Polytechnique Montréal - June 2021

LTTng and Related Projects Updates

EfficiOS

Outline

- LTTng 2.13
- LTTng 2.14’s ongoing development
- LTTng-modules Upstreaming
- Babeltrace 2.1
- CTF 2.0
- Restartable Sequences

EfficiOS

LTTng 2.13 – Release Candidate phase

- 2.13.0 release is imminent.

- Currently in Release Candidate 2 phase.
  - Released May 15 2021
    - Major documentation overhaul.

- Release Candidate 1
  - Released April 23 2021

EfficiOS

LTTng 2.13

Expanding the *trigger* mechanism is the major focus of this release

- Allows users to specify actions to be taken when a specific condition is met.

Triggers were introduced in LTTng 2.10 to:

- Notify external applications when tracing buffer usage reached a given threshold,
- Allows a controller to disable certain events of lesser importance.

LTTng 2.13

New actions and conditions indicated in bold.

Supported conditions:

- Buffer usage threshold
- Session consumed size
- Rotation ongoing or completed
- **Event rule matches**

Supported actions:

- Notify
- **Record snapshot**
- **Rotate session**
- **Start session**
- **Stop session**
- **List of actions**

LTTng 2.13

Allow the capture and transmission of specific event payload and context fields along with a notification.

- Allows external applications to use the context of an event to take a decision (such as taking a snapshot on another machine).

```
$ lttng add-trigger --condition event-rule-matches --type=user
                    --name “sample_component:*”
                    --capture field1
                    --capture field2[4]
                    --capture '$ctx.vtid'
                    --action notify
```

EfficiOS

LTTng 2.13 – LTTng-UST

Removed the dependency on `liburcu` from the LTTng user space tracer:

- Avoid compatibility issues between applications and LTTng-UST which may be linked against different `liburcu` versions.

User space RCU is not stable yet (0.x release), major bumps will most likely still occur

- Linking two versions of `liburcu` in against an application results in a number of conflicts/symbol clashes,
- We can't expect applications which use `liburcu` to be updated in lock-step with the tracing infrastructure to use the same library version.

The subset of liburcu needed by LTTng UST is integrated in the project directly.

LTTng 2.13 – LTTng-UST Library ABI

- Introduce first LTTng-UST library ABI breaking change in 10 years.
- Requires instrumented applications and tracepoint probe providers to be recompiled against LTTng-UST 2.13.
- Motivation for bumping the LTTng-UST tracer library major soname from 0 to 1:
  - Since LTTng-UST 2.0, many internal data structures and symbols were exposed publicly.
  - This prevented introducing changes required for the event notifier and captures, as well as upcoming trace hit counters features, without a lot of code duplication, and would have made it challenging to maintain the code base over time.
- We used this ABI break opportunity to clean up the ABI and remove everything that should be private from public headers. This should hopefully remove the need for LTTng-UST ABI breaks in the future.
- LTTng-Tools 2.13 interacts with LTTng-UST 2.11, 2.12 and 2.13, thus allowing gradual rebuild of instrumented applications by keeping shared objects of LTTng-UST 2.11/2.12 and 2.13 installed in parallel.

LTTng 2.13 – LTTng-UST API

- Introduce new instrumentation API (v1) with proper namespacing, e.g. “tracepoint()” is renamed to “lttng_ust_tracepoint()”.
- Original instrumentation API (v0) still available.
- Compatibility with old API (v0) can be removed with the following define before including LTTng-UST headers:

```
#define LTTNG_UST_COMPAT_API_VERSION 1
```

LTTng 2.14 – Ongoing Development

Trace Hit Counters:

- Per-CPU array of counters
- New back-end (counting instead of serializing events)
- Non-blocking

Use cases:

- Count the number of event rule hits, without tracing to a ring buffer.
- Estimate the impact of a given tracing configuration on a live production workload.
- Collect statistics.

LTTng 2.14 – Ongoing Development

Implemented as part of `libcounter`

- Per-CPU split-counters implementation:
  - In kernel memory (LTTng kernel tracer),
  - In shared memory (LTTng user space tracer).

In summary:

- Mapping between events and keys is dynamically configurable by the user.
- Mapping between keys and counter indexes is done on a slow-path (when instrumentation is registered).
- Fast-path (in application/kernel) only needs to increment a per-cpu counter indexed within an array.
- Sum per-CPU counters for a given key when viewed.

EfficiOS

libcounter

Multi-dimension array of counters

- Only one dimension exposed initially, but additional dimensions will be useful for future use-cases
  - e.g. Aggregation based on field value

Counters are configurable:

- Array length for each dimension,
- Per-CPU/Global,
- Modulo or saturation arithmetic,
- Size of each counter: 8, 16, 32, or 64-bit (limited by architecture atomic operation size).

LTTng will initially only expose the needed subset of the libcounter features to end users.

EfficiOS

Trace Hit Counter (1/3)

- Trace hit counter are specified using
  - One `event-rule-matches` condition, and
  - One or more `incr-value` action(s).
    - Applying to a specific session and map.

- Flexible key description
  - Arbitrary keys created using the key syntax
    - Literal string,
    - event name variable (`${EVENT_NAME}`), and
    - provider name variable (`${PROVIDER_NAME}`).
  - Key rule examples:
    - `--key "Event category #2"`
    - `--key "${EVENT_NAME}_postfix"`
    - `--key "${PROVIDER_NAME}:${EVENT_NAME}"`
  -

EfficiOS

Trace Hit Counter (2/3)

Example of two trace hit counters on the same event rule:

```
$ lttng add-trigger --id $TRIGGER_NAME \
    --condition event-rule-matches \
      --type=user --name="incr_value_ex:*" \
    --action incr-value \
      --session $SESSION_NAME \
      --map $MAP_NAME \
      --key 'Total number of events' \
    --action incr-value \
      --session $SESSION_NAME \
      --map $MAP_NAME \
      --key '${PROVIDER_NAME} -> ${EVENT_NAME}'
```

Trace Hit Counter (3/3)

```
$ lttng view-map $MAP_NAME

Session: 'incr_value_ex_sess', map: 'incr_value_ex_map', map
bitness: 64
UID: 1000, CPU: ALL
+--------------------------+-----+----+----+
| key                      | val | uf | of |
+--------------------------+-----+----+----+
| Total number of events   |  19 |  0 |  0 |
+--------------------------+-----+----+----+
| incr_value_ex -> event1 |  10 |  0 |  0 |
+--------------------------+-----+----+----+
| incr_value_ex -> event2 |   5 |  0 |  0 |
+--------------------------+-----+----+----+
| incr_value_ex -> event3 |   4 |  0 |  0 |
+--------------------------+-----+----+----+
```

```
Reminder
    --key 'Total number of events'
    --key '${PROVIDER_NAME} -> ${EVENT_NAME}'
```

EfficiOS

LTTng-modules Upstreaming

Faultable tracepoints v2

- LKML: [RFC] Faultable tracepoints (v2) - 2021-02-18
- https://lore.kernel.org/lkml/20210218222125.46565-1-mjeanson@efficios.com/

Overall plan:

- Focus on presenting LTTng-modules differentiator as doing efficient tracing of system calls with user-space argument capture. Break LTTng-modules down into generally useful infrastructure pieces for upstreaming.
- Taking page faults is required in order to capture data from user-space reliably.
- Having a ring buffer allowing preemption to stay enabled between reserve and commit is also needed unless the overhead of an extra copy is acceptable, thus justifying LTTng's ring buffer approach compared to Ftrace.

EfficiOS

Babeltrace 2.1 — Ongoing Development

Development work on Babeltrace 2.1 is ongoing:

- Changes required to support CTF 2.0.
  - Introduce C++ 11 to the code base,
  - Refactoring of the CTF Filesystem Source plugin based on C++ 11 interfaces.

CTF 2.0

EfficiOS is leading the effort to address CTF 1.8 shortcomings (mostly related to lack of extensibility of the metadata format) with a new version of the specification:

- Gathered feedback following the publication of the first draft,
- A second draft of the CTF 2.0 specification is available: https://diamon.org/ctf/files/CTF2-PROP-2.0.html

The ease of adoption of the new tracing format is a key concern. So far, our transition plan is in three phases:

1. Babeltrace 2 will support reading both CTF 1.8 and CTF 2.0,
2. LTTng will produce both CTF 1.8 and CTF 2.0,
3. CTF 1.8 is eventually phased-out and becomes unsupported by LTTng.

CTF 2.0

No major progress has been achieved since January 2021.
Effort was diverted to LTTng 2.13 and other projects.

EfficiOS

Restartable Sequences

Overview

- Restartable Sequence (`rseq`) is a Linux system call implemented by EfficiOS,
- Allow fast per-CPU operations in user space,
- End goal is to eliminate atomic operations from the user space tracer's fast-path,
- Useful for other use-cases (e.g. memory allocators),
- Merged in Linux 4.18.

Restartable Sequences

Missing pieces for LTTng-UST integration:

- Integration of rseq into the GNU C Library.
- Missing Linux kernel feature allowing modification of per-cpu user space data from remote CPUs safely against concurrent CPU-hotplug and cgroup `cpuset` configuration changes, without hurting partitioned latency-sensitive workloads.

Overall, no major progress has been achieved since January 2021. Effort was diverted to LTTng 2.13 and other projects.

Questions?

- lttng.org
- lttng-dev@lists.lttng.org
- @lttng_project
- #lttng OFTC

EfficiOS