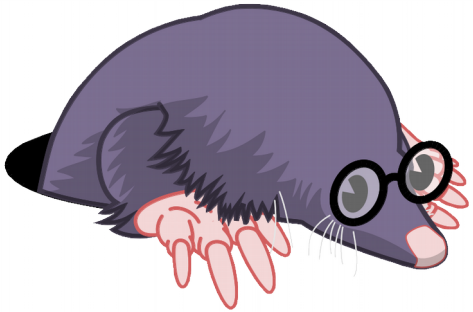
Polytechnique Montréal
May 2020



LTTng and Related Projects Updates

EfficiOS

Outline

- LTTng 2.12
- Babeltrace 2.0
- Restartable Sequences
- Upcoming LTTng features
  - LTTng 2.13 & 2.14

LTTng 2.12 – Release Status

LTTng v2.12.0 released on April 7th 2020 (National Beer Day)

- New Features:
  - UID/GID tracker,
  - File descriptor pooling (relay daemon),
  - Fast clear,
  - Container support (namespace contexts),
  - Working directory override (relay daemon),
  - Trace hierarchy by session or host name (relay daemon),
  - Version tracking.

UID/GID Tracker

- Specialized filtering mechanism for UID/GID tracking:
  - Makes it possible to create tracing buffers only for some users/groups (or applications, in per-PID buffering mode),
  - Works in the same way as the existing PID tracker functionality,

- Reduces memory use on multi-user setups when tracing in *per-UID* mode.

File Descriptor Pooling

- Impose a hard cap on the number of file descriptors opened by the relay daemon (--fd-pool-size),
- The LTTng file format causes many files to be opened simultaneously:
  - Metadata file + one file per data stream (i.e. per CPU),
  - Doubled when a live client is consuming the trace (files opened for writing and reading),
- Many support cases reported file descriptor exhaustion:
  - Not always possible to increase the system limit for administrative reasons (team doesn’t have the necessary permissions on the system).

Clear command

- Discard the data recorded for a session,
- Builds on the work done in 2.11 for session rotations,
- Tracing setup time is greatly reduced for teams running multiple test runs:
  - Run test, read trace, clear,
  - No need to re-create the session, channels, etc.
- Works with live clients:
  - Live clients will skip-ahead to the newest data after a clear,
- Useful when debugging:
  - Try to reproduce a problem, clear between attempts,

```
$ lttng clear --session my_session
```

- Use of clear can be disallowed per relayd process:
  - LTTNG_RELAYD_DISALLOW_CLEAR environment variable.

Container Support (namespace contexts)

- Allow the capture of the namespaces of the current process when an event occurs (available from both kernel and user space tracers):
  - Cgroup,
  - IPC,
  - Mount,
  - Network,
  - PID,
  - User,
  - UTS (hostname and domain name).
- It is then possible to map the events back to a container name (e.g. Docker or LXD user-visible name),
- Namespace hierarchy can be dumped to the trace *on-demand.*

Working Directory Override (Relay Daemon)

- New `--working-directory` option changes the working directory of the relay daemon,
- Helpful for teams who launch the relay daemon from a drive that should be un-mountable,
- Used to set the working directory to a writeable directory so that core dumps can be written.

Trace hierarchy by session or host name

- Two new options for the relay daemon:
  - `--group-output-by-session`,
  - `--group-output-by-host`.
- Allows users to control the path hierarchy of traces produced by the relay daemon:
  - By hostname (default):
    - `relayd_output/host_name/session_name/`
  - By session name:
    - `relayd_output/session_name/host_name/`
- Makes it easier to collect all traces from a cluster.

Version Tracking

- Introduced a mechanism to register out-of-tree changes applied on top of LTTng,
- Objective is to make it easy to know the exact version of LTTng running on systems when a support ticket is created,
- Vendors often add custom patches which can cause problems that are hard to track for us,
- Requires the cooperation of the vendors to “register” those patches at build time:

```
$ lttng --version
```

Babeltrace 2.0

- Babeltrace 2.0.0 released on January 22nd, 2020 (National Hot Sauce Day) after 5 years of development.
- Babeltrace is a trace manipulation toolkit:
  - Library with a C API,
  - Python 3 bindings,
  - Command-line tool.
- Makes it easy to view, convert, transform, and analyze traces.
- Reference parser implementation of the Common Trace Format (CTF).
- Can read and write CTF traces.

Restartable Sequences

- Restartable sequence system call:
  - 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.
- Integration of rseq into the GNU C Library is imminent (likely for glibc 2.32).
- Still working on the missing pieces for LTTng-UST integration:
  - 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.

LTTng 2.13 – Ongoing Work

- Dynamic Snapshots (triggers) is the major focus of this release,
- A new top-level concept will be introduced: triggers
  - Triggers can be associated to an event rule and *trigger* an action when that event rule is met,
- Supported actions:
  - Start tracing,
  - Stop tracing,
  - Rotate session,
  - Record snapshot,
  - Notify.

Dynamic Snapshot / Triggers

```
$ lttng create-trigger --id my_id
        --userspace
        --tracepoint provider:hello
        --filter ‘caller_id == 1422432’
        --action stop session_name
        --action snapshot session_name
```

- When the `hello` event occurs with `caller_id` 1422432, a session is *stopped* and a snapshot is *recorded.*

Dynamic Snapshot / Triggers

- The *notify* action allows external applications to receive the contents of an event associated to a trigger,
- Allows complex scenarios that reach beyond the scope of LTTng, for example:
  - A communication error occurs in a code path instrumented with an LTTng tracepoint,
  - An application can listen for that specific event and receive a notification when it occurs,
  - Inspect the payload of the event to connect to the machine that was involved and take a snapshot on *that* machine.

Dynamic Snapshot / Triggers

- Like regular events, triggers can be *dropped* when the system is overloaded:
  - Dropped events are accounted for in aggregation maps,
- Triggers can be associated to counters:
  - Trigger once after *n* matches,
  - Trigger after every *n* matches.

LTTng 2.14 – Planned Features

- Trace Hit Counters
  - Allow doing a dry-run of tracing without producing events into a ring buffer.
  - Figure out if the impact of a given tracing configuration is too high for a live production workload.
  - Count (aggregate) number of events encountered, without tracing to a ring buffer.
  - Per-CPU split-counters:
    - In kernel memory (LTTng kernel tracer),
    - In shared memory (LTTng userspace tracer).
  - Per-CPU array of counters in shared memory.
  - Flexible mapping between events and keys (strings), dynamically configurable.
  - Mapping between keys and counter index done entirely within session daemon.
  - Sum per-CPU counters for a given key when viewed.

Questions ?

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