EFFICIENT C/C++ TRACING
WITH ECLIPSE

DOMINIQUE <DOT> TOUPIN <AT> ERICSSON <DOT> COM
ABOUT ME

› Developer Tool Manager at Ericsson, helping Ericsson sites to develop better software efficiently

› Telecommunication systems
  – Open, standards-based common platform
  – High availability, 99.999 %
  – Broad range of support for both infrastructure and value-added applications
  – Multimedia, network and application processing capabilities
ABOUT ME

› GDB improvements, non-stop, multi-process, global breakpoint, dynamic tracepoint, core awareness, OS awareness, PTC set, …

› LTTng improvements, user space, common trace format, multicore association, CEWG, …

› Eclipse GDB and LTTng integration, analysis

› Linux tracing research project with Ecole Polytechnique

› Linux Tracing Summit Organization

› CDT Summit Organization
TRACING CONCEPTS

**Static Tracepoints** (e.g. linux kernel trace event, LTTng UST)
- Inserted at *compile* time, enable/disable at run-time
- *Low* overhead
- Represent the wisdom of developers who are most familiar with the code
- Rest of the world can use them to extract a great deal of useful information without having to know the code

**Dynamic Tracepoint** (e.g. Linux kernel kprobes, GDB tracepoint)
- Inserted at *run-time*, enable/disable at run-time
- *Big* overhead compared to static tracepoint
- Use it when a static tracepoint is missing
- If used a lot over a long period of time, think about converting them to static
TRACING USEFULNESS

- Very efficient compared to logging
- Performance tuning
- Diagnose multicore programming bug
- Monitoring a live system
LTTNG

› Linux Tracing Toolkit next generation (LTTng)

› Included in several Linux Distros, more than 90 contributors from 20 different organizations

› System-wide tracing across:
  - Kernel
  - Hypervisor
  - VM
  - Library
  - Application

› Precise and fast clock sources with near cycle-level timestamps, i.e. 1-100 ns
LTTNG DISTINCTIVE FEATURES

- Efficient trace data transport
  - Compact Binary format with Common Trace Format
  - Per-core buffers ensures scalability
  - Fast-paths in caller context, amortized synchronization
  - Zero-copy using splice and mmap system calls, over disk, network or consumed in-place

- Low-intrusiveness instrumentation and control
  - Per-process control thread with low footprint
  - Fast user-space process registration
  - Static Tracepoints enabled with static jump patching
LTTNG DISTINCTIVE FEATURES

› Multi-session support with per host or per user daemon
› Algorithms based on RCU verified by model checking
› Designed to meet real-time constraints
› Supports live streaming of the trace data
COMMON TRACE FORMAT

- Ericsson and Linux Foundation CE Linux Workgroup
- Reviewed by Linux kernel developers and SystemTAP communities
- 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, ...

- Requirement, specification, reference implementation http://www.efficios.com/ctf
COMMON TRACE FORMAT

- Self describing
- Very compact binary trace format
- System-wide and multi-system trace representation in a common format, for integrated analysis:
  - Software traces
    - Across multiple CPUs
    - Across the software stack, e.g. hypervisor, kernel, library, applications
  - Hardware traces
    - DSPs, device-specific tracing components.
    - GPUs.
ECLIPSE TRACING MONITORING FRAMEWORK (TMF)

- Eclipse Linux Tools Project
- Framework to build trace visualization and analysis tool
- Scalability allows to handle traces exceeding memory
- Enable trace analysis from different sources
- LTTng Eclipse integration is an implementation on top of TMF
Tracing projects integrate into Project Explorer view using Common Navigator Framework (CNF)

Tracing projects are used to group traces you are interested in

Traces folder contains the traces of interest

Experiments are the way to group and correlate traces
KERNEL TRACE CONTROL

- Configuration, control and collection of traces for analysis
- Streaming of traces
EVENTS VIEW

› ‘Raw’ merged events in chronological order

› Supports following functionalities
  – Searching
  – Filtering
  – Highlighting
HISTOGRAM VIEW

› Event distribution over full traceset and selected window

› Controls to modify current event and event window

› Zoom in/out using mouse scroll
CONTROL FLOW VIEW

› Displays processes state changes (color-coded) over time
› State 'tooltips'
› Zooming and filtering
› Quick navigation between processes, states
RESOURCE VIEW

› Displays system resource states (color-coded) over time
› State 'tooltips'
› Zooming and filtering
› Quick navigation between resources, states
STATISTICS VIEW

- Displays **basic trace statistics**
  - Per process, event type
  - Per CPU, mode

<table>
<thead>
<tr>
<th>Statistics View</th>
<th>Number of Events</th>
<th>CPU Time</th>
<th>Cumulative CPU Time</th>
<th>Elapsed Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Trace1-15316</td>
<td>15316</td>
<td>0.060617118</td>
<td>0.876353906</td>
<td>0.762141301</td>
</tr>
<tr>
<td>CPUs</td>
<td>15316</td>
<td>0.060617118</td>
<td>8.497766916</td>
<td>0.762141301</td>
</tr>
<tr>
<td>Event Types</td>
<td>99</td>
<td>0.003638321</td>
<td>0.008165658</td>
<td>0.004082829</td>
</tr>
<tr>
<td>Modes</td>
<td>12432</td>
<td>0.001467848</td>
<td>0.00309494</td>
<td>0.00154747</td>
</tr>
<tr>
<td>IRQ</td>
<td>89</td>
<td>0.030586202</td>
<td>1.595780335</td>
<td>0.748304729</td>
</tr>
<tr>
<td>MODEUNKNOWN</td>
<td>1890</td>
<td>0.006543252</td>
<td>0.031454274</td>
<td>0.008206273</td>
</tr>
<tr>
<td>SOFTIRQ</td>
<td>319</td>
<td>0.018381495</td>
<td>0.</td>
<td>0.</td>
</tr>
<tr>
<td>SYSCALL</td>
<td>487</td>
<td>0.</td>
<td>0.</td>
<td>0.</td>
</tr>
</tbody>
</table>
GDB TRACEPOINT INTEGRATION
TEXT PARSER WIZARD

› Supports text and XML traces

› Customizable by the user via a wizard

› Regular expressions used for parsing text traces

› Parsed traces integrate with all TMF views
SEQUENCE DIAGRAM FRAMEWORK

- Supports any trace type
- Provides all tooling for sequence diagram handling
- Extendable for user specific sequence diagrams
CURRENT DEVELOPMENT

› Support for LTTng 2.0
  – CTF-based Kernel and UST traces
  – Trace control
  – Session management
  – Support for multiple trace sessions
  – Streaming

› Juno
  – Project graduation work (Linux Tools 1.0)
  – Uplift to Eclipse 4.X

› More analysis tools
  – Latency analysis, clock adjustment, trace comparison, etc.
SOME REFERENCES

› Linux Tools
  - Update Site: http://download.eclipse.org/technology/linuxtools/update

LTTng (Eclipse)
  - LTTng Eclipse Project: http://www.eclipse.org/linuxtools/projectPages/Lttng
  - LTTng Eclipse Wiki: http://wiki.eclipse.org/Linux_Tools_Project/LTTng

› LTTng project: http://lttng.org

› For more info, questions, discussions:
  - linuxtools-dev@eclipse.org