ftrace and Mulitple Buffers

Steven Rostedt
Red Hat

ftrace Today

- Located in /sys/kernel/debug/tracing
- The 'trace' file
- The 'trace-pipe' file
- per_cpu/cpu*/
  - stats
  - trace
  - trace_pipe
  - trace_pipe_raw

ftrace Today

available_events
options
trace
available_filter_functions
per_cpu
trace_clock
available_tracers
printk_formats
trace_marker
buffer_size_kb
README
trace_options
buffer_total_size_kb
saved_cmdlines
trace_pipe
current_tracer
set_event
trace_stat
dyn_ftrace_total_info
set_ftrace_filter
tracing_cpumask
enabled_functions
set_ftrace_notrace
tracing_enabled
events
set_ftrace_pid
tracing_max_latency
free_buffer
set_graph_function
tracing_on
function_profile_enabled
stack_max_size
tracing_thresh
kprobe_events
stack_trace
kprobe_profile
stack_trace_filter

ftrace Today

- One global buffer
- One tracer at a time (function, nop, function graph, latency: irqsoff, preemptoff, etc)
- All events go into the same buffer
  - Tracing two events
    - One is a hog
    - One seldom triggers
      - The hog hides this event

ftrace Today

- Was always designed to handle multiple buffers
- Ring buffer is agnostic to the tracer
- global_trace (the tracer array) was always static
- Each tracer has uses its own trace array pointer

ftrace Today

- TRACE_EVENT came along
- The macros were complex
  - Simple things were done to offset the complexity
- Called handlers to just use the global buffer

ftrace Today

- TRACE_EVENT events now can pass data
- Function tracer can now pass data
- The descriptor of where the events are written can be passed

ftrace Tomorrow

- Encompass data recording within a group
- Multiple buffers
- Different events recording in different buffers

What's the problem?

- The work is done (still needs testing)
- The hardest part needs to be solidified
  - The User Interface
- /debug/tracing/instances
- /debug/tracing/instances/new
- /debug/tracing/instances/delete (free?)
- /debug/tracing/instances/foobar/events
- /debug/tracing/instances/foobar/trace (etc)

What else can we do?

- ls /debug/tracing/foobar
  - trace
  - trace_pipe
  - tracing_enabled
  - tracing_on
  - trace_marker
  - buffer_size_kb
  - (etc)

What's done

- Just a prototype (for now)
- Just events
  - No tracers
    - function
    - latency
  - But, these are to come
- Filter on processes
  - Currently filter is global

Perf?

- Add syscall interface to create ftrace buffers
- Use perf ioctl method
- allow perf to read the ftrace buffers
- Interleave the events from perf and ftrace

Discussion and Demo!