FOSDEM 2018

The LTTng Approaches to Solving Complex Problems in Production

EfficiOS

jdesfossez@efficios.com

Content

- Trace buffering, aggregation and sampling.
- What is LTTng ?
- Why LTTng compared to other tracing solutions ?
- LTTng trace extraction modes with use-cases and examples:
  - Disk and streaming,
  - Live,
  - Snapshot,
  - Rotation.
- Conclusion.

Biography

- Julien Desfossez
  - Software Developer at EfficiOS,
  - Works on LTTng kernel and user-space tracers, Babeltrace,
  - Author and maintainer of the latency-tracker and LTTng-Analyses projects.

EfficiOS

Trace Buffering

- Fast and efficient logging:
  - Generate events at specific locations in the code,
  - Extract parameters for later analysis,
  - Application-specific or system-wide.
- Common trace buffering solutions on Linux:
  - ftrace (kernel tracing),
  - perf in some modes,
  - LTTng (kernel and user-space tracing).

Trace Buffering Use-Cases

- Understanding complex problems that require low-level and a high volume of information (e.g: concurrency issues),
- Requires deep knowledge of the operating system or internal behavior of the application,
- Usually the “last line of defense” to fix a problem,
- With LTTng analyses tools, monitoring and cloud use-cases become possible.

Trace Aggregation

- Aggregation tools are used to perform run-time measurements or statistics based on tracing information.
- Common aggregation tools on Linux:
  - SystemTap,
  - eBPF/BCC,
  - latency-tracker.

Sampling or Profiling

- Periodically take a snapshot of the current activity of a system,
- Extract statistics and hot spots,
- Commong profiling tools on Linux:
  - perf,
  - oprofile,
  - gprof.

LTTng Advantages

- **Fast kernel tracing** (same speed as ftrace but extracts the syscalls payload),
- **Fast user-space tracing** (does not rely on system calls at every event), native support for C/C++ applications, agents for Java and Python,
- Designed to **run continuously in production environments,**
- Multi-platform: x86, ARM, PPC, MIPS, s390, Tilera,
- Ability to merge kernel and user-space traces,
- Multi-host/clock support,
- Standard trace format (Common Trace Format),
- Packaged by the major distributions,
- Standalone kernel modules,
- **Vast ecosystem of analysis and post-processing tools.**

LTTng Trace Recording Modes

- Tracing to disk with all kernel events enabled can quickly generate huge traces:
  - 54k events/sec on an idle 4-cores laptop, 2.2 MB/sec
  - 2.7M events/sec on a busy 8-cores server, 95 MB/sec
- In addition to filtering and enabling specific events, LTTng offers various recording modes:
  - Local disk and streaming mode,
  - Live mode,
  - Snapshot mode,
  - Rotation mode (new in 2.11).

Disk and Streaming Modes

- Default mode,
- Write buffers to disk or the network when they are full,
- Only limited by disk space,
- Tracing session needs to be stopped to process the trace,
- Use-cases:
  - Understanding the complete life-cycle of a system or an application,
  - Trace exploration (need to identify what is relevant),
  - Post-mortem analyses,
  - Reverse engineering,
  - Continuous Integration.

Disk and Streaming Modes

```
$ lttng create # For streaming: -U net://<server>
$ lttng enable-event -k -a # All kernel events
$ lttng enable-event -u -a # All user-space events
$ lttng start
...
$ lttng stop
$ lttng view
$ lttng destroy
```

Disk and Streaming Mode - Example

- Sometimes users complain that the “website is slow”,
- We do not see anything in the monitoring tools (averages, percentiles, etc),
- Problem seems to happen periodically but we can only rely on users to report it,
- Methodology:
  - Record all the I/O, scheduling and system calls activity on the webserver,
  - When a problem is reported, run statistics tools on the trace.
- Full writeup on this case: https://lttng.org/blog/2015/02/04/web-request-latency-root-cause/

Top system call latencies

```
lttng-iolatencytop /path/to/trace --limit=3 --minsize=2
```

```
Checking the trace for lost events...
Timerange: [2015-01-15 12:18:37.216484041, 2015-01-15 12:18:53.821580313]
Top open syscall latencies (usec)
Begin              End                Name           Duration (usec)       Size  Proc          PID    Filename
[12:18:50.432950815,12:18:50.870648568] open              437697.753        N/A  apache2       31517  /var/lib/php5/sess_0ifir2hangm8gga
[12:18:52.946080165,12:18:52.946132278] open                  52.113        N/A  apache2       31588  /var/lib/php5/sess_mr9045p1k55vin1
[12:18:46.800846035,12:18:46.800874916] open                  28.881        N/A  apache2       31591  /var/lib/php5/sess_r7c12pccfvjtas1
[12:18:51.389797604,12:18:51.389824426] open                  26.822        N/A  apache2       31520  /var/lib/php5/sess_4sdb1rtjkhb78sa

Top read syscall latencies (usec)
Begin              End                Name           Duration (usec)       Size  Proc          PID    Filename
[12:18:37.256073107,12:18:37.256555967] read                 482.860      7.00 B  bash          10237  unknown (origin not found) (fd=3)
[12:18:52.000209798,12:18:52.000252304] read                  42.506     1.00 KB  irqbalance    1337   /proc/interrupts (fd=3)
[12:18:37.256559439,12:18:37.256601615] read                  42.176      5.00 B  bash          10237  unknown (origin not found) (fd=3)
[12:18:42.000281918,12:18:42.000320016] read                  38.098     1.00 KB  irqbalance    1337   /proc/interrupts (fd=3)

Top write syscall latencies (usec)
Begin              End                Name           Duration (usec)       Size  Proc          PID    Filename
[12:18:49.913241516,12:18:49.915908862] write               2667.346     95.00 B  apache2       31584  /var/log/apache2/access.log (fd=8)
[12:18:37.472823631,12:18:37.472859836] writev                36.205     21.97 KB  apache2       31544  unknown (origin not found) (fd=12)
[12:18:37.991578372,12:18:37.991612724] writev                34.352     21.97 KB  apache2       31589  unknown (origin not found) (fd=12)
[12:18:39.547778549,12:18:39.547812515] writev                33.966     21.97 KB  apache2       31584  unknown (origin not found) (fd=12)

Top sync syscall latencies (usec)
Begin              End                Name           Duration (usec)       Size  Proc          PID    Filename
[12:18:50.162776739,12:18:51.157522361] sync              994745.622        N/A  sync          22791  None (fd=None)
[12:18:37.227867532,12:18:37.232289687] sync_file_range     4422.155        N/A  lttng-consumerd 19964  /home/julien/lttng-traces/analysis
[12:18:37.238076585,12:18:37.239012027] sync_file_range      935.442        N/A  lttng-consumerd 19964  /home/julien/lttng-traces/analysis
[12:18:37.220974711,12:18:37.221647124] sync_file_range      672.413        N/A  lttng-consumerd 19964  /home/julien/lttng-traces/analysis
```

Live Mode

- Tracing sessions of arbitrary duration and size (same as streaming mode),
- Can attach to a running session and start processing the events while the session is still running,
- The trace is still written to disk but we can limit its size with the `tracefile-size` and `tracefile-count` options (on-disk ring buffer),
- Use-cases:
  - Low throughput logging with quick feedback,
  - Distributed or embedded systems,
  - Continuous monitoring (extracting metrics from events out-of-bound).

Live Mode

```
$ lttng create --live # optional: -U net://<server>
$ lttng enable-event -k -a
$ lttng enable-event -u -a
$ lttng start
$ lttng view
$ lttng stop
$ lttng destroy
```

EfficiOS

Live Mode - Bounded Disk Usage

```
$ lttng create --live # optional: -U net://<server>
$ lttng enable-channel -k chan --tracefile-size 10M --tracefile-count 4
$ lttng enable-event -k -a -c chan
$ lttng start
$ lttng view
$ lttng stop
$ lttng destroy
```

Snapshot Mode

- Memory-only tracing (ring-buffer),
- Low overhead while tracing (no I/O),
- On demand, "`lttng snapshot record`" extracts tracing buffers content from memory to disk or the network,
- Triggers to extract the snapshots can be errors detected by an application, high latencies measured, segmentation faults, time-based sampling, etc,
- The time span covered by a snapshot depends on the buffer size configuration, number of events enabled and the event rate.

Snapshot Mode

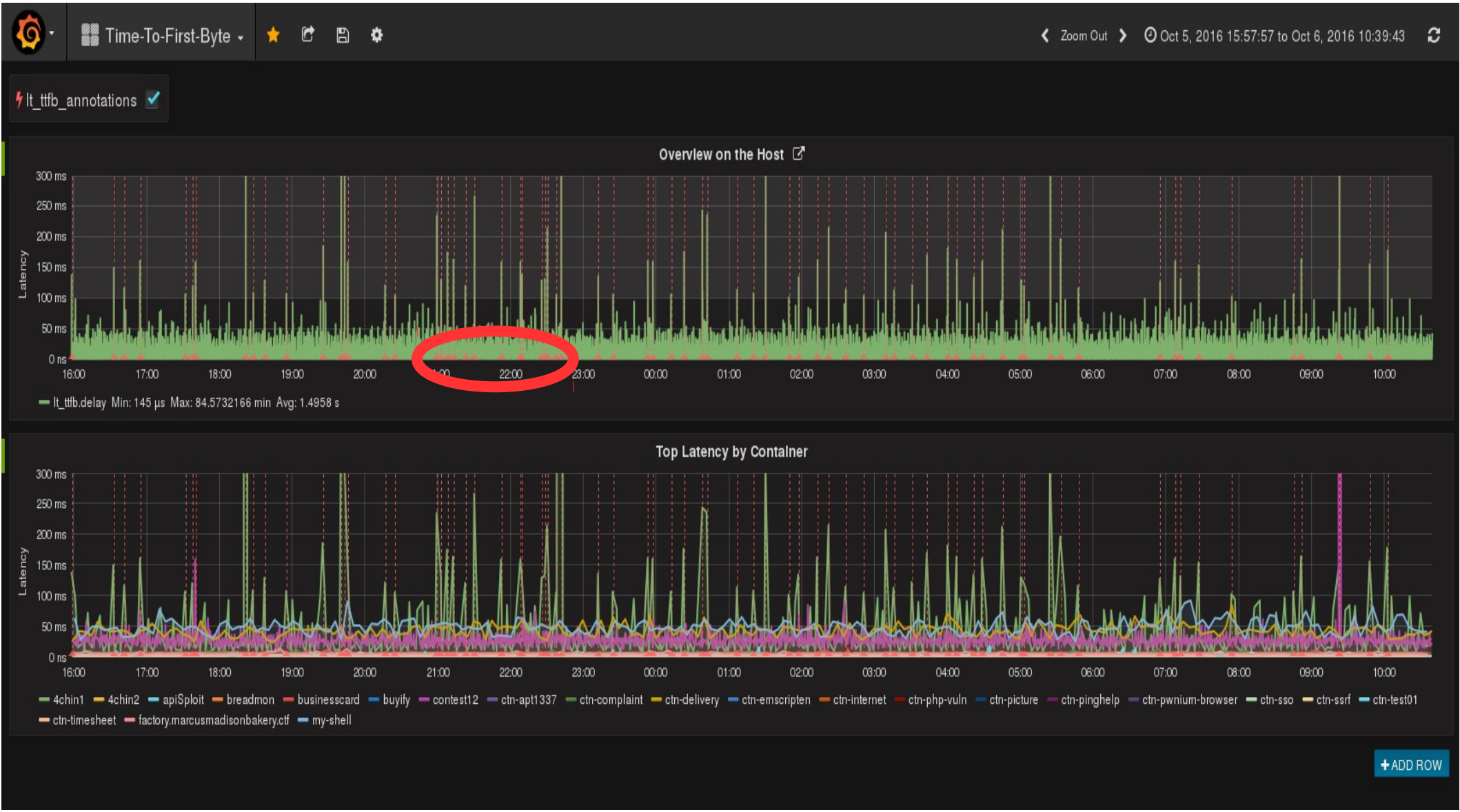
- Use-cases:
  - Fault investigation: get the full activity a few seconds before an error or high latency occured,
  - Profiling: get a sense of the machine activity periodically,
  - When a Continuous Integration worker detects an error.

Snapshot Mode

```
$ lttng create --snapshot # optional: -U net://<server>
$ lttng enable-event -k -a
$ lttng enable-event -u -a
$ lttng start
...
$ lttng snapshot record
...
$ lttng snapshot record
...
$ lttng snapshot record
```

Snapshot Mode - Example

- We sometimes measure high response times with an aggregation tool (latency-tracker),
- We want to know what is happening around the time the latencies are detected,
- Methodology:
  - Start a snapshot session with scheduling, I/O, and system calls events,
  - Every time a high latency is detected, record a snapshot,
  - Send the snapshot to an automated post-processing tool that generates activity reports,
  - Plot all the response times in Grafana and link the spikes to the snapshot analyses.


Time-To-First-Byte
Zoom Out
Oct 5, 2016 15:57:57 to Oct 6, 2016 10:39:43
lt_ttfb_annotations
Overview on the Host
Latency
300 ms
250 ms
200 ms
150 ms
100 ms
50 ms
0 ns
16:00
17:00
18:00
19:00
20:00
22:00
23:00
00:00
01:00
02:00
03:00
04:00
05:00
06:00
07:00
08:00
09:00
10:00
lt_ttfb.delay Min: 145 µs Max: 84.5732166 min Avg: 1.4958 s
Top Latency by Container
Latency
300 ms
250 ms
200 ms
150 ms
100 ms
50 ms
0 ns
16:00
17:00
18:00
19:00
20:00
21:00
22:00
23:00
00:00
01:00
02:00
03:00
04:00
05:00
06:00
07:00
08:00
09:00
10:00
4chin1
4chin2
apiSploit
breadmon
businesscard
buyify
contest12
ctn-apt1337
ctn-complaint
ctn-delivery
ctn-emscripten
ctn-internet
ctn-php-vuln
ctn-picture
ctn-pinghelp
ctn-pwnium-browser
ctn-sso
ctn-ssrf
ctn-test01
ctn-timesheet
factory.marcusmadisonbakery.ctf
my-shell
ADD ROW

I/O request latency distribution

```
lttng-iolatencyfreq /path/to/trace
```

```
Timerange: [2015-01-06 10:58:26.140545481, 2015-01-06 10:58:27.229358936]
Open latency distribution (usec)
################################################################################
 5.562 ████████████████████████████████████████████████████████████████████ 25
 9.168 ████████                                                              4
12.774 ████████████████                                                      8
16.380 ███████                                                               3
19.986 ████                                                                  2
23.592                                                                       0
27.198                                                                       0
30.804                                                                       0
34.410 ▌                                                                     1
38.016                                                                       0
41.623                                                                       0
45.229                                                                       0
48.835                                                                       0
52.441                                                                       0
56.047                                                                       0
59.653                                                                       0
63.259                                                                       0
66.865                                                                       0
70.471                                                                       0
74.077 ████                                                                  2
```

Rotation Mode

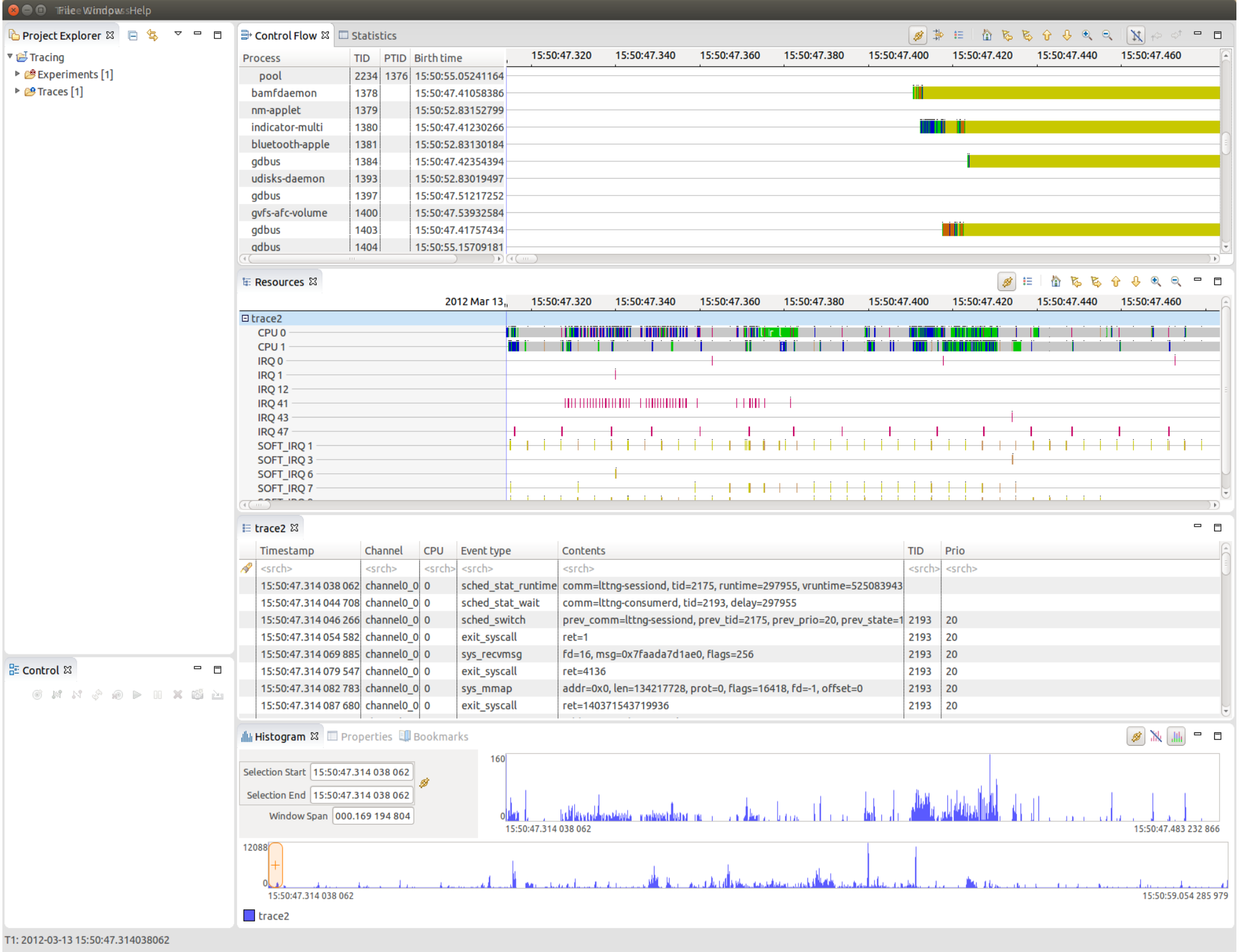
- New in LTTng 2.11 (expected to be released in March 2018),
- Archive a tracing session’s current chunk,
- Allows to process/archive/delete/compress a chunk of a trace while it is still writing in a separate directory,
- The trace can run indefinitely but the chunks can be processed like offline traces (disk or streaming mode),
- Timer-based or size-based auto-rotation available.

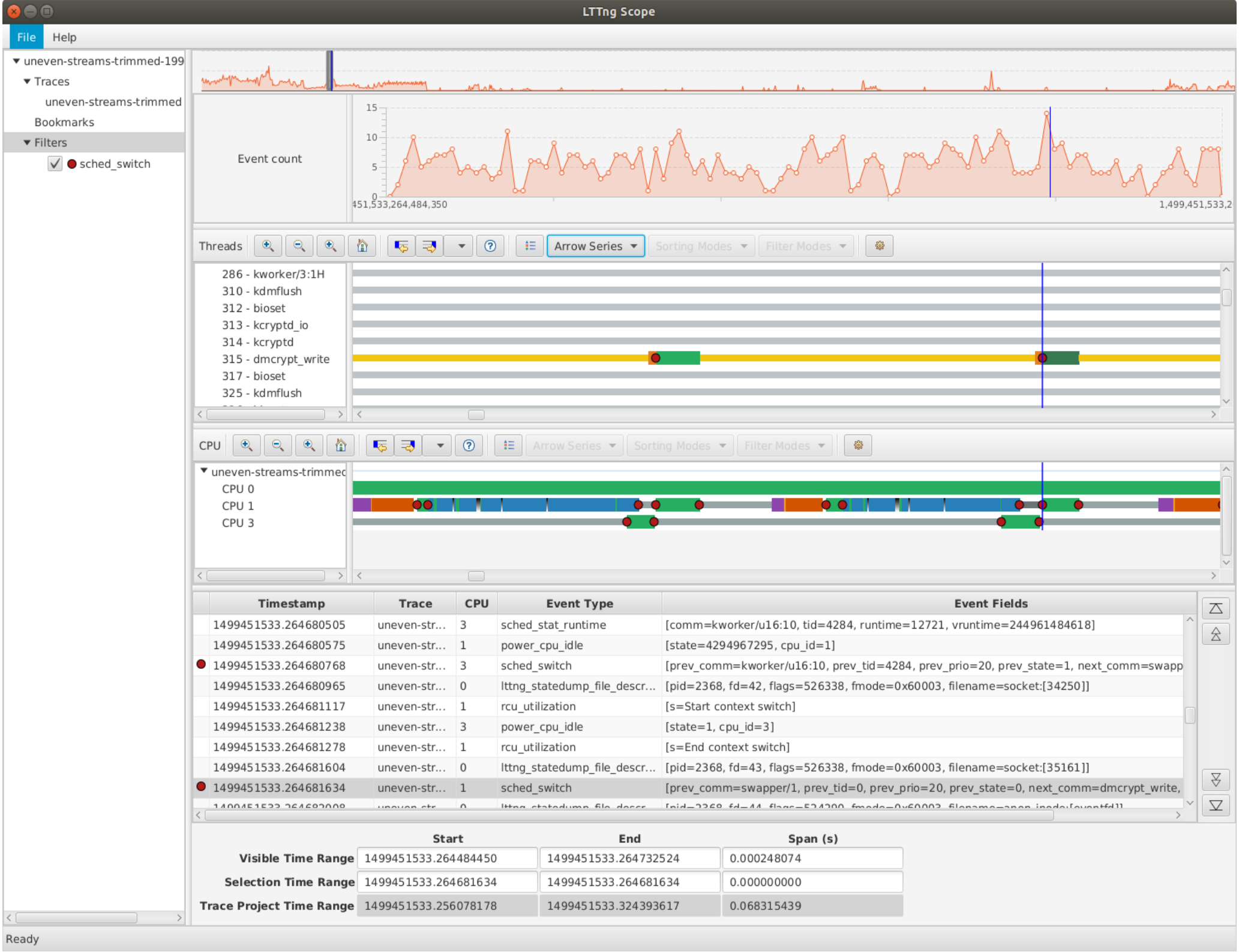
Rotation Mode

- Use-cases:
  - Continuous monitoring: periodically rotate and extract/plot low-level metrics from the trace,
  - Smaller traces to process than with the default mode,
  - Spreading the post-processing load (send chunks for analysis to available worker servers),
  - Archiving/Compression.

Rotation Mode

```
$ lttng create # optional: -U net://<server>

$ lttng enable-event -k -a

$ lttng enable-event -u -a

$ lttng start

...

$ lttng rotate

Output files of session auto-20180125-155317 rotated to
/home/julien/lttng-traces/auto-20180125-
155317/20180125T155319-0500-20180125T155320-0500-1

$ lttng rotate

...

$ lttng rotate
```


File Windows Help
Project Explorer
Tracing
Experiments [1]
Traces [1]
Control
Control Flow
Statistics
Process
TID
PTID
Birth time
pool 2234 1376 15:50:55.05241164
bamfdaemon 1378 15:50:47.41058386
nm-applet 1379 15:50:52.83152799
indicator-multi 1380 15:50:47.41230266
bluetooth-apple 1381 15:50:52.83130184
gdbus 1384 15:50:47.42354394
udisks-daemon 1393 15:50:52.83019497
gdbus 1397 15:50:47.51217252
gvfs-afc-volume 1400 15:50:47.53932584
gdbus 1403 15:50:47.41757434
gdbus 1404 15:50:55.15709181
15:50:47.320
15:50:47.340
15:50:47.360
15:50:47.380
15:50:47.400
15:50:47.420
15:50:47.440
15:50:47.460
Resources
2012 Mar 13
trace2
CPU 0
CPU 1
IRQ 0
IRQ 1
IRQ 12
IRQ 41
IRQ 43
IRQ 47
SOFT_IRQ 1
SOFT_IRQ 3
SOFT_IRQ 6
SOFT_IRQ 7
trace2
Timestamp
Channel
CPU
Event type
Contents
TID
Prio
15:50:47.314 038 062 channel0_0 0 sched_stat_runtime comm=lttng-sessiond, tid=2175, runtime=297955, vruntime=525083943
15:50:47.314 044 708 channel0_0 0 sched_stat_wait comm=lttng-consumerd, tid=2193, delay=297955
15:50:47.314 046 266 channel0_0 0 sched_switch prev_comm=lttng-sessiond, prev_tid=2175, prev_prio=20, prev_state=1 2193 20
15:50:47.314 054 582 channel0_0 0 exit_syscall ret=1 2193 20
15:50:47.314 069 885 channel0_0 0 sys_recvmsg fd=16, msg=0x7faada7d1ae0, flags=256 2193 20
15:50:47.314 079 547 channel0_0 0 exit_syscall ret=4136 2193 20
15:50:47.314 082 783 channel0_0 0 sys_mmap addr=0x0, len=134217728, prot=0, flags=16418, fd=-1, offset=0 2193 20
15:50:47.314 087 680 channel0_0 0 exit_syscall ret=140371543719936
Histogram
Properties
Bookmarks
Selection Start 15:50:47.314 038 062
Selection End 15:50:47.314 038 062
Window Span 000.169 194 804
160
0
15:50:47.314 038 062
15:50:47.483 232 866
12088
0
15:50:47.314 038 062
15:50:59.054 285 979
trace2
T1: 2012-03-13 15:50:47.314038062


LTTng Scope
File
Help
uneven-streams-trimmed-199
Traces
uneven-streams-trimmed
Bookmarks
Filters
sched_switch
Event count
15
10
5
0
451,533,264,484,350
1,499,451,533,2
Threads
Arrow Series
Sorting Modes
Filter Modes
286 - kworker/3:1H
310 - kdmflush
312 - bioset
313 - kcryptd_io
314 - kcryptd
315 - dmcrypt_write
317 - bioset
325 - kdmflush
CPU
uneven-streams-trimmed
CPU 0
CPU 1
CPU 3
Timestamp
Trace
CPU
Event Type
Event Fields
1499451533.264680505 uneven-str... 3 sched_stat_runtime [comm=kworker/u16:10, tid=4284, runtime=12721, vruntime=244961484618]
1499451533.264680575 uneven-str... 1 power_cpu_idle [state=4294967295, cpu_id=1]
1499451533.264680768 uneven-str... 3 sched_switch [prev_comm=kworker/u16:10, prev_tid=4284, prev_prio=20, prev_state=1, next_comm=swapp
1499451533.264680965 uneven-str... 0 lttng_statedump_file_descr... [pid=2368, fd=42, flags=526338, fmode=0x60003, filename=socket:[34250]]
1499451533.264681117 uneven-str... 1 rcu_utilization [s=Start context switch]
1499451533.264681238 uneven-str... 3 power_cpu_idle [state=1, cpu_id=3]
1499451533.264681278 uneven-str... 1 rcu_utilization [s=End context switch]
1499451533.264681604 uneven-str... 0 lttng_statedump_file_descr... [pid=2368, fd=43, flags=526338, fmode=0x60003, filename=socket:[35161]]
1499451533.264681634 uneven-str... 1 sched_switch [prev_comm=swapper/1, prev_tid=0, prev_prio=20, prev_state=0, next_comm=dmcrypt_write,
Start
End
Span (s)
Visible Time Range 1499451533.264484450 1499451533.264732524 0.000248074
Selection Time Range 1499451533.264681634 1499451533.264681634 0.000000000
Trace Project Time Range 1499451533.256078178 1499451533.324393617 0.068315439
Ready

Conclusion

- LTTng allows to extract low-level, high volume tracing information in production environments,
- Efficient kernel and user-space combined tracing,
- Used for monitoring and fault investigation in at least cloud, telecommunication and automotive environments,
- There are five main ways to extract LTTng traces, flexibility based on the use-case,
- Not just a tracer to use when all else has failed.

Questions ?

EfficiOS

www.efficios.com

lttng.org

lttng-dev@lists.lttng.org

@lttng_project

OFTC / #lttng