LTTng and Related Projects Update

DORSAL Progress Meeting

December 2022

EfficiOS

Outline

- LTTng 2.14 (next release)
  - Aggregation maps / Trace Hit Counters
- LTTng 2.15 and Babeltrace 2.1
  - Common Trace Format 2 (CTF 2)
- Restartable Sequences: Concurrency IDs
  - Ongoing discussions
- libside

LTTng 2.14

- LTTng is used in production by most of our customers
  - We have identified a few common pain points that we're addressing
- Key limitations of ring-buffer tracing
  - Memory overhead (size and bandwidth)
  - CPU overhead (reading the current time is not always cheap)
  - Requires a post-processing phase to be useful
- Any trade-offs we can explore?

Recording vs. aggregation: level of details

- Recording: exact recording, order of events, precise timing, context from event payloads, …

```
[18:11:50.275355561] (+0.000000873) carbonara syscall_entry_recvmsg:
                                    { cpu_id = 5 }, { fd = 20, msg = 140676324897776, flags = 0 }
[18:11:50.275356143] (+0.000000582) carbonara kmem_kfree:
                                    { cpu_id = 5 }, { call_site = 0xFFFFFFFF94F5179D, ptr = 0x0 }
[18:11:50.275356397] (+0.000000254) carbonara syscall_exit_recvmsg:
                                    { cpu_id = 5 }, { ret = -11, msg = 140676324897776 }
[18:11:50.275358773] (+0.000002376) carbonara syscall_entry_recvmsg:
                                    { cpu_id = 5 }, { fd = 20, msg = 140676324897792, flags = 0 }
[18:11:50.275359412] (+0.000000639) carbonara kmem_kfree:
                                    { cpu_id = 5 }, { call_site = 0xFFFFFFFF94F5179D, ptr = 0x0 }
[18:11:50.275359733] (+0.000000321) carbonara syscall_exit_recvmsg:
                                    { cpu_id = 5 }, { ret = -11, msg = 140676324897792 }
```

Recording vs. aggregation: level of details

- Aggregation: simply count occurrences of event rule matches

```
+---------------------------------------+-----------+----+----+
| key                                   |       val | uf | of |
+---------------------------------------+-----------+----+----+
| syscall_entry_recvmsg                 | 3,404,391 |  0 |  0 |
+---------------------------------------+-----------+----+----+
| kmem_kfree                            |   611,014 |  0 |  0 |
+---------------------------------------+-----------+----+----+
```

Per-CPU arrays of counters

- Associate a key (string) to a value
- Configurable width (32/64 bits)
- Configurable size (number of counters)
- Indicates underflow/overflow

- Not a new concept for kernel users
  - `BPF_MAP_TYPE_PERCPU_ARRAY`
  - Now available to the user space tracer too

Maps are presented like a regular back-end

- Create a user space map named **`my_map`** with session **`my_session`**

```
$ lttng add-map --userspace --session=my_session
                --bitness=64 --max-key-count=1024
                my_map
```

```
Every 2.0s: lttng view-map my_map                carbonara: Tue Dec  6 14:55:01 2022

Session: 'my_session', map: 'my_map', map bitness: 64
Kernel global map, CPU: ALL
+-----------------------+---------+----+----+
| key                   |     val | uf | of |
+-----------------------+---------+----+----+
| syscall count         |   60363 |  0 |  0 |
+-----------------------+---------+----+----+
| syscall error count   |    4744 |  0 |  0 |
+-----------------------+---------+----+----+
```

```
                                                --action incr-value
                                                --session my_session
                                                --map my_map
                                                --key "[X, Y[" 
Adding trigger with filter: 'size >= 0 && size < 2'
Adding trigger with filter: 'size >= 2 && size < 4'
Adding trigger with filter: 'size >= 4 && size < 8'
Adding trigger with filter: 'size >= 8 && size < 16'
Adding trigger with filter: 'size >= 16 && size < 32'
Adding trigger with filter: 'size >= 32 && size < 64'
Adding trigger with filter: 'size >= 64 && size < 128'
Adding trigger with filter: 'size >= 128 && size < 256'
Adding trigger with filter: 'size >= 256 && size < 512'
Adding trigger with filter: 'size >= 512 && size < 1024'
Adding trigger with filter: 'size >= 1024 && size < 2048'
Adding trigger with filter: 'size >= 2048 && size < 4096'
Adding trigger with filter: 'size >= 4096 && size < 8192'
Adding trigger with filter: 'size >= 8192 && size < 16384'
Adding trigger with filter: 'size >= 16384 && size < 32768'
Adding trigger with filter: 'size >= 32768 && size < 65536'
Adding trigger with filter: 'size >= 65536'
$ watch lttng view-map histogram_map
```

In [4]:

Performance of aggregation maps

- As expected, they are a lot cheaper to use than ring-buffer tracing

Method	**Time per event (ns)**	**σ (stdev)**
LTTng-UST ring-buffer (4 × 8 MiB)	158	0.222
LTTng-UST map	43.3	0.656
LTTng-modules ring-buffer (4 × 8 MiB)	151	0.824
LTTng-modules maps	44.8	0.219
eBPF per-CPU array	57.0	0.683

Benchmark code available, see reference slide

Future work for aggregation maps

- Native histogram support
- Decrement value
- Use event payload in the `incr-value` action
- Use event size in the `incr-value` action (dry run mode)

Common Trace Format 2.0

- Implementation ongoing. Planned release in Babeltrace (2.1) and LTTng (2.15)
  - Allowed us to validate the specification (produce and consume)

- `CTF2-SPECRC-6.0` was released on 8 July 2022 (simplifications)
  - Remove variable-length bit array, keeping only variable-length integers
  - Made metadata stream UUID optional
  - Remove trace environment data stream

Restartable Sequences (rseq) ABI extensions

- NUMA node id (`node_id`)
  - Implement a faster getcpu(2) in libc
  - Implement fast node-local memory allocation
- Per memory-map concurrency id (`mm_cid`)
  - Ideal scaling of user space per-cpu data structures
  - Concurrency id is bounded by the number of concurrently running threads for a given memory map at any given time.
- Per memory-map NUMA cid (`mm_numa_cid`)
  - Maintain NUMA-locality of per-cpu data structures
- Positive reception by the community
  - Possibly upstreamed during the next merge window

libside: Software Instrumentation Dynamically Enabled

- New project
  - Tracer-agnostic application instrumentation framework
  - Usable from the purely user space tracers and from the kernel
- Declare events statically without code generation
  - Reduced code footprint (less impact on the instruction cache)
  - More flexible type system (variants, nested types, dynamic compound types)
- Spurred by the upstreaming of *User events* (Microsoft) into the Linux kernel
  - The infrastructure has a few shortcomings
  - Currently marked a *broken* to prevent its use before feedback is addressed

ROCgdb: GDB for AMD GPUs

- Working with AMD to add GPU debugging support into gdb
- First set of 12 patches has been posted for upstreaming
  - Preliminary support: break on GPU code, show GPU and CPU thread state
- Future work
  - AMDGPU DWARF extensions (e.g. stack unwinding),
  - Lane divergence support.

```
(gdb) b func
Breakpoint 2 at 0x7ffff551280c: file bit_extract.cpp, line 40.
(gdb) c
Continuing.
[Switching to thread 43, lane 0 (AMDGPU Lane 2:1:1:38/0 (9,0,0)[64,0,0])]

Thread 43 "bit_extract" hit Breakpoint 1, with lanes [0-63], bit_extract_kernel (C_d=0x7ffdef000000, A_d=0x7ff
def600000, N=1000000) at bit_extract.cpp:48
```

ROCm Tools: CTF output plug-in

- Working with AMD to add CTF production support to ROCm Tools
- Currently produces a text-based trace format
  - Hard to use for viewers
- CTF is more suited to storing large traces
  - Binary format
  - Preserves fields' types and semantics
- Producing an output plug-in based on barectf
  - Initial prototype by Yoann Heitz (DORSAL) in 2021

barectf

Roadmap

- LTTng 2.14: March 2023
- LTTng 2.15 and Babeltrace 2.1: September 2023
  - Depends on changes to CTF 2 (requirements of libside and user events)
- RSEQ Concurrency ID: Next kernel merge window?
- libside: Unknown, still evolving rapidly

References

- Aggregation maps benchmark repository

  https://github.com/jgalar/LinuxCon2022-Benchmarks

- Preliminary AMDGPU gdb support patch set

  https://inbox.sourceware.org/gdb-patches/20221206135729.3937767-1-simon.marchi@efficios.com/T/

- AMDGPU DWARF extensions

  https://llvm.org/docs/AMDGPUDwarfExtensionsForHeterogeneousDebugging.html

- CTF 2 Release Candidate 6

  https://diamon.org/ctf/files/CTF2-SPECRC-6.0.html

- RSEQ node id and mm concurrency id extensions path set

  https://lkml.org/lkml/2022/11/22/1176

- User events – but not quite yet

  https://lwn.net/Articles/889607/

- libside repository

  https://github.com/efficios/libside