LTTng-UST - Bug #1242

SEGV on process exit with shared library

03/04/2020 07:35 PM - Stephen Hemminger

Status:	New	**Start date:**	03/04/2020
Priority:	Normal	**Due date:**	
Assignee:	Mathieu Desnoyers	**% Done:**	0%
Category:		**Estimated time:**	0.00 hour
Target version:			

Description

Our application has a main process and a dynamically loaded library.
Both are using Lttng userspace tracepoints.

On process exit (initiated by ctrl-C) the main process does its cleanup and calls dlclose() on the dynamic libary.
That part is handled normally.

The issue is that later the main process gets a SEGV in the lttng-ust library cleanup logic.
Does the lttng internals still have references to the unloaded memory.

[Switching to Thread 0xfffff722b010 (LWP 18895)]
0x0000fffff7ea3b1c in ?? () from /lib/liblttng-ust.so.0
(gdb) where
#0 0x0000fffff7ea3b1c in ?? () from /lib/liblttng-ust.so.0
#1 0x0000fffff7fdcb1c in *dl_fini () at dl-fini.c:138*
#2 0x0000fffff79e12d0 in __run_exit_handlers (status=0, listp=0xfffff7b125c8 <_exit_funcs>,
run_list_atexit=run_list_atexit@entry=true, run_dtors=run_dtors@entry=true) at exit.c:108
#3 0x0000fffff79e1434 in __GI_exit (status=<optimized out>) at exit.c:139
#4 0x0000fffff79cdce8 in __libc_start_main (main=0xaaaaaaab2a60 <main>, argc=3, argv=0xffffffffbf8, init=<optimized out>,
fini=<optimized out>, rtld_fini=<optimized out>, stack_end=<optimized out>) at ../csu/libc-start.c:342
#5 0x0000aaaaaaab336c in _start () at ../sysdeps/aarch64/start.S:94
Backtrace stopped: previous frame identical to this frame (corrupt stack?)
(gdb) q
A debugging session is active.

```
Inferior 1 [process 18895] will be killed.
```

History

#1 - 03/05/2020 10:07 AM - Mathieu Desnoyers

- Assignee set to Mathieu Desnoyers

- Project changed from LTTng to LTTng-UST

Let's start with a likely probable cause. As documented in lttng-ust(3):

```
      Note that it is not safe to use dlclose(3) on a tracepoint provider
      shared object that is being actively used for tracing, due to a lack of
      reference counting from LTTng-UST to the shared object.
```

So a few questions about this specific application:

- Does the dlclose'd library contain a tracepoint provider object, or depends on a .so which contains a tracepoint provider object ?
- Is there an active tracing session targeting the UST (userspace) tracing domain when this happens ? Does the problem show up with both tracing enabled and disabled ?
- Can you provide the log reproducing the issue with the application launched with the environment variable LTTNG_UST_DEBUG=1 set ?
- Can you provide a gdb backtrace of the SIGSEGV with the symbols corresponding to the addresses for the lttng-ust.so.0 symbols ?