# Comparison between perf, Ftrace, LTTng and GDB tracepoints

Rafik Fahem
Department of Computer and Software Engineering

*June 29, 2010*
*École Polytechnique, Montreal*

# Content

- **Perf**
- **Ftrace**
- **LTTng**
- **GDB tracepoints**
- **Discussion**
- **Questions**

# Perf

- Used for performance analysis
- Static tracepoints:
  - Makes use of the tracepoints available in the kernel(version 2.6.32 and higher)
- Dynamic tracepoints can be defined with:
  - Symbols and registers without debuginfo
  - C line numbers, function names and local variables with debuginfo

# Perf

- Dynamic tracepoints are implemented with kprobes
- Vmlinux containing the debuginfo has to be a Dwarf binary
- Traces only the events generated by a specific program

# Perf

- Available data:
  - Number of times the tracepoint was hit
  - Execution time
  - Calling functions
- Trace is available only after execution

# Ftrace

- Can be used for debugging and latency analysis
- Includes six different tracers: function calls, context switches...etc
- Uses the static tracepoints available in the kernel source
- Dynamic tracepoints unavailable

# Ftrace

- Available data depends on the tracer enabled: execution time, calling functions, PIDs...
- Thousands of filters available to trace only some selected events
- Trace is recorded in debugfs
- Trace can be viewed when tracing

# LTTng

- Optimized for static tracing
- Uses the static tracepoints available in the kernel
- Dynamic tracepoints implemented using kprobes
- Several probes can be connected to the same tracepoint
- Live monitoring under development

# GDB tracepoints

- Can only be used in user space
- Only dynamic tracepoints are supported in the current version
- UST tracepoints can be used
- Tracepoints inserted using traps
- Fast tracepoints:
  - Faster but restricted in where they may be installed
  - Inserted with a jump

# GDB tracepoints

- Data collected can include registers, local variables and global data.
- Tracepoints can be defined with line numbers, function names and addresses
- A trace snapshot is collected every time a tracepoint is hit
- These snapshots are stored in buffers and can be examined later

# Discussion

- Perf:
  - Tracing a single program
  - Statistics: execution time, calling functions, number of times the tracepoint was hit
  - Hardware performance counters
- Ftrace:
  - Filters
  - Trace in debugfs

# Discussion

- LTTng:
  - Optimized static tracing
  - User-space tracing
  - Trace streaming
- GDB tracepoints:
  - Fast tracepoints using jumps
- Debuginfo is needed to access variables(SystemTap)

# Questions?