LTTng and Related Projects Update

DORSAL Progress Meeting

December 2024

EfficiOS

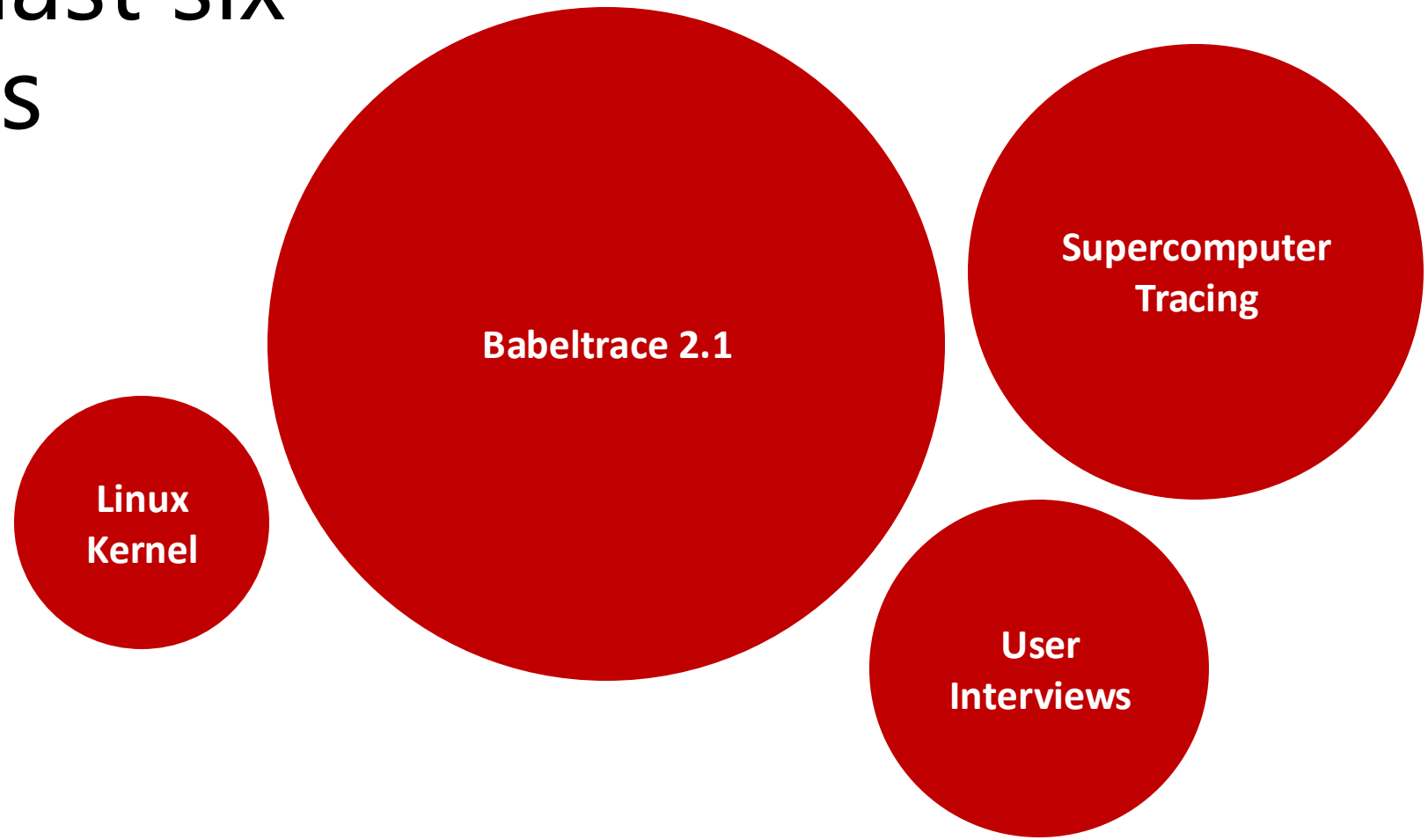
Outline

- Last Six Months
- Upcoming Work

In the

Last six months

R&D Activities in the last six months



Babeltrace 2.1 – Moving towards CTF 2

Babeltrace 2.1 – Add reading and producing CTF 2 traces

- Release: rc-1 within the next few weeks!

LTTng 2.15 – Producing CTF 2 traces

- Release: TBD (aiming for Q4 2025)

What is CTF 2 ?

The Common Trace Format (CTF) is:

- A binary trace format
- Fast to write
- Flexible

CTF 2 is a major revision of CTF 1, bringing many improvements.

Field classes common to CTF 1 and CTF 2

Field class	CTF 1.8	CTF 2
Fixed-length integer	✓	✓
UTF-8 string	✓	✓
Floating point number	✓	✓
Fixed-length array	✓	✓
Dynamic-length array	✓	✓
Structure	✓	✓
Variant	✓	✓

What does CTF 2 do better than CTF 1?

	CTF 1.8	CTF 2
Metadata format	TSDL (custom DSL) • Non-trivial to parse.	JSON text sequences • Widely used standard format with pre-existing parser libraries in various languages.
Augment events and fields with user-defined metadata	✗ 😿	✓ 😺 • Associate user-defined name to a value. • Used to tailor analysis or pretty printing of trace data.

What does CTF 2 do better than CTF 1?

Field class	CTF 1.8	CTF 2
BLOB	✗	✓ • Record opaque binary blobs • IANA media type attribute
Optional	✗	✓
LEB128 variable length integer	✗	✓ • Values > 64-bit range • Common need in scientific computing
UTF-16 and UTF-32 string character encoding	✗	✓ • Native string encoding on some platforms • E.g. Windows, Java VM
Fixed-length bit map	✗	✓ • Associate names to specific bits in a bitmap • Useful to represent flags
Boolean	✗	✓

Supercomputer Tracing

1st & 3rd fastest in TOP500

El Capitan Supercomputer – Lawrence Livermore National Lab

- Instrumentation of necessary libraries (e.g. MPI)
- Integration with existing AMD tooling (ROCm)

Aurora Supercomputer – Argonne National Lab

- Enable on the fly analysis of massive amounts of trace data
- Via Babeltrace performance optimizations, limited-size trace footprint

Linux Kernel & Community Work

Laying the foundation to...

Reduce userspace tracer CPU and memory overhead

- Reduce CPU execution constraints by replacing hardware atomic instructions with kernel-managed software transactions
- Bound memory allocation to max number of concurrently running threads (rather than allocate for each CPU)

Enable kernel tracer to have previously unavailable data

- Handle page faults while tracing system calls

User Interviews

User Interviews

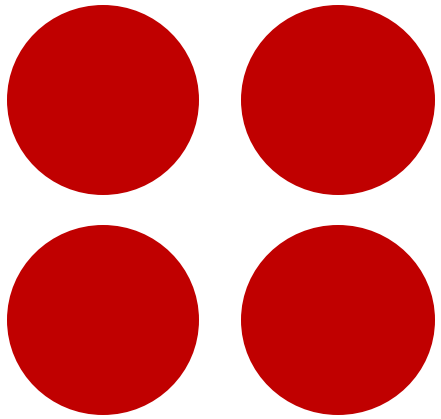
Develop active connections with tracing users to...

- Shorten feedback loops
- Improve feedback accuracy

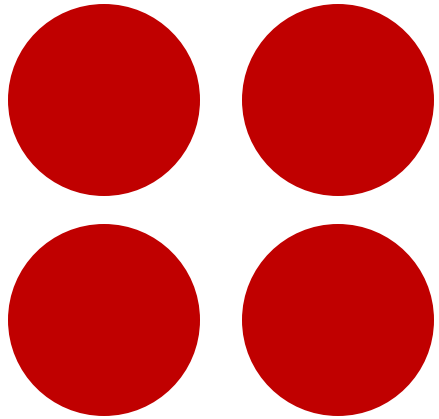
The aim is to deliver more impact in less time.

User Interviews

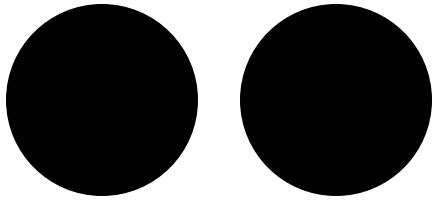
Troubleshooters



Feature Developers



Tool Developers



6/10

Offered (unsolicited!) to give input again

Upcoming* Work

Core Projects

Babeltrace

- Reading and producing CTF 2 traces (2.1, **next few weeks!**)

LTTng

- Trace Hit Counters (upcoming 2.14, 2025)
- Producing CTF 2 traces (upcoming 2.15, TBD)

Important Problems

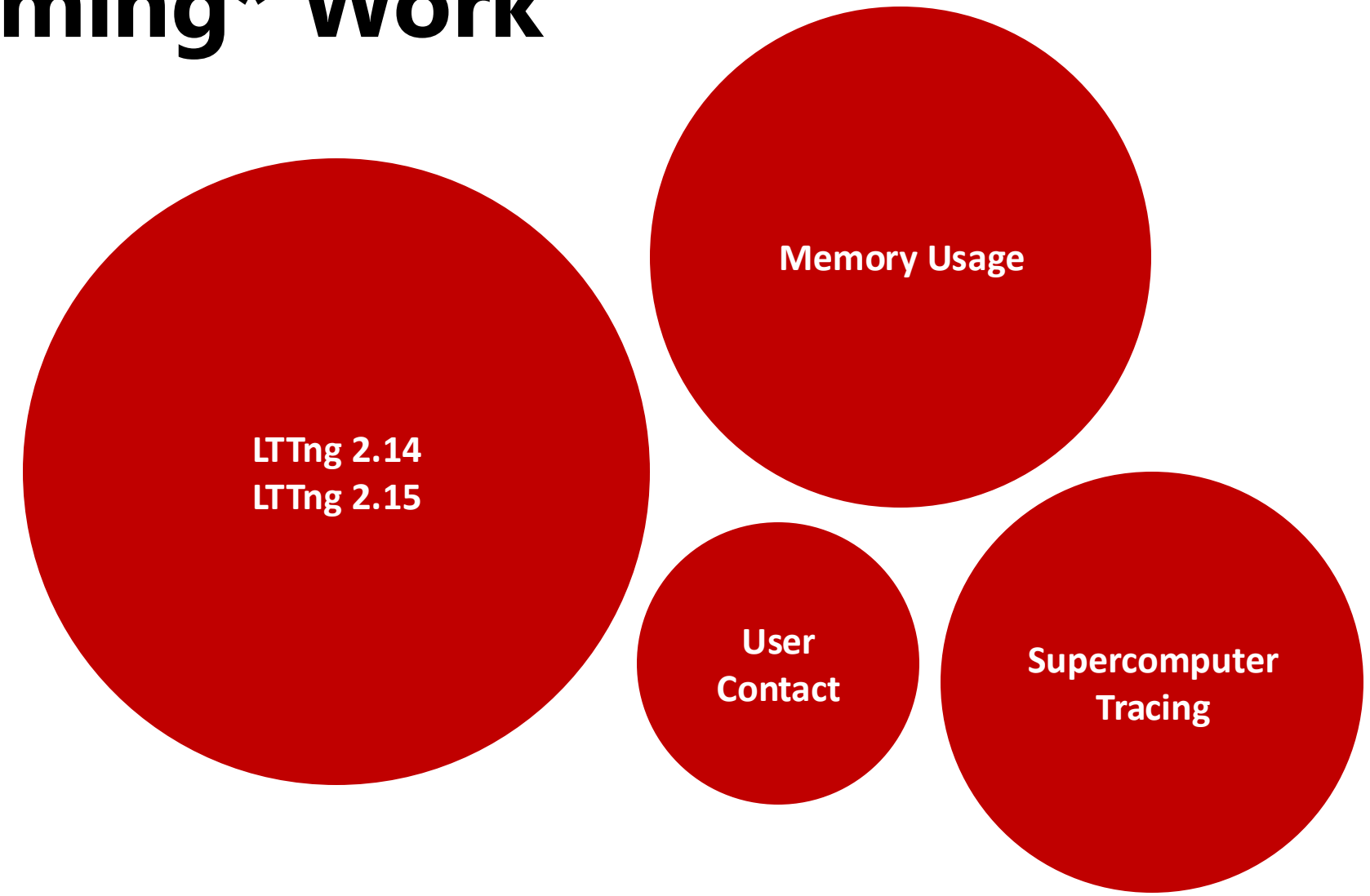
High memory usage of tracing buffers

- Buffers currently allocated per CPU and per container
- Quickly not viable in a typical containerized environment

Wasting time processing incomplete traces

- Know as early as possible if the information you want was not captured, so you can adjust and retry
- Maximize detail without overwhelming the system

Upcoming* Work



Questions ?

- Links:
  - https://www.efficios.com
  - https://lttng.org
  - https://babeltrace.org
  - https://diamon.org
  - https://barectf.org

Annex

References

- Common Trace Format 2 Specification

  https://diamon.org/ctf

- libside repository

  https://github.com/efficios/libside

SIDE ABI RFC (libside)

- The SIDE ABI is currently at RFC stage, aiming to create a specification.
  - https://github.com/efficios/libside/blob/master/doc/rfc-side-abi.txt
- Runtime/language agnostic,
- Supports multiple concurrent tracers,
- Instrumentation is not specific to a tracer,
  - No need to rebuild applications if using a different tracer,
- Instrumentation can be either static or dynamic,
- Supports complex/nested types,
- Supports both static and dynamic types,
- Libside is a C/C++ reference implementation for the System V ELF ABI.