Polytechnique Montréal
December 2019

LTTng Project Updates

EfficiOS

Outline

- LTTng 2.11
- Upcoming LTTng features
  - LTTng 2.12 & 2.13
- Babeltrace 2.0
- Restartable Sequences

LTTng 2.11 – Release Status

Released on October 19th 2019 (v2.11.0)

Very big release:

- Two years of development,
- Lots of new features,
- Required significant re-engineering:
  - Protocols (no breaking changes),
  - Internal file management.

Spent ~1 year in Release Candidate (beta) to ensure a smooth release:

- Fixing issues uncovered in testing,
- Developing 2.12 in parallel.

LTTng 2.11 – New Features

- Session rotation (*details on following slides*),
- Dynamic tracing of user-space (from kernel, Uprobe-based),
- Support of arrays and bit-wise binary operators in filters,
- User and kernel space call-stack capture (from kernel-space),
- Improved performance of relay daemon:
  - Handling of slow clients and network errors,
- NUMA-aware buffer allocations by the user-space tracer,
- Support unloading of user-space probe providers (dlclose).

Session Rotation

Motivation:

- Tracing can be left running for a long time,
- Resulting traces can be huge,
- Want to process traces as they are being produced,

Apply the concept of log rotations to traces:

- Provide trace archives (“chunks”) that can be processed independently.

Session Rotation – Use-cases

- Process traces before the end of a test run,
- Read traces without stopping traces (without using “live”),
- Pipeline and/or shard trace analysis (scale-out),
- Encryption,
- Compression,
- Clean-up of old chunks (keep a bounded backlog of traces),
- Integration with external message buses (Kafka, ZeroMQ, etc.)

Rotating a tracing session

Immediate rotation:

```
$ lttng rotate --session my_session
```

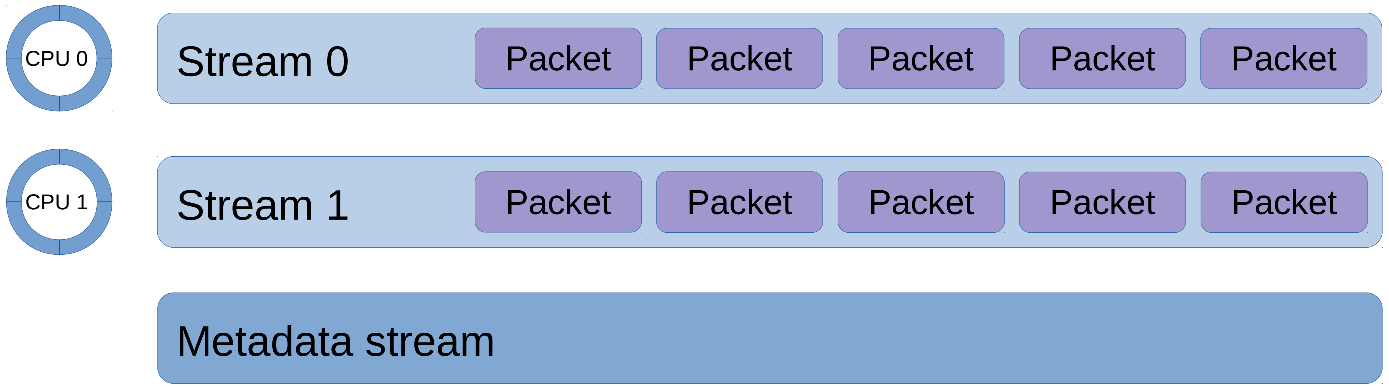
Scheduled rotation:

```
$ lttng enable-rotation --session my_session --timer 30s
$ lttng enable-rotation --session my_session --size 500M
```

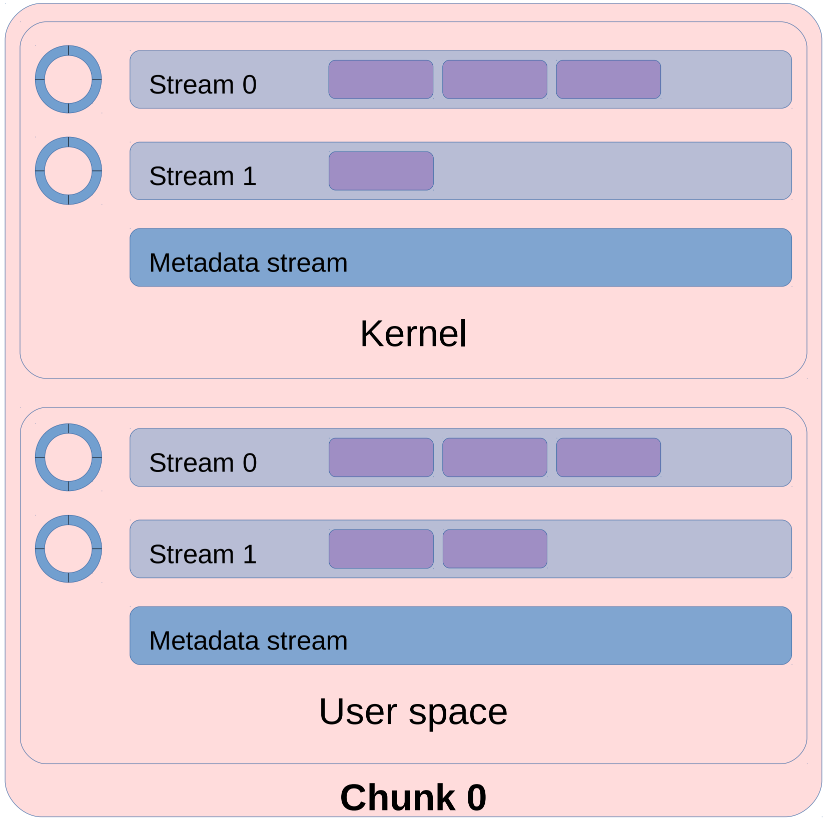
Session Rotation

As produced by LTTng, a CTF trace is a *set* of files

- One *event stream* file per CPU
- A *metadata* file describing the layout of the event streams



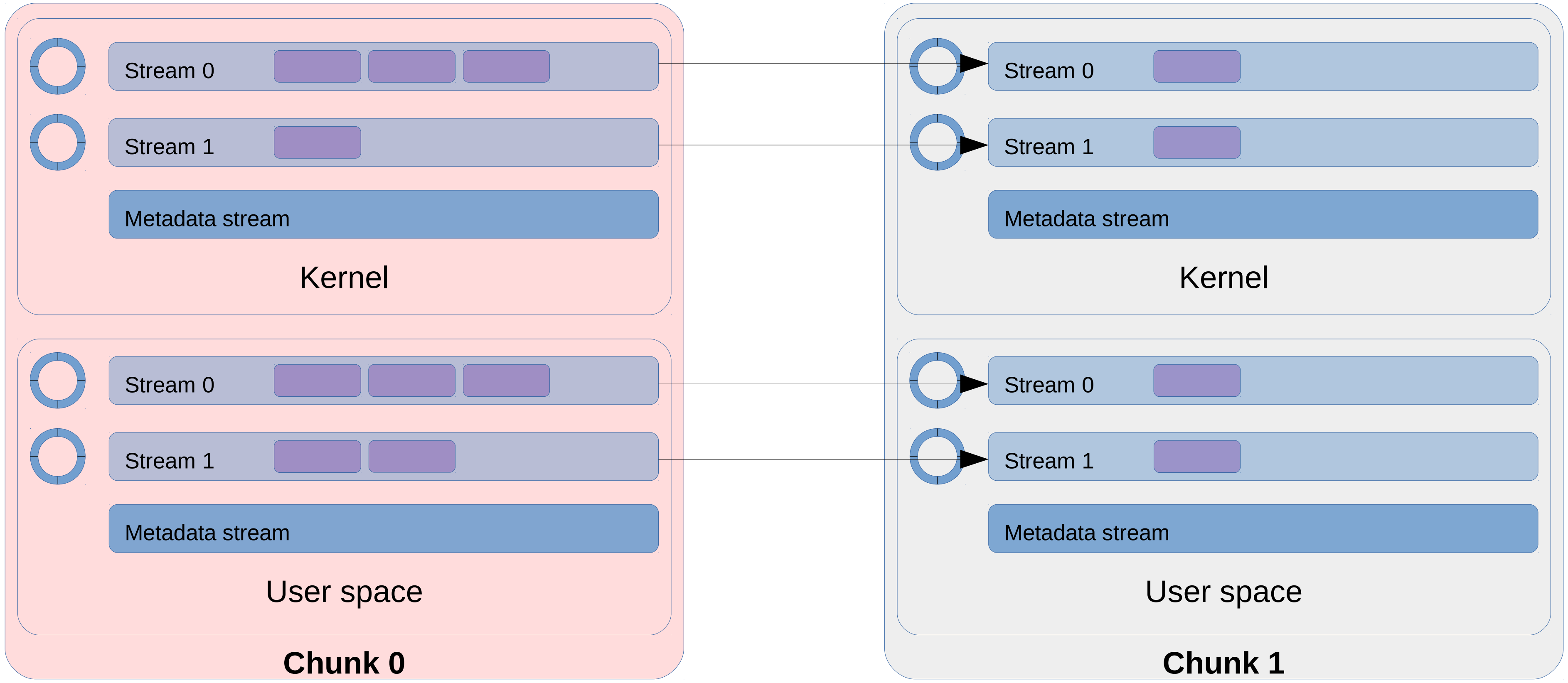
Session rotation – step by step



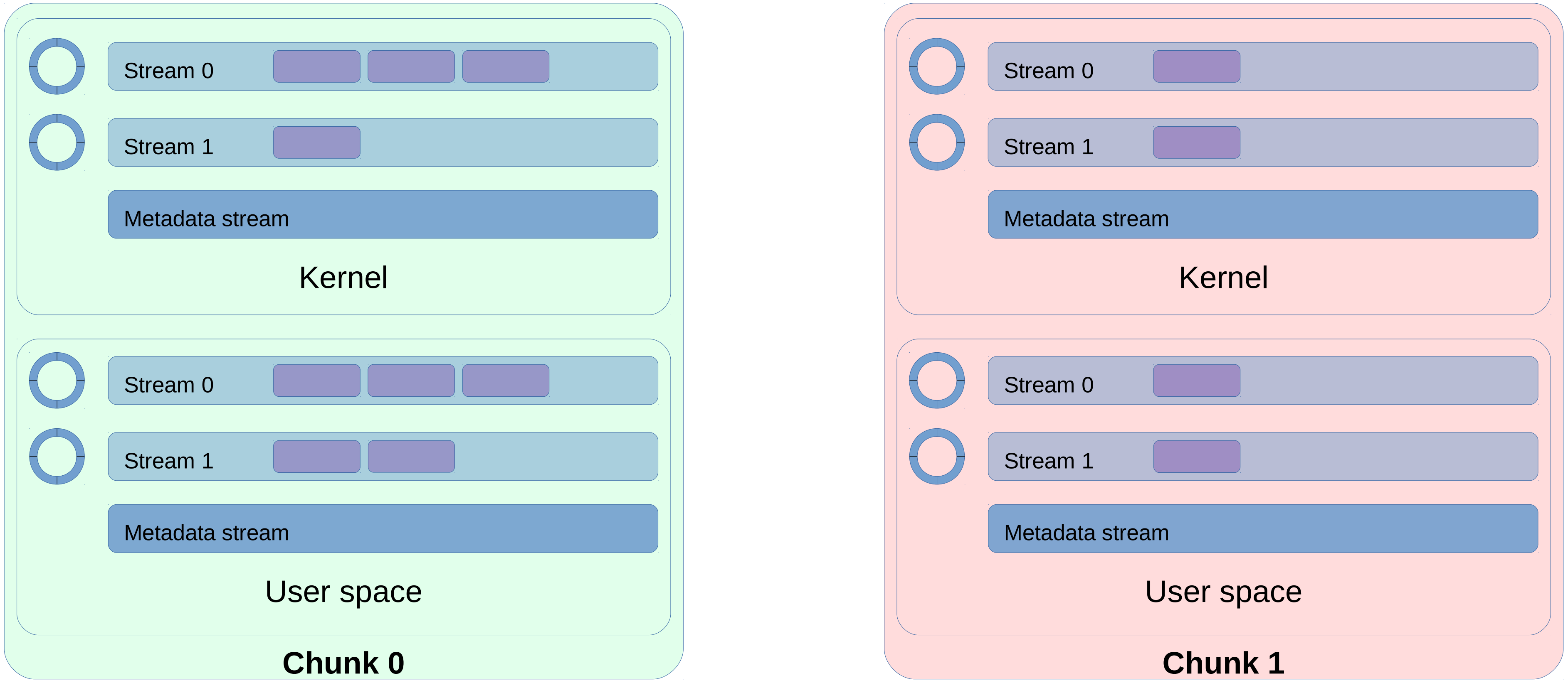
```
$ lttng rotate --session my_session
```

- Sample production position of every stream
- Establish a per-stream “switch-over” point
- Flush the layout description of all events declared up to the “switch-over” point
- Consume tracing data up to the “switch-over” point
- Notify user of trace archive chunk availability

Session rotation



Session rotation



LTTng 2.12 – 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
```

LTTng 2.12 – Release Status

- Currently putting the finishing touches to the clear command:
  - Fixing issues following internal testing.
- Most of the features are present upstream (master branch),
- Release Candidate planned by the end of the year (before December 20th):
  - Final release date depends on the feedback we get,
  - We expect this phase to be fairly short as the changes were not as invasive as previous releases.

LTTng 2.13 – New Features

- 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.

Babeltrace 2.0

- Reaching a stable release after 5 years of development,
- Last year was mostly performance improvements and API clean-ups,
- Focus on easing the transition from Babeltrace 1:
  - Performance is now slightly better than Babeltrace 1,
  - Can co-exist with Babeltrace 1 on the same machine.
- Documentation is the only remaining milestone for release.

Restartable Sequences

- Restartable sequence system call:
  - Allow 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.
- Integrating the syscall in glibc is crucial for adoption,
- Still working on the missing pieces for LTTng-ust integration.

Questions ?

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