Cloud Monitoring and Distribution
Bug Reporting with Live Streaming
and Snapshots

mathieu.desnoyers@efficios.com
Mathieu Desnoyers

EfficiOS

- http://www.efficios.com

Author/Maintainer of

- Userspace RCU,
- LTTng kernel and user-space tracers,
- Babeltrace.
New and upcoming features since Tracing Summit 2012

- LTTng
- Babeltrace

Cloud monitoring

Enhanced bug reports

LTTng project roadmap
• LTTng 2.x does

\textbf{NOT}

require any kernel modification.
LTTng Reminder

- LTTng 2.x does

\textbf{NOT}

require any kernel modification.
LTTng Reminder

- LTTng 2.x does not require any kernel modification.
LTtng Reminder

• LTtng 2.x does

\textbf{NOT}

require any kernel modification.
LTTng Features Since Tracing Summit 2012

- LTTng 2.1 Basse Messe (December 2012)
  - Network Streaming (TCP)
  - Session daemon health check
  - Event field filtering (LTTng-UST)
  - ARM, MIPS system call tracing (LTTng modules)
LTTng Features Since Tracing Summit 2012

- LTTng 2.2 Cuda (June 2013)
  - Per user ID buffers (LTTng-UST)
  - On disk file rotation (maximum stream file size)
LTTng Features Since Tracing Summit 2012

- LTTng 2.3 Dominus Vobiscum (September 2013)
  - Flight recorder tracing
    - Stop and non-stop snapshots
    - Core dump handler integration
      - LTTng Tools extras/
Flight recorder session + snapshot

$ lttng create --snapshot
$ lttng enable-event -k sched_switch
$ lttng enable-event -k --syscall -a
$ lttng start
$ ...
$ lttng snapshot record

Snapshot recorded successfully for session auto-20131019-113803

$ babeltrace /home/julien/lttng-traces/auto-20131019-113803/snapshot-1-20131019-113813-0/kernel/
At any point in time, a snapshot can be taken of the current trace buffers.

Overwrite mode meaning flight recorder

\texttt{lttng\_snapshot\_record(..)}
LT Tng Features Since Tracing Summit 2012

- LT Tng 2.4 Époque Opaque (upcoming)
  - Java Util Logging (JUL) tracing
  - Live streaming
    - Analysis of live traces
  - Consumer and relay daemon health check
  - Packet index generated by consumer daemon
    - Faster load of large traces in viewers afterward
Live Network Streaming Deployment

Server A (lttng-sessiond) → Server B (lttng-sessiond) → Server C (lttng-sessiond)

lttng-relayd

Viewer

TCP

TCP
Live Network Streaming Session

On the server to trace:
$ lttng create --live 2000000 -U net://10.0.0.1
$ lttng enable-event -k sched_switch
$ lttng enable-event -k --syscall -a
$ lttng start

On the receiving server (10.0.0.1):
$ lttng-relayd -d

On the viewer machine:
$ lttngtop -r 10.0.0.1
As the trace is being **created**, you **extract** and can **analyze** the data.

**Continuous Analysis**
- Extract data with live streaming for analysis on another machine

**Cluster-level analysis**
- Gather traces from multiple machines
  - Load balancing analysis
  - Latency detection

**System Administration**
- Get data of faulty machine “on-demand”
Performance Results

Number of database requests vs Number of threads

Dedicated disk for the DB

requests/sec vs number of threads

- No tracing
- Flight recorder
- Streaming
- Tracing to disk
- strace mysql
Babeltrace Features Since Tracing Summit 2012

- Babeltrace 1.0 (initial release, October 2012)
- Babeltrace 1.1 (API namespacing fix, March 2013)
- Babeltrace 1.2 (upcoming)
  - Common Trace Format (CTF) Writer API
  - Python bindings
  - Nexus to CTF converter
  - Live trace stream read support
    - Connect to LTTng relay daemon
Cloud Monitoring

- Live network streaming
- Flight recorder tracing and snapshots
- Bytecode interpreter
  - On traced target or separate dedicated machine,
  - Triggers:
    - Start tracing
    - Stop tracing
    - Gather snapshot
  - Aggregation
Enhanced Bug Reports

- Flight recorder tracing
- In production
- Extremely low overhead
- When error is encountered
  - Gather snapshot
  - “Do you want to send a detailed bug report?"
  - Very detailed trace of trace leading to the problem sent along with bug report
LTTng Project Roadmap

- Save/restore trace session configuration to/from files
- Support Perf PMU counters from LTTng-UST
- Dynamic instrumentation of user-space (dyninst)
- Listing libraries shared objects (LD_PRELOAD)
- Hardware tracing (ARM, Freescale, Intel, ...)
- Triggers and aggregation in LTTng-UST bytecode interpreter
- LTTng modules bytecode interpreter
- Android port for LTTng modules and UST