LTTng and Related Projects Update

DORSAL Progress Meeting

June 2023

EfficiOS

Outline

- New members joined EfficiOS team
- Ongoing collaborations
- 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
- Libside
- Userspace RCU library
- ROCgdb
- ROCm LTTng-UST instrumentation
- Tracing Summit

New members joined EfficiOS team

- Olivier Dion,
- Erica Bugden,
- Kienan Stewart.

Ongoing collaborations

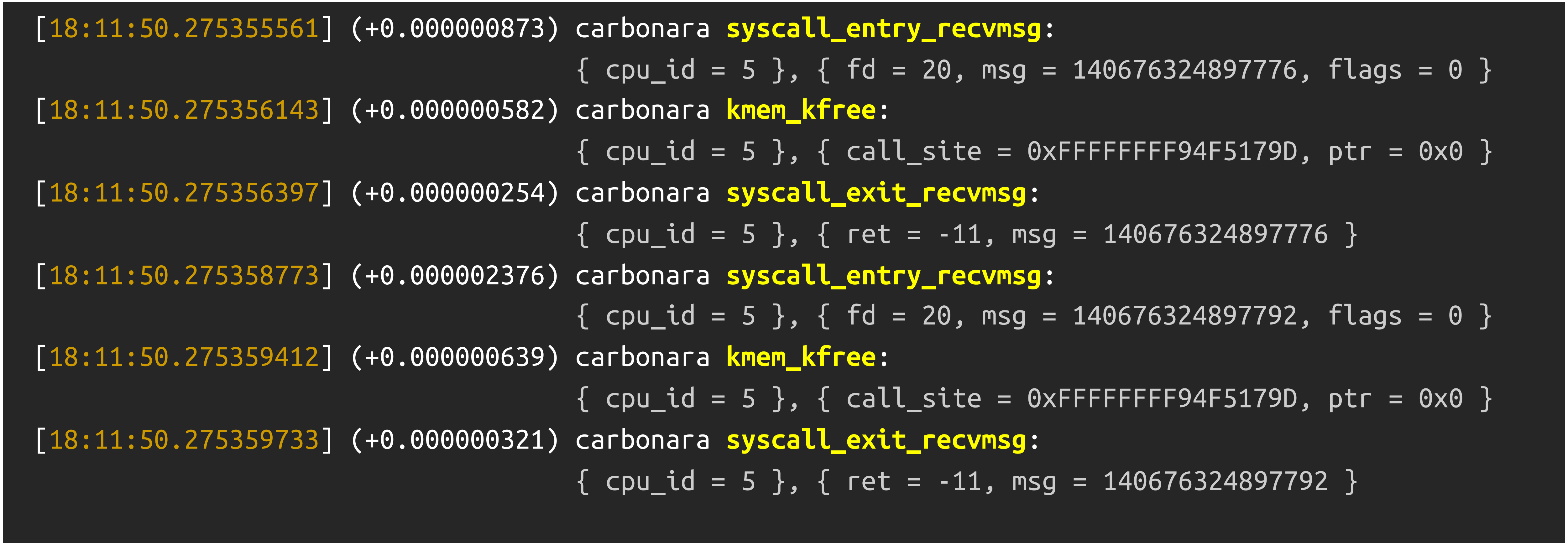
- Ericsson,
- Ciena,
- AMD and Lawrence Livermore National Laboratory,
- Argonne National Laboratory,
- Internet Systems Consortium (ISC).

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

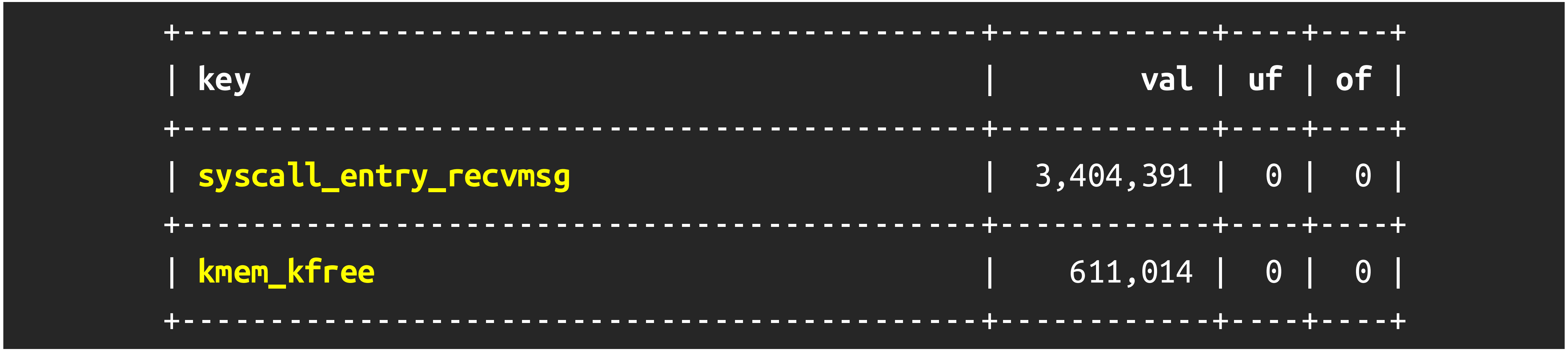
- Recording: exact recording, order of events, precise timing, context from event payloads, …



```
[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
```

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)
  - Allows us to validate the specification (produce and consume)
- `CTF2-SPECRC-7.0` was released on April 7, 2023
  - Add field class alias,
  - Add relative field location,
  - Make it possible to specify user-defined clock origins,
  - Replace UUID property of trace class fragment with UID property (any string).

Restartable Sequences (rseq) ABI extensions

- NUMA node id (`node_id`) (merged in Linux 6.3)
  - Implement a faster getcpu(2) in libc
  - Implement fast node-local memory allocation
- Per memory-map concurrency id (`mm_cid`) (merged in Linux 6.3)
  - 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.
  - Caused scheduler performance regression on Intel Sapphire Rapids fixed in Linux 6.4.
- Per memory-map NUMA cid (`mm_numa_cid`) (work in progress)
  - Maintain NUMA-locality of per-cpu data structures
- Expose scheduler state and thread ID for userspace adaptative mutexes. (work in progress)
- Per-namespace (shared memory) concurrency id (future work)

libside: Software Instrumentation Dynamically Enabled

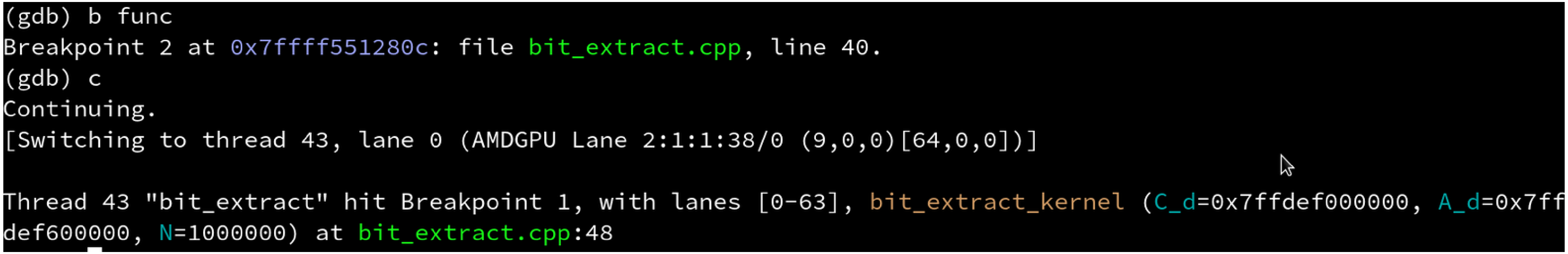
- New project
  - Tracer-agnostic application instrumentation framework
  - Usable from the purely user space tracers and from the kernel
- Declare events statically without code generation
  - Reduced code footprint (less impact on the instruction cache)
  - More flexible type system (variants, nested types, dynamic compound types)
- Spurred by the upstreaming of *User events* (Microsoft) into the Linux kernel

Userspace RCU library

- Now used by the BIND name server,
- Requirement that Userspace RCU QSBR and the liburcu-cds data structures support ThreadSanitizer (TSAN),
    - Moving liburcu memory model to C11 atomics,
    - Deprecating liburcu-signal
    - Add annotation infrastructure to validate multiple stores/loads associated with a single release/acquire barrier:
        - Acquire group,
        - Release group.

ROCgdb: GDB for AMD GPUs

- Basic support for ROCm / AMD GPU merged upstream
- Working on subsequent pieces
- To be truly useful, need to add support for the AMDGPU DWARF extensions, scheduled to be discussed by the DWARF standards committee



```
(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

- ROCm 5.5 adds CTF production support to ROCm Tools
    - Output plug-in based on barectf
- Integration of LTTng-UST with ROCm (ongoing work)
    - Exa-Tracer project

Roadmap

- LTTng 2.14: September 2023
- Babeltrace 2.1: September 2023
- LTTng 2.15: January 2024
- libside: Unknown, still evolving rapidly
- Userspace RCU 0.15: August 2023

Submit a talk for

Tracing Summit 2023

Two weeks remaining for proposals

September 17-18
With OSS Europe in
Bilbao, Spain

Info and submission at
tracingsummit.org

Talk proposal deadline
June 16th

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 7

  https://diamon.org/ctf/files/CTF2-SPECRC-7.0.html

- RSEQ node id and mm concurrency id extensions patch set

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

- sched: Fix performance regression introduced by mm_cid

  https://lore.kernel.org/lkml/20230420145548.232747-1-mathieu.desnoyers@efficios.com/

- User trace events – one year later

  https://lwn.net/Articles/927595/

- libside repository

  https://github.com/efficios/libside