LINUX系统高级调试和优化

--事件追踪

张银奎

2017/8/12

1

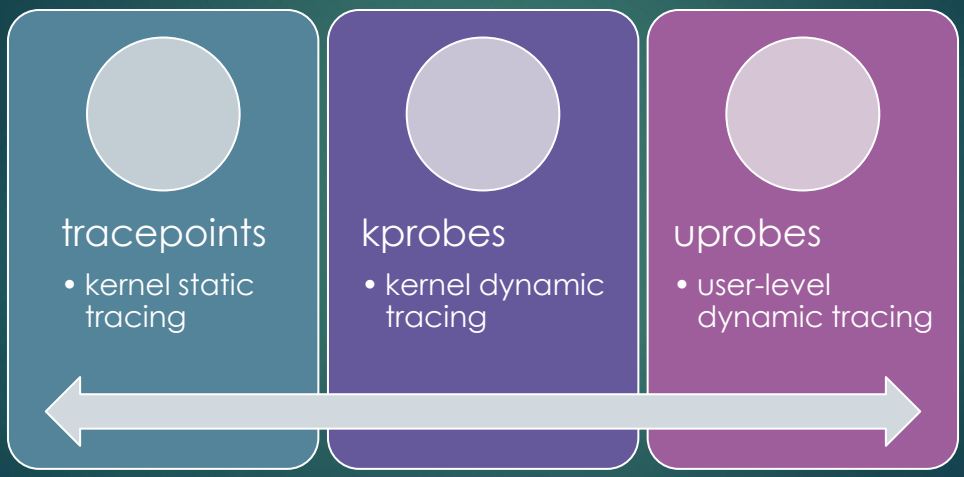
Gregg的追踪机制时间表

1990's: Static tracers, prototype dynamic tracers
• 2004: Linux kprobes (2.6.9)
– Dynamic kernel tracing, difficult interface
• 2005: Solaris DTrace (s10)
– Static & dynamic tracing, user & kernel level, production ready, easy to use, far better than anything of the time, and, marketed
• 2008: Linux ftrace (2.6.27)
• 2009: Linux perf (2.6.31)
• 2009: Linux tracepoints (2.6.32)
– Static kernel tracing
• 2010-2014: ftrace & perf_events enhancements
• 2014: eBPF patches
– Fast (JIT'd) in kernel aggregations and programs

http://www.brendangregg.com/Slides/LISA2014_LinuxPerfAnalysisNewTools.pdf

2

三类追踪信息源



3

c:\bench\linux-4.4.14\Documentation\trace*.*

↑Name	Ext	Size	Date	Attr
[..]		<DIR>	2016/06/25 01:18	----
[postprocess]		<DIR>	2016/06/25 01:18	----
coresight	txt	15,011	2016/06/25 01:18	-a--
events	txt	17,546	2016/06/25 01:18	-a--
events-kmem	txt	5,389	2016/06/25 01:18	-a--
events-nmi	txt	1,561	2016/06/25 01:18	-a--
events-power	txt	3,494	2016/06/25 01:18	-a--
ftrace	txt	109,566	2016/06/25 01:18	-a--
ftrace-design	txt	14,383	2016/06/25 01:18	-a--
function-graph-fold	vim	1,355	2016/06/25 01:18	-a--
intel_th	txt	3,665	2016/06/25 01:18	-a--
kprobetrace	txt	7,082	2016/06/25 01:18	-a--
mmiotrace	txt	6,804	2016/06/25 01:18	-a--
ring-buffer-design	txt	30,648	2016/06/25 01:18	-a--
stm	txt	3,941	2016/06/25 01:18	-a--
tracepoint-analysis	txt	12,243	2016/06/25 01:18	-a--
tracepoints	txt	5,227	2016/06/25 01:18	-a--
uprobetracer	txt	6,447	2016/06/25 01:18	-a--

4

ftrace

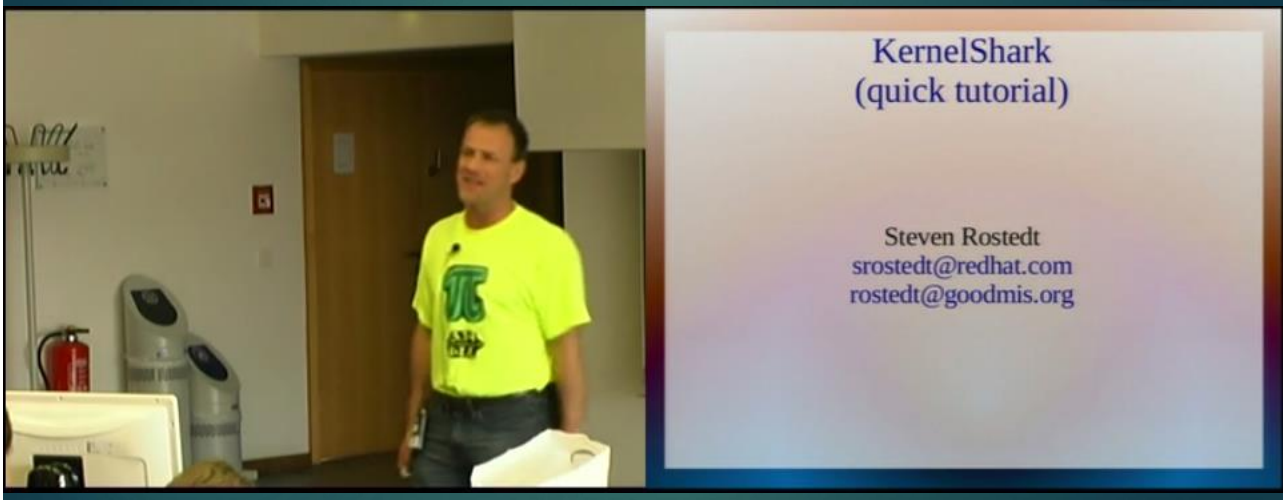
- Linux内核中的踪迹工具
  - 最初在 2.6.27 中出现
- 最初只能产生函数调用的踪迹
- 目前已发展为框架，支持扩展
  - **Power tracer**：记录系统电源管理相关的信息
  - **Hardware branch tracer**：利用处理器的分支跟踪能力，实现硬件级别的指令跳转记录
  - **Schedule switch tracer**: 跟踪进程调度情况
  - **Initcall tracer**：记录系统在 boot 阶段所调用的 init call

5

5

```
            ftrace - Function Tracer
            ========================

Copyright 2008 Red Hat Inc.
  Author:   Steven Rostedt <srostedt@redhat.com>
 License:   The GNU Free Documentation License, Version 1.2
            (dual licensed under the GPL v2)
Reviewers:  Elias Oltmanns, Randy Dunlap, Andrew Morton,
            John Kacur, and David Teigland.
Written for: 2.6.28-rc2
Updated for: 3.10

Introduction
------------

Ftrace is an internal tracer designed to help out developers and
designers of systems to find what is going on inside the kernel.
It can be used for debugging or analyzing latencies and
performance issues that take place outside of user-space.

Although ftrace is typically considered the function tracer, it
is really a frame work of several assorted tracing utilities.
There's latency tracing to examine what occurs between interrupts
disabled and enabled, as well as for preemption and from a time
```

6



https://chemnitzer.linux-tage.de/2012/vortraege/media/videos/v4sa1500.mp4

7

实现：kernel/trace

blktrace.c		
bpf_trace.c	ftrace.c	Kconfig
Makefile	power-traces.c	ring_buffer.c
ring_buffer_benchmark.c	rpm-traces.c	trace.c
trace.h	trace_benchmark.c	trace_benchmark.h
trace_branch.c	trace_clock.c	trace_entries.h
trace_events.c	trace_events_filter.c	trace_events_filter_test.h
trace_events_trigger.c	trace_event_perf.c	trace_export.c
trace_functions.c	trace_functions_graph.c	trace_irqsoff.c
trace_kdb.c	trace_kprobe.c	trace_mmiotrace.c
trace_nop.c	trace_output.c	trace_output.h
trace_printk.c	trace_probe.c	trace_probe.h
trace_sched_switch.c	trace_sched_wakeup.c	trace_selftest.c
trace_selftest_dynamic.c	trace_seq.c	trace_stack.c
trace_stat.c	trace_stat.h	trace_syscalls.c
trace_uprobe.c		
44 个文件	1,109,715 字节	

8

VFS接口

Ftrace uses the debugfs file system to hold the control files as well as the files to display output.

When debugfs is configured into the kernel (which selecting any ftrace option will do) the directory /sys/kernel/debug will be created. To mount this directory, you can add to your /etc/fstab file:

debugfs /sys/kernel/debug debugfs defaults 0 0

Or you can mount it at run time with:

mount -t debugfs nodev /sys/kernel/debug

9

文件接口

```
root@gewubox:/sys/kernel/debug/tracing# ls
available_events            max_graph_depth     stack_trace_filter
available_filter_functions  options             trace
available_tracers           per_cpu             trace_clock
buffer_size_kb              printk_formats      trace_marker
buffer_total_size_kb        README              trace_options
current_tracer              saved_cmdlines      trace_pipe
dyn_ftrace_total_info       set_event           trace_stat
enabled_functions           set_ftrace_filter   tracing_cpumask
events                      set_ftrace_notrace  tracing_max_latency
free_buffer                 set_ftrace_pid      tracing_on
function_profile_enabled    set_graph_function  tracing_thresh
instances                   snapshot            uprobe_events
kprobe_events               stack_max_size      uprobe_profile
kprobe_profile              stack_trace
```

10

使用ftrace

- 选择 tracer
- 启用
- 执行目标程序，促发产生踪迹信息
- 关闭 ftrace
- 查看 trace 文件

11

追踪器(tracer)

- 将追踪源的原始信息搜集起来，计算加工，并以标准接口提供给前端

12

数据结构

```
(gdb) pt function_trace
type = struct tracer {
    const char *name;
    int (*init)(struct trace_array *);
    void (*reset)(struct trace_array *);
    void (*start)(struct trace_array *);
    void (*stop)(struct trace_array *);
    void (*open)(struct trace_iterator *);
    void (*pipe_open)(struct trace_iterator *);
    void (*wait_pipe)(struct trace_iterator *);
    void (*close)(struct trace_iterator *);
    void (*pipe_close)(struct trace_iterator *);
    ssize_t (*read)(struct trace_iterator *, struct file *, char *, size_t,  loff_t *);
    ssize_t (*splice_read)(struct trace_iterator *, struct file *, loff_t *,
    struct pipe_inode_info *, size_t, unsigned int);
    void (*print_header)(struct seq_file *);
    enum print_line_t (*print_line)(struct trace_iterator *);
    int (*set_flag)(u32, u32, int);
    int (*flag_changed)(struct tracer *, u32, int);
    struct tracer *next;
    struct tracer_flags *flags;
    bool print_max;
    bool enabled;
    bool use_max_tr;
}
```

13

格物

```
(gdb) p function_trace
$1 = {name = 0xc18752a4 "function", init = 0xc10ffb80 <function_trace_init>,
  reset = 0xc10ffb40 <function_trace_reset>,
  start = 0xc10ffb30 <function_trace_start>, stop = 0, open = 0,
  pipe_open = 0, wait_pipe = 0xc10fbc20 <poll_wait_pipe>, close = 0,
  pipe_close = 0, read = 0, splice_read = 0, print_header = 0, print_line = 0,
  set_flag = 0xc10ffc90 <func_set_flag>, flag_changed = 0, next = 0xc19e11e0,
  flags = 0xc197224c, print_max = false, enabled = false, use_max_tr = false}
```

14

观察可用的tracer

root@gewubox:/sys/kernel/debug/tracing# cat available_tracers
blk mmiotrace function_graph wakeup_rt wakeup function nop

15

选用tracer

- # sudo su
- # cd /sys/kernel/debug/tracing
- # echo function > current_tracer
- #cat trace

- 快速禁止或者启用
- # echo 0 > tracing_on : quick way to disable tracing
- # echo 1 > tracing_on : quick way to re-enable tracing

- 停用
- # echo nop > current_tracer

16

过滤

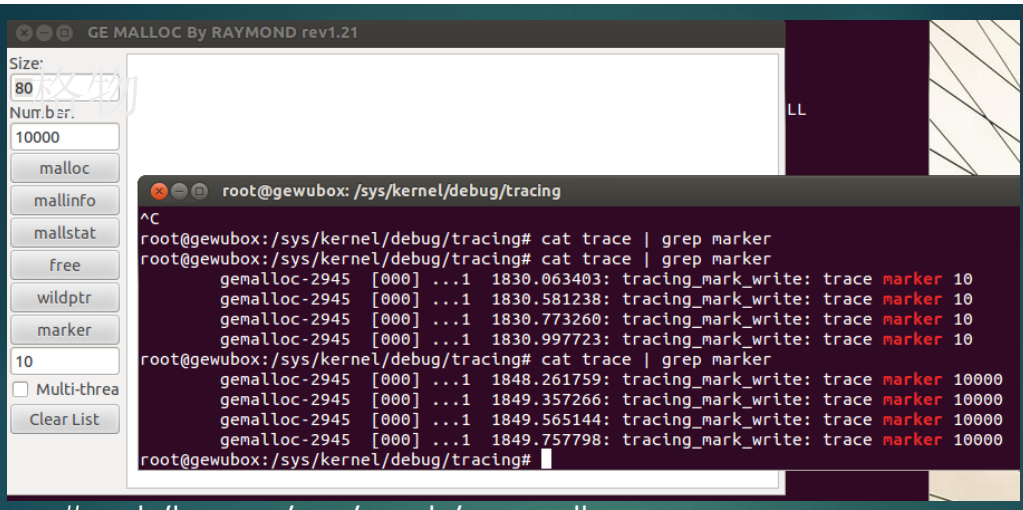
- tracing_cpumask:
  - This is a mask that lets the user only trace on specified CPUs. The format is a hex string representing the CPUs.
- set_ftrace_notrace:
  - This has an effect opposite to that of
- set_ftrace_filter.
  - Any function that is added here will not be traced. If a function exists in both set_ftrace_filter and set_ftrace_notrace, the function will _not_ be traced.
- set_ftrace_pid:
  - Have the function tracer only trace a single thread.
- set_event_pid:
  - Have the events only trace a task with a PID listed in this file.
  - Note, sched_switch and sched_wake_up will also trace events listed in this file.

17

trace_marker

```
void trace_write(const char *fmt, ...)
{
	va_list ap;
	char buf[256];
	int n;

	if (trace_fd < 0)
		return;

	va_start(ap, fmt);
	n = vsnprintf(buf, 256, fmt, ap);
	va_end(ap);

	write(trace_fd, buf, n);
}

start:

	trace_fd = open("trace_marker", WR_ONLY);
```

18




```
# cd /home/ge/work/gemalloc
# sudo ./gemalloc
#echo function > current_tracer
# echo pid_of_gemalloc > set_ftrace_pid
# cat trace | grep marker
```

19

```
# tracer: function
#
# entries-in-buffer/entries-written: 140080/250280   #P:4
#
#                              _-----=> irqs-off
#                             / _----=> need-resched
#                            | / _---=> hardirq/softirq
#                            || / _--=> preempt-depth
#                            ||| /     delay
#           TASK-PID   CPU#  ||||    TIMESTAMP  FUNCTION
#              | |       |   ||||       |         |
          bash-1977  [000] .... 17284.993652: sys_close <-system_call_fastpath
          bash-1977  [000] .... 17284.993653: __close_fd <-sys_close
          bash-1977  [000] .... 17284.993653: _raw_spin_lock <-__close_fd
          sshd-1974  [003] .... 17284.993653: __srcu_read_unlock <-fsnotify
          bash-1977  [000] .... 17284.993654: add_preempt_count <-_raw_spin_lock
          bash-1977  [000] ...1 17284.993655: _raw_spin_unlock <-__close_fd
          bash-1977  [000] ...1 17284.993656: sub_preempt_count <-_raw_spin_unlock
          bash-1977  [000] .... 17284.993657: filp_close <-__close_fd
          bash-1977  [000] .... 17284.993657: dnotify_flush <-filp_close
          sshd-1974  [003] .... 17284.993658: sys_select <-system_call_fastpath
                            --------
```

Note: all time values are in microseconds.

20

```
# tracer: irqsoff
#
# irqsoff latency trace v1.1.5 on 3.8.0-test+
# --------------------------------------------------------------------
# latency: 16 us, #4/4, CPU#0 |  (M:preempt VP:0, KP:0, SP:0 HP:0 #P:4)
#    -----------------
#    | task: swapper/0-0 (uid:0 nice:0 policy:0 rt_prio:0)
#    -----------------
#  => started at: run_timer_softirq
#  => ended at:   run_timer_softirq
#
#
#                  _------=> CPU#
#                 / _-----=> irqs-off
#                | / _----=> need-resched
#                | | / _---=> hardirq/softirq
#                | | | / _--=> preempt-depth
#                | | | | /     delay
#  cmd     pid   | | | | |  time  |   caller
#     \   /      | | | | |  \     |   /
  <idle>-0       0d.s2    0us+: _raw_spin_lock_irq <-run_timer_softirq
  <idle>-0       0dNs3   17us : _raw_spin_unlock_irq <-run_timer_softirq
  <idle>-0       0dNs3   17us+: trace_hardirqs_on <-run_timer_softirq
  <idle>-0       0dNs3   25us : <stack trace>
 =>   raw_spin_unlock_irq
```

21

trace-cmd: A front-end for Ftrace

trace-cmd [COMMAND] ...

- record - record a trace into a trace.dat file
- start - start tracing without recording into a file
- extract - extract a trace from the kernel
- stop - stop the kernel from recording trace data
- reset - disable all kernel tracing and clear the trace buffers
- report - read out the trace stored in a trace.dat file
- split - parse a trace.dat file into smaller file(s)
- listen - listen on a network socket for trace clients
- list - list the available events, plugins or options

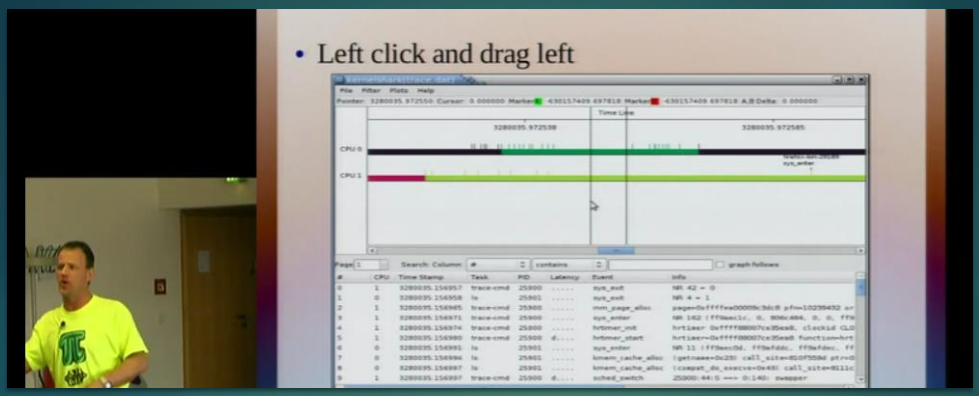
22

trace-cmd start -p function

```
#0  function_trace_start (tr=0xc1b5f180) at kernel/trace/trace_functions.c:46
#1  0xc10f58d0 in rb_simple_write (filp=<optimized out>, ubuf=<optimized out>,
    cnt=1, ppos=0xe9193f98) at kernel/trace/trace.c:5845
#2  0xc1178645 in vfs_write (file=0xe28919c0, buf=0x806389b "1", count=1,
    pos=0xe9193f98) at fs/read_write.c:473
#3  0xc1178af7 in SYSC_write (count=1, buf=0x806389b "1", fd=<optimized out>)
    at fs/read_write.c:522
#4  SyS_write (fd=3, buf=134625435, count=1) at fs/read_write.c:515
#5  <signal handler called>
#6  0xb77ad424 in ?? ()
#7  0xb7602533 in ?? ()
```

plugin function

23



KernelShark

24

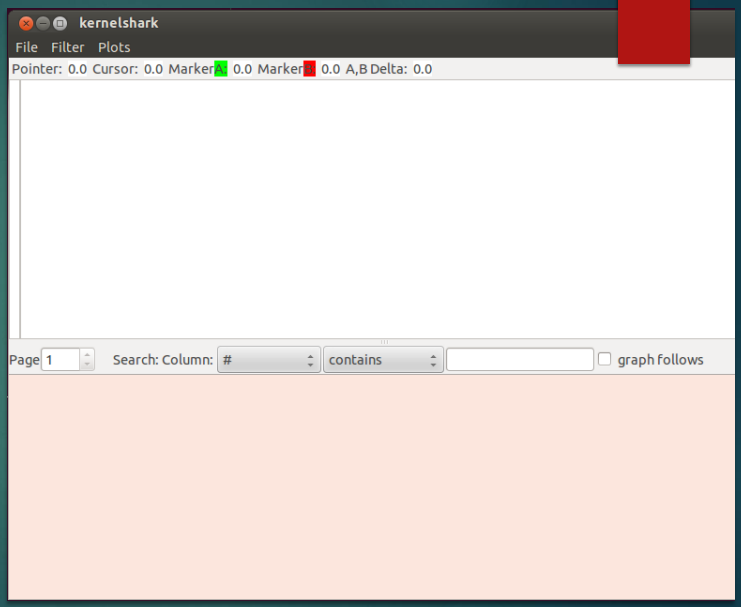
安装

```
root@gewubox:~# sudo apt-get install kernelshark
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following NEW packages will be installed:
  kernelshark
0 upgraded, 1 newly installed, 0 to remove and 358 not upgraded.
Need to get 263 kB of archives.
After this operation, 586 kB of additional disk space will be used.
Get:1 http://cn.archive.ubuntu.com/ubuntu/ precise/universe kernelshark i386 1.0.3-0ubuntu2 [263 kB]
Fetched 263 kB in 0s (730 kB/s)
Selecting previously unselected package kernelshark.
(Reading database ... 151009 files and directories currently installed.)
Unpacking kernelshark (from .../kernelshark_1.0.3-0ubuntu2_i386.deb) ...
Setting up kernelshark (1.0.3-0ubuntu2) ...
```

25

不完整版本

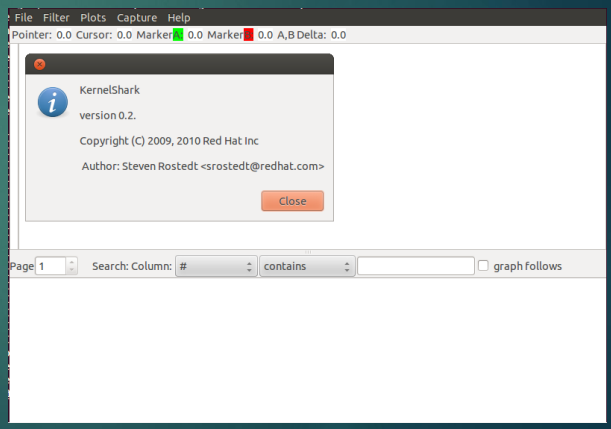
- 缺少Record和help菜单



26

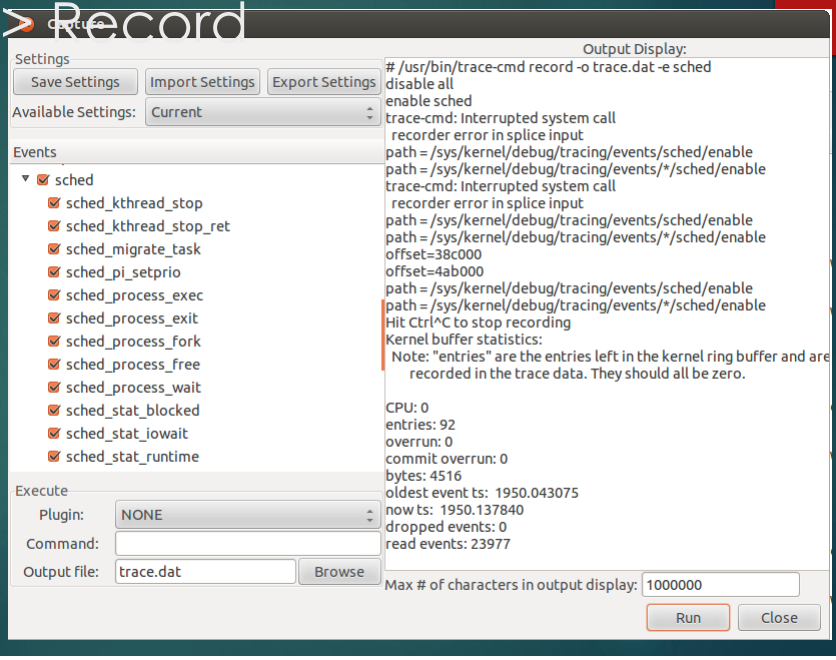
自己build

- git clone git://git.kernel.org/pub/scm/linux/kernel/git/rostedt/trace-cmd.git
- sudo apt-get install swig
- sudo apt-get install libxml2-dev
- make gui



27

Capture > Record

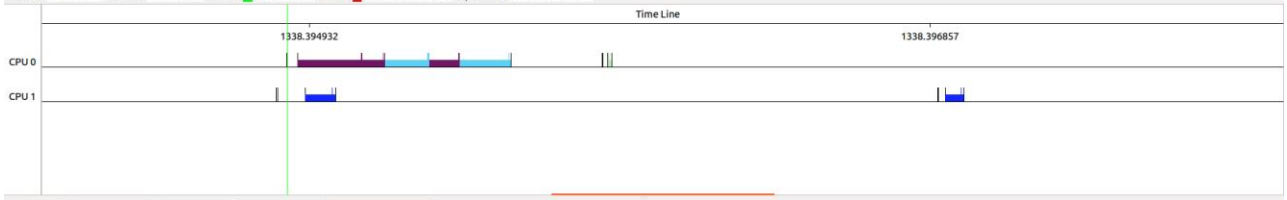


28

调度细节

File Filter Plots Capture Help

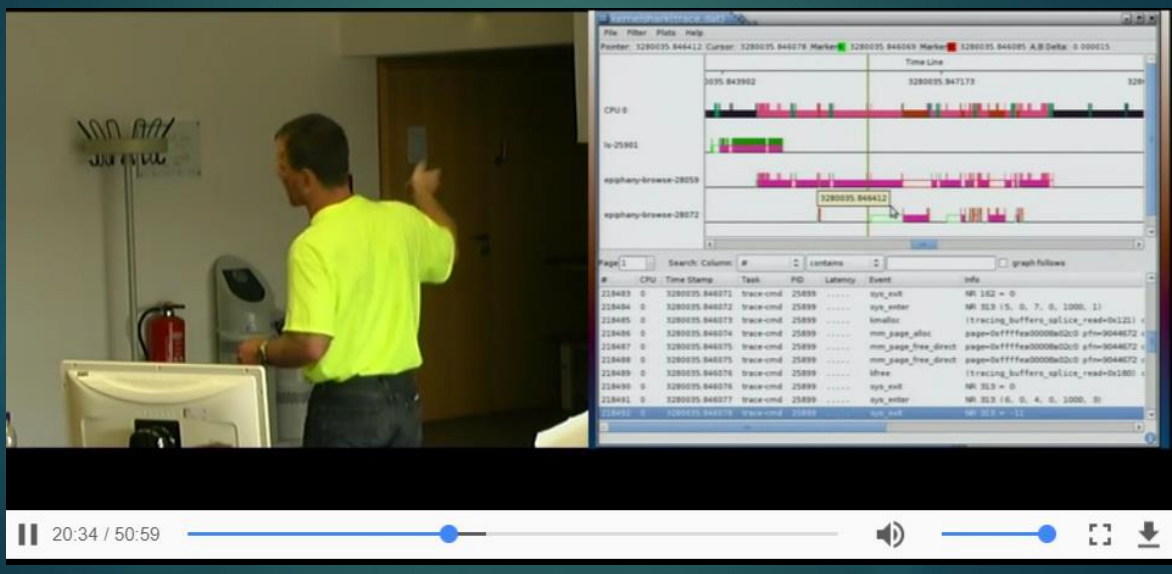
Pointer: 1338.394154 Cursor: 1338.347307 Marker A: 1338.394865 Marker B: -633436106.662061 A,B Delta: 633437444.652623



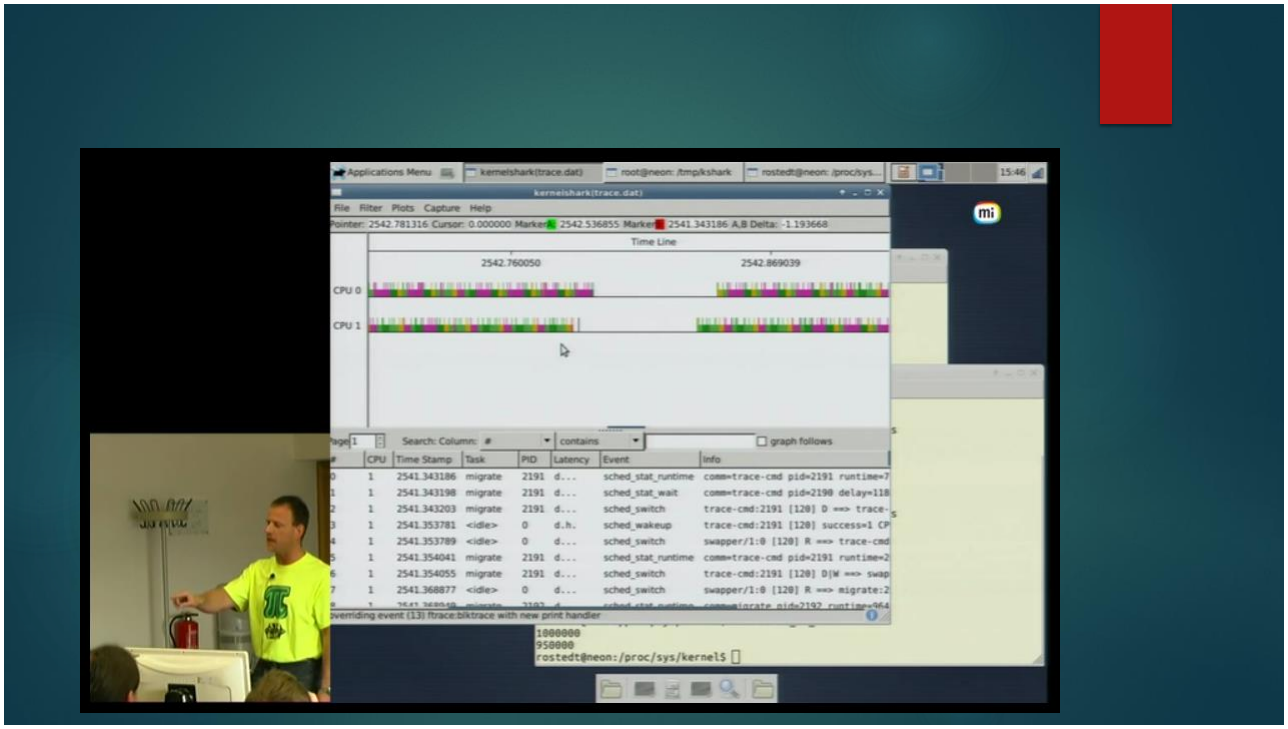
Page 1 Search: Column: CPU contains graph follows

#	CPU	Time Stamp	Task	PID	Latency	Event	Info
275	0	1338.347163	Xorg	991	d...	sched_stat_runtime	comm=Xorg pid=991 runtime=288940 [ns] vruntime=12464107532 [ns]
276	0	1338.347167	Xorg	991	d...	sched_stat_wait	comm=kernelshark pid=6714 delay=93375 [ns]
277	0	1338.347167	Xorg	991	d...	sched_switch	prev_comm=Xorg prev_pid=991 prev_prio=120 prev_state=S ==> next_comm=kernelshark next_pid=6714 next_prio=120
278	0	1338.347301	kernelshark	6714	d...	sched_stat_sleep	comm=Xorg pid=991 delay=137979 [ns]
279	0	1338.347304	kernelshark	6714	dN..	sched_wakeup	comm=Xorg pid=991 prio=120 success=1 target_cpu=000
280	0	1338.347305	kernelshark	6714	dN..	sched_stat_runtime	comm=kernelshark pid=6714 runtime=137979 [ns] vruntime=19999115272 [ns]
281	0	1338.347307	kernelshark	6714	dN..	sched_stat_wait	comm=Xorg pid=991 delay=0 [ns]
282	0	1338.347307	kernelshark	6714	d...	sched_switch	prev_comm=kernelshark prev_pid=6714 prev_prio=120 prev_state=R ==> next_comm=Xorg next_pid=991 next_prio=120
283	0	1338.347396	Xorg	991	d...	sched_stat_runtime	comm=Xorg pid=991 runtime=95434 [ns] vruntime=12464202966 [ns]
284	0	1338.347398	Xorg	991	d...	sched_stat_wait	comm=kernelshark pid=6714 delay=95434 [ns]
285	0	1338.347399	Xorg	991	d...	sched_switch	prev_comm=Xorg prev_pid=991 prev_prio=120 prev_state=S ==> next_comm=kernelshark next_pid=6714 next_prio=120
286	0	1338.347493	kernelshark	6714	d...	sched_stat_runtime	comm=kernelshark pid=6714 runtime=97238 [ns] vruntime=19999212510 [ns]
287	0	1338.347501	kernelshark	6714	d...	sched_switch	prev_comm=kernelshark prev_pid=6714 prev_prio=120 prev_state=S ==> next_comm=swapper/0 next_pid=0 next_prio=120
288	0	1338.354948	<idle>	0	d.h.	sched_stat_sleep	comm=Xorg pid=991 delay=7548869 [ns]
289	0	1338.354952	<idle>	0	d.h.	sched_wakeup	comm=Xorg pid=991 prio=120 success=1 target_cpu=000
290	0	1338.354995	<idle>	0	dN..	sched_stat_wait	comm=Xorg pid=991 delay=0 [ns]
291	0	1338.354996	<idle>	0	d...	sched_switch	prev_comm=swapper/0 prev_pid=0 prev_prio=120 prev_state=R ==> next_comm=Xorg next_pid=991 next_prio=120
292	1	1338.355028	<idle>	0	d.h.	sched_stat_sleep	comm=ssh-agent pid=1819 delay=10005619955 [ns]
293	1	1338.355031	<idle>	0	dNh.	sched_wakeup	comm=ssh-agent pid=1819 prio=120 success=1 target_cpu=001
294	1	1338.355057	<idle>	0	dN..	sched_stat_wait	comm=ssh-agent pid=1819 delay=0 [ns]

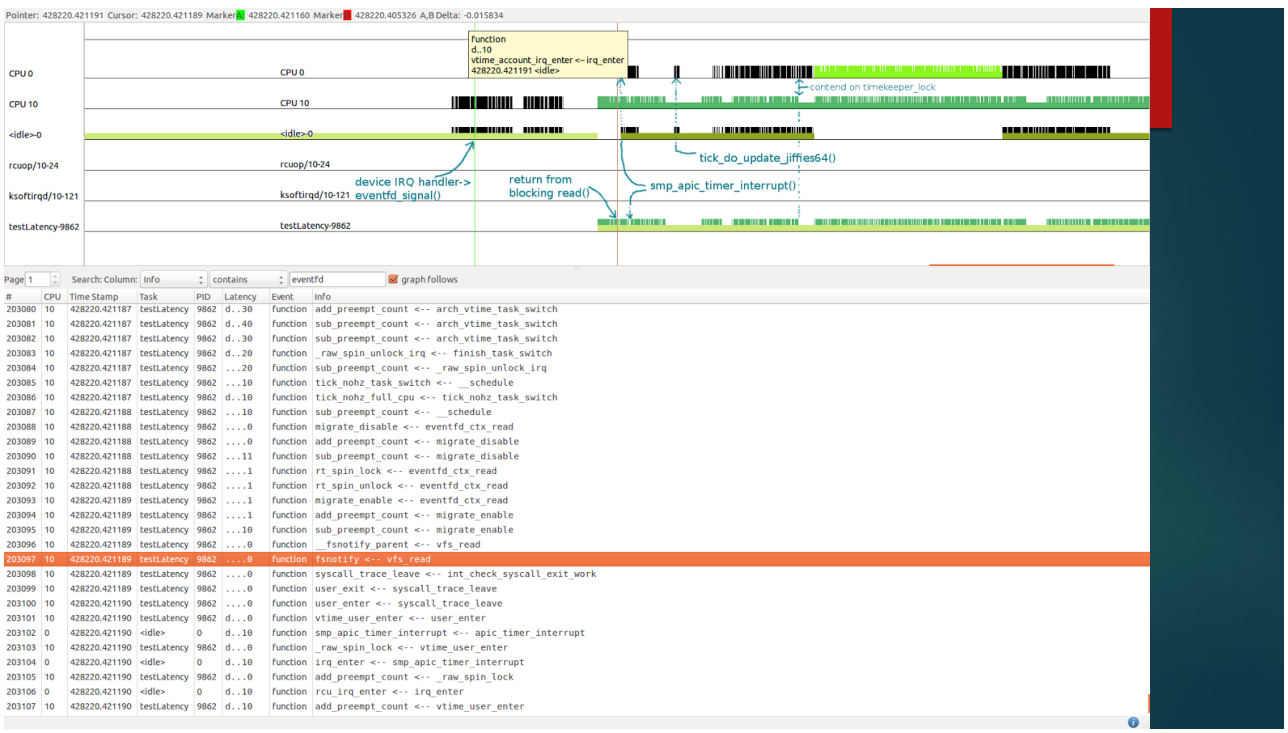
29



30


Applications Menu
kernelshark(trace.dat)
root@neon: /tmp/kshark
rostedt@neon: /proc/sys...
15:46
kernelshark(trace.dat)
File Filter Plots Capture Help
Time Line
2542.760050
2542.869039
CPU 0
CPU 1
Search: Column: # contains graph follows
CPU Time Stamp Task PID Latency Event Info
0 1 2541.343186 migrate 2191 d... sched_stat_runtime comm=trace-cmd pid=2191 runtime=7
1 1 2541.343198 migrate 2191 d... sched_stat_wait comm=trace-cmd pid=2190 delay=118
2 1 2541.343203 migrate 2191 d... sched_switch trace-cmd:2191 [120] D ==> trace-
3 1 2541.353781 <idle> 0 d.h. sched_wakeup trace-cmd:2191 [120] success=1 CP
4 1 2541.353789 <idle> 0 d... sched_switch swapper/1:0 [120] R ==> trace-cmd
5 1 2541.354041 migrate 2191 d... sched_stat_runtime comm=trace-cmd pid=2191 runtime=2
6 1 2541.354055 migrate 2191 d... sched_switch trace-cmd:2191 [120] D|W ==> swap
7 1 2541.368877 <idle> 0 d... sched_switch swapper/1:0 [120] R ==> migrate:2
overriding event (13) ftrace:blktrace with new print handler
1000000
950000
rostedt@neon:/proc/sys/kernel$

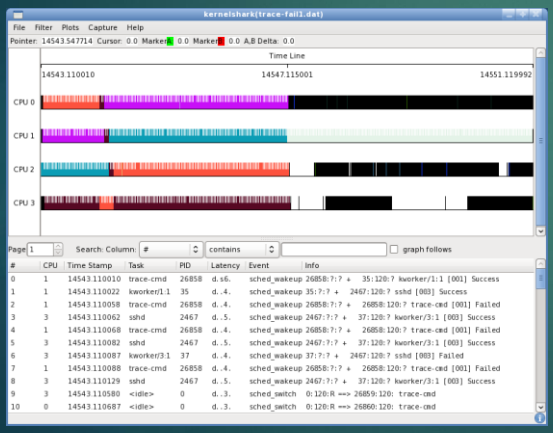
31

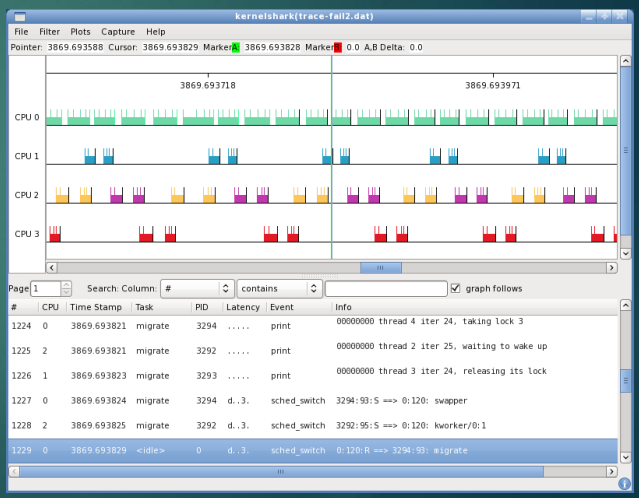

Pointer: 428220.421191 Cursor: 428220.421189 Marker: 428220.421160 Marker: 428220.405326 A,B Delta: -0.015834
function
d..10
vtime_account_irq_enter <-- irq_enter
428220.421191 <idle>
CPU 0
CPU 10
<idle>-0
rcuop/10-24
ksoftirqd/10-121
testLatency-9862
contend on timekeeper_lock
tick_do_update_jiffies64()
device IRQ handler->
eventfd_signal()
return from
blocking read()
smp_apic_timer_interrupt()
Page 1 Search: Column: Info contains eventfd graph follows
CPU Time Stamp Task PID Latency Event Info
203080 10 428220.421187 testLatency 9862 d..30 function add_preempt_count <-- arch_vtime_task_switch
203081 10 428220.421187 testLatency 9862 d..40 function sub_preempt_count <-- arch_vtime_task_switch
203082 10 428220.421187 testLatency 9862 d..30 function sub_preempt_count <-- arch_vtime_task_switch
203083 10 428220.421187 testLatency 9862 d..20 function _raw_spin_unlock_irq <-- finish_task_switch
203084 10 428220.421187 testLatency 9862 ...20 function sub_preempt_count <-- _raw_spin_unlock_irq
203085 10 428220.421187 testLatency 9862 ...10 function tick_nohz_task_switch <-- __schedule
203086 10 428220.421187 testLatency 9862 d..10 function tick_nohz_full_cpu <-- tick_nohz_task_switch
203087 10 428220.421188 testLatency 9862 ...10 function sub_preempt_count <-- __schedule
203088 10 428220.421188 testLatency 9862 ....0 function migrate_disable <-- eventfd_ctx_read
203089 10 428220.421188 testLatency 9862 ....0 function add_preempt_count <-- migrate_disable
203090 10 428220.421188 testLatency 9862 ...11 function sub_preempt_count <-- migrate_disable
203091 10 428220.421188 testLatency 9862 ....1 function rt_spin_lock <-- eventfd_ctx_read
203092 10 428220.421188 testLatency 9862 ....1 function rt_spin_unlock <-- eventfd_ctx_read
203093 10 428220.421189 testLatency 9862 ....1 function migrate_enable <-- eventfd_ctx_read
203094 10 428220.421189 testLatency 9862 ....1 function add_preempt_count <-- migrate_enable
203095 10 428220.421189 testLatency 9862 ...10 function sub_preempt_count <-- migrate_enable
203096 10 428220.421189 testLatency 9862 ....0 function __fsnotify_parent <-- vfs_read
203097 10 428220.421189 testLatency 9862 ....0 function fsnotify <-- vfs_read
203098 10 428220.421189 testLatency 9862 ....0 function syscall_trace_leave <-- int_check_syscall_exit_work
203099 10 428220.421189 testLatency 9862 ....0 function user_exit <-- syscall_trace_leave
203100 10 428220.421190 testLatency 9862 ....0 function user_enter <-- syscall_trace_leave
203101 10 428220.421190 testLatency 9862 d...0 function vtime_user_enter <-- user_enter
203102 0 428220.421190 <idle> 0 d..10 function smp_apic_timer_interrupt <-- apic_timer_interrupt
203103 10 428220.421190 testLatency 9862 d...0 function _raw_spin_lock <-- vtime_user_enter
203104 0 428220.421190 <idle> 0 d..10 function irq_enter <-- smp_apic_timer_interrupt
203105 10 428220.421190 testLatency 9862 d...0 function add_preempt_count <-- _raw_spin_lock
203106 0 428220.421190 <idle> 0 d..10 function rcu_irq_enter <-- irq_enter
203107 10 428220.421190 testLatency 9862 d..10 function add_preempt_count <-- vtime_user_enter

32

Using KernelShark to analyze the real-time scheduler

- https://lwn.net/Articles/425583/
- Steven Rostedt的文章





33

资源

- LWN two-part article on ftrace basics.
- Roasted's LWN.net article on how to use kernelshark.
- Documentation distributed with the Linux source code. Look in Documentation/trace.
- Roasted's LWN article on trace-cmd.

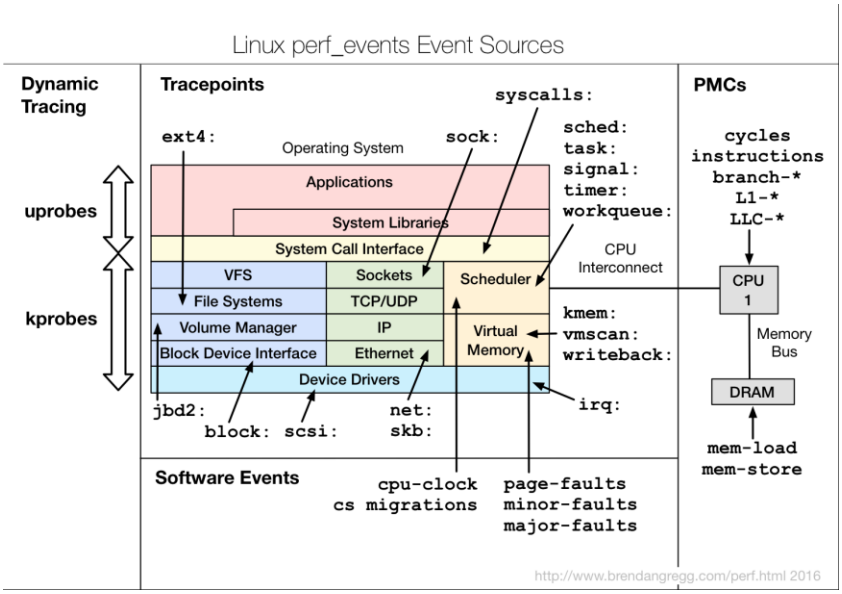
34

Perf events

- perf Linux profiler
- Performance Counters for Linux (PCL), Linux perf events (LPE), or perf_events
- Both ftrace and perf are core Linux tracing tools, included in the kernel source. Your system probably has ftrace already, and perf is often just a package add.

- 安装linux-tools-common
- 或者自己构建
  - Source code is in Linux: tools/perf

35



36

构建

- cd linux-3.12.2/kernel/tools/perf
- sudo apt-get install flex
- sudo apt-get install bison
- make

```
  CC tests/hists_link.o
  CC tests/python-use.o
  CC tests/bp_signal.o
  CC tests/bp_signal_ov      w.o
  CC tests/task-exit.o
  CC tests/sw-clock.o
  CC tests/perf-time-to-tsc.o
  CC tests/code-reading.o
  CC tests/sample-parsing.o
  CC tests/parse-no-sample-id-all.o
  CC arch/x86/util/header.o
  CC arch/x86/util/tsc.o
  CC util/symbol-minimal.o
  CC tests/keep-tracking.o
  AR libperf.a
  SUBDIR /home/ge/work/linux-3.12.2/t
  CC debugfs.o
  AR liblk.a
  SUBDIR /home/ge/work/linux-3.12.2/t
  * new build flags or cross compiler
CC FPIC            event-parse.o
CC FPIC            trace-seq.o
CC FPIC            parse-filter.o
CC FPIC            parse-utils.o
CC FPIC            kbuffer-parse.o
BUILD STATIC LIB   libtraceevent.a
  LINK perf
  GEN perf-archive
```

37

```
perf [--version] [--help] COMMAND [ARGS]

The most commonly used perf commands are:
  annotate        Read perf.data (created by perf record) and display annotated code
  archive         Create archive with object files with build-ids found in perf.data file
  bench           General framework for benchmark suites
  buildid-cache   Manage build-id cache.
  buildid-list    List the buildids in a perf.data file
  diff            Read perf.data files and display the differential profile
  evlist          List the event names in a perf.data file
  inject          Filter to augment the events stream with additional information
  kmem            Tool to trace/measure kernel memory(slab) properties
  kvm             Tool to trace/measure kvm guest os
  list            List all symbolic event types
  lock            Analyze lock events
  mem             Profile memory accesses
  record          Run a command and record its profile into perf.data
  report          Read perf.data (created by perf record) and display the profile
  sched           Tool to trace/measure scheduler properties (latencies)
  script          Read perf.data (created by perf record) and display trace output
  stat            Run a command and gather performance counter statistics
  test            Runs sanity tests.
  timechart       Tool to visualize total system behavior during a workload
  top             System profiling tool.
  trace           strace inspired tool
```

38

```
ge@gewubox:~/work/linux-3.12.2/tools/perf$ ./perf list

List of pre-defined events (to be used in -e):
  cpu-clock                                          [Software event]
  task-clock                                         [Software event]
  page-faults OR faults                              [Software event]
  context-switches OR cs                             [Software event]
  cpu-migrations OR migrations                       [Software event]
  minor-faults                                       [Software event]
  major-faults                                       [Software event]
  alignment-faults                                   [Software event]
  emulation-faults                                   [Software event]
  dummy                                              [Software event]

  rNNN                                               [Raw hardware event descriptor]
  cpu/t1=v1[,t2=v2,t3 ...]/modifier                  [Raw hardware event descriptor]
   (see 'man perf-list' on how to encode it)

  mem:<addr>[:access]                                [Hardware breakpoint]
```

39

perf record -e page-faults -a

```
ge@gewubox:/tmp/kshark$ sudo perf report
# ========
# captured on: Thu Aug 17 17:51:48 2017
# hostname : gewubox
# os release : 3.12.2
# perf version : 3.12.2
# arch : i686
# nrcpus online : 2
# nrcpus avail : 2
# cpudesc : Intel(R) Core(TM) i5-7200U CPU @ 2.50GHz
# cpuid : GenuineIntel,6,142,9
# total memory : 1544896 kB
# cmdline : /usr/bin/perf record -e page-faults -a
# event : name = page-faults, type = 1, config = 0x2, config1 = 0x0, config2 = 0x0, excl_usr = 0, excl_kern
# HEADER_CPU_TOPOLOGY info available, use -I to display
# pmu mappings: software = 1, tracepoint = 2, breakpoint = 5
# ========
#
```

40

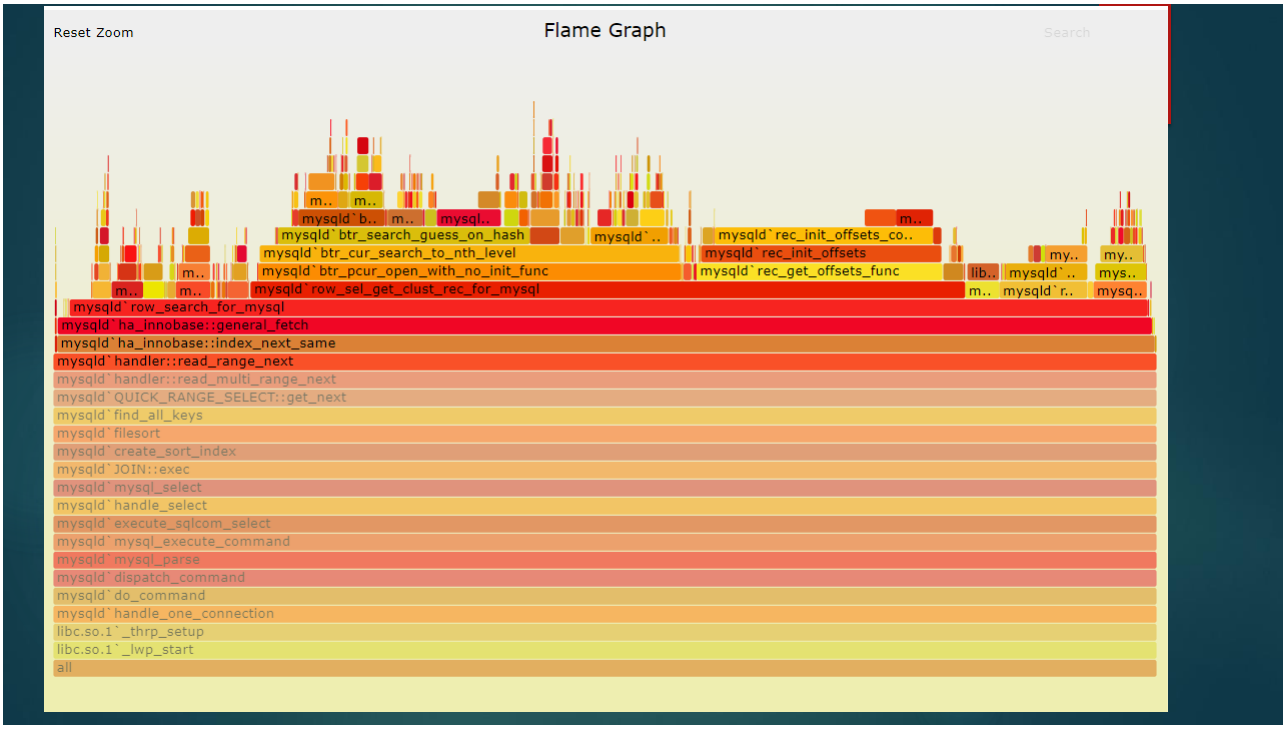
续

```
# Samples: 1K of event 'page-faults'
# Event count (approx.): 115486
#
# Overhead         Command                Shared Object          Symbol
# ........  ..............  ...........................  ..............
#
    82.58%            Xorg  libpixman-1.so.0.30.2        [.] 0x0005f6eb
     4.16%            bash  bash                         [.] 0x00046040
     2.20%            bash  libc-2.15.so                 [.] 0x000768db
     1.36%             sed  libc-2.15.so                 [.] 0x000d9910
     1.23%            bash  ld-2.15.so                  [.] 0x0000b9d9
     1.13%             sed  ld-2.15.so                  [.] 0x00005048
     0.71%  gnome-terminal  ld-2.15.so                  [.] 0x00018907
     0.66%  gnome-terminal  libc-2.15.so                [.] 0x00137331
     0.63%             sed  bash                        [.] 0x0008b0b0
     0.48%  gnome-terminal  libcairo.so.2.11000.2       [.] 0x00042725
     0.37%              ls  libc-2.15.so                [.] 0x0007d260
     0.31%            Xorg  libc-2.15.so                [.] 0x00077341
     0.27%        metacity  libc-2.15.so                [.] 0x00074e22
     0.26%  gnome-terminal  libgobject-2.0.so.0.3200.4  [.] 0x0000cbbc
     0.21%             sed  [kernel.kallsyms]           [k] 0xc132b138
     0.19%           uname  ld-2.15.so                  [.] 0x0000e180
     0.18%             sed  libselinux.so.1             [.] 0x00015930
     0.17%              ls  ld-2.15.so                  [.] 0x000112f0
     0.15%         logging  ld-2.15.so                  [.] 0x00009467
```

41

```
root@gewubox:/tmp# perf record -e page-faults -a
^C[ perf record: Woken up 1 times to write data ]
[ perf record: Captured and wrote 0.469 MB perf.data (~20500 samples) ]
root@gewubox:/tmp# ls
at-spi2          perf.data           pulse-PKdhtXMmr18n  ssh-HGRLEuYw1735
keyring-UHcMhw   pulse-2L9K88eMlGn7  pulse-sYOmDRVLhJlw  unity_support_test.1
root@gewubox:/tmp# sudo perf report
# ========
# captured on: Fri Dec  1 21:50:47 2017
# hostname : gewubox
# os release : 3.12.2
# perf version : 3.12.2
# arch : i686
# nrcpus online :
# nrcpus avail :
# cpudesc : Intel(
# cpuid : GenuineIntel,6,142,9
# total memory : 1544952 kB
# cmdline : /usr/bin/perf record -e page-faults -a
# event : name = page-faults, type = 1, config = 0x2, config1 = 0x0, config2 = 0x0, excl_usr = 0, excl_kern = 0, excl
# HEADER_CPU_TOPOLOGY info available, use -I to display
# pmu mappings: software = 1, tracepoint = 2, breakpoint = 5
# ========
#
# Samples: 49  of event 'page-faults'
# Event count (approx.): 5533
#
# Overhead   Command      Shared Object       Symbol
# ........  .......  .................  ..............
#
    95.57%     Xorg  libpixman-1.so.0.30.2  [.] 0x0005f6f5
     3.65%  gemalloc  [kernel.kallsyms]      [.] 0x00076b1e
     0.58%  gemalloc  libcairo.so.2.11000.2  [.] 0x0000cfbe
     0.14%     perf  libc-2.15.so            [.] 0x000e3080
```


42



43

Try it

```
# git clone https://github.com/brendangregg/FlameGraph  # or download it from github
# cd FlameGraph
# perf record -F 99 -a -g -- sleep 60
# perf script | ./stackcollapse-perf.pl > out.perf-folded
# ./flamegraph.pl out.perf-folded > perf-kernel.svg

ge@gewubox:~/work/FlameGraph$ sudo perf record -F 99 -a -g -- sleep 60
[ perf record: Woken up 5 times to write data ]
[ perf record: Captured and wrote 1.759 MB perf.data (~76871 samples) ]
[vboxguest] with build id 90d523cfb7053db5f4024d1c38d2e9a2b7bfa50e not found, continuing without
ge@gewubox:~/work/FlameGraph$ perf script | ./stackcollapse-perf.pl > out.perf_folded
failed to open perf.data: Permission denied
ge@gewubox:~/work/FlameGraph$ sudo perf script | ./stackcollapse-perf.pl > out.perf_folded
Filtering for events of type: cpu-clock
ge@gewubox:~/work/FlameGraph$ ./flamegraph.pl out.perf_folded > perf-kernel.svg
ge@gewubox:~/work/FlameGraph$ firefox perf-kernel.svg
```

44

切问而近思

欢迎关注格友公众号

45