Real time debugging: using non-intrusive tracepoints to debug live systems

Marc Khouzam, Ericsson Canada

marc.khouzam@ericsson.com

Agenda

- DSF-GDB today
- Tracepoints
  - The need
  - The solution
    - Dynamic Tracepoints and Eclipse demo
    - Static tracepoints
    - Trace data visualization and Eclipse demo
- Questions

DSF-GDB today

- Optimized GDB integration
- Standard Debugging features
- Multi-thread/Multi-process
- Non-stop debugging

- Extendable Pretty-printing with GDB 7.0
- Linux, Windows, Mac
- Reverse Debugging
- Any-binary debugging (no need for a project)

DSF-GDB demo

- Any-binary debugging
  - Debugging GDB itself

TRACEPOINTS

The need

- Extremely low intrusiveness
  - For live sites
  - For race conditions
  - For Real-time
  - For UI bugs

The solution

- Highly efficient tracing tool using tracepoints
- Dynamic Tracepoints
  - Added dynamically while code is executing
- Static Tracepoints
  - Added in the source code, before compilation
- Disconnected tracing

The solution

- GDB (GNU Debugger)
  - Enhanced dynamic tracepoints
  - New control of static tracepoints
- LTTng and UST (new User Space Tracing)
  - Can be controlled at run-time by GDB
- Eclipse CDT
  - Extending the existing DSF-GDB integration

GDB's New Tracepoint Feature

- Tracepoint support using gdbserver (on Linux)
  - Tracing on the host can still be done using gdbserver
- Tracepoints implemented by
  - Breakpoints (slow dynamic tracepoints)
  - Jump-patching (fast dynamic tracepoints)
  - User-space LTTng (static tracepoints)
- Observer-mode to enforce tracing instead of debugging

Dynamic Tracepoints (DSF-GDB Demo)

- Creation of tracepoint as is done as for breakpoints
- Enable/Disable tracepoints
- Dynamic condition can be assigned to a tracepoint
- Specification of data to be gathered using symbolic expressions and memory addresses (actions)
- Pass count per tracepoint to stop tracing automatically
- Trace-state variables that can be used in conditions and actions
- Tracepoints are only in effect if tracing is enabled

Dynamic Tracepoints

- Possible to define global actions (affecting all tracepoints)
- Option to use a finite trace buffer or circular trace buffer
- Disconnected data gathering

- On-disk trace data storage for ‘small’ amounts of data
- Automatic timestamp collection on successful tracepoint hit

Trace Data Visualization (DSF-GDB Demo)

- Navigation through data records using GDB
- Each data record is a snapshot of debug information
- Records can be examined using standard debugger views
  - As if debugger was attached at a specific point in time
  - Only collected information can be shown
  - Highlighting of the tracepoint of interest
- All collected data of a record can also be dumped as plain text

- Trace data can be saved to file
- Saved trace data can be examined offline

Next Tracepoint Features

- Disabling tracepoints during Tracing
- Tracepoints Enhanced Visualization:
  - Currently the user must have an idea of what has been collected
  - Goal is to directly and only show what has been collected
- Fast Tracepoints on 3-byte instruction
  - Currently fast tracepoints are 5-byte jumps insert in the code
  - New 3-byte jump to a nearby location to the 5-byte jump

Future DSF-GDB work

- Multi-core awareness
  - Reporting to the user which threads run on which cores
- Enhanced multi-process support
  - Currently limited to single address-space targets
  - Will be extended to Linux
- Flexible-hierarchy breakpoint view usage
  - Helios brings a new Flexible-hierarchy breakpoint view

Future DSF-GDB work

- Bringing more GDB features to DSF-GDB
  - Code patching (hot-swap?)
  - Checkpoints
  - Enhanced debugging console
  - Scripting
  - …
  - ➢ GDB is full of debugging feature, we just have to tap into that

Relevant Links

- CDT Tracepoint wiki
  - http://wiki.eclipse.org/CDT/designs/Tracepoints
- Features and screen shots
  - http://www.eclipse.org/dsdp/dd/development/relnotes/dd_news-1.1.html
  - http://wiki.eclipse.org/CDT/cdt-debug-default-integration
- DSF-GDB feature-parity effort
  - http://wiki.eclipse.org/CDT/cdt-debug-feature-parity-effort
- Reverse Debugging Webinar
  - http://live.eclipse.org/node/723

Questions?