* **[PATCH] can: bcm: fix infoleak in struct bcm_msg_head**
@ 2021-06-12 21:09 Norbert Slusarek
  2021-06-13 9:51 ` Oliver Hartkopp
  2021-06-14 7:20 ` Marc Kleine-Budde
  0 siblings, 2 replies; 8+ messages in thread
From: Norbert Slusarek @ 2021-06-12 21:09 UTC (permalink / raw)
  To: socketcan; **+Cc:** mkl, davem, kuba, linux-can, netdev

```
From: Norbert Slusarek <nslusarek@gmx.net>
Date: Sat, 12 Jun 2021 22:18:54 +0200
Subject: [PATCH] can: bcm: fix infoleak in struct bcm_msg_head

On 64-bit systems, struct bcm_msg_head has an added padding of 4 bytes between
struct members count and ival1. Even though all struct members are initialized,
the 4-byte hole will contain data from the kernel stack. This patch zeroes out
struct bcm_msg_head before usage, preventing infoleaks to userspace.

Fixes: ffd980f976e7 ("[CAN]: Add broadcast manager (bcm) protocol")
Signed-off-by: Norbert Slusarek <nslusarek@gmx.net>

---
 net/can/bcm.c | 3 +++
 1 file changed, 3 insertions(+)

diff --git a/net/can/bcm.c b/net/can/bcm.c
index 909b9e684e04..b03062f84fe7 100644
--- a/net/can/bcm.c
+++ b/net/can/bcm.c
@@ -402,6 +402,7 @@ static enum hrtimer_restart bcm_tx_timeout_handler(struct hrtimer *hrtimer)
                if (!op->count && (op->flags & TX_COUNTEVT)) {

                        /* create notification to user */
+                       memset(&msg_head, 0, sizeof(msg_head));
                        msg_head.opcode  = TX_EXPIRED;
                        msg_head.flags   = op->flags;
                        msg_head.count   = op->count;
@@ -439,6 +440,7 @@ static void bcm_rx_changed(struct bcm_op *op, struct canfd_frame *data)
        /* this element is not throttled anymore */
        data->flags &= (BCM_CAN_FLAGS_MASK|RX_RECV);

+       memset(&head, 0, sizeof(head));
        head.opcode  = RX_CHANGED;
        head.flags   = op->flags;
        head.count   = op->count;
@@ -560,6 +562,7 @@ static enum hrtimer_restart bcm_rx_timeout_handler(struct hrtimer *hrtimer)
        }

        /* create notification to user */
+       memset(&msg_head, 0, sizeof(msg_head));
        msg_head.opcode  = RX_TIMEOUT;
        msg_head.flags   = op->flags;
        msg_head.count   = op->count;
--
2.30.2
```

^ permalink raw reply related [**flat**|nested] 8+ messages in thread

---

* **Re: [PATCH] can: bcm: fix infoleak in struct bcm_msg_head**
  2021-06-12 21:09 [PATCH] can: bcm: fix infoleak in struct bcm_msg_head Norbert Slusarek
@ 2021-06-13 9:51 ` Oliver Hartkopp
  2021-06-13 11:18   ` Patrick Menschel
  2021-06-14 7:20 ` Marc Kleine-Budde
  1 sibling, 1 reply; 8+ messages in thread
From: Oliver Hartkopp @ 2021-06-13 9:51 UTC (permalink / raw)
  To: Norbert Slusarek; **+Cc:** mkl, davem, kuba, linux-can, netdev

On 12.06.21 23:09, Norbert Slusarek wrote:
> From: Norbert Slusarek <nslusarek@gmx.net>
> Date: Sat, 12 Jun 2021 22:18:54 +0200
> Subject: [PATCH] can: bcm: fix infoleak in struct bcm_msg_head
>
> On 64-bit systems, struct bcm_msg_head has an added padding of 4 bytes between
> struct members count and ival1. Even though all struct members are initialized,
> the 4-byte hole will contain data from the kernel stack. This patch zeroes out
> struct bcm_msg_head before usage, preventing infoleaks to userspace.
>
> Fixes: ffd980f976e7 ("[CAN]: Add broadcast manager (bcm) protocol")
> Signed-off-by: Norbert Slusarek <nslusarek@gmx.net>

Acked-by: Oliver Hartkopp <socketcan@hartkopp.net>

Thanks Norbert!

Yes, when this data structure was created in 2003 either 64 bit machines
were far away for me and infoleaks were not a hot topic like today.

Would be interesting to check where data structures are used in the
Linux UAPI that became an infoleak in the 32-to-64-bit compilation
transistion.

Thanks for the heads up!

Best regards,
Oliver

>
> ---
>   net/can/bcm.c | 3 +++
>   1 file changed, 3 insertions(+)
>
> diff --git a/net/can/bcm.c b/net/can/bcm.c
> index 909b9e684e04..b03062f84fe7 100644
> --- a/net/can/bcm.c
> +++ b/net/can/bcm.c
> @@ -402,6 +402,7 @@ static enum hrtimer_restart bcm_tx_timeout_handler(struct hrtimer *hrtimer)
>   if (!op->count && (op->flags & TX_COUNTEVT)) {
>
>   /* create notification to user */
> +   memset(&msg_head, 0, sizeof(msg_head));
>   msg_head.opcode = TX_EXPIRED;
>   msg_head.flags   = op->flags;
>   msg_head.count   = op->count;
> @@ -439,6 +440,7 @@ static void bcm_rx_changed(struct bcm_op *op, struct canfd_frame *data)
>   /* this element is not throttled anymore */
>   data->flags &= (BCM_CAN_FLAGS_MASK|RX_RECV);
>
> +   memset(&head, 0, sizeof(head));
>   head.opcode = RX_CHANGED;
>   head.flags   = op->flags;
>   head.count   = op->count;
> @@ -560,6 +562,7 @@ static enum hrtimer_restart bcm_rx_timeout_handler(struct hrtimer *hrtimer)
>   }
>
>   /* create notification to user */
> +   memset(&msg_head, 0, sizeof(msg_head));
>   msg_head.opcode = RX_TIMEOUT;
>   msg_head.flags   = op->flags;
>   msg_head.count   = op->count;
> --
> 2.30.2
>
```

^ permalink raw reply  [**flat**|nested] 8+ messages in thread
```

* **Re: [PATCH] can: bcm: fix infoleak in struct bcm_msg_head**
  2021-06-13 9:51 ` Oliver Hartkopp
@ 2021-06-13 11:18   ` Patrick Menschel
  2021-06-13 13:35   ` Norbert Slusarek
  0 siblings, 1 reply; 8+ messages in thread
From: Patrick Menschel @ 2021-06-13 11:18 UTC (permalink / raw)
  To: Oliver Hartkopp, Norbert Slusarek; **+Cc:** mkl, davem, kuba, linux-can, netdev

Am 13.06.21 um 11:51 schrieb Oliver Hartkopp:
>
>
> On 12.06.21 23:09, Norbert Slusarek wrote:
>> From: Norbert Slusarek <nslusarek@gmx.net>
>> Date: Sat, 12 Jun 2021 22:18:54 +0200
>> Subject: [PATCH] can: bcm: fix infoleak in struct bcm_msg_head
>>
>> On 64-bit systems, struct bcm_msg_head has an added padding of 4 bytes
>> between
>> struct members count and ival1. Even though all struct members are
>> initialized,
>> the 4-byte hole will contain data from the kernel stack. This patch
>> zeroes out
>> struct bcm_msg_head before usage, preventing infoleaks to userspace.
>>
>> Fixes: ffd980f976e7 ("[CAN]: Add broadcast manager (bcm) protocol")
>> Signed-off-by: Norbert Slusarek <nslusarek@gmx.net>
>
> Acked-by: Oliver Hartkopp <socketcan@hartkopp.net>
>
> Thanks Norbert!
>
> Yes, when this data structure was created in 2003 either 64 bit machines
> were far away for me and infoleaks were not a hot topic like today.
>
> Would be interesting to check where data structures are used in the
> Linux UAPI that became an infoleak in the 32-to-64-bit compilation
> transistion.
>
Hi,

1.
Are you sure this leak really happens on 64-bit and not on 32-bit instead?

I remember I got the problems with bcm msg head on the 32bit raspberry
pi because I missed the alignment by accident.

When I calculate the size of msg head on a Ryzen 1800X with Python
3.9.5, I get:

struct.calcsize("IIIllllII"),struct.calcsize("IIIllllII0q")
(56, 56)

First Value is raw, the second value is the alignment hack with the zero
length quad word "0q".

On the 32bit raspberry pi, same op results in the gap.

struct.calcsize("IIIllllII"),struct.calcsize("IIIllllII0q")
(36, 40)

2.
Finding stucts with non-zero-ed gaps should be easy with a skript or
even better with a GCC directive. I believe Syzbot does such a thing too.

Kind Regards,
Patrick Menschel

^ permalink raw reply [**flat**|nested] 8+ messages in thread

---

* **Re: [PATCH] can: bcm: fix infoleak in struct bcm_msg_head**
  2021-06-13 11:18   ` Patrick Menschel
@ 2021-06-13 13:35   ` Norbert Slusarek
  2021-06-13 15:36   ` Patrick Menschel
  0 siblings, 1 reply; 8+ messages in thread
From: Norbert Slusarek @ 2021-06-13 13:35 UTC (permalink / raw)
  To: Patrick Menschel; **+Cc:** Oliver Hartkopp, mkl, davem, kuba, linux-can, netdev

>Hi,
>
>1.
>Are you sure this leak really happens on 64-bit and not on 32-bit instead?
>
>I remember I got the problems with bcm msg head on the 32bit raspberry
>pi because I missed the alignment by accident.
>
>When I calculate the size of msg head on a Ryzen 1800X with Python
>3.9.5, I get:
>
>struct.calcsize("IIIllllII"),struct.calcsize("IIIllllII0q")
>(56, 56)
>
>First Value is raw, the second value is the alignment hack with the zero
>length quad word "0q".
>
>On the 32bit raspberry pi, same op results in the gap.
>
>struct.calcsize("IIIllllII"),struct.calcsize("IIIllllII0q")
>(36, 40)

Hey Patrick,

having reproduced this leak I could only observe the issue on 64-bit systems.
I've just tested it on a 32-bit OS running on a raspberry pi and I couldn't observe
any leak. The offset difference on 32-bit between count and ival1 is 4.
On 64-bit systems, it's 8:

```
(gdb) ptype struct bcm_msg_head
type = struct bcm_msg_head {
    __u32 opcode;
    __u32 flags;
    __u32 count;
    struct bcm_timeval ival1;
    struct bcm_timeval ival2;
    canid_t can_id;
    __u32 nframes;
    struct can_frame frames[0];
}
(gdb) p/x &((struct bcm_msg_head *)0x0)->count
$1 = 0x8
(gdb) p/x &((struct bcm_msg_head *)0x0)->ival1
$2 = 0x10
(gdb) p sizeof(((struct bcm_msg_head *)0x0)->count)
$3 = 4
```

>2.
>Finding stucts with non-zero-ed gaps should be easy with a skript or
>even better with a GCC directive. I believe Syzbot does such a thing too.
>
>Kind Regards,
>Patrick Menschel

I didn't notice any syzbot report about this leak, nor did I find it with syzkaller.

Norbert

^ permalink raw reply [**flat**|nested] 8+ messages in thread

* **Re: [PATCH] can: bcm: fix infoleak in struct bcm_msg_head**
  2021-06-13 13:35 ` Norbert Slusarek
@ 2021-06-13 15:36 ` **Patrick Menschel**
  2021-06-13 18:33 ` Norbert Slusarek
  0 siblings, 1 reply; 8+ messages in thread
From: Patrick Menschel @ 2021-06-13 15:36 UTC (permalink / raw)
  To: Norbert Slusarek; **+Cc:** Oliver Hartkopp, mkl, davem, kuba, linux-can, netdev

Am 13.06.21 um 15:35 schrieb Norbert Slusarek:
>> Hi,
>>
>> 1.
>> Are you sure this leak really happens on 64-bit and not on 32-bit instead?
>>
>> I remember I got the problems with bcm msg head on the 32bit raspberry
>> pi because I missed the alignment by accident.
>>
>> When I calculate the size of msg head on a Ryzen 1800X with Python
>> 3.9.5, I get:
>>
>> struct.calcsize("IIIllllII"),struct.calcsize("IIIllllII0q")
>> (56, 56)
>>
>> First Value is raw, the second value is the alignment hack with the zero
>> length quad word "0q".
>>
>> On the 32bit raspberry pi, same op results in the gap.
>>
>> struct.calcsize("IIIllllII"),struct.calcsize("IIIllllII0q")
>> (36, 40)
>
> Hey Patrick,
>
> having reproduced this leak I could only observe the issue on 64-bit systems.
> I've just tested it on a 32-bit OS running on a raspberry pi and I couldn't observe
> any leak. The offset difference on 32-bit between count and ival1 is 4.
> On 64-bit systems, it's 8:
>
> (gdb) ptype struct bcm_msg_head
> type = struct bcm_msg_head {
>   __u32 opcode;
>   __u32 flags;
>   __u32 count;
>   struct bcm_timeval ival1;
>   struct bcm_timeval ival2;
>   canid_t can_id;
>   __u32 nframes;
>   struct can_frame frames[0];
> }
> (gdb) p/x &((struct bcm_msg_head *)0x0)->count
> $1 = 0x8
> (gdb) p/x &((struct bcm_msg_head *)0x0)->ival1
> $2 = 0x10
> (gdb) p sizeof(((struct bcm_msg_head *)0x0)->count)
> $3 = 4
>

Ouch,

I should not skip lines while reading.
We're talking about different gaps as it seems. I didn't realize the gap
in front of ival1 before.

There is also a gap in between nframes and frames[0].
That one is caused by align(8) of data in struct can_frame.
It propagates upwards into that gap on 32bit arch.
You can find it if you actually fill frames[] with a frame.

I found it while concatenating bcm_msg_head and a can frame into a
python bytearray which was too short for the raspberry pi as I forgot
the alignment.

I came up with a format string "IIIllllII0q" for bcm_msg_head.

Kind Regards,
Patrick

^ permalink raw reply [**flat**|nested] 8+ messages in thread

---

* **Re: [PATCH] can: bcm: fix infoleak in struct bcm_msg_head**
  2021-06-13 15:36 ` Patrick Menschel
@ 2021-06-13 18:33 ` **Norbert Slusarek**
  0 siblings, 0 replies; 8+ messages in thread
From: Norbert Slusarek @ 2021-06-13 18:33 UTC (permalink / raw)
  To: Patrick Menschel; **+Cc:** Oliver Hartkopp, mkl, davem, kuba, linux-can, netdev

>Ouch,
>
>I should not skip lines while reading.
>We're talking about different gaps as it seems. I didn't realize the gap
>in front of ival1 before.
>
>There is also a gap in between nframes and frames[0].
>That one is caused by align(8) of data in struct can_frame.
>It propagates upwards into that gap on 32bit arch.
>You can find it if you actually fill frames[] with a frame.
>
>I found it while concatenating bcm msg head and a can frame into a
>python bytearray which was too short for the raspberry pi as I forgot
>the alignment.
>
>I came up with a format string "IIIllllII0q" for bcm_msg_head.
>
>Kind Regards,
>Patrick

I confirm that there is a similar 4-byte leak happening on 32-bit systems.
It's possible to retrieve kernel addresses etc. which allows for a KASLR
bypass. I will request a CVE and publish a notice regarding
this on oss-security where I will mention Patrick too.

Anyways, this patch seems to be working for the leak on 32-bit systems as well.

Norbert

^ permalink raw reply [**flat**|nested] 8+ messages in thread

---

* **Re: [PATCH] can: bcm: fix infoleak in struct bcm_msg_head**
  2021-06-12 21:09 [PATCH] can: bcm: fix infoleak in struct bcm_msg_head Norbert Slusarek
  2021-06-13 9:51 ` Oliver Hartkopp
@ 2021-06-14 7:20 ` **Marc Kleine-Budde**
  2021-06-15 20:40 ` Norbert Slusarek
  1 sibling, 1 reply; 8+ messages in thread
From: Marc Kleine-Budde @ 2021-06-14 7:20 UTC (permalink / raw)
  To: Norbert Slusarek; **+Cc:** socketcan, davem, kuba, linux-can, netdev

[-- Attachment #1: Type: text/plain, Size: 1103 bytes --]

On 12.06.2021 23:09:26, Norbert Slusarek wrote:
> From: Norbert Slusarek <nslusarek@gmx.net>
> Date: Sat, 12 Jun 2021 22:18:54 +0200
> Subject: [PATCH] can: bcm: fix infoleak in struct bcm_msg_head
>
> On 64-bit systems, struct bcm_msg_head has an added padding of 4 bytes between
> struct members count and ival1. Even though all struct members are initialized,
> the 4-byte hole will contain data from the kernel stack. This patch zeroes out

> struct bcm_msg_head before usage, preventing infoleaks to userspace.
>
> Fixes: ffd980f976e7 ("[CAN]: Add broadcast manager (bcm) protocol")
> Signed-off-by: Norbert Slusarek <nslusarek@gmx.net>

Added to linux-can/testing.

regards,
Marc

P.S.: I think the gmx web interface mangled the patch and converted tabs
to spaces. Try to use git send-mail to avoid this.

--
Pengutronix e.K. | Marc Kleine-Budde |
Embedded Linux | https://www.pengutronix.de |
Vertretung West/Dortmund | Phone: +49-231-2826-924 |
Amtsgericht Hildesheim, HRA 2686 | Fax: +49-5121-206917-5555 |

[-- Attachment #2: signature.asc --]
[-- Type: application/pgp-signature, Size: 488 bytes --]

^ permalink raw reply [**flat**|nested] 8+ messages in thread

---

* **Re: [PATCH] can: bcm: fix infoleak in struct bcm_msg_head**
  2021-06-14 7:20 ` Marc Kleine-Budde
@ 2021-06-15 20:40   ` Norbert Slusarek
  0 siblings, 0 replies; 8+ messages in thread
From: Norbert Slusarek @ 2021-06-15 20:40 UTC (permalink / raw)
  To: Marc Kleine-Budde; **+Cc:** socketcan, davem, kuba, linux-can, netdev

The issue has been assigned CVE-2021-34693 and the announcement on
oss-security is available in the link below.
https://www.openwall.com/lists/oss-security/2021/06/15/1

Norbert

^ permalink raw reply [**flat**|nested] 8+ messages in thread

---

end of thread, other threads:[~2021-06-15 20:40 UTC | newest]

Thread overview: 8+ messages (download: mbox.gz / follow: Atom feed)
-- links below jump to the message on this page --
2021-06-12 21:09 [PATCH] can: bcm: fix infoleak in struct bcm_msg_head Norbert Slusarek
2021-06-13 9:51 ` Oliver Hartkopp
2021-06-13 11:18   ` Patrick Menschel
2021-06-13 13:35   ` Norbert Slusarek
2021-06-13 15:36   ` Patrick Menschel
2021-06-13 18:33   ` Norbert Slusarek
2021-06-14 7:20 ` Marc Kleine-Budde
2021-06-15 20:40   ` Norbert Slusarek

This is a public inbox, see mirroring instructions
for how to clone and mirror all data and code used for this inbox;
as well as URLs for NNTP newsgroup(s).