ESPILON
SECURITY RESEARCH

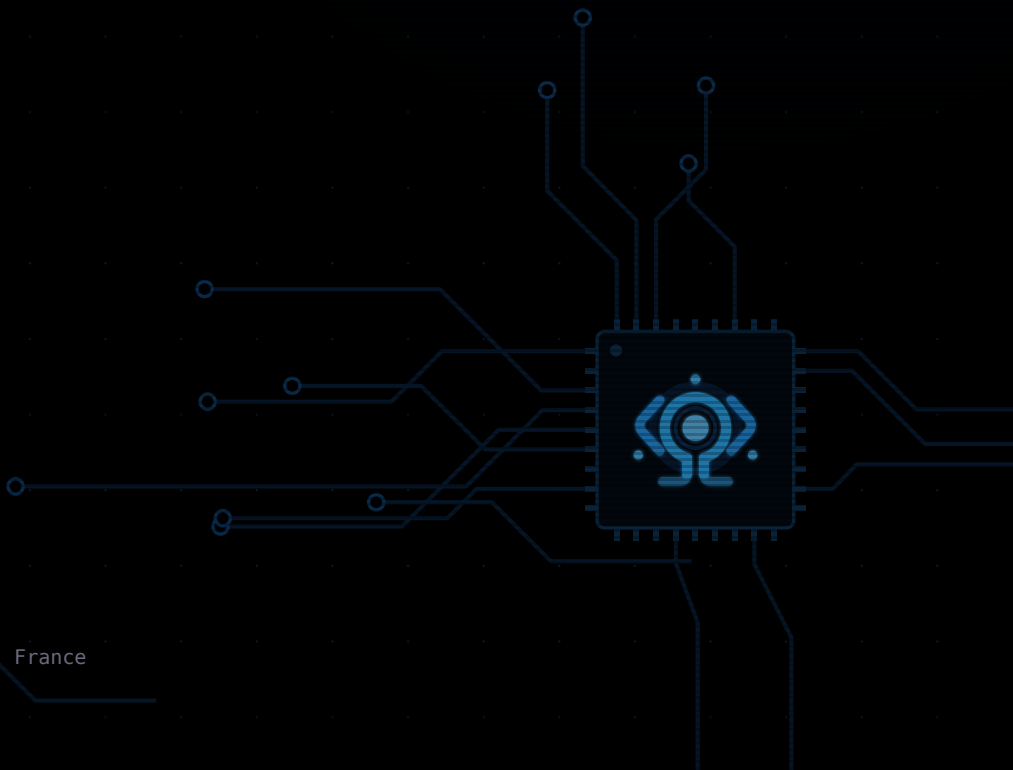
SECURITY ADVISORY

Out-of-bounds write in the ALSA USB-MIDI Novation output routine

CVE-2026-80829 NOT SCORED

Linux kernel · ALSA usb-audio

CVE PUBLISHED **04 Sep 2026**



Marouane El Moufid (Eun0us)

espilon.net · Association loi 1901 · France

01 Vulnerability

N/A

NOT SCORED

CVSS · LINUX KERNEL (CNA)

No CVSS score assigned by the CNA

The CNA record carries no metrics and no CWE. NVD lists the CVE as “Received”, with no analysis (checked 2026-10-28).

CVE	CVE-2026-80829 (CNA: Linux kernel), CVE record
Component	`snd_usbmidi_novation_output()` in `sound/usb/midi.c`
Type	Out-of-bounds write through a negative length passed to `memcpy()`
Attacker	Physical: a USB device that presents itself as a Novation USB-MIDI interface, plus a local process sending MIDI output to it
Fix	`1035a8f63bae` (mainline), backported to eight stable branches
Credit	Espilon · Marouane El Moufid (Eun0us), author of the fix

Summary

For the Novation quirk, the USB-MIDI driver reserves a two-byte header in the URB transfer buffer and asks the rawmidi core for `ep->max_transfer - 2` bytes of payload. `max_transfer` is the output endpoint’s `wMaxPacketSize`, chosen by the device. A device that advertises a bulk OUT endpoint of one byte turns the request into `-1`, which the rawmidi core carries into a `memcpy()` with a length of `(size_t)-1` and a one-byte destination. The akai output routine in the same file had the same class of defect and was fixed in July 2026 (`0970274613fb`); the Novation routine was left without a guard. We found it as an incomplete-fix sibling and sent the fix as a patch.

Affected and fixed versions

Per the CNA record, the code has been affected since the start of the git history (2.6.12, `1da177e4c3f4`). Fixed releases:

5.10	5.15	6.1	6.6	6.12	6.18	7.1	7.2	MAIN
5.10.269	5.15.220	6.1.187	6.6.156	6.12.108	6.18.49	7.1.13	7.2.3	7.3

Timeline

DATE	EVENT
2026-07-26	Akai fix `0970274613fb` applied upstream (same class, other reporters)
2026-08-18	Unguarded Novation sibling identified; confirmed with a host ASan harness
2026-08-23	Patch sent to the ALSA maintainers and linux-sound@vger.kernel.org
2026-08-24	Patch applied by Takashi Iwai (`1035a8f63bae`)
2026-08-26	CVE-2026-80829 reserved by the Linux kernel CNA
2026-09-04	CVE record published

02 Technical analysis

Vulnerable code

sound/usb/midi.c at 17056a36d276 (parent of 1035a8f63bae)

```
870  static void snd_usbmidi_novation_output(struct snd_usb_midi_out_endpoint *ep,
871                                          struct urb *urb)
872  {
873          uint8_t *transfer_buffer;
874          int count;
875
876          if (!ep->ports[0].active)
877                  return;
878          transfer_buffer = urb->transfer_buffer;
879          count = snd_rawmidi_transmit(ep->ports[0].substream,
880                                       &transfer_buffer[2],
881                                       ep->max_transfer - 2);
882          if (count < 1) {
883                  ep->ports[0].active = 0;
884                  return;
885          }
886          transfer_buffer[0] = 0;
887          transfer_buffer[1] = count;
888          urb->transfer_buffer_length = 2 + count;
889  }
```

`ep->max_transfer` is an `int` taken from the endpoint descriptor for every device outside a short list of hard-coded exceptions, and the URB buffers are allocated with exactly that size:

sound/usb/midi.c at 17056a36d276, snd_usbmidi_out_endpoint_create() (abridged)

```
1440          switch (umidi->usb_id) {
1441          default:
1442                  ep->max_transfer = usb_maxpacket(umidi->dev, pipe);
1443                  break;
...
1465          for (i = 0; i < OUTPUT_URBS; ++i) {
1466                  buffer = usb_alloc_coherent(umidi->dev,
1467                                              ep->max_transfer, GFP_KERNEL,
1468                                              &ep->urbs[i].urb->transfer_dma);
```

Root cause

With `max_transfer` equal to 1, line 881 passes `count = -1`, and the destination `&transfer_buffer[2]` on line 880 already lies beyond the end of the one-byte buffer. `snd_rawmidi_transmit()` hands the value to `__snd_rawmidi_transmit_peek()` unchanged. There, `-1` misses the special case `count == 1` (line 1370), the clamp on line 1375 keeps the smaller (negative) value, the next clamp can only lower it, and `get_aligned_size()` leaves it as it is for a byte-stream substream, whose alignment is 0:

sound/core/rawmidi.c at 17056a36d276, __snd_rawmidi_transmit_peek() and get_aligned_size() (abridged)

```
197   #define get_aligned_size(runtime, size) ((size) & ~get_align(runtime))
...
1373          } else {
1374                  count1 = runtime->buffer_size - runtime->hw_ptr;
1375                  if (count1 > count)
1376                          count1 = count;
1377                  if (count1 > (int)(runtime->buffer_size - runtime->avail))
1378                          count1 = runtime->buffer_size - runtime->avail;
1379                  count1 = get_aligned_size(runtime, count1);
1380                  if (!count1)
1381                          goto __skip;
1382                  memcpy(buffer, runtime->buffer + runtime->hw_ptr, count1);
```

`count1` is `-1` at line 1382, which `memcpy()` receives as `(size_t)-1`. The only precondition inside the rawmidi core is that some output is pending (`avail < buffer_size`), which is the case as soon as a process writes MIDI data to the port.

Reachability

`snd_usbmidi_novation_ops` is installed for the `QUIRK_MIDI_NOVATION` entries of `sound/usb/quirks-table.h`: vendor 0x1235, products 0x0001, 0x0002 and 0x4661. An attacker needs a USB device, or a programmable USB gadget, that enumerates with one of these IDs and a bulk OUT endpoint whose `wMaxPacketSize` is 1; the patch notes that the USB core only clamps this value downwards, so 1 survives enumeration. The write happens when a local process sends MIDI output to the resulting `/dev/snd/midiC*D*` port. No crafted input beyond the descriptors is needed.

Impact

The copy targets a one-byte coherent DMA buffer with a length of `(size_t)-1`. The first bytes copied are the pending MIDI data written by the local process; past that, the copy runs through adjacent kernel memory on both the source and the destination side. The consequence the code implies is kernel memory corruption from a malicious USB device. We did not test the trigger on a running kernel with a real or emulated device, and we make no code-execution claim. The CNA did not publish a severity.

Proof

A host harness compiles rawmidi.c lines 1354 to 1397 (`peek.inc`) and midi.c lines 870 to 889 (`novation.inc`) verbatim; on 2026-10-28 both extracts were byte-identical to the parent of the fix. The rawmidi wrapper (without its spinlock), `get_align()` and the structures are re-declared. Setup: `max_transfer = 1`, a 4096-byte ring with pending data, alignment 0. `peek.inc:29` is rawmidi.c line 1382 and `novation.inc:10` is midi.c line 879. ASan stops the call on its negative size before any byte is copied; the region it describes is the source ring. Output of the harness binary from the original finding, re-run on 2026-10-28 (trimmed: interceptor and libc frames, the allocation stack and the summary lines removed):

• AddressSanitizer: negative-size-param in __snd_rawmidi_transmit_peek (max_transfer = 1), trimmed

```
[poc] max_transfer=1 -> count arg=-1
=================================================================
==1359589==ERROR: AddressSanitizer: negative-size-param: (size=-1)
[...]
    #1 0x55825afa5e1d in __snd_rawmidi_transmit_peek peek.inc:29:3
    #2 0x55825afa595c in snd_rawmidi_transmit harness_novation.c:85:10
    #3 0x55825afa574b in snd_usbmidi_novation_output novation.inc:10:10
    #4 0x55825afa55f8 in setup_runtime harness_novation.c:145:2
    #5 0x55825afa50be in main harness_novation.c:162:2
[...]
0x621000000100 is located 0 bytes inside of 4096-byte region [0x621000000100,0x621000001100)
```

The negative control, the same harness built with `max_transfer = 64`, prints `[negctl] max_transfer=64 -> count arg=62` and `returned without fault`, with no sanitizer report.

03 Fix and recommendation

Upstream fix

Commit `1035a8f63bae` (author Marouane El Moufid, applied by Takashi Iwai on 2026-08-24) returns early when the endpoint cannot hold the two-byte header plus one payload byte, mirroring the guard added to the akai routine. Full diff, verbatim:

• torvalds/linux commit 1035a8f63bae28e498b0e7b5ac91d749844a7158, sound/usb/midi.c

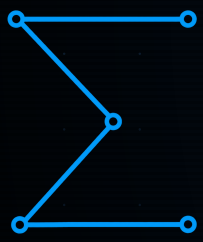
```
@@ -875,6 +875,8 @@ static void snd_usbmidi_novation_output(struct snd_usb_midi_out_endpoint *ep,

        if (!ep->ports[0].active)
                return;
+       if (ep->max_transfer < 3)
+               return;
        transfer_buffer = urb->transfer_buffer;
        count = snd_rawmidi_transmit(ep->ports[0].substream,
                                     &transfer_buffer[2],
```

Recommendation

Update to the fixed release of your branch (table in section 01) or later. Until then, restricting which USB devices the machine accepts removes the attacker's way in, since the bug needs a device the attacker controls.

04 Credit and references

- Credit: Marouane El Moufid (Eun0us), Espilon, author of the fix commit. The kernel CNA record has no credit field.
- CVE record: www.cve.org/CVERecord?id=CVE-2026-80829
- Fix (mainline): git.kernel.org/stable/c/1035a8f63bae28e498b0e7b5ac91d749844a7158
- Patch submission: patch.msgid.link/178749334830.543645.13722252148340572274@espilon.net
- Stable backports: the eight commits listed in the CVE record · prior akai fix: git.kernel.org, commit 0970274613fb



ESPILON

SECURITY RESEARCH

Embedded, IoT and OT security research

espilon.net

eun0us@espilon.net

Association loi 1901 · France