Virtual and real machines simultaneous tracing and monitoring with LTTngTop

Julien Desfossez
Michel Dagenais

Dec 6, 2012
École Polytechnique, Montreal

Content

- Live streaming
- Live analysis
- LTTngTop overview
- Early results
- Conclusion and Future work
- References

Live Streaming

- Implemented in lttng-tools 2.1
- Allow to send the trace data over the network
- No local storage required
- 2 ports : control and data
- IPv4/IPv6
- TCP-only for now

Live Streaming usage

- Start a lttng-relay on the receiving host :

```
$ lttng-relayd -d
```

- Create a LTTng session with the streaming parameter :

```
$ lttng create -U net://receiving-host
$ lttng enable-event -a --syscall -k
$ lttng start
$ lttng stop
$ lttng destroy
```

Live Analysis

- Viewing/analysing a trace while it is active
- Requirements/use-cases :
  - Language/platform independant (C/C++, Java, Unix, Windows, etc)
  - Local or remote sessions
  - Read trace data from disk (tracefiles) or directly from the memory of the tracer
  - Attach/detach to the sessions
  - Multiple viewers on the same session
  - Efficient synchronisation algorithm

Live Analysis Status

- Work in progress
- Index generation complete for the kernel tracer
- First iteration : most compatible approach with a TCP port
- Next step : give access to mmap regions to the viewers (need a consumer library and might put restrictions on the platform/language, but more efficient for the best scenario)

Live Analysis Challenges

- Metadata streaming before the data
- Synchronisation of streams :
  - Each stream is independant (no synchronisation at the tracer level)
  - We process traces sorted by time
  - We need to have data for each stream to be sure we won't receive data “from the past”
  - Inactive/slow channels won't produce data as often as active/high troughput channels
  - We don't want to force the flushing of inactive streams for energy efficiency
  - We don't want to have a “grace period” on the viewer side

LTTngTop Overview

- Sysadmin-oriented kernel trace viewer
- Gather statistics only from kernel events (no constant /proc hammering)
- Lightweight and console-based (ncurses)
- top-like look and feel
- CPU usage, I/O statistics, Perf PMU counters evolution per-process

LTTngTop Overview

- Replay the recorded trace at the same rate it happened
- Pause and navigate back and forth in the history
- Display the state of the system at any point in time (CPU usage, opened files, bandwidth, process creation/termination, etc)

LTTngTop use-cases

- Hard to reproduce bugs (happens sometimes and disappears by itself)
- Detailed statistics per-process at any point in time (including opened files)
- Quickly isolate interesting events from a given trace and then read its text dump?

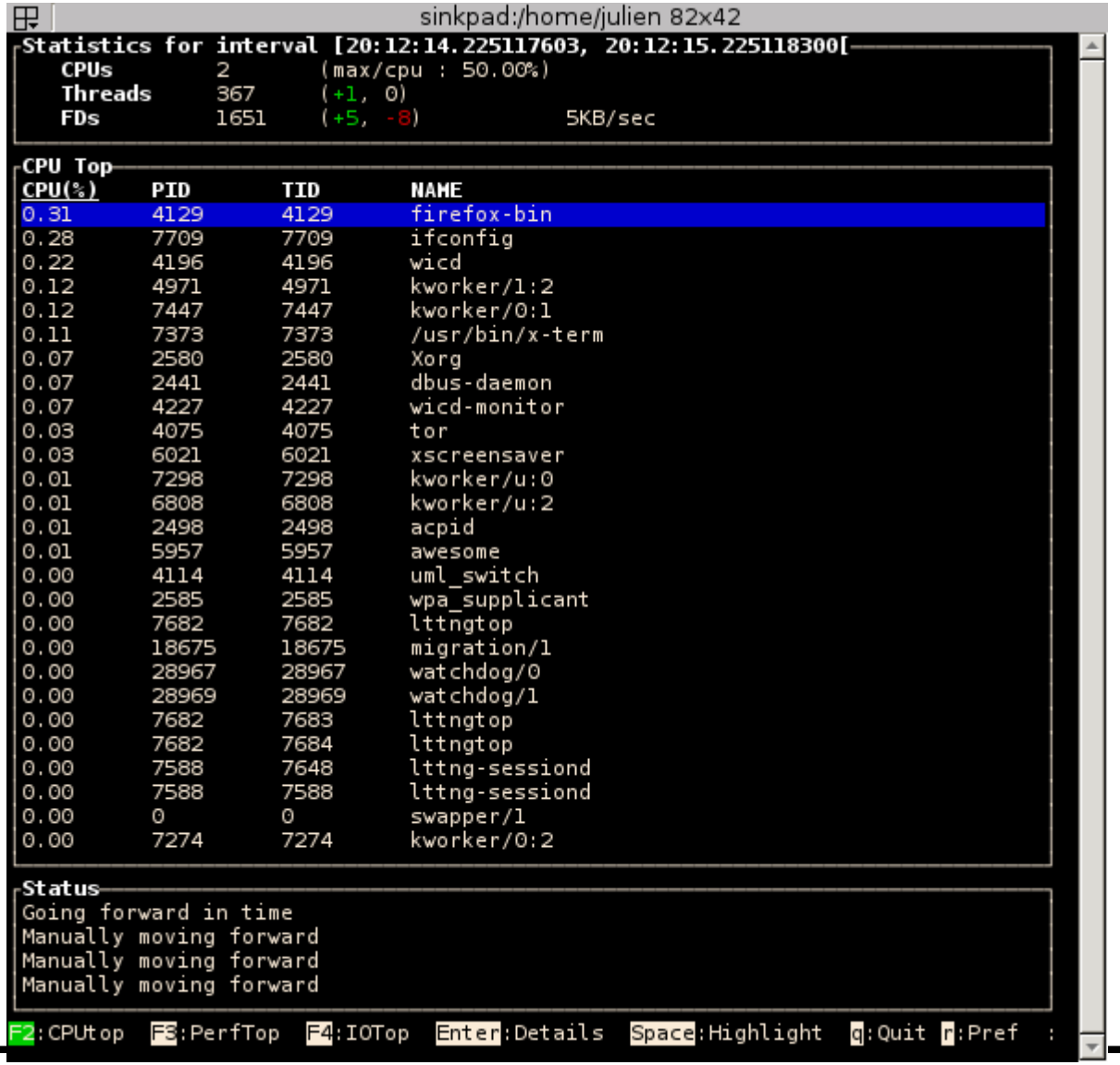
LTTngTop Work in Progress

- Live “in-memory” tracing : proof-of-concept of accessing the trace buffer from a viewer application
- Containers support (LXC) :
  - vpid/vtid/vppid
  - nesting and hierarchy support
- Viewer-side filtering :
  - selected processes (with/without childs)
  - selected container (with/without nesting)

LTTngTop Work in Progress

- “Attach” to a live process (graphical and textdump like `strace`, still faster than `ptrace` even though we trace the whole system)
- CPU hotplug support
- kprobe support (hit stats)

LTTngTop Demo



LTTngTop Future

- Large-scale data center use-cases :
  - Remote traces live analysis (monitoring, trending and debugging)
  - Distributed analysis computation
  - High-level to fine-grained view

Questions ?

Branches used in this demo :

```
$ git clone -b lttngtop-live \
git://git.dorsal.polymtl.ca/~jdesfossez/lttng-tools
```

```
$ git clone -b lttngtop-live \
git://git.dorsal.polymtl.ca/~jdesfossez/babeltrace
```

```
$ git clone -b live git://git.lttng.org/lttngtop.git
```