Babeltrace - Bug #1276

Empty ctf_sequence(uint64_t) on aarch64 makes babeltrace fail

07/21/2020 07:49 AM - Stefan Schuermans

Status:	Waiting for review	**Start date:**	07/21/2020
Priority:	Normal	**Due date:**	
Assignee:	Philippe Proulx	**% Done:**	0%
Category:		**Estimated time:**	0.00 hour
Target version:			

Description

Empty ctf_sequence(uint64_t) on aarch64 makes babeltrace fail

I am using LTTng 2.12 (liblttng-ust0 2.12.0, lttng-modules 2.12.1) on aarch64 Linux version 4.9.140-tegra to trace sequences of uint64_t from userspace (unsigned int for the length of the sequence). When a sequence is empty, the resulting CTF data cannot be parsed by babeltrace in most cases.

```
[error] Unexpected end of packet. Either the trace data stream is corrupted or metadata descriptio
n does not match data layout.
[error] Reading event failed.
Error printing trace.
```

System and OS Details

System and OS details:

```
$ lsb_release -a
No LSB modules are available.
Distributor ID: Ubuntu
Description:    Ubuntu 18.04.4 LTS
Release:        18.04
Codename:       bionic
$ uname -a
Linux jetson-tx2-tb31 4.9.140-tegra #1 SMP PREEMPT Mon Dec 9 22:52:02 PST 2019 aarch64 aarch64 aar
ch64 GNU/Linux
$ lscpu
Architecture:          aarch64
Byte Order:            Little Endian
CPU(s):                6
On-line CPU(s) list:   0,3-5
Off-line CPU(s) list:  1,2
Thread(s) per core:    1
Core(s) per socket:    4
Socket(s):             1
Vendor ID:             ARM
Model:                 3
Model name:            Cortex-A57
Stepping:              r1p3
CPU max MHz:           2035,2000
CPU min MHz:           345,6000
BogoMIPS:              62.50
L1d cache:             32K
L1i cache:             48K
L2 cache:              2048K
Flags:                 fp asimd evtstrm aes pmull sha1 sha2 crc32
$ lttng --version
lttng (LTTng Trace Control) 2.12.1 - (Ta) Meilleure
$ dpkg --list | grep lttng
ii  liblttng-ctl-dev:arm64                              2.12.1-1
       arm64        LTTng control and utility library (development files)
ii  liblttng-ctl0:arm64                                 2.12.1-1
```

```
       arm64        LTTng control and utility library
ii  liblttng-ust-ctl4:arm64                              2.12.0-2
       arm64        LTTng 2.0 Userspace Tracer (trace control library)
ii  liblttng-ust-dev:arm64                               2.12.0-2
       arm64        LTTng 2.0 Userspace Tracer (development files)
ii  liblttng-ust-python-agent0:arm64                     2.12.0-2
       arm64        LTTng 2.0 Userspace Tracer (Python agent native library)
ii  liblttng-ust0:arm64                                  2.12.0-2
       arm64        LTTng 2.0 Userspace Tracer (tracing libraries)
ii  lttng-modules-dkms                                   2.12.1-1slx4806
       all          Linux Trace Toolkit (LTTng) kernel modules (DKMS)
ii  lttng-tools                                          2.12.1-1
       arm64        LTTng control and utility programs
$ babeltrace --help | head -n 1
BabelTrace Trace Viewer and Converter 1.5.5
$ dpkg --list | grep babel
ii  babeltrace                                           1.5.5-1
       arm64        Trace conversion program
ii  libbabeltrace-dev:arm64                              1.5.5-1
       arm64        Babeltrace development files
ii  libbabeltrace1:arm64                                 1.5.5-1
       arm64        Babeltrace conversion libraries
(I also tried more recent babeltrace versions, see below.)
```

How to Reproduce

This is my tracepoint definition:

```
#include <stdint.h>

TRACEPOINT_EVENT(
  myseq,
  mytp,
  TP_ARGS(
    uint64_t *, values_arg,
    unsigned int, len_arg,
    uint32_t, marker_arg
  ),
  TP_FIELDS(
    ctf_sequence(uint64_t, values, values_arg, unsigned int, len_arg)
    ctf_integer(uint32_t, marker, marker_arg)
  )
)
```

The "marker" is used to find the traced data in a hexdump of the CTF files. The error also occurs when the marker field is left out.

This is a small application using the tracepoint to trace the integer values of its command line arguments usint the tracepoint:

```
#include <myseq.h>
#include <stdlib.h>
#include <stdint.h>

int main(int argc, char **argv) {
  int i;
  uint64_t values[argc - 1];

  for (i = 1; i < argc; ++i) {
    values[i - 1] = atoi(argv[i]);
  }
  tracepoint(myseq, mytp, values, argc - 1, 0x11223344);

  return 0;
}
```

I use this script on aarch64 to reproduce the error:

```
#! /bin/bash

set -eux -o pipefail

cd "$(dirname "$0")"

mkdir -p build
cd build

lttng-gen-tp ../myseq.tp -o myseq.c -o myseq.h
gcc -Wall -Wextra -Werror -I . -o myapp ../main.c myseq.c -llttng-ust -ldl

# avoid empty sequence -> works
lttng create mysession --output mysession
lttng enable-event -s mysession -u 'myseq:*'
lttng start mysession
./myapp 1 2 3
./myapp 1 2 3 4 5
lttng stop mysession
lttng destroy mysession
babeltrace mysession

# empty sequence -> breaks
lttng create mysession2 --output mysession2
lttng enable-event -s mysession2 -u 'myseq:*'
lttng start mysession2
./myapp 1 2 3
./myapp # empty sequence
./myapp 1 2 3 4 5
lttng stop mysession2
lttng destroy mysession2
babeltrace mysession2
```

The babeltrace command to dump the CTF data of mysession2 produces an error in most executions:

```
[15:21:00.555952242] (+?.??????????) jetson-tx2-tb31 myseq:mytp: { cpu_id = 5 }, { _values_length =
 3, values = [ [0] = 1, [1] = 2, [2] = 3 ], marker = 287454020 }
[15:21:00.569996583] (+0.014044341) jetson-tx2-tb31 myseq:mytp: { cpu_id = 0 }, { _values_length =
 0, values = [ ], marker = 0 }
[error] Unexpected end of packet. Either the trace data stream is corrupted or metadata descriptio
n does not match data layout.
[error] Reading event failed.
Error printing trace.
```

The error occurs in more than half of the experiments when running the LTTng tracing on aarch64. It does not matter if babeltrace is executed on aarch64 or on x86_64@with the trace data recorded on @aarch64.

The error occurs with the following versions of babeltrace:

- babeltrace 1.5.5
- babeltrace 1.5.8
- babeltrace git 64b7b73406 (origin/stable-1.5, stable-1.5)
- babeltrace git 6488b1761f (master, origin/master)

For the babeltrace2 version, the output is as follows:

```
07-20 17:54:11.039 27615 27615 E PLUGIN/CTF/MSG-ITER set_current_event_class@msg-iter.c:1266 [auto
-disc-source-ctf-fs] No event class with ID of event class ID to use in stream class: msg-it-addr=
0x55844033a7a0, stream-class-id=0, event-class-id=13124, trace-class-addr=0x55844033fc60
07-20 17:54:11.039 27615 27615 E PLUGIN/CTF/MSG-ITER ctf_msg_iter_get_next_message@msg-iter.c:2883
 [auto-disc-source-ctf-fs] Cannot handle state: msg-it-addr=0x55844033a7a0, state=AFTER_EVENT_HEAD
ER
07-20 17:54:11.039 27615 27615 E PLUGIN/SRC.CTF.FS ctf_fs_iterator_next_one@fs.c:91 [auto-disc-sou
rce-ctf-fs] Failed to get next message from CTF message iterator.
07-20 17:54:11.039 27615 27615 W LIB/MSG-ITER bt_message_iterator_next@iterator.c:868 Component in
put port message iterator's "next" method failed: iter-addr=0x55844033a630, iter-upstream-comp-nam
e="auto-disc-source-ctf-fs", iter-upstream-comp-log-level=WARNING, iter-upstream-comp-class-type=S
```

```
OURCE, iter-upstream-comp-class-name="fs", iter-upstream-comp-class-partial-descr="Read CTF traces
 from the file sy", iter-upstream-port-type=OUTPUT, iter-upstream-port-name="2a7e7597-25fb-4596-9f
9e-cfaa07ad4dcf | 0 | 0", status=ERROR
07-20 17:54:11.039 27615 27615 E PLUGIN/FLT.UTILS.MUXER muxer_upstream_msg_iter_next@muxer.c:448 [
muxer] Upstream iterator's next method returned an error: status=ERROR
07-20 17:54:11.039 27615 27615 E PLUGIN/FLT.UTILS.MUXER validate_muxer_upstream_msg_iters@muxer.c:
994 [muxer] Cannot validate muxer's upstream message iterator wrapper: muxer-msg-iter-addr=0x55844
0339f60, muxer-upstream-msg-iter-wrap-addr=0x55844033ab70
[15:21:00.555952242] (+?.?????????) jetson-tx2-tb31 myseq:mytp: { cpu_id = 5 }, { _values_length =
 3, values = [ [0] = 1, [1] = 2, [2] = 3 ], marker = 287454020 }
[15:21:00.569996583] (+0.014044341) jetson-tx2-tb31 myseq:mytp: { cpu_id = 0 }, { _values_length =
 0, values = [ ], marker = 0 }
07-20 17:54:11.039 27615 27615 E PLUGIN/FLT.UTILS.MUXER muxer_msg_iter_next@muxer.c:1422 [muxer] C
annot get next message: comp-addr=0x558440338860, muxer-comp-addr=0x5584403429d0, muxer-msg-iter-a
ddr=0x558440339f60, msg-iter-addr=0x558440339620, status=ERROR
07-20 17:54:11.039 27615 27615 W LIB/MSG-ITER bt_message_iterator_next@iterator.c:868 Component in
put port message iterator's "next" method failed: iter-addr=0x558440339620, iter-upstream-comp-nam
e="muxer", iter-upstream-comp-log-level=WARNING, iter-upstream-comp-class-type=FILTER, iter-upstre
am-comp-class-name="muxer", iter-upstream-comp-class-partial-descr="Sort messages from multiple in
pu", iter-upstream-port-type=OUTPUT, iter-upstream-port-name="out", status=ERROR
07-20 17:54:11.039 27615 27615 W LIB/GRAPH consume_graph_sink@graph.c:466 Component's "consume" me
thod failed: status=ERROR, comp-addr=0x5584403389a0, comp-name="pretty", comp-log-level=WARNING, c
omp-class-type=SINK, comp-class-name="pretty", comp-class-partial-descr="Pretty-print messages (@t
ext@ fo", comp-class-is-frozen=0, comp-class-so-handle-addr=0x55844031c450, comp-class-so-handle-p
ath="/usr/local/stow/babeltrace2/lib/babeltrace2/plugins/babeltrace-plugin-text.la", comp-input-po
rt-count=1, comp-output-port-count=0
07-20 17:54:11.039 27615 27615 E CLI cmd_run@babeltrace2.c:2530 Graph failed to complete successfu
lly

ERROR:    [Babeltrace CLI] (babeltrace2.c:2530)
  Graph failed to complete successfully
CAUSED BY [libbabeltrace2] (graph.c:466)
  Component's "consume" method failed: status=ERROR, comp-addr=0x5584403389a0,
  comp-name="pretty", comp-log-level=WARNING, comp-class-type=SINK,
  comp-class-name="pretty", comp-class-partial-descr="Pretty-print messages
  (@text@ fo", comp-class-is-frozen=0, comp-class-so-handle-addr=0x55844031c450,
  comp-class-so-handle-path="/usr/local/stow/babeltrace2/lib/babeltrace2/plugins/babeltrace-plugin
-text.la",
  comp-input-port-count=1, comp-output-port-count=0
CAUSED BY [libbabeltrace2] (iterator.c:868)
  Component input port message iterator's "next" method failed:
  iter-addr=0x558440339620, iter-upstream-comp-name="muxer",
  iter-upstream-comp-log-level=WARNING, iter-upstream-comp-class-type=FILTER,
  iter-upstream-comp-class-name="muxer",
  iter-upstream-comp-class-partial-descr="Sort messages from multiple inpu",
  iter-upstream-port-type=OUTPUT, iter-upstream-port-name="out", status=ERROR
CAUSED BY [muxer: 'filter.utils.muxer'] (muxer.c:1422)
  Cannot get next message: comp-addr=0x558440338860,
  muxer-comp-addr=0x5584403429d0, muxer-msg-iter-addr=0x558440339f60,
  msg-iter-addr=0x558440339620, status=ERROR
CAUSED BY [muxer: 'filter.utils.muxer'] (muxer.c:994)
  Cannot validate muxer's upstream message iterator wrapper:
  muxer-msg-iter-addr=0x558440339f60,
  muxer-upstream-msg-iter-wrap-addr=0x55844033ab70
CAUSED BY [muxer: 'filter.utils.muxer'] (muxer.c:448)
  Upstream iterator's next method returned an error: status=ERROR
CAUSED BY [libbabeltrace2] (iterator.c:868)
  Component input port message iterator's "next" method failed:
  iter-addr=0x55844033a630, iter-upstream-comp-name="auto-disc-source-ctf-fs",
  iter-upstream-comp-log-level=WARNING, iter-upstream-comp-class-type=SOURCE,
  iter-upstream-comp-class-name="fs",
  iter-upstream-comp-class-partial-descr="Read CTF traces from the file sy",
  iter-upstream-port-type=OUTPUT,
  iter-upstream-port-name="2a7e7597-25fb-4596-9f9e-cfaa07ad4dcf | 0 | 0",
  status=ERROR
CAUSED BY [auto-disc-source-ctf-fs (2a7e7597-25fb-4596-9f9e-cfaa07ad4dcf | 0 | 0): 'source.ctf.fs'
] (fs.c:91)
```

```
  Failed to get next message from CTF message iterator.
CAUSED BY [auto-disc-source-ctf-fs: 'source.ctf.fs'] (msg-iter.c:2883)
  Cannot handle state: msg-it-addr=0x55844033a7a0, state=AFTER_EVENT_HEADER
CAUSED BY [auto-disc-source-ctf-fs: 'source.ctf.fs'] (msg-iter.c:1266)
  No event class with ID of event class ID to use in stream class:
  msg-it-addr=0x55844033a7a0, stream-class-id=0, event-class-id=13124,
  trace-class-addr=0x55844033fc60
```

Some Ideas

I think the issue happens only if the length type of the sequence has a smaller alignment than the array element type of the sequence. liblttng-ust apparently aligns the offset of the begin of the (empty) sequence array to the expected array element alignment, even when writing no elements.

My current hypothesis is that babeltrace does not expect the array element alignment to happen when in case there are zero elements. However, I do not have any idea if this alignment should be there and babeltrace should expect it or if there should be no alignment in case of zero elements and liblttng-ust should not perform it.

History

#1 - 07/21/2020 10:22 AM - Mathieu Desnoyers

Indeed, something appears to be fishy. CTF 1.8.3 states that sequences behave like arrays:

"The sequence elements follow the array specifications."

And the array specification states:

"Arrays are always aligned on their element alignment requirement."

So I think your analysis is correct: the LTTng tracer is correct in aligning the 0-sized sequence on the alignment of the element type. Babeltrace 1 and 2 should expect this. We may have a similar issue with 0-sized arrays as well.

#2 - 07/21/2020 01:41 PM - Philippe Proulx

- Target version deleted (Babeltrace - stable 1.x)

- Assignee set to Philippe Proulx

- File ctf-1.8-array-align.tar added

Attaching a minimalist trace which makes both Babeltrace 1.5.8 and Babeltrace 2 @ 6488b1761 (*Fix: source.ctf.lttng-live: muxing failure on clear (unit conversion)*) fail.

The output of Babeltrace 2 is:

```
ev: { a = 1, b = [ ], c = 2 }
```

with many errors (as reported above); we expect:

```
ev: { a = 1, b = [ ], c = 3 }
```

Working on it.

Also resetting the target version as this bug targets all Babeltrace versions.

#3 - 07/21/2020 02:29 PM - Philippe Proulx

- Status changed from New to Waiting for review

See https://review.lttng.org/c/babeltrace/+/3805.

Files

ctf_sequence_empty.tar.xz	2.41 KB	07/21/2020	Stefan Schuermans
ctf-1.8-array-align.tar	10 KB	07/21/2020	Philippe Proulx