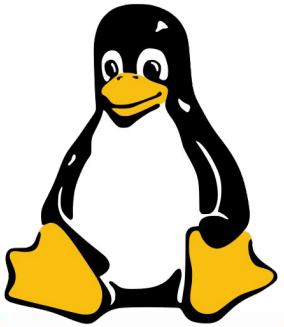
Tracing With ftrace

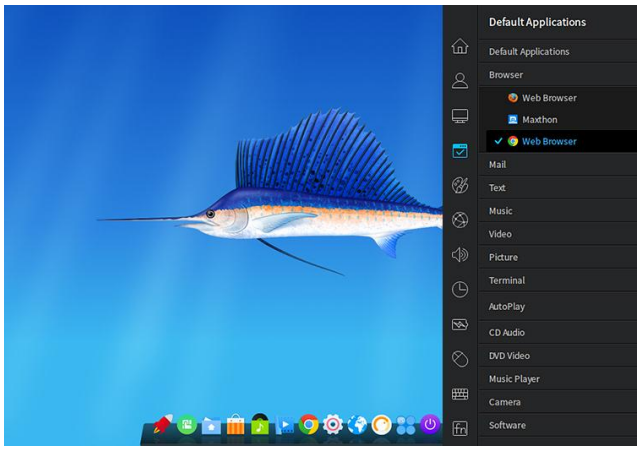
Derek Dai
Community Develop
Product Team
dailidu@linuxdeepin.com
daiderek@gmail.com

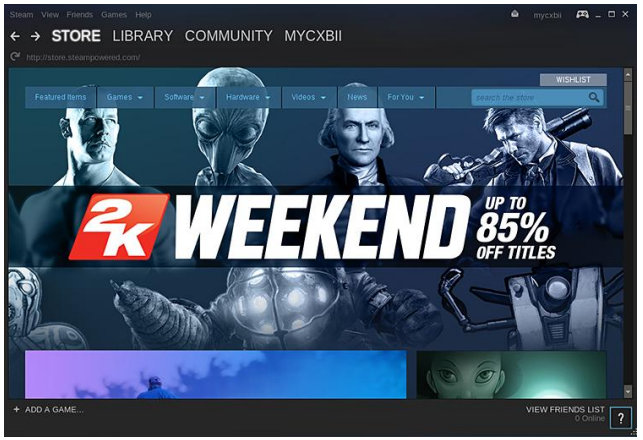


deepin

deepin

A Linux distribution committed to providing an elegant, user-friendly, safe and stable operating system for users all over the world.





Agenda

- What is ftrace?
- What info can ftrace provide to us?
- Where to get ftrace?
- Basic usage
- Tracers & filters
- Case study

What is ftrace?

- A tracing framework for the Linux kernel
- Primarily developed by Steven Rostedt
- Released on October 9, 2008

ftrace

http://www.brendangregg.com/Slides/Velocity2015_LinuxPerfTools.pdf

What is ftrace?

- You can tell ftrace what events to trace
- ftrace logs the events into kernel ring buffer
  - All by kernel itself
- Then you can dump the log from kernel, analyzes it, visualizes it, ... you name it

What is ftrace?

- ftrace have Integrated with four kinds of event sources
  - Static tracepoints
  - mcount()ed kernel functions
  - kprobe
  - uprobe

What info can ftrace provide to us?

- Entry and exit of a function
- Event/function name and arguments
- Return value
- Task name, PID
- CPU#
- Interrupt/schedule/preempt state
- Timestamp
- Stack depth, call graph, latency
- [...]

What info can ftrace provide to us?

This is a event looks like

```
# tracer: nop
#
# entries-in-buffer/entries-written: 3497/3497   #P:4
#
#                              _-----=> irqs-off
#                             / _----=> need-resched
#                            | / _---=> hardirq/softirq
#                            || / _--=> preempt-depth
#                            ||| /     delay
#           TASK-PID   CPU#  ||||    TIMESTAMP  FUNCTION
#              | |       |   ||||       |         |
 systemd-shutdow-1     [001] ....   122.135229: sys_kill(pid: 6ea, sig: f)
```

Where to get ftrace?

- Already built-in in modern Linux kernel
- Check with your kernel config

```
$ grep FTRACE /boot/config-4.3.0-0.bpo.1-amd64
CONFIG_KPROBES_ON_FTRACE=y
CONFIG_HAVE_KPROBES_ON_FTRACE=y
# CONFIG_PSTORE_FTRACE is not set
CONFIG_HAVE_DYNAMIC_FTRACE=y
CONFIG_HAVE_DYNAMIC_FTRACE_WITH_REGS=y
CONFIG_HAVE_FTRACE_MCOUNT_RECORD=y
CONFIG_FTRACE=y
CONFIG_FTRACE_SYSCALLS=y
CONFIG_DYNAMIC_FTRACE=y
CONFIG_DYNAMIC_FTRACE_WITH_REGS=y
CONFIG_FTRACE_MCOUNT_RECORD=y
# CONFIG_FTRACE_STARTUP_TEST is not set
```

Basic usage

- Enable ftrace through debugfs

```
# cd /sys/kernel/debug/tracing
# echo 'sys_enter_open' >set_event
# echo 1 >tracing_on
```

- Or from kernel command line

root=... ro **trace_event=sys_enter_open**

Basic usage - list events

```
## events in this file are static tracepoints
# cat available_events
v4l2:v4l2_dqbuf
v4l2:v4l2_qbuf
v4l2:vb2_buf_done
v4l2:vb2_buf_queue
v4l2:vb2_dqbuf
v4l2:vb2_qbuf
kvmmmu:kvm_mmu_pagetable_walk
kvmmmu:kvm_mmu_paging_element
kvmmmu:kvm_mmu_set_accessed_bit
kvmmmu:kvm_mmu_set_dirty_bit
kvmmmu:kvm_mmu_walker_error
kvmmmu:kvm_mmu_get_page
kvmmmu:kvm_mmu_sync_page
[...]
```

Basic usage - trace syscall open()

```
# grep 'sys_[^_]*_open$' available_events
syscalls:sys_exit_open
Syscalls:sys_enter_open

# echo sys_exit_open >set_event
# cat set_event
syscalls:sys_exit_open

# echo sys_enter_open >>set_event
# cat set_event
syscalls:sys_exit_open
syscalls:sys_enter_open
```

Basic usage - trace all syscalls

```
# echo syscalls >set_event
# cat set_event
syscalls:sys_exit_iopl
syscalls:sys_enter_iopl
syscalls:sys_exit_mmap
syscalls:sys_enter_mmap
syscalls:sys_exit_unshare
syscalls:sys_enter_unshare
syscalls:sys_exit_set_tid_address
syscalls:sys_enter_set_tid_address
syscalls:sys_exit_personality
syscalls:sys_enter_personality
syscalls:sys_exit_wait4
syscalls:sys_enter_wait4
Syscalls:sys_exit_waitid
[...]
```

Basic usage - start tracing

```
# echo 1 >tracing_on
## dump event log
# cat trace
# tracer: nop
#
# entries-in-buffer/entries-written: 174105/9551275   #P:4
#
#                              _-----=> irqs-off
#                             / _----=> need-resched
#                            | / _---=> hardirq/softirq
#                            || / _--=> preempt-depth
#                            ||| /     delay
#           TASK-PID   CPU#  ||||    TIMESTAMP  FUNCTION
#              | |       |   ||||       |         |
          chrome-17948 [003] .... 10372.244365: sys_recvmsg(fd: c, msg: 7fffdac4c4d0, flags:
0)
          chrome-17948 [003] .... 10372.244366: sys_recvmsg -> 0xfffffffffffffff5
          chrome-17948 [003] .... 10372.244367: sys_poll(ufds: 15c48b3127e0, nfds: 3,
timeout_msecs: ffffffff)
          [...]
```

Basic usage - stop and clean up

```
## stop tracing
# echo 0 >tracing_on

## clean up ring buffer
# echo >trace

## remove all the tracing events
# echo >set_event
```

Tracers

```
## list available tracers
# cat available_tracers
blk mmiotrace function_graph function nop

## check current tracer
# cat current_tracer
nop
```

Tracers - function

```
## trace almost every kernel function calls
# wc -l available_filter_functions
41596 available_filter_functions
# echo function >current_tracer
# cat trace
# tracer: function
#
# entries-in-buffer/entries-written: 205048/12051258   #P:4
#
#                              _-----=> irqs-off
#                             / _----=> need-resched
#                            | / _---=> hardirq/softirq
#                            || / _--=> preempt-depth
#                            ||| /     delay
#           TASK-PID   CPU#  ||||    TIMESTAMP  FUNCTION
#              | |       |   ||||       |         |
 gnome-terminal--2196  [003] .... 16660.619043: skb_copy_datagram_iter <-
unix_stream_read_actor
 gnome-terminal--2196  [003] .... 16660.619044: consume_skb <-unix_stream_read_generic
 gnome-terminal--2196  [003] .... 16660.619044: mutex_unlock <-unix_stream_read_generic
```

Tracers - function

```
## trace only the functions that match given patterns
## sched, *sched, *sched*, sched*
# echo 'sched*' >set_ftrace_filter
# cat trace
[...]
          <idle>-0     [003] .N.. 26548.183873: sched_ttwu_pending <-cpu_startup_entry
          <idle>-0     [003] .N.. 26548.183873: schedule_preempt_disabled <-cpu_startup_entry
          <idle>-0     [003] .N.. 26548.183873: schedule <-schedule_preempt_disabled
            Xorg-949   [001] .... 26548.183880: schedule_hrtimeout_range <-
poll_schedule_timeout
            Xorg-949   [001] .... 26548.183881: schedule_hrtimeout_range_clock <-
poll_schedule_timeout
            Xorg-949   [001] .... 26548.183881: schedule_hrtimeout_range_clock.part.23 <-
poll_schedule_timeout
            Xorg-949   [001] .... 26548.183882: schedule <-schedule_hrtimeout_range_clock.
part.23
          <idle>-0     [001] d... 26548.183885: sched_idle_set_state <-cpuidle_enter_state
[...]
```

Tracers - function

```
## filter out the functions that match given patterns
# echo function >current_tracer
# echo '*lock' >set_ftrace_notrace
# echo '*rcu*' >set_ftrace_notrace
# cat trace
[...]
        haveged-821   [003] d... 27263.056357: put_prev_entity <-put_prev_task_fair
        haveged-821   [003] d... 27263.056357: check_cfs_rq_runtime <-put_prev_entity
           cat-7038   [001] .... 27263.056357: do_set_pte <-filemap_map_pages
           cat-7038   [001] .... 27263.056357: add_mm_counter_fast <-do_set_pte
           cat-7038   [001] .... 27263.056357: page_add_file_rmap <-do_set_pte
           cat-7038   [001] .... 27263.056357: mem_cgroup_begin_page_stat <-
page_add_file_rmap
         <idle>-0     [003] d... 27263.056357: finish_task_switch <-__schedule
           cat-7038   [001] .... 27263.056357: mem_cgroup_end_page_stat <-do_set_pte
         <idle>-0     [003] .... 27263.056357: tick_nohz_idle_enter <-cpu_startup_entry
           cat-7038   [001] .... 27263.056357: unlock_page <-filemap_map_pages
[...]
```

Tracers - function_graph

```
# echo function_graphc >current_tracer
# cat trace
# tracer: function_graph
#
# CPU  DURATION                  FUNCTION CALLS
# |     |   |                     |   |   |   |
 0)               |              mutex_lock() {
 0)   0.048 us    |                _cond_resched();
 0)   0.449 us    |              }
 0)   0.048 us    |              _raw_spin_lock();
 0)   0.041 us    |              mutex_unlock();
 0)   0.043 us    |              put_pid();
 0)   2.323 us    |            } /* unix_stream_read_generic */
 0)   2.702 us    |          } /* unix_stream_recvmsg */
 0)   3.508 us    |        } /* sock_recvmsg */
 0)   0.048 us    |        kfree();
 0)   5.157 us    |      } /* ___sys_recvmsg */
 0)   0.047 us    |      fput();
 0)   7.391 us    |    } /* __sys_recvmsg */
 [...]
```

Case study: what is blocking the shutdown

A user session refuses to terminate. systemd waits until it timed out, then kills it forcefully.

Case study: what is blocking the shutdown

- Since the system is shutting down, it's difficult to trace from the user space except from the init (systemd) itself.
- A user session is composed by a variety of programs... but which ones is causing the problem?

Case study: what is blocking the shutdown

Collect information below to narrow down the problem, then gdb it if necessary

- Signal related
  - Who sent, what signal, when, to whom, when was the signal received
- Processes lifecycle related
  - Who terminated, when, for what reason

Case study: what is blocking the shutdown

Create a hook script which is executed by systemd to dump log into a file

```
# cat <<END >/lib/systemd/system-shutdown/debug.sh
#!/bin/sh
mount -o remount,rw /
cat /sys/kernel/debug/tracing/trace >/shutdown.log
mount -o remount,ro /
END
# chmod +x /lib/systemd/system-shutdown/debug.sh
```

Case study: what is blocking the shutdown

Enable ftrace before shutdown

```
# echo syscalls:sys_enter_kill   >>set_event
# echo syscalls:sys_enter_tgkill >>set_event
# echo signal:signal_deliver     >>set_event
# echo sched:sched_process_exit  >>set_event
# echo global                    >trace_clock
# echo 40960                     >buffer_size_kb
# echo nop                       >current_tracer
# echo                           >trace
# echo 1                         >trcing_on
```

Case study: what is blocking the shutdown

wc -l /shutdown.log

3508 /shutdown.log

cat /shutdown.log

```
[...]
   scim-im-agent-2274  [003] ....   123.570977: sched_process_exit: comm=scim-im-agent
pid=2274 prio=120
  scim-panel-gtk-2275  [002] ....   123.571340: sched_process_exec: filename=/usr/lib/x86_64-
linux-gnu/scim-1.0/scim-panel-gtk pid=2275 old_pid=2275
   scim-im-agent-2055  [002] ....   124.272921: sched_process_exit: comm=scim-im-agent
pid=2055 prio=120
 systemd-shutdow-1     [000] ....   212.220956: sys_kill(pid: 8e3, sig: 9)
  scim-panel-gtk-2275  [002] d...   212.221019: signal_deliver: sig=9 errno=0 code=0
sa_handler=0 sa_flags=0
 systemd-shutdow-1     [000] ....   212.221022: sys_kill -> 0x0
  scim-panel-gtk-2275  [002] ....   212.221585: sched_process_exit: comm=scim-panel-gtk
pid=2275 prio=120
[...]
```

Case study: what is blocking the shutdown

Process log with script

```
[...]
  scim-im-agent(2274) exited at  4.532707s, got signals:

  scim-im-agent(2055) exited at  5.234651s, got signals:  1(3.125658) 15
(3.125659)

 scim-panel-gtk(2275) exited at 93.183315s, got signals:  9(93.182749)

 systemd-logind( 509) exited at 93.189001s, got signals:  9(93.188841)

[...]
```

Resources

- Dynamic probes with ftrace
- Debugging the kernel using Ftrace - part 1
- Debugging the kernel using Ftrace - part 2
- Measuring Function Duration
- Secrets of the Ftrace function tracer
- perf-tools Project
- Documentation/trace

Questions?

What info cat ftrace provide to us?

- ftrace + kprobe
  - kprobe can trigger events wherever you want in the kernel
- ftrace + uprobe
  - Just like kprobe but for the user space
- Both probe types can extract register value, memory content, ... when an event occurs
- We can discuss it after this session