedule_preempt_disabled+0x14
    [<c0d33f68>] __mutex_lock_slowpath+0x228
    [<c0d34168>] mutex_lock+0x20
    [<c06b6690>] cam_smmu_iommu_fault_handler+0xa4
    [<c0a513e8>] msm_iommu_secure_fault_handler_v2+0x33c
    [<c0176308>] irq_thread+0xcc
    [<c0143ba8>] kthread+0xa0
    [<c01062e0>] ret_from_fork+0x14
Code in cam_smmu_iommu_fault_handler
                /* check wether it is in the table */
                for (i = 0; i < iommu_cb_set.cb_num; i++) {
                        if (!strcmp(iommu_cb_set.cb_info[i].name, cb_name)) {
                                mutex_lock(&iommu_cb_set.cb_info[i].lock);
                                if (!(iommu_cb_set.cb_info[i].fault_handler)) {
Code in msm_iommu_secure_fault_handler_v2
static void _iommu_lock_acquire(unsigned int need_extra_lock)
{
                mutex_lock(&msm_iommu_lock);
}
```

Deadlock Example 2

2346170

- Phenomenon:
  - LCD cannot be lightened when pressing power key on low-battery devices.
  - Devices out of 3 samples are seeing the issue
  - adb shell works; logcat /trace/dmesg are all available for debugging

Deadlock Example 2 (cont.)

```
[  94.907394] cfinteractive   D ffffffc000204c50 11864   301      2 0x00000000
[  94.914347] Call trace:
[  94.916780] [<ffffffc000204c50>] __switch_to+0x70/0x7c
[  94.921903] [<ffffffc000d54314>] __schedule+0x55c/0x784
[  94.927110] [<ffffffc000d545a0>] schedule+0x64/0x70
[  94.931972] [<ffffffc000d54908>] schedule_preempt_disabled+0x10/0x24
[  94.938308] [<ffffffc000d5581c>] __mutex_lock_slowpath+0x1bc/0x304
[  94.944471] [<ffffffc000d5598c>] mutex_lock+0x28/0x48
[  94.949507] [<ffffffc00022042c>] get_online_cpus+0x4c/0x70
[  94.954977] [<ffffffc00058a5e8>] cpr_regulator_set_voltage+0x19c/0x364
[  94.961486] [<ffffffc00058c7c8>] cpr_regulator_set_voltage_op+0x30/0x50
[  94.968086] [<ffffffc000575b48>] _regulator_do_set_voltage+0x104/0x35c
[  94.974593] [<ffffffc000575e64>] regulator_set_voltage+0xc4/0xf8
[  94.980585] [<ffffffc000a2c684>] update_vdd+0xc4/0x28c
[  94.985704] [<ffffffc000a2cf78>] unvote_vdd_level+0x78/0xb0
[  94.991258] [<ffffffc000a2cfd8>] unvote_rate_vdd+0x28/0x38
[  94.996729] [<ffffffc000a2d354>] clk_set_rate+0x18c/0x280
[  95.002113] [<ffffffc000a61004>] dev_target+0x88/0x94
[  95.007146] [<ffffffc000a59778>] update_devfreq+0xc0/0x114
[  95.012615] [<ffffffc000a5b62c>] update_node+0x44/0xb8
[  95.017736] [<ffffffc000a5b6cc>] update_all_devfreqs+0x2c/0x44
[  95.023552] [<ffffffc000a5bb24>] cpufreq_trans_notifier+0x54/0x74
[  95.029629] [<ffffffc000242898>] notifier_call_chain+0x44/0x80
[  95.035444] [<ffffffc000242d2c>] __srcu_notifier_call_chain+0x48/0x70
[  95.041868] [<ffffffc000242d64>] srcu_notifier_call_chain+0x10/0x1c
[  95.048120] [<ffffffc0009283bc>] cpufreq_notify_transition+0x1e4/0x224
[  95.054630] [<ffffffc0009321f0>] set_cpu_freq.isra.0+0xf8/0x178
[  95.060532] [<ffffffc00093238c>] msm_cpufreq_target+0x11c/0x158
[  95.066434] [<ffffffc000928978>] __cpufreq_driver_target+0x90/0x20c
[  95.072684] [<ffffffc00092f31c>] cpufreq_interactive_speedchange_task+0x244/0x314
```

- blocked cpu_hotplug.lock in get_online_cpus

Hold l2_clk.prepare_lock

Deadlock Example 2 (cont.)

PowerManagerService is also blocked by the mutex and have the lock 0x06ba578e

```
"PowerManagerService" prio=5 tid=18 Native
  | group="main" sCount=1 dsCount=0 obj=0x12e96f90 self=0x556dee2bf0
  | sysTid=1686 nice=-4 cgrp=default sched=0/0 handle=0x7f6ba42450
  | state=D schedstat=( 306646035 97057298 676 ) utm=12 stm=18 core=3 HZ=100
  | stack=0x7f6b940000-0x7f6b942000 stackSize=1037KB
  | held mutexes=
  kernel: __switch_to+0x70/0x7c
  kernel: get_online_cpus+0x4c/0x70
  kernel: show+0x20/0xc0
  kernel: sysfs_read_file+0xe0/0x170
  kernel: vfs_read+0xa0/0x12c
  kernel: SyS_read+0x44/0x74
  kernel: cpu_switch_to+0x48/0x4c
  native: (backtrace::Unwind failed for thread 1686)
  at com.android.server.power.PowerManagerService.nativeSetInteractive(Native method)
  at com.android.server.power.PowerManagerService.setHalInteractiveModeLocked(PowerManagerService.java:2214)
  at com.android.server.power.PowerManagerService.updateSuspendBlockerLocked(PowerManagerService.java:2146)
  at com.android.server.power.PowerManagerService.updatePowerStateLocked(PowerManagerService.java:1321)
  at com.android.server.power.PowerManagerService.access$1100(PowerManagerService.java:89)
  at com.android.server.power.PowerManagerService$2.onStateChanged(PowerManagerService.java:2033)
  - locked <0x06ba578e> (a java.lang.Object)
  at com.android.server.display.DisplayPowerController$4.run(DisplayPowerController.java:1030)
  at android.os.Handler.handleCallback(Handler.java:739)
  at android.os.Handler.dispatchMessage(Handler.java:95)
  at android.os.Looper.loop(Looper.java:148)
  at android.os.HandlerThread.run(HandlerThread.java:61)
  at com.android.server.ServiceThread.run(ServiceThread.java:46)
```

Block by same mutex

Hold this java lock

Deadlock Example 2 (cont.)

- Other tasks is blocked by the lock 0x06ba578e

```
"android.ui" prio=5 tid=12 Blocked
  | group="main" sCount=1 dsCount=0 obj=0x12ce0580 self=0x556deb89a0
  | sysTid=1491 nice=-2 cgrp=default sched=0/0 handle=0x7f6c060450
  | state=S schedstat=( 241992623 112246205 1045 ) utm=14 stm=10 core=0 HZ=100
  | stack=0x7f6bf5e000-0x7f6bf60000 stackSize=1037KB
  | held mutexes=
  at com.android.server.power.PowerManagerService.uidGoneInternal(PowerManagerService.java:2380)
  - waiting to lock <0x06ba578e> (a java.lang.Object) held by thread 18
  at
    com.android.server.power.PowerManagerService$LocalService.uidGone(PowerManagerService.java:358
    4)
  at
    com.android.server.am.ActivityManagerService.dispatchUidsChanged(ActivityManagerService.java:3858)
  at com.android.server.am.ActivityManagerService.access$600(ActivityManagerService.java:268)
  at
    com.android.server.am.ActivityManagerService$UiHandler.handleMessage(ActivityManagerService.java:1
    638)
  at android.os.Handler.dispatchMessage(Handler.java:102)
  at android.os.Looper.loop(Looper.java:148)
  at android.os.HandlerThread.run(HandlerThread.java:61)
  at com.android.server.ServiceThread.run(ServiceThread.java:46)
```

Deadlock Example 2 (cont.)

```
"android.fg" prio=5 tid=13 Blocked
  | group="main" sCount=1 dsCount=0 obj=0x12ce0660 self=0x556deb91f0
  | sysTid=1493 nice=0 cgrp=default sched=0/0 handle=0x7f6bf5b450
  | state=S schedstat=( 23066867 8480575 47 ) utm=2 stm=0 core=1
HZ=100
  | stack=0x7f6be59000-0x7f6be5b000 stackSize=1037KB
  | held mutexes=
  at
com.android.server.power.PowerManagerService.monitor(PowerManagerS
ervice.java:2597)
  - waiting to lock <0x06ba578e> (a java.lang.Object) held by thread 18
  at com.android.server.Watchdog$HandlerChecker.run(Watchdog.java:175)
  at android.os.Handler.handleCallback(Handler.java:739)
  at android.os.Handler.dispatchMessage(Handler.java:95)
  at android.os.Looper.loop(Looper.java:148)
  at android.os.HandlerThread.run(HandlerThread.java:61)
  at com.android.server.ServiceThread.run(ServiceThread.java:46)
```

Deadlock Example 2 (cont.)

```
[ 6437.521891] msm_thermal:hot D ffffffc000204c50 0 321 2 0x00000000
[ 6437.521909] Call trace:
[ 6437.521922] [<ffffffc000204c50>] __switch_to+0x70/0x7c
[ 6437.521936] [<ffffffc000c33f84>] __schedule+0x558/0x75c
[ 6437.521949] [<ffffffc000c341ec>] schedule+0x64/0x70
[ 6437.521962] [<ffffffc000c34540>] schedule_preempt_disabled+0x20/0x3c
[ 6437.521976] [<ffffffc000c34fec>] __mutex_lock_slowpath+0x180/0x1f8
[ 6437.521989] [<ffffffc000c3508c>] mutex_lock+0x28/0x48
[ 6437.522002] [<ffffffc000969c74>] clk_unprepare+0x30/0xdc
[ 6437.522017] [<ffffffc00088ad2c>] msm_cpufreq_cpu_callback+0x78/0xe4
[ 6437.522030] [<ffffffc000241534>] notifier_call_chain+0x44/0x80
[ 6437.522043] [<ffffffc0002415a0>] __raw_notifier_call_chain+0x8/0x14
[ 6437.522057] [<ffffffc00021ff84>] __cpu_notify+0x2c/0x50
[ 6437.522073] [<ffffffc000c2acfc>] _cpu_up+0x154/0x1cc
[ 6437.522086] [<ffffffc000c2add4>] cpu_up+0x60/0x7c
[ 6437.522100] [<ffffffc000c296c8>] update_offline_cores+0x24c/0x3cc
[ 6437.522114] [<ffffffc000c29c38>] do_hotplug+0x1cc/0x2bc
[ 6437.522128] [<ffffffc00023d6d0>] kthread+0xac/0xb8
```

Blocked by l2_clk.prepare_clk

Hold cpu_hotplug.lock mute

Deadlock Example 2 - Summary

- Many task blocked by the same mutex
- Could find a cycle loop of lock waiting
- If debug switch is open, could find current owner of the lock
- If debug switch is not open, need check code call path

Fix

https://us.codeaurora.org/cgit/quic/la//kernel/msm-3.10/commit/?id=8b070e64db7cecd0c10d92a73b38230dc0f58e4d

Deadlock Example 3

2465640

- Phenomenon
  - Crash
  - Non-secure watchdog bite

Deadlock Example 3 - log1

- Watchdog Context
  ==========
  Non-secure watchdog timeout on CPU 2 in function _raw_spin_lock()
  CPU |World |Received WDT Int? |Received SGI Int? |In Warm Boot? |WDT Status
  0 |non-secure world |no |yes |no |0x600000B3
  1 |secure world |no |no |no |0xFE82600C
  2 |non-secure world |yes |no |no |0xFE80BBE9
  3 |non-secure world |no |yes |no |0x0

- CPU 0 Call Stacks:
  -000|_raw_spin_lock()
  -001|vprintk_emit()
  -002|printk()
  -003|__dev_printk()
  -004|dev_err()
  -005|qup_i2c_interrupt()
  -006|handle_irq_event_percpu()
  -007|handle_irq_event()
  -008|handle_fasteoi_irq()
  -009|generic_handle_irq()
  -010|handle_IRQ()
  -011|gic_handle_irq()
  -012|__irq_svc()
  -013|format_decode()
  -014|vsnprintf()
  -015|sprintf()
  -016|devkmsg_read()
  -017|vfs_read()
  -018|sys_read()
  -019|ret_fast_syscall()

CPU 2 Call Stacks:
-000|_raw_spin_lock()
-001|vprintk_emit()
-002|printk()
-003|tcp_connect()
-004|tcp_v4_connect()
-005|tcp_v6_connect()
-006|inet_stream_connect()
-007|sys_connect()
-008|ret_fast_syscall()
----|end of frame

```
asmlinkage int vprintk_emit
  ,..
  local_irq_save(flags);
  …
  lockdep_off();
  raw_spin_lock(&logbuf_lock);
  logbuf_cpu = this_cpu;
```

CPU 3 Call Stacks:
-000|smp_call_function_many()
-001|uncached_logk_pc()
-002|contextidr_notifier()
-003|notifier_call_chain()
-
004|__atomic_notifier_call_chain()
-005|atomic_notifier_call_chain()
-006|__switch_to()
----|end of frame

Deadlock Example 3 - log2

- CP1 call stack:
  -000|__kmalloc(size = 0, flags = 0)
  -001|ext4_ext_find_extent(inode = 0xD2D8B990, block = 882, ?)
  -002|ext4_ext_map_blocks(?, inode = 0xD2D8B990, map = 0xF315FC38, ?)
  -003|cyc_to_sched_clock(inline)
  -003|sched_clock()
  -004|cyc_to_sched_clock(inline)
  -004|sched_clock()
  -005|msm_rtb_write_timestamp(inline)
  -005|uncached_logk_pc_idx(inline)
  -005|uncached_logk_pc(log_type = LOGK_READL, caller = 0xC074CC00, data = 0xC5DEE930, caller = 0xC074CC00,
  -006|sdhci_readl(inline)
  -006|sdhci_irq(?, dev_id = 0xF56DCCC0)
  -007|printk(fmt = 0xC0D5EFA7)
  -008|__dev_printk(level = 0xC0CE3A75, dev = 0xF612FA08, vaf = 0xF315FD30)
  -009|dev_err(?, fmt = 0xC0DA64B2)
  -010|qup_i2c_interrupt(irq = 132, devid = 0xF622A400)
  -011|handle_irq_event_percpu(desc = 0xF6010480, action = 0xF6242400)
  -012|handle_irq_event(desc = 0xF6010480)
  -013|handle_fasteoi_irq(?, desc = 0xF6010480)
  -014|generic_handle_irq(irq = 132)
  -015|current_thread_info(inline)
  -015|set_irq_regs(inline)
  -015|set_irq_regs(inline)
  -015|handle_IRQ(irq = 132, ?)
  -016|gic_handle_irq(regs = 0xF315FE38)
  -017|__irq_svc(asm)
  -->|exception
  -018|format_decode(fmt = 0xC0D0F85D, spec = 0xF315FEB0)
  -019|vsnprintf(buf = 0xF36CC020, ?, fmt = 0xF5FFD880, ?)
  -020|sprintf(?, fmt = 0xC0D0F85D)
  -021|devkmsg_read(?, buf = 0x0, count = 3067129856, ?)
  -022|vfs_read(?, buf = 0xB6D0B000, ?, pos = 0xF315FF88)
  -023|file_pos_write(inline)
  -023|sys_read(?, buf = 0xB6D0B000, ?)
  -024|ret_fast_syscall(asm)
  -->|exception
  -025|NUR:0xB6E1B380(asm)
  ---|end of frame

Deadlock Example 3 - Summary

- On SMP, same spin_lock can not be called more than once.
- If a lock could be used in irq/softirq, need to use spin_lock_irq
- devkmsg_read use logbuf_lock, printk can be called in irq handler

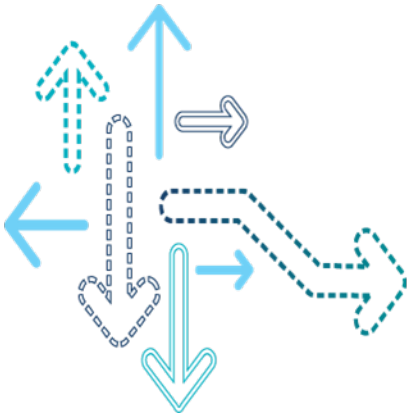
Fix:

```
@@ -430,20 +430,20 @@ static ssize_t devkmsg_read(struct file *file, char __user *buf,
ret = mutex_lock_interruptible(&user->lock);
if (ret)
return ret;
- raw_spin_lock(&logbuf_lock);
+ raw_spin_lock_irq(&logbuf_lock);
```

Full patch:

- https://us.codeaurora.org/cgit/quic/la/kernel/msm-3.18/commit/kernel/printk.c?h=LA.BR.1.1.3.c3-06600-8x16.0&id=5c53d819c71c63fdc91f30a59164583f68e2d63a

Analysis of Userspace Task Lockup Issues



Userspace Task Lock Example

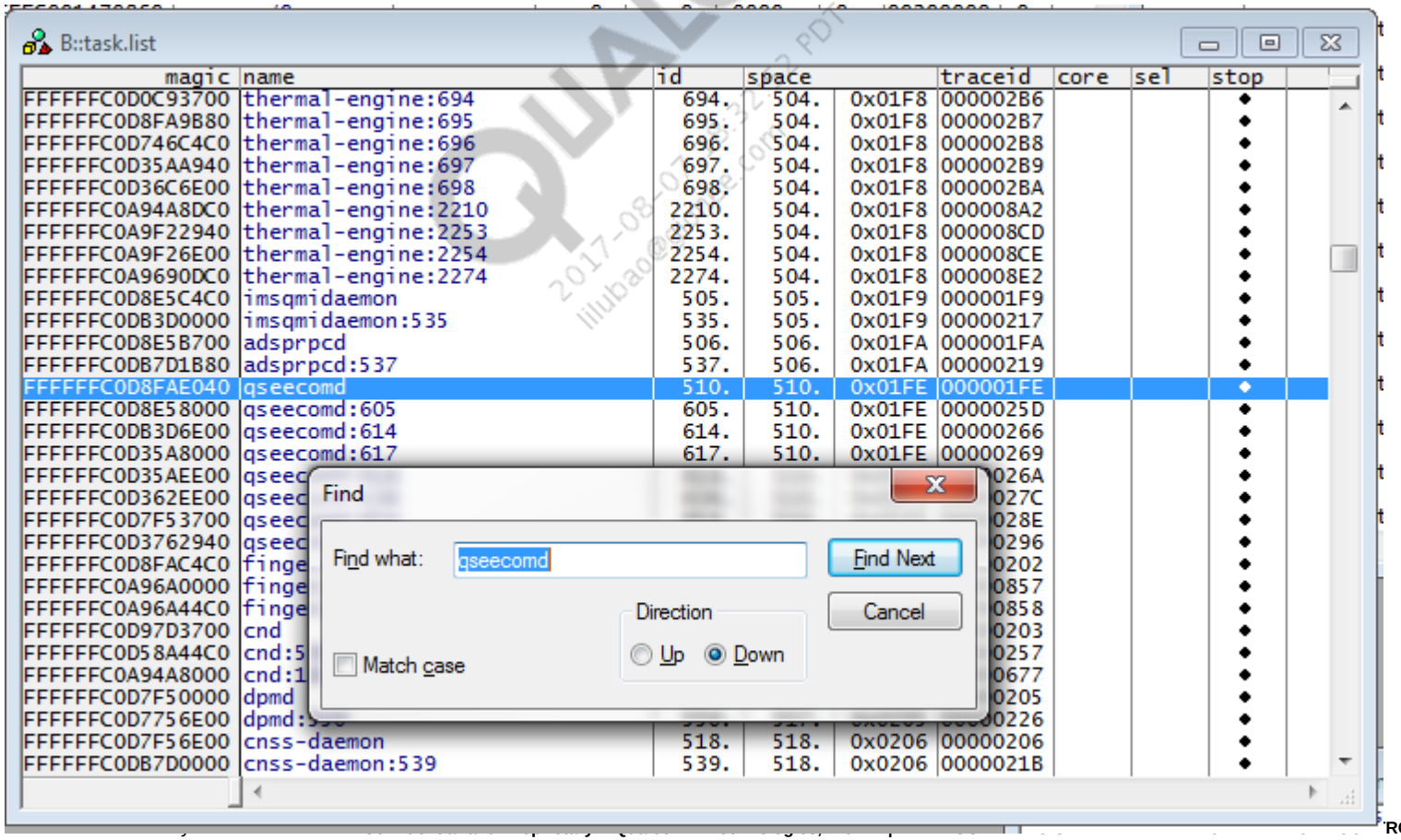
0370348

- Normally a process hang is due to the deadlocks caused by the programming error, such as when unlock is not done. These types of problems are difficult to debug in large systems, but if one dumps the stack trace of the hang process then it is easier to find out all the threads blocked on a lock; this is enough to get an idea about the lock that caused the deadlock, then it’s all about the code walk-through and analysis. So, for userspace task lock, the main task is to get this process userspace task stack.
- When the user process is found to hang, logcat is needed to get the process PID, and use sysrq to get ramdump

Userspace Task Lock Example (cont.)

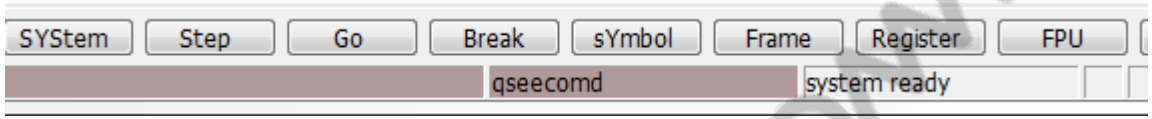
1. Execute task.list to get task list, then double-click selected task; it will switch to context of qseecomd
2. Get failed space task's PID from logcat

   In this example was found qseecomd:510 cannot "fread" a file



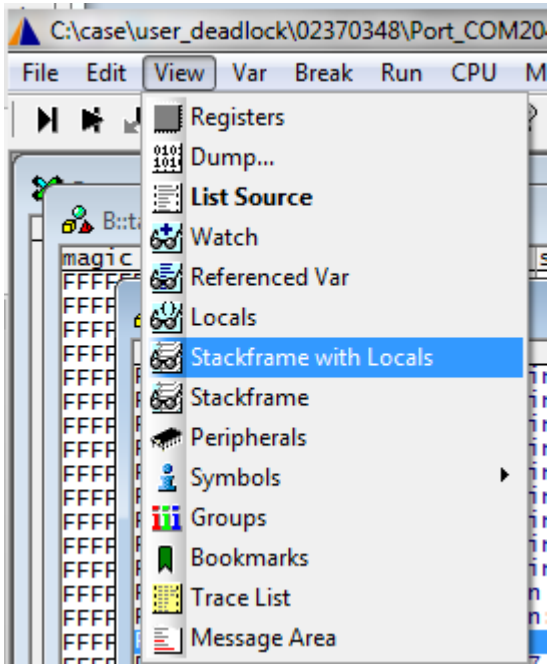
Userspace Task Lock Example (cont.)

1. Make sure the task is qseecomd in the status bar



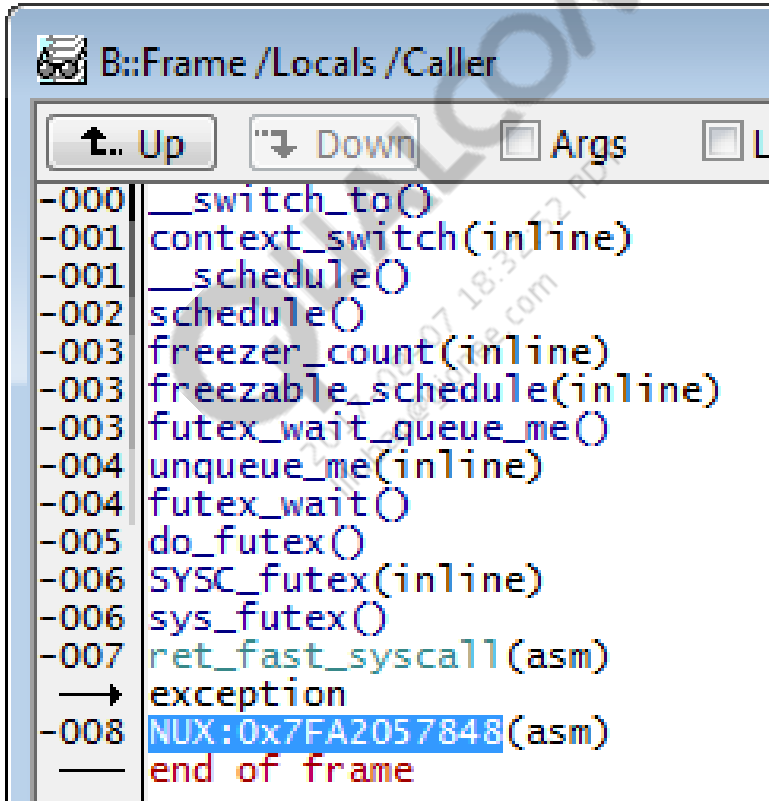
2. Get task stack of qseecomd in kernel space by View:Stackframe with local
3. If the task address is known, use this command:

   v.frame /task 0xffffffc0d8fae040



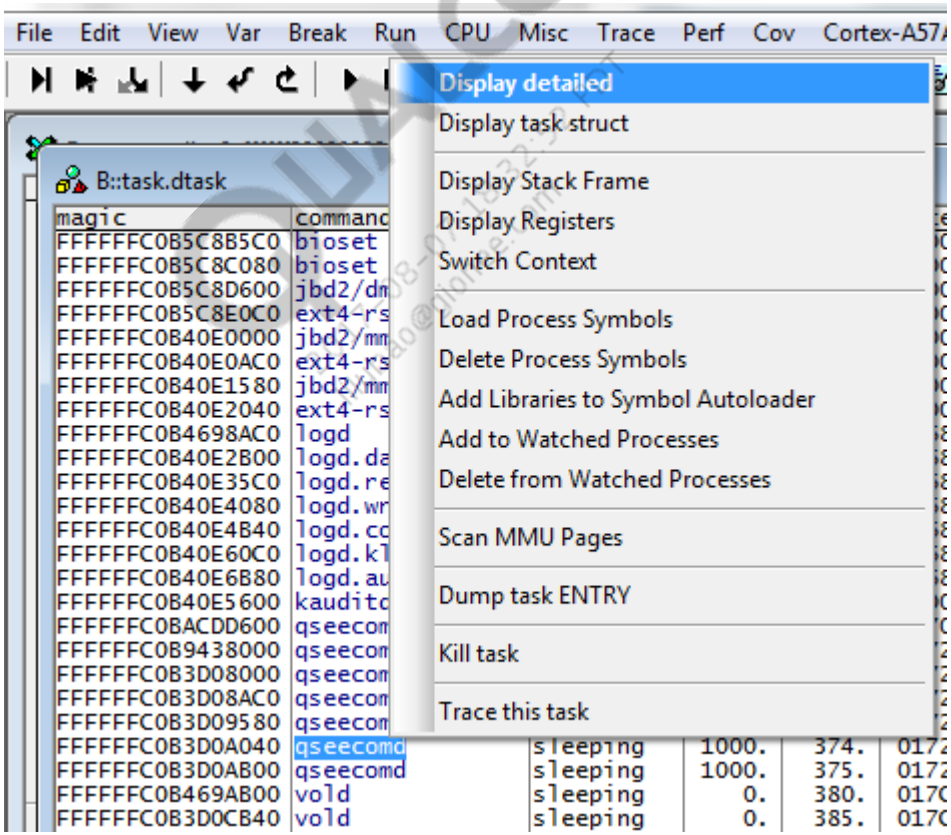
Userspace Task Lock Example (cont.)

- The line below “exception” is a userspace function; 0x7FA2057848 is the userspace address



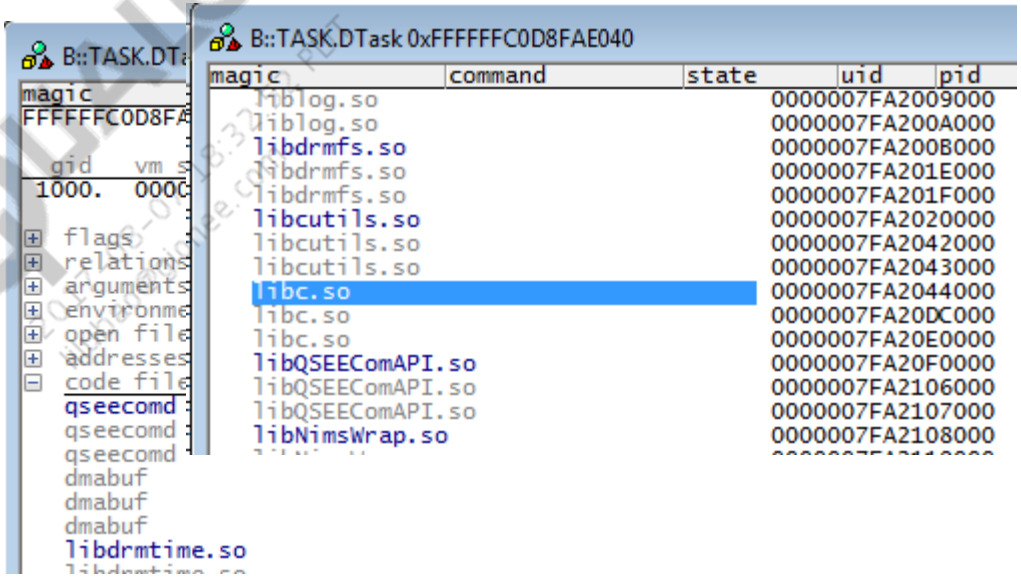
Userspace Task Lock Example (cont.)

1. Load user task symbol file, like libc.so; first get the libc.so mapping address in the user task
2. Select the task, right-click, then select Display detailed



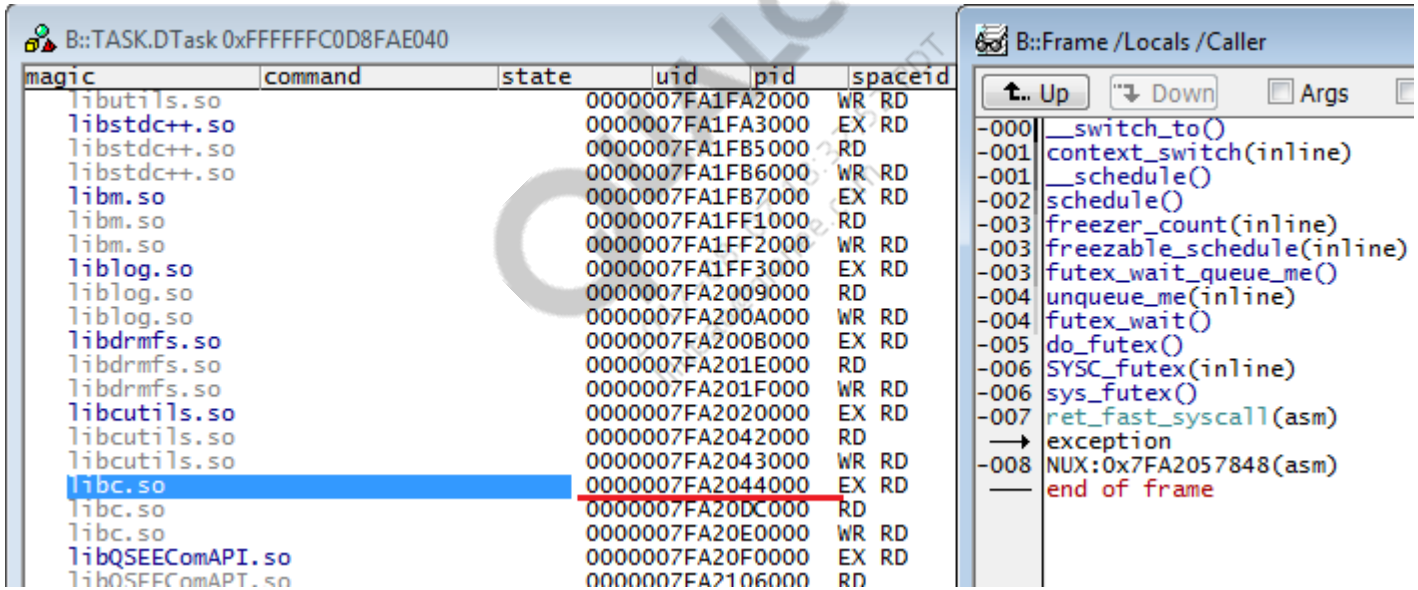
Userspace Task Lock Example (cont.)

- Unclasp “code file” and find which lib is mapping into 0x7FA2057848; found libc.so; need symbol file



Userspace Task Lock Example (cont.)

- Data.LOAD C:\case\user_deadlock\02370348\out\target\product\msm8952_64\symbols\system\lib64\libc.so 0x7fa2044000 /nocode /noclear

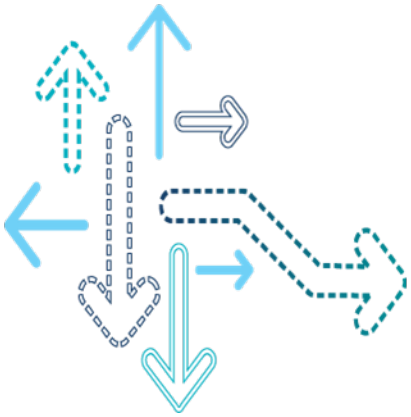


Userspace Task Lock Example (cont.)

1. Get the process stack of the user space
   - QSEECOMD is locked at pthread_mutex_lock, as there is no owner member in pthread_mutex_lock, we have to check all QSEECOMD task stack and to find out who is holding pthread_mutex_lock.
2. If there are any QSEECOMD, use ramdump parser tools to get task list, then check all QSEECOMD kernel space tasks

```
-->|exception
-008|syscall(asm)
-009|__futex(inline)
-009|__futex_wait_ex(inline)
-009|normal_lock(inline)
-009|pthread_mutex_lock(
    |     (register pthread_mutex_t *) mutex = 0x0000007FA20E5D58 -> (
    |       (int) value = 2,  //blocked by this mutex. I don't which proces
    |       (char [36]) __reserved = (0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
    |   (register int) shared = 0
    |   (register int) shared = 0
    |   (register pthread_mutex_t *) mutex = 0x0000007FA20E5D58
    |   (register int) unlocked = 0
    |   (register int32_t *) ptr = 0x0000007FA20E5D58
    |   (register int32_t) old_value = 0
    |   (register int) value = 2
    |   (register void *) ftx = 0x0000007FA20E5D58
    |   (register int) value = 2
    |   (auto int) op = 4096
    |   (register void *) ftx = 0x0000007FA20E5D58
    |   (register int) saved_errno = 2
    |   (register int) result = 0
    |   (register int32_t *) ptr = 0x0000007FA20E5D58
    |   (register int32_t) new_value = 2
-010|__atexit_register_cleanup(
    |     (register void (*)()) func = 0x0000007FA20AED64)
-011|__smakebuf(
    |     (register FILE *) fp = 0x0000007FA20ECB30)
    |   (register void *) p = 0x000000559A933850
    |   (register int) flags = 1152
    |   (auto size_t) size = 16384
    |   (auto int) couldbetty = 0
-012|__srefill(
    |     (register FILE *) fp = 0x0000007FA20ECB30)
-013|fread(
    |     (register size_t) size = 64,
    |     (register size_t) count = 1,
    |     (register FILE *) fp = 0x0000007FA20ECB30)
    |   (register size_t) resid = 64
```

Analysis of Memory Overflow Issues



Repro Steps

02187943

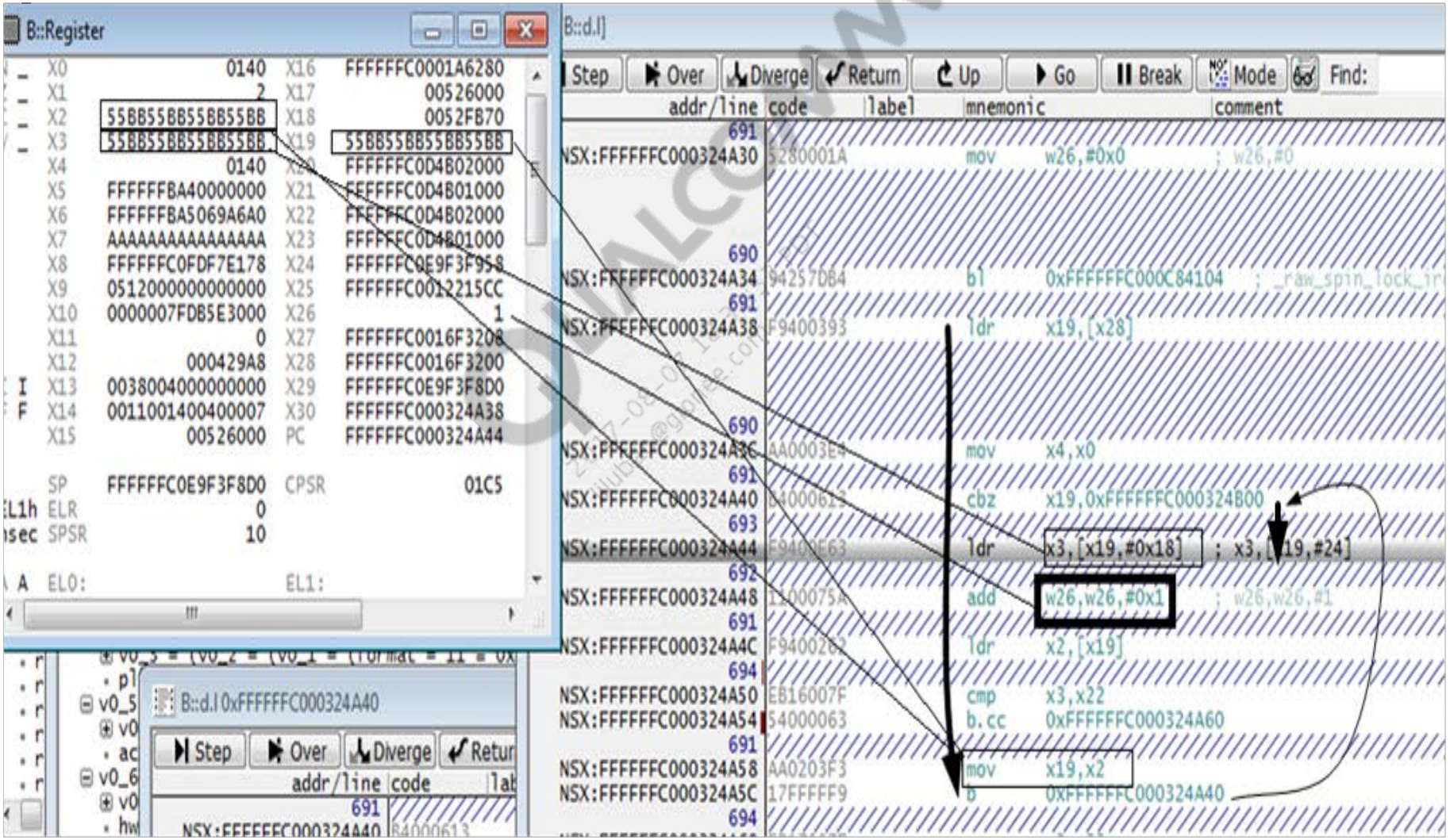
- Copy file to DUT by USB MTP function device
- Set the content of file (words of 0x55bb55bb)
- Boot the phone with USB cable plugged to PC
- Copy file from phone to PC by MTP device

Memory Overflow Example

- [110.732382] Unable to handle kernel paging request at virtual address 55bb55bb55bb55d3

- [110.732659] Hardware name: Qualcomm Technologies, Inc. MSM 8996 v3 + PMI8994 CDP (DT)
- [110.732689] task: ffffffc071140e00 ti: ffffffc0e9f3c000 task.ti: ffffffc0e9f3c000
- [110.732755] PC is at debug_check_no_obj_freed+0x84/0x1c8
- [110.732790] LR is at debug_check_no_obj_freed+0x78/0x1c8
- [110.732816] pc : [<ffffffc000324a44>] lr : [<ffffffc000324a38>] pstate: 000001c5
- [110.732842] sp : ffffffc0e9f3f8d0
- [110.732862] x29: ffffffc0e9f3f8d0 x28: ffffffc0016f3200
- [110.732903] x27: ffffffc0016f3208 x26: 0000000000000001
- [110.732941] x25: ffffffc0012215cc x24: ffffffc0e9f3f958
- [110.732979] x23: ffffffc0d4b01000 x22: ffffffc0d4b02000
- [110.733014] x21: ffffffc0d4b01000 x20: ffffffc0d4b02000
- [110.733051] x19: 55bb55bb55bb55bb x18: 000000000052fb70
- [110.733087] x17: 0000000000526000 x16: ffffffc0001a6280
- [110.733122] x15: 0000000000526000 x14: 0011001400400007
- [110.733158] x13: 0038004000000000 x12: 00000000000429a8
- [110.733193] x11: 0000000080000000 x10: 0000007fdb5e3000
- [110.733230] x9 : 0512000000000000 x8 : ffffffc0fdf7e178
- [110.733265] x7 : aaaaaaaaaaaaaaaa x6 : ffffffba5069a6a0
- [110.733300] x5 : ffffffba40000000 x4 : 0000000000000140
- [110.733334] x3 : 55bb55bb55bb55bb x2 : 55bb55bb55bb55bb
- [110.733370] x1 : 0000000000000002 x0 : 0000000000000140

Analyze the Panic Context



Analyze the Panic Context (cont.)

- 0xffffffc000324a38 <+120>: ldr x19, [x28]
- 0xffffffc000324a3c <+124>: mov x4, x0
- **0xffffffc000324a40 <+128>: cbz x19, 0xffffffc000324b00 <debug_check_no_obj_freed+320>**
- **0xffffffc000324a44 <+132>: ldr x3, [x19,#24]**
- 0xffffffc000324a48 <+136>: add w26, w26, #0x1
- 0xffffffc000324a4c <+140>: ldr x2, [x19]
- 0xffffffc000324a50 <+144>: cmp x3, x22
- 0xffffffc000324a54 <+148>: b.cc 0xffffffc000324a60 <debug_check_no_obj_freed+160>
- 0xffffffc000324a58 <+152>: mov x19, x2
- **0xffffffc000324a5c <+156>: b 0xffffffc000324a40 <debug_check_no_obj_f-**

- x28=0xFFFFFFC0016F3200 , [0x28]=0xffffffc071355870 , [[0x28]]=0x55bb55bb55bb

- x3=[x19,#0x18]=[[x28],#0x18]=0x55bb55bb55bb
- x2=[x19]=[[28]]=0x55bb55bb55bb
- x19=x2=0x55bb55bb
- [x19,#0x18] = 0x55bb55bb55bb55d3

- **Unable to handle kernel paging request at virtual address 55bb55bb55bb55d3**

Analyze the MTP File Transfer Context

1. Allocate 8 buffers for the usb gadget request

```
static struct usb_request *mtp_request_new(struct usb_ep *ep, int buffer_size)
{…; req->buf = kmalloc(buffer_size, GFP_KERNEL);…}
```

```
0xFFFFFFC071334000->0xFFFFFFC071337FFF
0xFFFFFFC071338000->0xFFFFFFC07133BFFF
0xFFFFFFC07133c000->0xFFFFFFC07133FFFF
0xFFFFFFC071340000->0xFFFFFFC071343FFF
0xFFFFFFC071344000->0xFFFFFFC071347FFF
0xFFFFFFC071348000->0xFFFFFFC07134BFFF
0xFFFFFFC07134c000->0xFFFFFFC07134FFFF
0xFFFFFFC071350000->0xFFFFFFC071353FFF
```

- Each buffer size is 16384 bytes by default

```
* #define MTP_BULK_BUFFER_SIZE       16384
* unsigned int mtp_tx_req_len = MTP_BULK_BUFFER_SIZE;
```

Analyze the MTP File Transfer Context (cont.)

2. Dynamic change mtp_tx_req_len from 16384 to 131072 in init.qcom.usb.sh

```
echo 131072 > /sys/module/g_android/parameters/mtp_tx_req_len
echo 131072 > /sys/module/g_android/parameters/mtp_rx_req_len
```

3. /* read from a local file and write to USB */

static void send_file_work(struct work_struct *data)

will result in [0xFFFFFFC071334000, 0xFFFFFFC07136FFFF] all filled by file content (0x55bb55bb) .

```
_______________address|_______0_______4_______8_______C_0123456789ABCDEF
NSD:FFFFFFC071354000|>55BB55BB 55BB55BB 55BB55BB 55BB55BB .U.U.U.U.U.U.U.U
NSD:FFFFFFC071354010| 55BB55BB 55BB55BB 55BB55BB 55BB55BB .U.U.U.U.U.U.U.U
NSD:FFFFFFC071354020| 55BB55BB 55BB55BB 55BB55BB 55BB55BB .U.U.U.U.U.U.U.U
NSD:FFFFFFC071354030| 55BB55BB 55BB55BB 55BB55BB 55BB55BB .U.U.U.U.U.U.U.U
NSD:FFFFFFC071354040| 55BB55BB 55BB55BB 55BB55BB 55BB55BB .U.U.U.U.U.U.U.U
NSD:FFFFFFC071354050| 55BB55BB 55BB55BB 55BB55BB 55BB55BB .U.U.U.U.U.U.U.U
NSD:FFFFFFC071354060| 55BB55BB 55BB55BB 55BB55BB 55BB55BB .U.U.U.U.U.U.U.U
NSD:FFFFFFC071354070| 55BB55BB 55BB55BB 55BB55BB 55BB55BB .U.U.U.U.U.U.U.U
NSD:FFFFFFC071354080| 55BB55BB 55BB55BB 55BB55BB 55BB55BB .U.U.U.U.U.U.U.U
NSD:FFFFFFC071354090| 55BB55BB 55BB55BB 55BB55BB 55BB55BB .U.U.U.U.U.U.U.U
NSD:FFFFFFC0713540A0| 55BB55BB 55BB55BB 55BB55BB 55BB55BB .U.U.U.U.U.U.U.U
```

Analyze the MTP File Transfer Context (cont.)

- [0xFFFFFFC071334000, 0xFFFFFFC071353FFF] belongs to MTP USB request buffers
- [0xFFFFFFC071334000, 0xFFFFFFC07136FFFF] is filled by 0x55BB
- So, [0xFFFFFFC071354000, 0xFFFFFFC07136FFFF] are overwritten by 0x55BB; the module accessing a memory address located in this range will get an incorrect value.

For example:

- When memory address 0xffffffc071355870 is accessed, it gets the value 0x55bb55bb55bb55bb, because 0xFFFFFFC071355870 is located at the overwritten memory range [0xFFFFFFC0071354000->0xFFFFFFC07136FFFF]

x28=0xFFFFFFC0016F3200

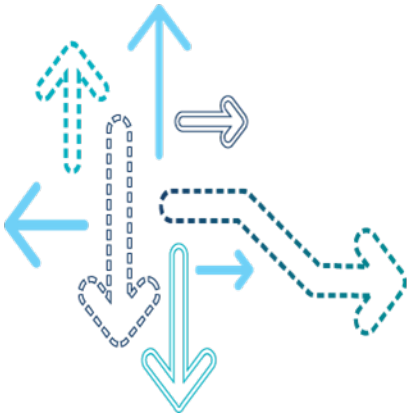
[0x28]=0xffffffc071355870

[[0x28]]=0x55bb55bb55bb

Summary

- USB request buffer allocated 16384 bytes, but uses it as 131072 bytes, which results in memory overflow
- When reading assembler code, need to consider the code flow jump

Analysis of Watchdog Bark Issues



Dmesg Log and Watchdog Driver Data

02187943

- Dmesg log:

```
[   30.628216] dhd_prot_ioctl: status ret value is -5
[   30.630816] dhd_preinit_ioctls lpc fail WL_DOWN : 0, lpc = 1
[   30.632130] dhd_preinit_ioctls Set lpc ret --> 0
[   30.634401] dhd_prot_ioctl: status ret value is -23
[   30.655547] Firmware version = wl0: Sep 23 2015 20:45:47 version 7.112.100.34 (r588403) FWID 01-ed85b532
[   30.658240] dhd_prot_ioctl: status ret value is -14
[   33.941396] msm_thermal:update_cpu_freq Unable to update policy for cpu:4. err:-16
[   42.238696] Watchdog bark! Now = 42.238687
[   42.238719] Watchdog last pet at 31.238399
```

- Watchdog driver data:

```
B::v.v wdog_data
⊟ wdog_data = 0xFFFFFFC00E9FB100 ≙ __bss_stop+0xCE78758 → (
  ▪ phys_base = 4177620992,
  ▪ size = 4096,
  ⊞ base = 0xFFFFFF8000836000 ≙ ,
  ⊞ wdog_absent_base = 0x0 ≙ ,
  ⊞ dev = 0xFFFFFFC0B950F910 ≙ __bss_stop+0xB798CF68,
  ▪ pet_time = 10000,
  ▪ bark_time = 11000,
  ▪ bark_irq = 35,
  ▪ bite_irq = 36,
  ▪ do_ipi_ping = TRUE,
  ▪ last_pet = 31,
  ▪ min_slack_ticks = 29646,
  ▪ min_slack_ns = 464277520,
  ⊞ scm_regsave = 0x0 ≙ ,
  ⊞ alive_mask = (bits = (255)),
  ⊞ disable_lock = (count = (counter = 1), wait_lock = (rlock = (raw_lock
  ⊞ init_dogwork_struct = (data = (counter = 96), entry = (next = 0xFFFFF
  ⊞ dogwork_struct = (work = (data = (counter = -271752222715), entry = (
  ▪ irq_ppi = FALSE,
  ⊞ wdog_cpu_dd = 0x0 ≙ ,
  ⊞ panic_blk = (notifier_call = 0xFFFFFFC0005049C8 ≙ panic_wdog_handler,
  ▪ enabled = TRUE)
```

Timer List and Work Queue

- Timer list:

```
    Jiffies:4294941650    timer_jiffies:4294940366     next_timer:4294940366     running_timer:0
====timer tv1 @ffffffc0019c4b38 ====
149.00 expire: 4294940565  defer:0 data: ffffffc0b6fc0360 callback: mdss_fb_no_update_notify_timer_cb
207.00 expire: 4294940367  defer:1 data:        0 callback: cpufreq_interactive_timer
240.00 expire: 4294940400  defer:1 data: ffffffc0ba4ce588 callback: delayed_work_timer_fn  vmstat_update work_data:       1
243.00 expire: 4294940403  defer:0 data: ffffffc0b565d508 callback: pm_wakeup_timer_fn

====timer tv2 @ffffffc0019c5b38 ====
 23.00 expire: 4294940576  defer:0 data: ffffffc0b642dc88 callback: delayed_work_timer_fn  bdi_writeback_workfn work_data:
 23.01 expire: 4294940624  defer:0 data:        0 callback: cpufreq_interactive_nop_timer
 24.00 expire: 4294940766  defer:0 data: ffffffc0b681e1d8 callback: delayed_work_timer_fn  check_dsi_ctrl_status work_data:
 24.01 expire: 4294940902  defer:0 data: ffffffc0a440e200 callback: commit_timeout
 25.00 expire: 4294941176  defer:0 data: ffffffc0b6785200 callback: delayed_work_timer_fn  update_temp_data work_data:
 60.00 expire: 4294949920  defer:0 data:        0 callback: addrconf_verify

====timer tv3 @ffffffc0019c5f38 ====
  1.00 expire: 4294998016  defer:0 data: ffffffc001b5dca8 callback: flow_cache_new_hashrnd
  1.01 expire: 4294998016  defer:0 data: ffffffc001b70140 callback: inet_frag_secret_rebuild
  1.02 expire: 4294998016  defer:0 data: ffffffc001b78c00 callback: inet_frag_secret_rebuild

====timer tv4 @ffffffc0019c6338 ====
  8.00 expire: 4303519744  defer:0 data: ffffffc0b5fab700 callback: ipv6_regen_rndid
  8.01 expire: 4303519744  defer:0 data: ffffffc0b5484d00 callback: ipv6_regen_rndid

====timer tv5 @ffffffc0019c6738 ====
```

- Work Queue 02187943

```
CPU 0
pool 0
IDLE Workqueue worker: kworker/0:0 current_work: (None)
IDLE Workqueue worker: kworker/0:1 current_work: (None)
IDLE Workqueue worker: kworker/0:3 current_work: (None)
IDLE Workqueue worker: kworker/0:4 current_work: (None)
IDLE Workqueue worker: kworker/0:2 current_work: (None)
pool 1
IDLE Workqueue worker: kworker/0:1H current_work: (None)
IDLE Workqueue worker: kworker/0:0H current_work: (None)

CPU 1
pool 0
BUSY Workqueue worker: kworker/1:2 current_work: (None)
BUSY Workqueue worker: kworker/1:1 current_work: (None)
BUSY Workqueue worker: kworker/1:0 current_work: (None)
IDLE Workqueue worker: kworker/1:3 current_work: (None)
Pending entry: sensor_update_work
Pending entry: packet_arrival_worker
Pending entry: vmstat_update
pool 1
BUSY Workqueue worker: kworker/1:2 current_work: (None)
BUSY Workqueue worker: kworker/1:1 current_work: (None)
BUSY Workqueue worker: kworker/1:0 current_work: (None)
IDLE Workqueue worker: kworker/1:1H current_work: (None)
IDLE Workqueue worker: kworker/1:0H current_work: (None)
Pending entry: pet_watchdog_work
```

Run Queue

```
CPU0 7 process is running
 |--curr: AsyncTask #1(3107)
 |--idle: swapper/0(0)
 |--stop: migration/0(7)
CFS 1 process is pending
 |--curr: 5 process is grouping
 |    |--curr: AsyncTask #1(3107)
 |    |--next: None(0)
 |    |--last: None(0)
 |    |--skip: None(0)
 |    |--pend: HeapTaskDaemon(2761)
 |    |--pend: HeapTaskDaemon(2722)
 |    |--pend: externalstorage(2751
 |    |--pend: d.process.media(2712
 |--next: None(0)
 |--last: None(0)
 |--skip: None(0)
RT 1 process is pending
 |--pend: irq/3-usbin-src(271)
```

```
CPU1 15 process is running
 |--curr: migration/1(11)
 |--idle: swapper/1(0)
 |--stop: migration/1(11)
CFS 12 process is pending
 |--curr: None(0)
 |--next: POSIX timer 0(426)
 |--last: msm_irqbalance(386)
 |--skip: None(0)
 |--pend: POSIX timer 0(426)
 |--pend: DispSync(406)
 |--pend: ksoftirqd/1(12)
 |--pend: kworker/1:1H(30)
 |--pend: EventThread(427)
 |--pend: msm_irqbalance(386)
 |--pend: surfaceflinger(382)
 |--pend: perfd(567)
 |--pend: Binder_4(2641)
 |--pend: POSIX timer 9(2686)
 |--pend: kworker/1:0(13)
 |--pend: 2 process is grouping
 |    |--curr: None(0)
 |    |--next: None(0)
 |    |--last: None(0)
 |    |--skip: None(0)
 |    |--pend: logd.reader.per(515)
 |    |--pend: logd.writer(353)
RT 1 process is pending
 |--pend: irq/216-tsens_i(233)
```

```
CPU2 4 process is running
 |--curr: migration/2(15)
 |--idle: swapper/2(0)
 |--stop: migration/2(15)
CFS 2 process is pending
 |--curr: None(0)
 |--next: None(0)
 |--last: None(0)
 |--skip: None(0)
 |--pend: thermal-engine(608)
 |--pend: kworker/2:1(27)
RT 1 process is pending
 |--pend: msm_thermal:hot(292)
```

```
CPU3 4 process is running
 |--curr: migration/3(19)
 |--idle: swapper/3(0)
 |--stop: migration/3(19)
CFS 2 process is pending
 |--curr: None(0)
 |--next: None(0)
 |--last: None(0)
 |--skip: None(0)
 |--pend: thermal-engine(589)
 |--pend: kworker/3:1(29)
RT 1 process is pending
 |--pend: irq/215-fc38800(444)
```

```
CPU4 7 process is running
 |--curr: migration/4(40)
 |--idle: swapper/4(0)
 |--stop: migration/4(40)
CFS 6 process is pending
 |--curr: None(0)
 |--next: None(0)
 |--last: None(0)
 |--skip: None(0)
 |--pend: kworker/4:0(42)
 |--pend: HeapTaskDaemon(3029)
 |--pend: kworker/u16:6(449)
 |--pend: HubConnection(2590)
 |--pend: kworker/u17:0(35)
 |--pend: ksoftirqd/4(41)
RT 0 process is pending
```

```
CPU5 6 process is running
 |--curr: migration/5(430)
 |--idle: swapper/5(0)
 |--stop: migration/5(430)
CFS 5 process is pending
 |--curr: None(0)
 |--next: None(0)
 |--last: None(0)
 |--skip: None(0)
 |--pend: thermal-engine(605)
 |--pend: kworker/5:0(432)
 |--pend: AudioService(2665)
 |--pend: HeapTaskDaemon(947)
 |--pend: rcu_sched(10)
RT 0 process is pending
```

```
CPU6 7 process is running
 |--curr: migration/6(434)
 |--idle: swapper/6(0)
 |--stop: migration/6(434)
CFS 6 process is pending
 |--curr: None(0)
 |--next: None(0)
 |--last: None(0)
 |--skip: None(0)
 |--pend: ATFWD-daemon(551)
 |--pend: thermal-engine(603)
 |--pend: kworker/6:1(458)
 |--pend: PowerManagerSer(965)
 |--pend: BootAnimation(484)
 |--pend: Error dump: sys(2859)
RT 0 process is pending
```

```
CPU7 12 process is running
 |--curr: migration/7(439)
 |--idle: swapper/7(0)
 |--stop: migration/7(439)
CFS 11 process is pending
 |--curr: None(0)
 |--next: None(0)
 |--last: None(0)
 |--skip: None(0)
 |--pend: WifiStateMachin(2658)
 |--pend: d.process.acore(3005)
 |--pend: com.android.nf(3124)
 |--pend: FLP Service Cal(3044)
 |--pend: system_server(937)
 |--pend: rcu_preempt(8)
 |--pend: putmethod.latin(3019)
 |--pend: qmuxd(3125)
 |--pend: healthd(379)
 |--pend: SoundPoolThread(2984)
 |--pend: 1 process is grouping
 |    |--curr: None(0)
 |    |--next: None(0)
 |    |--last: None(0)
 |    |--skip: None(0)
 |    |--pend: ContactsProvide(3034)
RT 0 process is pending
```

Debug Images

```
Core 0 PC: hrtimer_interrupt+d0 <ffffffc000243668>
Core 0 LR: hrtimer_interrupt+8c <ffffffc000243624>

[<ffffffc000243668>] hrtimer_interrupt+0xd0
[<ffffffc00090d0b0>] arch_timer_handler_virt+0x2c
[<ffffffc00026dcc0>] handle_percpu_devid_irq+0xdc
[<ffffffc00026a384>] generic_handle_irq+0x2c
[<ffffffc0002040dc>] handle_IRQ+0x88
[<ffffffc0002006dc>] gic_handle_irq+0x58
[<ffffffc0002035a8>] el1_irq+0x68
[<ffffffc0006503ac>] dhd_os_general_spin_unlock+0x1c
[<ffffffc00069c514>] dhd_msgbuf_rxbuf_post_ctrlpath+0x2b4
[<ffffffc00069ca20>] dhd_prot_event_process+0x70
[<ffffffc00069da6c>] dhd_prot_process_msgtype+0x1a8
[<ffffffc00069ddd8>] dhd_prot_process_ctrlbuf+0x5c
[<ffffffc0006973a4>] dhdpcie_bus_process_mailbox_intr+0x1fc
[<ffffffc00069861c>] dhd_bus_dpc+0x128
[<ffffffc000653244>] dhd_dpc+0x20
[<ffffffc000224914>] tasklet_action+0x90
[<ffffffc000224268>] __do_softirq+0x154
[<ffffffc000224428>] do_softirq+0x44
[<ffffffc00022464c>] irq_exit+0x7c
[<ffffffc0002040e0>] handle_IRQ+0x8c
[<ffffffc0002006dc>] gic_handle_irq+0x58
[<ffffffc000203a10>] el0_irq_naked+0x14
```

```
Core 1 PC: arch_counter_get_cntvct+1c <ffffffc00090d3bc>
Core 1 LR: __delay+1c <ffffffc00042be0c>

[<ffffffc00090d3bc>] arch_counter_get_cntvct+0x1c
[<ffffffc00042be50>] __const_udelay+0x28
[<ffffffc000504ebc>] msm_trigger_wdog_bite+0x60
[<ffffffc000504fcc>] wdog_bark_handler+0xd0
[<ffffffc00026ab3c>] handle_irq_event_percpu+0xb8
[<ffffffc00026ad14>] handle_irq_event+0x4c
[<ffffffc00026d9bc>] handle_fasteoi_irq+0xc4
[<ffffffc00026a384>] generic_handle_irq+0x2c
[<ffffffc0002040dc>] handle_IRQ+0x88
[<ffffffc0002006dc>] gic_handle_irq+0x58
[<ffffffc0002035a8>] el1_irq+0x68
[<ffffffc00028b880>] cpu_stopper_thread+0xa0
[<ffffffc0002470e8>] smpboot_thread_fn+0x1e0
[<ffffffc00023fab4>] kthread+0xb4
[<ffffffc000203bc0>] ret_from_fork+0x10
```

Core2, 3, 4 ,5 ,6 ,7

```
[<ffffffc00028b97c>] stop_machine_cpu_stop+0x68
[<ffffffc00028b880>] cpu_stopper_thread+0xa0
[<ffffffc0002470e8>] smpboot_thread_fn+0x1e0
[<ffffffc00023fab4>] kthread+0xb4
[<ffffffc000203bc0>] ret_from_fork+0x10
```

Core1 Call Stack Details Before Dog Bark

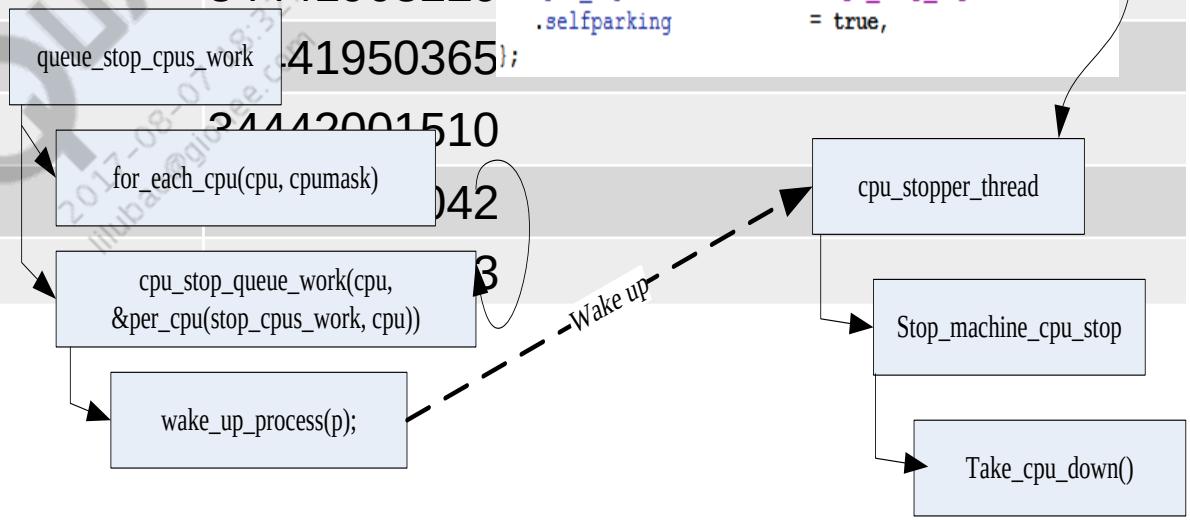
```
Up   Down   ☑ Args   ☑ Locals   ☐ Caller   Task:
-010 gic_handle_irq(
       ⊞ regs = 0xFFFFFFC006107BA0)
-011 el1_irq(asm)
→    exception
-012 stop_machine_cpu_stop(
       ⊟ data = 0xFFFFFFC0B5B8BC48 → )
     ▪ err = 0
     ▪ flags = 320
     ▪ is_active = FALSE
     ▪ tmp = 0
-013 cpu_stopper_thread(
       ?)
     ▪ ret = 0
     ⊞ fn = 0xFFFFFFC00028B914
     ⊞ arg = 0xFFFFFFC0B5B8BC48
     ⊞ done = 0xFFFFFFC0B5B8BB90
     ▪ __warned = FALSE
-014 smpboot_thread_fn(
       ⊞ data = 0xFFFFFFC0062C5F80)
     ⊞ ht = 0xFFFFFFC00171E808
-015 kthread(
       ⊞ _create = 0xFFFFFFC00E9EFC50)
     ⊞ threadfn = 0xFFFFFFC000246F08
     ⊞ data = 0xFFFFFFC0062C5F80
     ⊞ self = (flags = 1, cpu = 1, data = 0xFFFFFFC0062C5F80, parked = (done = 0, wait = (lock = (rlock = (raw_lock = (owner = 2,
     ▪ ret = 0
     ⊞ __key = ()
     ⊞ __key = ()
-016 ret_from_fork(asm)
```

```
B::v.v (struct stop_machine_data*)0xFFFFFFC0b5b8bc48
⊟ (struct stop_machine_data*)0xFFFFFFC0b5b8bc48 = 0xFFFFFFC0B5B8BC48 ≙ __bs
  ⊟ fn = 0xFFFFFFC000C824F4 ≙ take_cpu_down → ,
  ⊞ data = 0xFFFFFFC0B5B8BCD0 ≙ __bss_stop+0xB4009328,
  ▪ num_threads = 8,
  ⊞ active_cpus = 0xFFFFFFC000E009E8 ≙ cpu_bit_bitmap[8],
  ▪ state = STOPMACHINE_PREPARE,
  ⊞ thread_ack = (counter = 1))
```

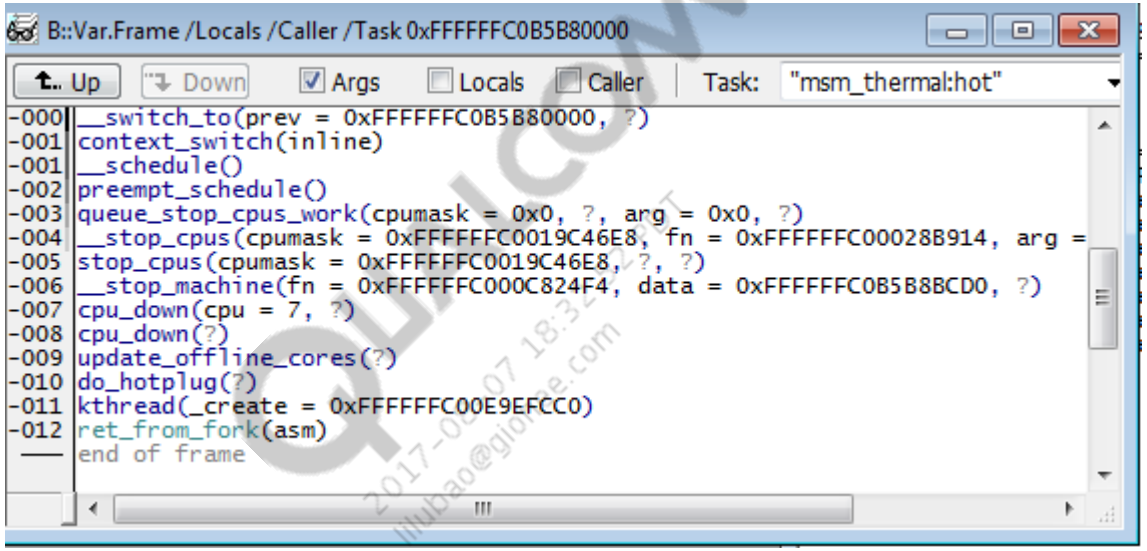
Why Migration Threads Run for Other Cores

Task name	Last_arrival
Migrate/1	34441826042
Migrate/2	34441866250
Migrate/3	34441880885
Migrate/4	34441908229
Migrate/5	41950365
Migrate/6	3442001510
Migrate/7	042
msm_thermal:hot	3

```
static struct smp_hotplug_thread cpu_stop_threads = {
    .store                  = &cpu_stopper_task,
    .thread_should_run      = cpu_stop_should_run,
    .thread_fn              = cpu_stopper_thread,
    .thread_comm            = "migration/%u",
    .create                 = cpu_stop_create,
    .setup                  = cpu_stop_unpark,
    .park                   = cpu_stop_park,
    .pre_unpark             = cpu_stop_unpark,
    .selfparking            = true,
};
```

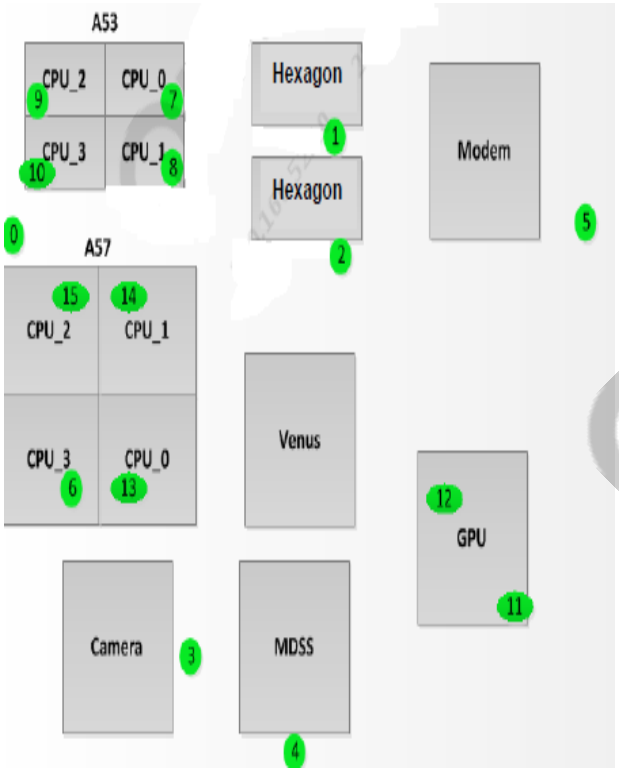


Msm_thermal:hot



KTM is one of the kernel drivers that are initialized very early to guarantee correct operation and protect the device from thermal damage.
Once the driver is initialized, it starts polling for the temperature, mitigating the CPU (or other devices below) when the temperature is above a certain threshold.

Thermal Sensors Locations



\\vmlinux\msm_thermal\cpus = (
(
cpu = 0,
sensor_type = 0xFFFFFFC002525E4C -> "**tsens_tz_sensor7",**
id_type = THERM_ZONE_ID,
sensor_id = 8,
offline = FALSE,
user_offline = FALSE,
hotplug_thresh_clear = FALSE,
threshold = ((temp = 0, trip = THERMAL_TRIP_ACTIVE, notify = 0x0, data = 0x0, active = 0, list = (next = 0x0, max_freq = FALSE,
user_max_freq = 4294967295,
shutdown_max_freq = 4294967295,
user_min_freq = 0,
limited_max_freq = 4294967295,
limited_min_freq = 0,
freq_thresh_clear = FALSE,
parent_ptr = 0xFFFFFFC0B9318D98),
(cpu = 1, sensor_type = 0xFFFFFFC002525E5D, id_type = THERM_ZONE_ID, sensor_id = 9, offline = FALSE, (cpu = 2, sensor_type = 0xFFFFFFC002525E6E, id_type = THERM_ZONE_ID, sensor_id = 10, offline = FALSE,
(cpu = 3, sensor_type = 0xFFFFFFC002525E7F, id_type = THERM_ZONE_ID, sensor_id = 11, offline = FALSE,
(cpu = 4, sensor_type = 0xFFFFFFC002525E91, id_type = THERM_ZONE_ID, sensor_id = 14, offline = FALSE,
(cpu = 5, sensor_type = 0xFFFFFFC002525EA3, id_type = THERM_ZONE_ID, sensor_id = 15, offline = FALSE,
(cpu = 6, sensor_type = 0xFFFFFFC002525EB5, id_type = THERM_ZONE_ID, sensor_id = 16, offline = FALSE, (cpu = 7, sensor_type = 0xFFFFFFC002525EC7, id_type = THERM_ZONE_ID, sensor_id = 7, offline = FALSE,

v.v tmdev To Know the Last Temperature

B::v.v tmdev

```
▪ tsens_len = 8192,
▪ calib_len = 4096,
⊞ res_tsens_mem = 0xFFFFFFC0B963BCC0,
⊞ res_calib_mem = 0xFFFFFFC0B963BCF8,
⊞ tsens_work = (data = (counter = 0), entry = (next = 0x0, prev = 0x0), func = 0x0),
▪ calib_mode = 7,
▪ tsens_type = 2,
▪ tsens_valid_status_check = TRUE,
⊞ tsens_thread_iq_dbg = (dbg_count = (11, 11, 11, 11, 11, 11, 11, 11, 11, 10), idx = 109, time_stmp = (30641709311, 30642687957, 30646659207, 30663019051, 31739159940
⊟ sensor_dbg_info = (
  ⊞ (temp = (38, 39, 39, 39, 71, 71, 0, 0, 0, 0), idx = 6, time_stmp = (5341312515, 6164861997, 6165101476, 6165330122, 13762890093, 13763574312, 0, 0, 0, 0)),
  ⊞ (temp = (37, 38, 38, 38, 50, 50, 0, 0, 0, 0), idx = 6, time_stmp = (5341936630, 6164877205, 6165114393, 6165342830, 13761544468, 13762909781, 0, 0, 0, 0)),
  ⊞ (temp = (38, 38, 38, 38, 52, 52, 53, 0, 0, 0), idx = 7, time_stmp = (5342441838, 6164890955, 6165127310, 6165355643, 13727840770, 13728524833, 13760782229, 0, 0,
  ⊞ (temp = (38, 38, 38, 38, 54, 54, 0, 0, 0, 0), idx = 6, time_stmp = (5342910432, 6164904757, 6165140070, 6165368507, 13759375718, 13760791083, 0, 0, 0, 0)),
  ⊞ (temp = (38, 38, 38, 38, 48, 48, 0, 0, 0, 0), idx = 6, time_stmp = (5343363817, 6164918768, 6165153299, 6165381528, 13758686395, 13759376187, 0, 0, 0, 0)),
  ⊞ (temp = (36, 36, 36, 36, 41, 41, 0, 0, 0, 0), idx = 6, time_stmp = (5343772567, 6164932935, 6165166007, 6165394445, 13758037124, 13758714676, 0, 0, 0, 0)),
  ⊞ (temp = (78, 89, 88, 89, 89, 89, 76, 76, 80, 81), idx = 636, time_stmp = (32839988328, 33691346404, 33791200050, 33841096821, 34191801197, 34192312968, 3238959509
  ⊞ (temp = (70, 72, 73, 70, 68, 68, 68, 67, 66, 70), idx = 521, time_stmp = (34192297812, 30622531759, 30672788791, 30838327593, 31839333430, 31839874627, 3183999363
  ⊞ (temp = (73, 71, 72, 77, 77, 78, 79, 76, 73, 73), idx = 525, time_stmp = (31839997075, 32290004577, 32739909578, 34191500885, 34192300156, 30622534572, 3067279165
  ⊞ (temp = (69, 67, 67, 71, 71, 73, 73, 72, 69, 69), idx = 525, time_stmp = (31839999680, 32290012702, 32739913380, 34191750676, 34192302343, 30622537072, 3067279446
  ⊞ (temp = (74, 72, 73, 78, 78, 79, 80, 77, 74, 74), idx = 525, time_stmp = (31840002075, 32290015671, 32739915828, 34191646145, 34192304531, 30622539676, 3067279696
  ⊞ (temp = (37, 37, 37, 37, 44, 44, 0, 0, 0, 0), idx = 6, time_stmp = (5346433140, 6165014810, 6165244549, 6165473091, 13751821343, 13752437437, 0, 0, 0, 0)),
  ⊞ (temp = (36, 36, 36, 36, 44, 44, 44, 0, 0, 0), idx = 7, time_stmp = (5346840380, 6165029132, 6165257674, 6165486007, 13722280353, 13727845770, 13751108843, 0, 0,
  ⊞ (temp = (79, 76, 76, 87, 87, 87, 74, 76, 77, 79), idx = 676, time_stmp = (32539796400, 32889826714, 32889854579, 33540790674, 34191766301, 34192306510, 3229001994
  ⊞ (temp = (89, 76, 77, 76, 80, 83, 77, 86, 87, 89), idx = 551, time_stmp = (34192308593, 32290022754, 32389695827, 32440228275, 32539753484, 32739919891, 3288977562
  ⊞ (temp = (89, 76, 77, 77, 81, 83, 79, 90, 89, 89), idx = 711, time_stmp = (34192310728, 32290028796, 32389645254, 32440176296, 32539671348, 32739923068, 3284010338
⊞ sensor = ())
```

Core0 Current Task

B::task.dtask

magic	command	state	uid	pid	spaceid	tty	flags	cpu
FFFFFFC033FAE200	ContactsProvide	running	10002.	3034.	0BBD	0	00400040	7.
FFFFFFC033FA8000	CallLogProvider	sleeping	10002.	3035.	0BBD	0	00400040	1.
FFFFFFC030AF4980	putmethod.latin	running	10058.	3019.	0BCB	0	00400140	7.
FFFFFFC030AFE200	Signal␣Catcher	sleeping	10058.	3024.	0BCB	0	40400040	6.
FFFFFFC030AF9880	JDWP	sleeping	10058.	3025.	0BCB	0	40400040	6.
FFFFFFC030AFD5C0	ReferenceQueueD	sleeping	10058.	3026.	0BCB	0	40400040	7.
FFFFFFC030AFA4C0	FinalizerDaemon	sleeping						
FFFFFFC030AFC980	FinalizerWatchd	sleeping						
FFFFFFC030AFB100	HeapTaskDaemon	running						
FFFFFFC030AFBD40	Binder_1	sleeping						
FFFFFFC033C955C0	Binder_2	sleeping						
FFFFFFC003A70000	AsyncTask␣#1	**current(0)**						
FFFFFFC02EAE0C40	re-initialized>	sleeping						
FFFFFFC02D84B100	Signal␣Catcher	sleeping						
FFFFFFC02D8A8000	JDWP	sleeping						
FFFFFFC02D8AE200	ReferenceQueueD	sleeping						
FFFFFFC02D8AD5C0	FinalizerDaemon	sleeping						
FFFFFFC02D8AC980	FinalizerWatchd	sleeping						
FFFFFFC02D8AB100	HeapTaskDaemon	sleeping						
FFFFFFC030AF3D40	Binder_1	sleeping						
FFFFFFC02E8C9880	Binder_2	sleeping						
FFFFFFC02D918000	re-initialized>	sleeping						
FFFFFFC02D91C980	Signal␣Catcher	sleeping						
FFFFFFC030AF8C40	JDWP	sleeping						
FFFFFFC02E8CC980	ReferenceQueueD	sleeping						
FFFFFFC02E8CB100	FinalizerDaemon	sleeping						
FFFFFFC02E8CBD40	FinalizerWatchd	sleeping						
FFFFFFC02EAE3100	HeapTaskDaemon	sleeping						
FFFFFFC02EAE6E40	Binder_1	sleeping						
FFFFFFC02EAE4980	Binder_2	sleeping						
FFFFFFC02D91BD40	com.android.nf	sleeping						
FFFFFFC02EAE1880	Signal␣Catcher	sleeping						
FFFFFFC02D84EE40	com.android.nf	running						
FFFFFFC030984980	kworker/1:3	sleeping						
FFFFFFC030983100	kworker/4:2	sleeping						
FFFFFFC02D848C40	kworker/4:3	sleeping						

B::Var.View %m %s (struct task_struct)*0xFFFFFFC003A70000

```
run_start = 30692721707,
⊞ sched_task_group = 0xFFFFFFC0B4E08B00,
fpu_counter = 0,
btrace_seq = 0,
policy = 0,
nr_cpus_allowed = 8,
⊞ cpus_allowed = (bits = (255)),
rcu_read_lock_nesting = 0,
rcu_read_unlock_special = 0,
⊞ rcu_node_entry = (next = 0xFFFFFFC003A70270, prev = 0xFFFFFFC003A70270),
⊞ rcu_blocked_node = 0x0,
⊟ sched_info = (
  pcount = 31,
  run_delay = 12514424,
  last_arrival = 30693070457,
  last_queued = 0),
⊞ tasks = (next = 0xFFFFFFC02EAE0EE8, prev = 0xFFFFFFC030982768),
⊞ pushable_tasks = (prio = 140, prio_list = (next = 0xFFFFFFC003A702C0, pre
⊞ mm = 0xFFFFFFC0A5634800,
⊞ active_mm = 0xFFFFFFC0A5634800,
brk_randomized = 1,
exit_state = 0,
exit_code = 0,
exit_signal = -1,
pdeath_signal = 0,
jobctl = 0,
```

Core0 is Struggling

- Core0 is struggling, because other cores are dying

B::Var.Frame /Locals /Caller /Task 0xFFFFFFC003A70000

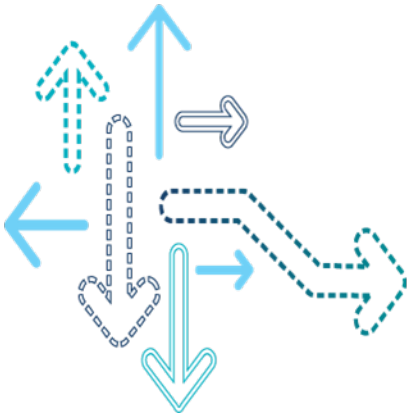
Up | Down | Args | Locals | Caller | Task: "AsyncTask #1:3107"

```
-004 handle_IRQ(irq = 20, ?)
-005 gic_handle_irq(regs = 0xFFFFFFC02DA6F9B0)
-006 el1_irq(asm)
 ⟶  exception
-007 __raw_spin_unlock_irqrestore(inline)
-007 raw_spin_unlock_irqrestore(?, flags = 320)
-008 dhd_os_general_spin_unlock(?, flags = 320)
-009 dhd_prot_rxbufpost_ctrl(inline)
-009 dhd_msgbuf_rxbuf_post_ctrlpath(dhd = 0xFFFFFFC0B5658000, event_buf
-010 dhd_msgbuf_rxbuf_post_event_bufs(inline)
-010 dhd_prot_event_process(dhd = 0xFFFFFFC0B5658000, buf = 0xFFFFFF800C
-011 dhd_process_msgtype(inline)
-011 dhd_prot_process_msgtype(dhd = 0xFFFFFFC0B5658000, ring = 0xFFFFFFC
-012 dhd_prot_process_ctrlbuf(dhd = 0x077F)
-013 dhdpci_bus_read_frames(inline)
-013 dhdpcie_bus_process_mailbox_intr(bus = 0xFFFFFFC0B5682000, intstatu
-014 dhd_bus_dpc(bus = 0xFFFFFFC0B5682000)
-015 dhd_dpc(data = 18446743801874972672)
-016 tasklet_unlock(inline)
-016 tasklet_action(?)
-017 static_key_false(inline)
-017 trace_softirq_exit(inline)
-017 __do_softirq()
-018 arch_local_irq_restore(inline)
-018 do_softirq()
-019 tick_irq_exit(inline)
-019 irq_exit()
-020 set_irq_regs(inline)
-020 handle_IRQ(irq = 304, ?)
-021 gic_handle_irq(regs = 0xFFFFFFC02DA6FED0)
-022 el0_irq_naked(asm)
 ⟶  exception
-023 NUX:0x7F821658C0(asm)
 —— end of frame
```

Summary

- Dog petting timer had expired; work was queued into core1 work queue, pending
- Core1/2/3/4/5/6/7 were all busy migrating tasks to other cores; core1 finally received watchdog bark IRQ
- Core0 seemed rather busy in handling IRQ/softIRQ; one current task had been running for over 10 seconds; no task switching seen
- Many migration threads were running simultaneously; KTM was causing them to run
- From thermal logs, performance cores all have been almost 90 degrees; some of power cores also have been over 70 degree, which is why KTM kicked in

Analysis of Watchdog Bite Issues



Dogbite example - 8939

02187943(1)

- QCAP tz diag log:

```
UIE: NO ENCRYPTION
(8)
(8)
(8)
(8)
(8)
(8)
(8)
(8)
(8)
Fatal Error: NON_SECURE_WDT
```

- Watchdog driver data:

```
B::v.v wdog_data
wdog_data = 0xFFFFFFC0AE0A9A00 → (
  phys_base = 184643584,
  size = 4096,
  base = 0xFFFFFF8000162000,
  wdog_absent_base = 0x0,
  dev = 0xFFFFFFC0AE072010,
  pet_time = 10000,
  bark_time = 15000,
  bark_irq = 35,
  bite_irq = 36,
  do_ipi_ping = TRUE,
  last_pet = 130940781,
  min_slack_ticks = 4294967295,
  min_slack_ns = 18446744073709551615,
  scm_regsave = 0x0,
  alive_mask = (bits = (0)),
  disable_lock = (count = (counter = 1), wait_lock = (rlock = (raw_lock = (own
  init_dogwork_struct = (data = (counter = 352), entry = (next = 0xFFFFFFC0AE0
  dogwork_struct = (work = (data = (counter = 352), entry = (next = 0xFFFFFFC0
  irq_ppi = FALSE,
  wdog_cpu_dd = 0x0,
  panic_blk = (notifier_call = 0xFFFFFFC00037B754, next = 0xFFFFFFC00106BB80,
  enabled = TRUE)
```

Dogbite Example

- QCAP tz counter

```
Reset Status
==========================
CPU   |Reset Reason                                    |Reset Count
0     |0x00000000    (TZBSP_ERR_FATAL_NONE             )|0x00000000
1     |0x00000000    (TZBSP_ERR_FATAL_NONE             )|0x00000000
2     |0x00000000    (TZBSP_ERR_FATAL_NONE             )|0x00000000
3     |0x00000000    (TZBSP_ERR_FATAL_NONE             )|0x00000000
4     |0x00000000    (TZBSP_ERR_FATAL_NONE             )|0x00000000
5     |0x00000000    (TZBSP_ERR_FATAL_NONE             )|0x00000000
6     |0x00000001    (TZBSP_ERR_FATAL_NON_SECURE_WDT   )|0x00000001
7     |0x00000000    (TZBSP_ERR_FATAL_NONE             )|0x00000000
```

- Call stack on core6 (TZ shows physical CPU6 is logical core3 in kernel)

```
Up   Down   Args   Locals   Caller   Task:
-000 msm_pm_swfi(from_idle = TRUE)
-001 msm_cpu_pm_enter_sleep(mode = MSM_PM_SLEEP_MODE_WAIT_FOR_INTERRUPT, from_id
-002 lpm_cpuidle_enter(dev = 0xFFFFFFC0B72E6000, ?, ?)
-003 cpuidle_enter_state(dev = 0xFFFFFFC0B72E6000, drv = 0xFFFFFFC00114A69E, ?)
-004 cpuidle_idle_call()
-005 arch_cpu_idle()
-006 arch_local_save_flags(inline)
-006 cpu_idle_loop(inline)
-006 cpu_startup_entry(?)
-007 secondary_start_kernel()
-008 NSX:0x40081168(asm)
-009 __secondary_switched(asm)
---  end of frame
```

Dogbite Example (cont.)

- Call stack on core1 (which core1 is logical core5)

```
Up   Down   Args   Locals   Caller   Task:
-000 csd_lock_wait(inline)
-000 generic_exec_single(cpu = 0, csd = 0xFFFFFFC0AE0AFCE8, wait = 1)
-001 smp_call_function_single(cpu = 0, func = 0xFFFFFFC00037B224, info = 0xFFFFFFC0AE0A9A00,
-002 ping_other_cpus(inline)
-002 pet_watchdog_work(work = 0xFFFFFFC0AE0A9AB0)
-003 static_key_false(inline)
-003 trace_workqueue_execute_end(inline)
-003 process_one_work(worker = 0xFFFFFFC0AE7B2D00, work = 0xFFFFFFC0AE0A9AB0)
-004 worker_thread(__worker = 0xFFFFFFC0AE7B2D00)
-005 kthread(_create = 0xFFFFFFC0AEB2BCB0)
-006 ret_from_fork(asm)
 —— end of frame
```

- Call stack on core4 (which core4 is the logical core0)

```
-000 __raw_remote_sfpb_spin_lock(lock = 0xFFFFFF8000593000)
-001 remote_spin_lock(?, ?)
-002 __smem_get_entry_secure(id = 28, size = 0xFFFFFFC04CBB3CAC, to_proc = 1, flags = 0, ?,
-003 smem_get_entry(id = 28, size = 0xFFFFFFC04CBB3CAC, to_proc = 1, flags = 0)
-004 smem_find(id = 28, size_in = 40, to_proc = 1, flags = 0)
-005 smd_alloc(inline)
-005 smd_alloc_channel(inline)
-005 scan_alloc_table(?, smd_ch_allocated = 0xFFFFFFC0011F76B0, table_id = 1, num_entries =
-006 smd_channel_probe_worker(work = 0xFFFFFFC0011F7680)
-007 static_key_false(inline)
-007 trace_workqueue_execute_end(inline)
-007 process_one_work(worker = 0xFFFFFFC04CB4A580, work = 0xFFFFFFC0011F7680)
-008 worker_thread(__worker = 0xFFFFFFC04CB4A580)
-009 kthread(_create = 0xFFFFFFC04CB67CA0)
-010 ret_from_fork(asm)
 —— end of frame
```

Detailed smp2p Analysis

- Smp2p ipc log

```
7.034733:   SMP2P Int modem(1)->Apps
7.069397:   SMP2P Int modem(1)->Apps
7.069427:   SMP2P Int modem(1)->Apps
7.069431:   'slave-kernel':1 GPIO bit 2 virq 495 (0->1,-) - edge 0->1 triggering
7.069474:   SMP2P Int modem(1)->Apps
7.069476:   'slave-kernel':1 GPIO bit 1 virq 494 (0->1,-) - edge 0->1 triggering
```

- Smp2p configurations in apps side

```
&soc {
   qcom,smp2p-modem {
           compatible = "qcom,smp2p";
           reg = <0x0b111008 0x4>;
           qcom,remote-pid = <1>;
           qcom,irq-bitmask = <0x4000>;
           interrupts = <0 27 1>;
   };

```

Modem smp2s irq Handling in Apps Side

```
B::v.v (struct irq_desc *)0xffffffc04cb0ec00
(struct irq_desc *)0xffffffc04cb0ec00 = 0xFFFFFFC04CB0EC00 → (
  irq_data = (
    irq = 59,
    hwirq = 59,
    node = 0,
    state_use_accessors = 16385,
    chip = 0xFFFFFFC001054FA8,
    domain = 0xFFFFFFC04CB03180,
    handler_data = 0x0,
    chip_data = 0xFFFFFFC0011B0380,
    msi_desc = 0x0,
    affinity = (
      (
        bits = (
          0xF1)))),
```

- For a mask val of **0xF1**, GIC driver chooses to define the irq to cpu 0 and programs the GIC HW accordingly. The cpu numbers referenced till now are cpu logical numbering which is what Linux uses
- TZ uses physical cpu numbering; cpu logical and physical numbering do not match on 8939

logical cpu 0 = physical cpu4
logical cpu 4 = physical cpu0

Call Stack Analysis

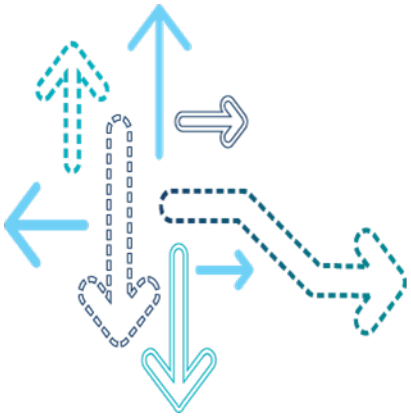
- Function smem_get_entry_secure calls remote_spin_lock_irqsave, which disables interrupts on the local cpu and waits for hw spinlock; since interrupts are disabled on logical cpu 0, the system will never receive the SMP2P interrupt
- Hence the system is in a deadlock; smp2p interrupt will be only be delivered to logical cpu 0; logical cpu0 waits for hw spinlock with interrupts disabled
- Change remote_spin_lock_irqsave to remote_spin_trylock_irqsave to resolve this issue

```
·000 __raw_remote_sfpb_spin_lock(lock = 0xFFFFFF8000593000)
·001 remote_spin_lock(?, ?)
·002 __smem_get_entry_secure(id = 28, size = 0xFFFFFFC04CBB3CAC, to_proc = 1, flags = 0, ?,
·003 smem_get_entry(id = 28, size = 0xFFFFFFC04CBB3CAC, to_proc = 1, flags = 0)
·004 smem_find(id = 28, size_in = 40, to_proc = 1, flags = 0)
·005 smd_alloc(inline)
·005 smd_alloc_channel(inline)
·005 scan_alloc_table(?, smd_ch_allocated = 0xFFFFFFC0011F76B0, table_id = 1, num_entries =
·006 smd_channel_probe_worker(work = 0xFFFFFFC0011F7680)
·007 static_key_false(inline)
·007 trace_workqueue_execute_end(inline)
·007 process_one_work(worker = 0xFFFFFFC04CB4A580, work = 0xFFFFFFC0011F7680)
·008 worker_thread(__worker = 0xFFFFFFC04CB4A580)
·009 kthread(_create = 0xFFFFFFC04CB67CA0)
·010 ret_from_fork(asm)
—— end of frame
```

Normal Steps for Dogbite Issue

- First check QCAP log, especially tz diag log, to resolve which core has met the dogbite
- Check last call stack of every core; if it is mostly a software issue, there are maybe clues, like pet watchdog failed, because some cores have no response to the ping; in this dump, logical core5 can not pet the dog because logical core0 can not respond to the ping
- Check the call stack that can not pet the dog normally, to see if it is:
  - Deadlocked
  - Waiting for some hardware event
  - Waiting for mutex
  - Waiting for spinlock
- In this example, it is waiting for remote spinlock

Analysis of Secure Watchdog Bite Issues



Secure Watchdog Bite Example

- Quick judge from GCC_RESET_STATUS
  - check GCC_RESET_STATUS from QCAP

```
Reset Registers (SDI)
    ==========================
    GCC_RESET_STATUS          : 0x23
```

- Check chip software interface document for GCC_RESET_STATUS bit map. Take 8939 as example, 0x23 mean bit 5 is 1, that is secure WDT expired.

Check TZ Counter

- ==========
  CPU |WarmEntry |WarmExit |PCEntry |PCExit |Warm JumpAddr |JumpInstr
  0 |0x002C6425 |0x002C6425 |0x003758E2 |0x000AF4BD |0x803BBF94 |0xEE100FB0
  1 |0x002CB5DC |0x002CB5DC |0x0032821E |0x0005CC42 |0x803BBF94 |0xEE100FB0
  2 |0x001D2973 |0x001D2973 |0x00237748 |0x00064DD5 |0x803BBF94 |0xEE100FB0
  3 |0x002443C5 |0x002443C5 |0x002577AA |0x000133E5 |0x803BBF94 |0xEE100FB0
  4 |0x00056177 |0x00056177 |0x00073C43 |0x0001DACC |0x803BBF94 |0xEE100FB0
  5 |0x0003C61B |0x0003C61B |0x00043489 |0x00006E6E |0x803BBF94 |0xEE100FB0
  6 |0x0002A254 |0x0002A254 |0x0002C9A6 |0x00002752 |0x803BBF94 |0xEE100FB0
  7 |0x0001E08E |0x0001E08E |0x0001F061 |0x00000FD3 |0x803BBF94 |0xEE100FB0

  CPU core 0 is ONLINE.
  CPU core 1 is POWER COLLAPSED.
  CPU core 2 is POWER COLLAPSED.
  CPU core 3 is POWER COLLAPSED.
  CPU core 4 is POWER COLLAPSED.
  CPU core 5 is POWER COLLAPSED.
  CPU core 6 is POWER COLLAPSED.
  CPU core 7 is POWER COLLAPSED.

Check AP Core Summary

- Kernel Counters

```
=========================
CPU  |PC Entry Cntr  |PC Exit Cntr   |Status
0    |0x73C44        |0x73C43        |POWER COLLAPSED
1    |0x43489        |0x43488        |POWER COLLAPSED
2    |0x2C9A6        |0x2C9A5        |POWER COLLAPSED
3    |0x1F061        |0x1F060        |POWER COLLAPSED
4    |0x3758E2       |0x3758E1       |POWER COLLAPSED
5    |0x32821E       |0x32821D       |POWER COLLAPSED
6    |0x237748       |0x237747       |POWER COLLAPSED
7    |0x2577AA       |0x2577A9       |POWER COLLAPSED

Watchdog Context
=========================
CPU  |World          |Received WDT Int?   |Received SGI Int?   |In Warm Boot?
4    |secure world   |no                  |no                  |no

CPU 4 Call Stacks (dumped by SDI):
-000|.krait_fixup()
----|end of frame
```

Check Core 4 Call Stack

- TZ call stack
  -000|tzbsp_mon_enter_pc(asm)
  ---|end of frame

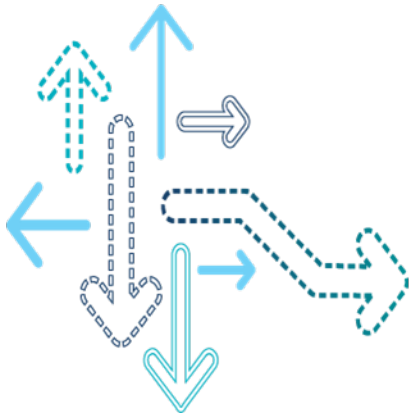
Summary

- AP side power status shows all cores in power collapse and core 4 in secure world
- TZ side power status shows core4 online; other cores in power collapse
- Only core4 online in secure world and in function tzbsp_mon_enter_pc; at this time it fails to respond to FIQ
- TZ team finds L2 GDHS request from HLOS is not cleared from TZ global variable; this may be a side effect for core power collapse

Secure WDT Summary

- Secure WDT pet by TZ side; if system happens to have secure WDT issue, it is caused by TZ failed to respond to FIQ
- Need SDI enabled; it can flash cache to save cpu context when TZ responds to FIQ failure; without CPU context, it is difficult to analyze this type of issue

Examples of TZNOC Issues



TZNOC Example

- From QCAP Report Summary

Root Cause Analysis

```
TZ - PCNOC Error: 0x32F02000   Source: 0x0C   Destination: 0x2F   MID: 0x00   BID: 0x02   PID: 0x00
```

- In TZ diag

```
(183)
SNOC ERROR: ERRLOG0 = 0x80030000
PCNOC ERROR: ERRLOG0 = 0x80030000
SNOC ERROR: ERRLOG1 = 0x05404000
PCNOC ERROR: ERRLOG1 = 0x32f02000
SNOC ERROR: ERRLOG3 = 0x078d9184
PCNOC ERROR: ERRLOG3 = 0x00019184
SNOC ERROR: ERRLOG4 = 0x00000000
PCNOC ERROR: ERRLOG4 = 0x00000000
SNOC ERROR: ERRLOG5 = 0x0000be11
PCNOC ERROR: ERRLOG5 = 0x0000be22
Fatal Error: NOC_ERROR
Fatal Error: NOC_ERROR
Error executing cmd 0 / slave: 0 / addr: 854 / len: 1 / rslt: 7
Error executing cmd 0 / slave: 0 / addr: 854 / len: 1 / rslt: 7
SPMI Read command failed.
SPMI Read command failed.
Error executing cmd 0 / slave: 0 / addr: 855 / len: 1 / rslt: 1
SPMI Read command failed.
Error executing cmd 0 / slave: 0 / addr: 856 / len: 1 / rslt: 1
SPMI Read command failed.


Noc Errors:
=====================================================================
PCNOC Error: 0x32F02000   Source: 0x0C   Destination: 0x2F   MID: 0x00   BID: 0x02   PID: 0x00
 SNOC Error: 0x05404000   Source: 0x02   Destination: 0x0A   MID: 0x00   BID: 0x02   PID: 0x00
```

TZNOC Example (cont.)

- Decode the SNOC Error Message

  - The snoc error is caused by pcnoc

```
---------------------
SNOC ERROR DECODED
---------------------
RouteID = 0x5404000

InitFlow = 0x2 qxm_bimc/I/O
TargFlow = 0xa qxs_pcnoc/T/O
targSubRange = 0x0
SrcId.BID = 0x2
SrcId.PID = 0x0
SrcId.MID = 0x0
seqId = 0x0

Address offset= 0x78d9184
TargFlow ( qxs_pcnoc/T/O ) Base Address -> Local Address: 0x0 Global Address: 0x
Complete Physical Address: 0x78d9184

OPC Decode - 0: RD Read
Errorcode 0: "Error detected by the slave with no information or no error reason
              for e.g. This can also be for an error is propagated back from ano

---------------------
```

TZNOC Example (cont.)

- Decode the PCNOC Error Message

  - The pcnoc error is caused by AP accessing usb_hs1

```
---------------------
PCNOC ERROR DECODED
---------------------
RouteID = 0x32f02000

InitFlow = 0xc qxm_snoc/I/O
TargFlow = 0x2f qhs8/T/usb_hs1
targSubRange = 0x0
SrcId.BID = 0x2 → BIMC
SrcId.PID = 0x0 → A53SS
SrcId.MID = 0x0
seqId = 0x0

Address offset= 0x19184
TargFlow ( qhs8/T/usb_hs1 ) Base Address -> Local Address: 0x0 Global Address: 0
Complete Physical Address: 0x78d9184

OPC Decode - 0: RD Read
Errorcode 0: "Error detected by the slave with no information or no error reason
              for e.g. This can also be for an error is propagated back from ano

---------------------
```

TZNOC Example (cont.)

- Restore call stack: (load dump in t32 simulator and run core0_regs.cmm)

```
-000 vectors(asm)
 ---> exception
-001 __raw_readl_no_log(inline)
-001 ehci_msm_runtime_resume(?)
-002 pm_generic_runtime_resume(dev = 0xFFFFFFC07B2D9C10)
-003 __rpm_callback(cb = 0xFFFFFFC00044B0A0, dev = 0xFFFFFFC07B2D9C10)
-004 rpm_callback(cb = 0xFFFFFFC00044B0A0, dev = 0xFFFFFFC07B2D9C10)
-005 rpm_resume(dev = 0xFFFFFFC07B2D9C10, rpmflags = 0)
-006 rpm_resume(dev = 0xFFFFFFC00512F888, rpmflags = 4)
-007 __pm_runtime_resume(dev = 0xFFFFFFC00512F888, rpmflags = 4)
-008 usb_autoresume_device(?)
-009 usb_remote_wakeup(udev = 0xFFFFFFC00512F800)
-010 device_unlock(inline)
-010 hcd_resume_work(?)
-011 static_key_false(inline)
-011 trace_workqueue_execute_end(inline)
-011 process_one_work(worker = 0xFFFFFFC0749C3280, work = 0xFFFFFFC07AB18180)
-012 worker_thread(__worker = 0xFFFFFFC0749C3280)
-013 kthread(_create = 0xFFFFFFC027E47CB0)
-014 ret_from_fork(asm)
 --- end of frame
```

- Related code:

```
static int ehci_msm_runtime_resume(struct device *dev)
{
    struct usb_hcd *hcd = dev_get_drvdata(dev);
    u32 portsc;

    dev_dbg(dev, "ehci runtime resume\n");

    portsc = readl_relaxed(USB_PORTSC);
```

Crash Here

TZNOC Example (cont.)

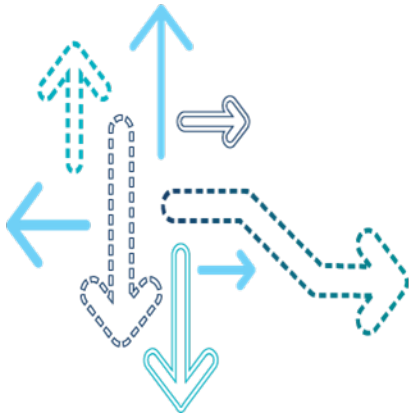
- Kernel Log:

```
[ 5056.945759]  0)msm_otg 78d9000.usb: USB exited from low power mode
[ 5056.963023]  0)PM: resume of devices complete after 32.873 msecs
[ 5056.965866]  (2)[323:psensord]apds9921_alsps_ps_min_store the buf is 43
[ 5056.974581]  (3)[262:healthd]healthd: battery l=27 v=3770 t=27.2 h=2 st=3 c=0 chg=
[ 5056.963945]  (0)[1194:system_server]Restarting tasks ... done.
[ 5056.979340]  (0)[1194:system_server]PM: suspend exit 2016-03-08 03:23:06.464841429 UTC
[ 5056.984488]  (0)[14856:kworker/u8:2 msm_otg 78d9000.usb: USB in low power mode
[ 5056.990646]  (0)[14856:kworker/u8:2 msm_otg 78d9000.usb: USB exited from low power mode
```

- Conclusion:

The AP accesses USB register but USB core is in LPM

Example of Bitflip Issues



DDR Bitflip

02158071

- Example 1 : Dmesg has messy data

- First check kernel log, to find messy data

```
[    3.388930] msm_cci_get_clk_info: clock-names[0][4] = camss_ahb_clk
[    3.388935] msm_cci_get_clk_info: clk_rate[0][4] = -1
--------- Corrupted Dmesg Bad Magic for record @ ffffffc001aeb57c ---------
ffffffc001aeb53c: 0000 0000 4400 2900 0000 0066 caef 7a5d  ....D.)....f..z]
ffffffc001aeb54c: 0000 0000 6d73 6d5f 6363 695f 6765 745f  ....msm_cci_get_ffffffc001aeb55c: 636c
   6b5f 696e 666f 3a20 636c 6b5f 7261  clk_info: clk_raffffffc001aeb56c: 7465 5b30 5d5b 345d 203d
   202d 3100 0000  te[0][4] = -1...ffffffc001aeb57c: 0425 ffc9 0000 0000 5400 3b00 0000 0066
   .%......T.;....ffffffffc001aeb58c: caef 7a5c 0000 0000 6d73 6d5f 6363 695f
   ..z\....msm_cci_ffffffc001aeb59c: 6765 745f 636c 6b5f 696e 666f 3a20 636c  get_clk_info:
   clffffffc001aeb5ac: 6f63 6b2d 6e61 6d65 735b 315d 5b30 5d20  ock-names[1][0] ffffffc001aeb5bc:
   3d20 6361 6d73 735f 746f 705f 6168 625f  = camss_top_ahb_ffffffc001aeb5cc: 636c 6b00 4634 ffc9
   0000 0000 4400 2900  clk.F4......D.).ffffffc001aeb5dc: 0000 0066 caef 7a5d 0000 0000 6d73 6d5f
   ...f..z]....msm_ffffffc001aeb5ec: 6363 695f 6765 745f 636c 6b5f 696e 666f  cci_get_clk_info
[    3.388945] msm_cci_get_clk_info: clk_rate[1][0] = -1
[    3.388951] msm_cci_get_clk_info: clock-names[1][1] = cci_src_clk
[    3.388955] msm_cci_get_clk_info: clk_rate[1][1] = 37500000
```

Quick judge from roareadiff.txt pattern

- Then check the roareadiff.txt:

```
detect RO area differences between vmlinux and DDR at 0xffffffc0009cf418
from DDR:
ffffffc0009cf418 *aa1403e0
from vmlinux:
ffffffc0009cf418 *aa1503e0
```

- It is 4 (100)->5 (101), one bit changed

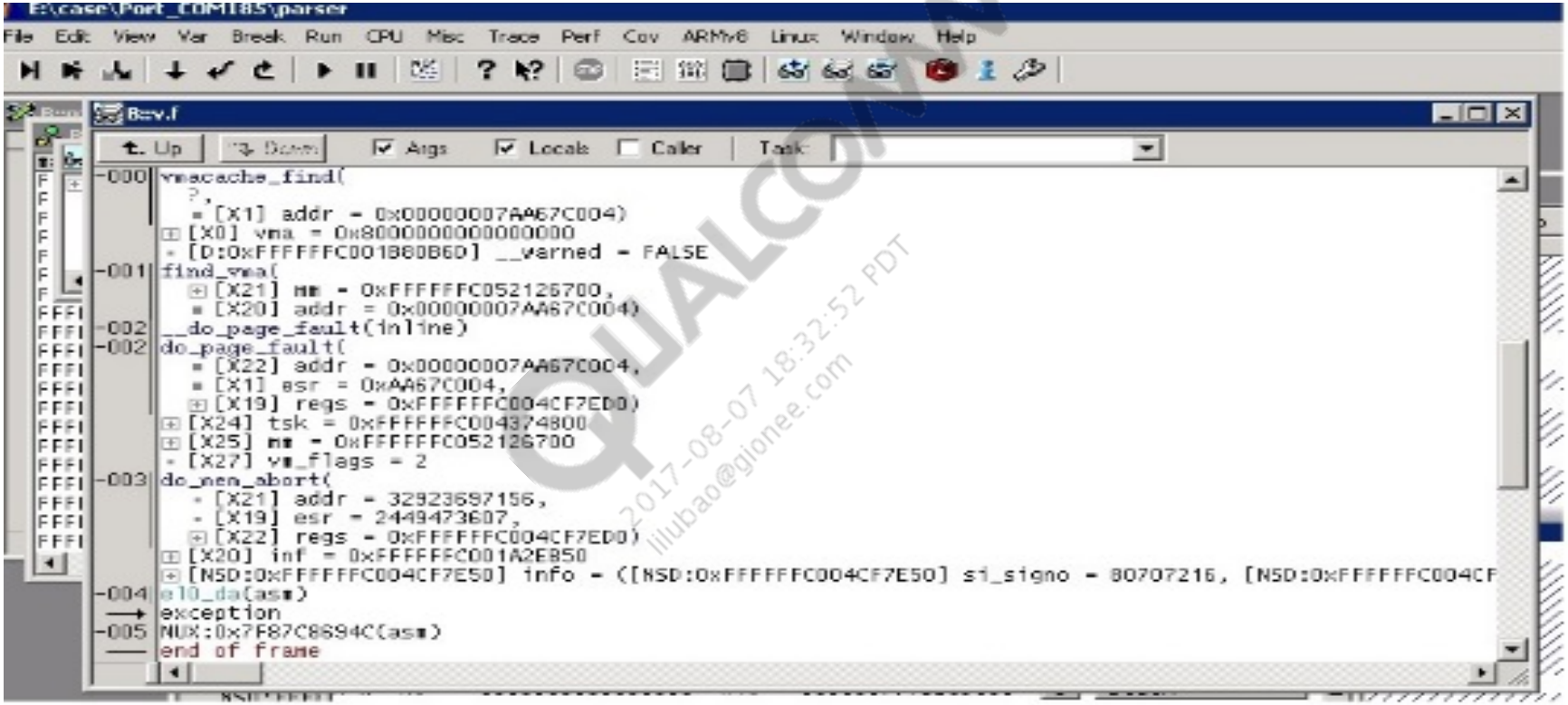
Cache Bitflip

02392677

- Example 2: Conclude bitflip from detail stack analysis
- Scenario: User space was accessing a virtual address 0x00000007AA67C004, and triggered page fault.

```
[ 2626.635186] Unable to handle kernel paging request at virtual address 8000000000000040
[ 2626.635208] pgd = ffffffc004822000
[ 2626.635216] [8000000000000040] *pgd=00000000b1191003, *pud=00000000b1191003, *pmd=0000000000000000
[ 2626.635242] Internal error: Oops: 96000006 [#1] PREEMPT SMP
[ 2626.635251] Modules linked in: wlan(O)
[ 2626.635273] CPU: 0 PID: 6400 Comm: RenderThread Tainted: G        W  O   3.18.20-perf-g463f887 #1
[ 2626.635283] task: ffffffc004374800 ti: ffffffc004cf4000 task.ti: ffffffc004cf4000
[ 2626.635300] PC is at vmacache_find+0x74/0xd8
[ 2626.635312] LR is at find_vma+0x1c/0x84
[ 2626.635319] pc : [<ffffffc0002eb55c>] lr : [<ffffffc0002f4db0>] pstate: 80000145
[ 2626.635326] sp : ffffffc004cf7c40
[ 2626.635332] x29: ffffffc004cf7c40 x28: ffffffc052126760
[ 2626.635347] x27: 0000000000000002 x26: 0000000092000047
[ 2626.635360] x25: ffffffc052126700 x24: ffffffc004374800
[ 2626.635374] x23: 0000000000000000 x22: 00000007aa67c004
[ 2626.635388] x21: ffffffc052126700 x20: ffffffc001b80000
[ 2626.635401] x19: ffffffc004cf7ed0 x18: 000000000000f01a
```

Call Stack



- vma parameter is stored in R0 register

Check the asm Code Logic

- Check how R0 get; looking at the asm code

```
B::Data.dump r(X3) /DIALOG
C:0xFFFFFFC004374C10   Find...   Modify...   Quad
address                          0                 8
NSD:FFFFFFC004374C10  0000000000000000  FFFFFFC0CC99DA50
NSD:FFFFFFC004374C20  0000000900000009  0000000000000001
NSD:FFFFFFC004374C30  0000000000000000  00000000FFFFFFFF
NSD:FFFFFFC004374C40  0000000000000000  0000000000000000
NSD:FFFFFFC004374C50  8000000000000001  000018EB00001900
NSD:FFFFFFC004374C60  0000000052134E20  FFFFFFC0CED5BC00

B::Register
N N  X0   8000000000000000  X16  0000007F7BB9D980
Z _  X1   00000007AA67C004  X17  00000007AA67C000
C _  X2   FFFFFFC004374C20  X18              F01A
V _  X3   FFFFFFC004374C10  X19  FFFFFFC004CF7ED0
     X4   FFFFFFC052126700  X20  FFFFFFC001B80000
     X5                0    X21  FFFFFFC052126700
     X6   0000007F77E80000  X22  00000007AA67C004
     X7                1    X23                 0
     X8                1    X24  FFFFFFC004374800
     X9               73    X25  FFFFFFC052126700
     X10  0000007F85826118  X26          92000047
     X11               0    X27                 2
     X12               2    X28  FFFFFFC052126760
I _  X13               5    X29  FFFFFFC004CF7C40
F F  X14  00000007AA640000  X30  FFFFFFC0002F4DB0
     X15  00000007AA67C000  PC   FFFFFFC0002EB55C

     SP   FFFFFFC004CF7C40  CPSR         80000145
EL1h ELR                0
nsec SPSR              10

[B::d.l]
Step  Over  Diverge  Return  Up  Go  Break  Mode  Find:  vmacache.c
addr/line  code  l  mnemonic  comment
93
NSX:FFFFFFC0002EB54C  EB02007F  cmp   x3,x2
NSX:FFFFFFC0002EB550  54FFFF20  b.eq  0xFFFFFFC0002EB534
94
NSX:FFFFFFC0002EB554  F9400060  ldr   x0,[x3]
96
NSX:FFFFFFC0002EB558  B4000260  cbz   x0,0xFFFFFFC0002EB5A4
98
NSX:FFFFFFC0002EB55C  F9402006  ldr   x6,[x0,#0x40]   ; x6,[x0,#64]
NSX:FFFFFFC0002EB560  EB0400DF  cmp   x6,x4
NSX:FFFFFFC0002EB564  54000140  b.eq  0xFFFFFFC0002EB58C
106
NSX:FFFFFFC0002EB568  D2800013  mov   x19,#0x0        ; x19,#0
98
NSX:FFFFFFC0002EB56C  35000225  cbnz  w5,0xFFFFFFC0002EB5B0
NSX:FFFFFFC0002EB570  D0008640  adrp  x0,0xFFFFFFC0013B5000
NSX:FFFFFFC0002EB574  52800C41  mov   w1,#0x62        ; w1,#98
NSX:FFFFFFC0002EB578  91175400  add   x0,x0,#0x5D5    ; x0,x0,#1493
```

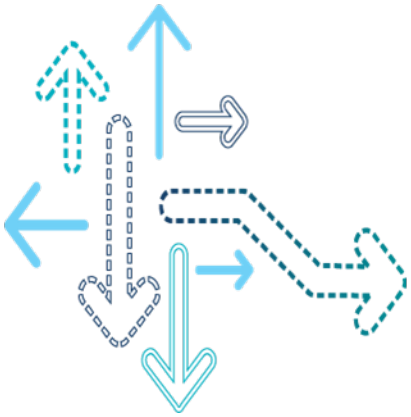
Code Reference

```
_____________addr/line|code_____|label_________________|mnemonic_________________|comment
NSX:FFFFFFC0002EB53C|B000C4A0 adrp x0,0xFFFFFFC001B80000
NSX:FFFFFFC0002EB540|91108042 add x2,x2,#0x420 ; x2,x2,#1056
NSX:FFFFFFC0002EB544|AA0003F4 mov x20,x0
NSX:FFFFFFC0002EB548|396DB405 ldrb w5,[x0,#0xB6D] ; w5,[x0,#2925]
|
| count_vm_vmacache_event(VMACACHE_FIND_CALLS);
|
93| for (i = 0; i < VMACACHE_SIZE; i++) {
NSX:FFFFFFC0002EB54C|EB02007F cmp x3,x2
NSX:FFFFFFC0002EB550|54FFFF20 b.eq 0xFFFFFFC0002EB534
94| struct vm_area_struct *vma = current->vmacache[i];
NSX:FFFFFFC0002EB554|F9400060 ldr x0,[x3]   //  [x3] is zero from memory dump.
|
96| if (!vma)
NSX:FFFFFFC0002EB558|B4000260 cbz x0,0xFFFFFFC0002EB5A4   //per normal logic, if x0 zero, we would jump.
| continue;
98|_________________if_(WARN_ON_ONCE(vma->vm_mm_!=_mm))
NSX:FFFFFFC0002EB55C|F9402006_____________________________ldr_____x6,[x0,#0x40]____;_x6,[x0,#64]
NSX:FFFFFFC0002EB560|EB0400DF cmp x6,x4
NSX:FFFFFFC0002EB564|54000140 b.eq 0xFFFFFFC0002EB58C
| count_vm_vmacache_event(VMACACHE_FIND_HITS);
| return vma;
| }
| }
```

Summary

- R0 is gotten from [R3]; while in memory it is 0x0, but register is 0x8000000000000000, so either bitflip happened in DDR reading or cache

Example of Memory Byte Shift Issues



Memory Bytes Shift Example

02187943

- Dmesg log:

```
<6>[ 5734.053929] update_heartbeat: update_heartbeat current:0
<6>[ 5734.069815] soc:100, soc_calib:100
<6>[ 5734.083630] soc:100, soc_calib:100
<6>[ 5738.341416] wlan: [1394:E :HDD] wlan_hdd_tdls_check_bmps: 1755:  pHddCtx or  pHddTdlsCtx points to NULL
<6>[ 5740.030296] qpnp_check_charger_uovp: qpnp_check_charger_uovp 5049 0
<6>[ 5740.036845] qpnp_check_battery_uovp: qpnp_check_battery_uovp bat vol:4318000
<6>[ 5740.044633] qpnp_check_battery_temp: qpnp_check_battery_temp temp:308
<6>[ 5740.051651] update_heartbeat: update_heartbeat current:0
<6>[ 5740.061677] soc:100, soc_calib:100
<6>[ 5740.070851] soc:100, soc_calib:100
<6>[ 5746.030910] qpnp_check_charger_uovp: qpnp_check_charger_uovp 5025 0
<6>[ 5746.040740] qpnp_check_battery_uovp: qpnp_check_battery_uovp bat vol:4319000
<6>[ 5746.048375] qpnp_check_battery_temp:
------------end Dmesg----------
```

Call Stack

core0_regs.cmm	5/20/2016 3:47 PM	CMM File	1 KB
core1_regs.cmm	5/20/2016 3:47 PM	CMM File	1 KB
core2_regs.cmm	5/20/2016 3:47 PM	CMM File	1 KB
core3_regs.cmm	5/20/2016 3:47 PM	CMM File	1 KB

Call Stack (cont.)

```
Up   Down   ☑ Args   ☑ Locals   ☐ Caller   Task:
    ⊞ old_regs = 0x0
    ⊞ __func__ = (0x68, 0x61, 0x6E, 0x64, 0x6C, 0x65, 0x5F, 0x49, 0x52, 0x51, 0x0)
-006 gic_handle_irq(
      ⊞ regs = 0xF29A1C20)
    ▪ irqstat = 0x13
    ⊞ cpu_base = 0xFA003000
-007 __irq_svc(asm)
 ⟶ exception
-008 __raw_spin_unlock_irq(inline)
-008 raw_spin_unlock_irq(
      ?)
    ▪ flag = 0x1
    ▪ nr = 0x1
-009 die(
      ?,
      ⊞ regs = 0xF29A1D30,
      ▪ err = 0x0)
    ⊞ thread = 0xF29A02F0
    ▪ ret = 0x1
    ▪ ret = 0x1
    ▪ die_counter = 0x2
-010 do_undefinstr(
      ⊞ regs = 0xF29A1D30)
    ⊞ info = (si_signo = 0x4, si_errno = 0x0, si_code = 0x00030001, _sifields = (_pad = (0xC093B9F8, 0xC0CA41F2, 0xC0CA41F4, 0x271AE91C, 0xF29A1D47, 0xF29A1D20,
    ⊞ pc = 0xC093B9F8
    ⊞ prog_cnt = 0xC093B9F8
    ⊞ regs = 0xF29A1D30
-011 __und_svc(asm)
 ⟶ exception
-012 spin_dump(
      ⊟ lock = 0xF1046D00 → (
        ⊞ raw_lock = (lock = 0x75399618),
        ▪ magic = 0x0,
        ▪ owner_cpu = 0xEBDAEBDA,
        ⊞ owner = 0xDEAD4EAD),
      ?)
    ⊞ owner = 0xDEAD4EAD
-013 spin_bug(
      ?,
      ?)
 —— end of frame
```

Kconfig

kernel.log | kassey.txt | dmesg_TZ.txt | analysis.txt | kconfig.txt

```
# CONFIG_DETECT_HUNG_TASK is not set
CONFIG_SCHED_DEBUG=y
# CONFIG_SYSRQ_SCHED_DEBUG is not set
CONFIG_SCHEDSTATS=y
CONFIG_TIMER_STATS=y
# CONFIG_DEBUG_OBJECTS is not set
# CONFIG_SLUB_DEBUG_ON is not set
# CONFIG_SLUB_STATS is not set
CONFIG_DEBUG_KMEMLEAK=y
CONFIG_DEBUG_KMEMLEAK_EARLY_LOG_SIZE=400
# CONFIG_DEBUG_KMEMLEAK_TEST is not set
CONFIG_DEBUG_KMEMLEAK_DEFAULT_OFF=y
CONFIG_DEBUG_PREEMPT=y
# CONFIG_DEBUG_RT_MUTEXES is not set
# CONFIG_RT_MUTEX_TESTER is not set
CONFIG_DEBUG_SPINLOCK=y
CONFIG_DEBUG_MUTEXES=y
# CONFIG_DEBUG_LOCK_ALLOC is not set
# CONFIG_PROVE_LOCKING is not set
# CONFIG_SPARSE_RCU_POINTER is not set
# CONFIG_LOCK_STAT is not set
```

Source

include/linux/rwlock_types.h

```
typedef struct {
        arch_rwlock_t raw_lock;
#ifdef CONFIG_GENERIC_LOCKBREAK
        unsigned int break_lock;
#endif
#ifdef CONFIG_DEBUG_SPINLOCK
        unsigned int magic, owner_cpu;
        void *owner;
#endif
#ifdef CONFIG_DEBUG_LOCK_ALLOC
        struct lockdep_map dep_map;
#endif
} rwlock_t;
#define RWLOCK_MAGIC            0xdeaf1eed
```

Source (cont.)

kernel/locking/spinlock_debug.c

```
void __rwlock_init(rwlock_t *lock, const char *name,
                   struct lock_class_key *key)
{
#ifdef CONFIG_DEBUG_LOCK_ALLOC
        /*
         * Make sure we are not reinitializing a held lock:
         */
        debug_check_no_locks_freed((void *)lock, sizeof(*lock));
        lockdep_init_map(&lock->dep_map, name, key, 0);
#endif
        lock->raw_lock = (arch_rwlock_t) __ARCH_RW_LOCK_UNLOCKED;
        lock->magic = RWLOCK_MAGIC;
        lock->owner = SPINLOCK_OWNER_INIT;
        lock->owner_cpu = -1;
}

#define RWLOCK_MAGIC            0xdeaf1eed
```

Summary

- Load the core3.cmm to see the core stack
- DDR is not stable, and caused memory bytes shift in the lock structure when checking with source code
- Recommend doing DDR stress first, then check DDR timing

Questions?

https://createpoint.qti.qualcomm.com