LTTng: Kernel and userspace tracing in Debian

mjeanson@efficios.com
Michael Jeanson, Software developer @ EfficiOS
- Debian Maintainer
- Ubuntu Member
- Fedora Packager
- Official and unofficial packager for other distros
• What is tracing?
• A description of the different tools involved
• LTTng compared to other tracing tools like strace
• The state of LTTng in Debian
• Basic use cases and workflows
• Analysis of kernel traces
What is tracing?

• Like a black box / flight recorder for your system
• Record runtime information
  - Syscalls
  - Function entry/exit
• Enable/Disable event(s) at runtime
• Low overhead
Why use tracing?

• Problems that are not easily diagnosed with traditional tools or debugging
  – Narrow down bug causes
  – Identify performance hogs
• Very low performance impact
  – Can be used on production systems
Tools

- Tracers
- Control utilities
- Viewers
- Post-processing / analysis
Tracers

- **lttng-modules**: OOT kernel tracer modules
  - compatible with kernels 2.6.38 to latest rc
  - you don’t need to recompile your kernel
  - lttng-modules-dkms in Debian

- **lttng-ust**: user-space tracer, in-process library
  - Java JUL and log4j agent
  - Python logging agent
Control utilities

- lttng-tools: cli utilities and daemons for trace control
  - lttng: main cli command
  - lttng-ctl: tracing control library
  - lttng-sessiond: tracing registry daemon
  - lttng-consumerd: extract trace data
  - lttng-relayd: network streaming daemon
Viewers

• babeltrace: cli text viewer and trace converter
• tracecompass:
  – GUI front-end for lttng
  – Collect, visualize and analyze traces
  – Eclipse plugin or standalone version
• lttngtop: ncurse top-like viewer

EfficiOS
Post-processing / Analysis

- Lttng-analyses
  - Record your system's activity
  - Do whatever it takes for your problem to occur.
  - Diagnose your problem's cause offline (when tracing is stopped).
Compare

- strace: syscall and signal tracer
- ftrace: in-kernel function and event tracer
- perf: in-kernel profiler and tracer
LTTng in Debian

- All the tools are packaged
- 2 maintainers
- Testing/unstable: latest 2.8 stack
- Stable: 2.5 stack, unsupported :( 
- Stable-backports: 2.8 stack coming soon
- Oldstable: Ancient stuff, don’t use it
LTTng in Ubuntu

- Xenial: 2.7 stack, supported
- Trusty: 2.4 stack, unsupported :
- PPAs
  - Daily builds
  - Stable branch builds
  - Release builds for latest LTS
Use cases

- Debugging complex and hard to reproduce problems
- Embedded development with remote tracing
- Use snapshot mode for difficult to reproduce bugs
- Low-level metric collection with network streaming of traces
- Low-overhead top-like monitoring with lttngtop
Workflows

- Given a reproducible problem
- Gather trace (high level at first)
- Analyze (narrow down the problem source)
- Add instrumentation if needed
- Rince, repeat
LTTng analyses

- Demo!
  - lttng create
  - lttng enable-event -k -a
  - lttng start
  - ...wait for the problem to appear...
  - lttng stop
  - lttng destroy
Demo: IO usage

$ lttng-iourgetop demo-trace/
Demo: IO latency

$ lttng-iolatencystats demo-trace/ --minsize 2
Demo: IO latency


- Wouldn’t be a demo if everything worked
- From the previous step, we know when the latency happened, look at the log
- Find the root cause
Questions

LTTng Project

🌐 https://{git | www}.lttng.org
✉ lttng-dev@lists.lttng.org
🐦 @lttng_project
👥 #lttng on irc.oftc.net