Linux Plumbers Conference 2011

LTTng 2.0 : Application, Library and Kernel tracing within your Linux distribution.

E-mail:
mathieu.desnoyers@efficios.com

> Presenter

- Mathieu Desnoyers
- EfficiOS Inc.
  - http://www.efficios.com
- Author/Maintainer of
  - LTTng, LTTng-UST, Babeltrace, LTTV, Userspace RCU

> LTTng 2.0 Toolchain Overview

- LTTng 2.0 kernel tracer
- LTTng-UST 2.0 user-space tracer
- LTTng tracing session daemon
- LTTng consumers
- “lttng” CLI / liblttngctl
- Babeltrace
- LTTng-top
- Common Trace Format (CTF)

> LTTng 2.0 Kernel Tracer

- Build against a vanilla or distribution kernel, without need for additional patches,
- Tracepoints, Function tracer, Perf CPU Performance Monitoring Unit (PMU) counters, kprobes, and kretprobes support,
- Supports multiple tracing sessions, flight recorder mode, snapshots, ...

> LTTng 2.0 Kernel Tracer

- ABI based on ioctl() returning anonymous file descriptors
  - implemented a top-level DebugFS “lttng” file.
- Lib Ring Buffer, initially developed generically for mainline Linux kernel (as a cleanup of the LTTng 0.x ring buffer) has been merged into LTTng 2.0.
- Exports trace data through the Common Trace Format (CTF).

> LTTng 2.0 Kernel Tracer

- Supports dynamically selectable “context” information to augment event payload
  - Any Perf PMU counter
  - PID, PPID, TID, executable name (comm), VPID, VTID, …
  - Dynamic Priority, nice value

> LTTng-UST 2.0 User-space Tracer

- TRACEPOINT_EVENT() API for application/library static instrumentation.
- libust linked with applications, listening for LTTng session daemon commands.
- Supports per-user and system-wide tracing.
- “tracing” group: no need to be root to perform system-wide tracing.

> TRACEPOINT_EVENT

In header:

```
TRACEPOINT_EVENT(ust_tests_hello_tptest,
        TP_PROTO(int anint, long *values,
                 char *text, size_t textlen,
                 double doublearg, float floatarg),
        TP_ARGS(anint, values, text, textlen,
                doublearg, floatarg),
        TP_FIELDS(
                ctf_integer(int, intfield, anint)
                ctf_integer_hex(int, intfield2, anint)
                ctf_array(long, arrfield1, values, 3)
                ctf_sequence(char, seqfield1, text,
                             size_t, textlen)
                ctf_string(stringfield, text)
                ctf_float(float, floatfield, floatarg)
                ctf_float(double, doublefield, doublearg)
        )
)
```

Tracepoint name convention

> User-level Tracepoint

Name convention

< [com_company_]project_[component_]event >

Where "company" is the name of the company,
"project" is the name of the project,
"component" is the name of the project component (which may include several levels of sub-components, e.g. ...component_subcomponent_...) where the tracepoint is located (optional),
"event" is the name of the tracepoint event.

Tracepoint invocation within the code:

```
void fct(void)
{
    tracepoint(ust_tests_hello_tptest, i, values,
                text, strlen(text), dbl, flt);
}
```

> Extended Tracepoint Declaration

- API planned, feature not implemented yet.
- TRACEPOINT_LOGLEVEL_ENUM()
  - Loglevels defined by the application implementor.
- Following a TRACEPOINT_EVENT():
  - TRACEPOINT_LOGLEVEL(name, level)
    - Optional selection of tracepoint activation on a per-loglevel basis.
  - TRACEPOINT_FORMAT(name, “format”)
    - Optional pretty-printing.

> tracepoint_printf()

- Feature planned
- tracepoint_printf(name, “fmt”, …);
- Augment Common Trace Format to store format strings
- Export only binary data through buffers.
- Pretty-printing performed at post-processing.

> LTTng-UST 2.0 Buffering

- Port of the lib ring buffer to user-space.
- Supports buffering between processes through POSIX shared memory maps.
- Wake-up though pipes.
- Buffers per process (for security), shared with consumer. Faster/lower memory consumption insecure global buffers feature planned too.
- Takes care of security concerns involved with sharing data structures between processes.

> LTTng Tracing Session Daemon

- Both centralized (system-wide) and per-user.
- Controls
  - LTTng kernel tracer (domain)
  - LTTng-UST application/library tracer (domain)
  - Right management by UNIX socket file access rights (tracing group).
  - File descriptor credentials passed through UNIX sockets
- Presents a unified notion of system-wide tracing session, with multiple “domains”.

> LTTng Consumers

- Spawned by the tracing sessions daemon
- Design guide-lines:
  - Minimal access, aiming at a design where sessiond opens all files, consumers just copy data between memory maps and file descriptors (received though UNIX socket credentials).
- Disk output (splice, mmap).
- In-place mmap buffer consumption (lttngtop).
- Planned network transport.

> LTTng CLI / liblttngctl

- Unified control interface for kernel and user-space tracing
  - “lttng” git-alike command line interface
  - All tracing control commands available through an API: liblttngctl and lttng.h

> LTTng UI examples

```
lttng list -k                          # list available kernel tracpoints
lttng create mysession                 # create session “mysession”
lttng enable-event -k -a               # enable all available tracepoints
lttng enable-event sched_switch,sys_enter -k
lttng enable-event aname -k --probe symbol+0xffff7260695
lttng enable-event aname -k --function <symbol_name>
lttng add-context -k -e sched_switch -t pid     # add PID context
lttng add-context -k -e sched_switch -t perf:cpu-cycles
lttng start                            # start tracing
...
lttng stop                             # stop tracing
lttng destroy                          # teardown session

# text output
babeltrace -n $HOME/lttng-traces/mysession-<date>-<time>
```

> LTTng 2.0 kernel tracer demo

> Common Trace Format

- Trace format specification
  - Funded by
    - Linux Foundation CE Linux Forum and Ericsson
  - In collaboration with Multi-Core Association Tool Infrastructure Workgroup
    - Freescale, Mentor Graphics, IBM, IMEC, National Instruments, Nokia Siemens Networks, Samsung, Texas Instruments, Tilera, Wind River, University of Houston, Polytechnique Montréal, University of Utah.
  - Gathered feedback from Linux kernel developers and SystemTAP communities.

> Common Trace Format

- Targets system-wide and multi-system trace representation in a common format, for integrated analysis:
  - Software traces
    - Across multiple CPUs
    - Across the software stack (Hypervisor, kernel, library, applications)
  - Hardware traces
    - DSPs, device-specific tracing components.
    - GPUs.

> Common Trace Format

- Babeltrace
  - Reference implementation trace conversion tool and read/seek API for trace collections.
  - Initially converts
    - From CTF to text
    - From dmesg text log to CTF
- LTTng kernel 2.0 and LTTng-UST 2.0
  - Native CTF producer reference implementation.
- Available at: http://www.efficios.com/ctf

> Distributions

- Distributions shipping LTTng 0.x
  - Wind River Linux, Montavista, STlinux, Linaro, Yocto, Mentor Embedded Linux, ELinOS, Novell SuSE Enterprise RT Linux.
- Packages
  - Debian and Ubuntu
    - UST, Userspace RCU, LTTV
- Working closely with Ubuntu and Debian to have LTTng 2.0 toolchain ready for the next Ubuntu LTS.

> Distributions

- Fedora
  - Fedora packages available for LTTng 0.x user-space tracing and trace analysis, LTTng 2.0 packages planned,
  - ***Actively looking for a sponsor.***
- RHEL 6
  - Interested in discussing backport of Steven's Tracepoint patches from 2.6.35:
    - “tracing: Let tracepoints have data passed to tracepoint callbacks”

> Questions ?

LTTng 2.0 prereleases available at
http://lttng.org/lttng2.0

EfficiOS

- http://www.efficios.com

- LTTng Information
  - http://lttng.org
  - ltt-dev@lists.casi.polymtl.ca