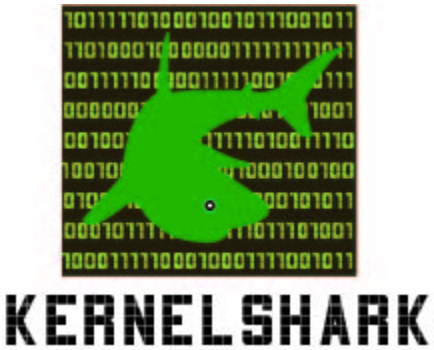
Swimming with the New KernelShark

Yordan Karadzhov

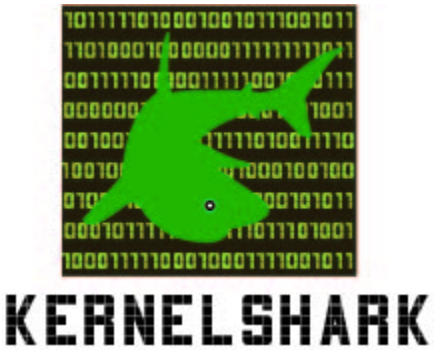
VMware Inc. - OSTC

What is KernelShark?



* **Front end reader of Linux kernel tracing data (Ftrace)**

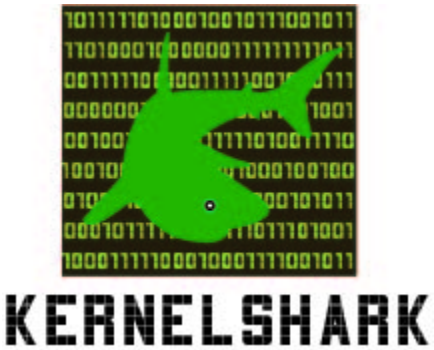
What is KernelShark?



* **Front end reader of Linux kernel tracing data (Ftrace)**

* The original version - started in 2009.
* Written in *Gtk+-2.0*
* Main goal: analyse and fully understood the performance of the Real-time scheduler.

What is KernelShark?



* **Front end reader of Linux kernel tracing data (Ftrace)**

* **New KernelShark**: use all lessons learned from the old version.
* Completely rewritten to use Qt.
* But not only this ...

The New KernelShark is

a. Optimized for processing significantly larger amounts of data.

b. New scalable data model - log(n) time complexity.

c. OpenGL-based visualization.

d. Preconfigurable - Json config I/O.

e. User modifiable - plugins.

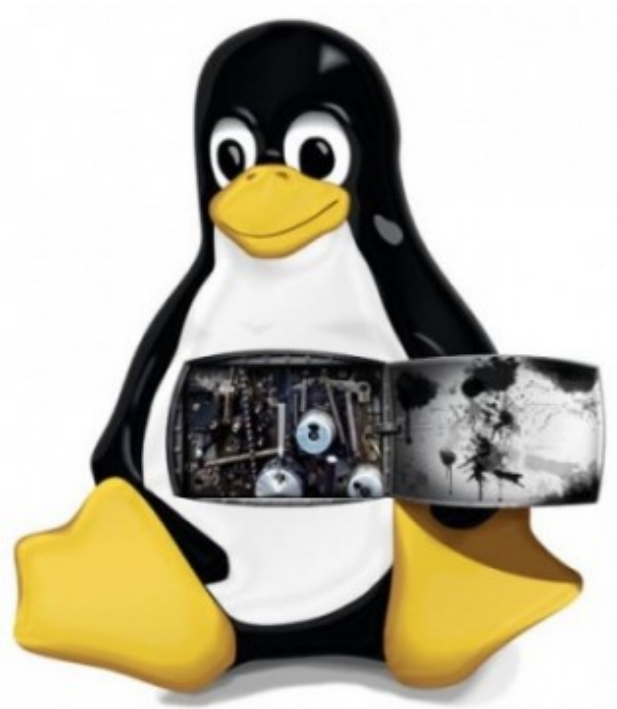
Why do we need KernelShark?

a. Ftrace - the official tracing infrastructure of the Linux kernel.

b. Extremely powerful instrument.

c. Allows to see what is happening in the kernel.

d. But you must know what you are looking for



Ftrace is extremely powerful but ...

```
     Compositor-1124  [001] .... 236364.639085: _cond_resched <-futex_wait_queue_me
     Compositor-1124  [001] .... 236364.639086: rcu_all_qs <-_cond_resched
chromium-browse-1117  [000] d... 236364.639086: do_syscall_64 <-entry_SYSCALL_64_after_hwframe
     Compositor-1124  [001] .... 236364.639086: drop_futex_key_refs.isra.14 <-futex_wait
chromium-browse-1117  [000] .... 236364.639087: syscall_trace_enter <-do_syscall_64
     Compositor-1124  [001] .... 236364.639087: hrtimer_cancel <-futex_wait
chromium-browse-1117  [000] .... 236364.639087: __secure_computing <-syscall_trace_enter
     Compositor-1124  [001] .... 236364.639087: hrtimer_try_to_cancel <-hrtimer_cancel
chromium-browse-1117  [000] .... 236364.639087: __seccomp_filter <-__secure_computing
     Compositor-1124  [001] .... 236364.639088: hrtimer_active <-hrtimer_try_to_cancel
chromium-browse-1117  [000] .... 236364.639088: seccomp_run_filters <-__seccomp_filter
     Compositor-1124  [001] .... 236364.639088: lock_hrtimer_base.isra.20 <-hrtimer_try_to_cancel
     Compositor-1124  [001] .... 236364.639088: _raw_spin_lock_irqsave <-lock_hrtimer_base.isra.20
chromium-browse-1117  [000] .... 236364.639088: SyS_futex <-do_syscall_64
     Compositor-1124  [001] d... 236364.639089: __remove_hrtimer <-hrtimer_try_to_cancel
     Compositor-1124  [001] d... 236364.639089: _raw_spin_unlock_irqrestore <-hrtimer_try_to_cancel
chromium-browse-1117  [000] .... 236364.639089: ktime_get <-SyS_futex
chromium-browse-1117  [000] .... 236364.639090: ktime_add_safe <-SyS_futex
chromium-browse-1117  [000] .... 236364.639090: do_futex <-SyS_futex
chromium-browse-1117  [000] .... 236364.639091: futex_wait <-do_futex
chromium-browse-1117  [000] .... 236364.639091: hrtimer_init <-futex_wait
chromium-browse-1117  [000] .... 236364.639092: __hrtimer_init <-hrtimer_init
chromium-browse-1117  [000] .... 236364.639092: hrtimer_init_sleeper <-futex_wait
chromium-browse-1117  [000] .... 236364.639092: ktime_add_safe <-futex_wait
chromium-browse-1117  [000] .... 236364.639093: futex_wait_setup <-futex_wait
chromium-browse-1117  [000] .... 236364.639093: get_futex_key <-futex_wait_setup
chromium-browse-1117  [000] .... 236364.639093: get_futex_key_refs.isra.13 <-get_futex_key
chromium-browse-1117  [000] .... 236364.639094: hash_futex <-futex_wait_setup
chromium-browse-1117  [000] .... 236364.639095: _raw_spin_lock <-futex_wait_setup
chromium-browse-1117  [000] .... 236364.639095: get_futex_value_locked <-futex_wait_setup
chromium-browse-1117  [000] .... 236364.639095: futex_wait_queue_me <-futex_wait
chromium-browse-1117  [000] .... 236364.639096: hrtimer_start_range_ns <-futex_wait_queue_me
     Compositor-1124  [001] d... 236364.639097: do_syscall_64 <-entry_SYSCALL_64_after_hwframe
chromium-browse-1117  [000] .... 236364.639097: lock_hrtimer_base.isra.20 <-hrtimer_start_range_ns
```

Ftrace is extremely powerful but ...

```
     Compositor-1124  [001] .... 236364.639085: _cond_resched <-futex_wait_queue_me
     Compositor-1124  [001] .... 236364.639086: rcu_all_qs <-_cond_resched
chromium-browse-1117  [000] d... 236364.639086: do_syscall_64 <-entry_SYSCALL_64_after_hwframe
```

... it can be hard to find what you need. You must be **True (kernel) detective**.

```
     Compositor-1124  [001] .... 236364.639088: hrtimer_active <-hrtimer_try_to_cancel
chromium-browse-1117  [000] .... 236364.639088: seccomp_run_filters <-__seccomp_filter
     Compositor-1124  [001] .... 236364.639088: lock_hrtimer_base.isra.20 <-hrtimer_try_to_cancel
     Compositor-1124  [001] .... 236364.639088: _raw_spin_lock_irqsave <-lock_hrtimer_base.isra.20
chromium-browse-1117  [000] .... 236364.639088: SyS_futex <-do_syscall_64
     Compositor-1124  [001] d... 236364.639089: __remove_hrtimer <-hrtimer_try_to_cancel
     Compositor-1124  [001] d... 236364.639089: _raw_spin_unlock_irqrestore <-hrtimer_try_to_cancel
chromium-browse-1117  [000] .... 236364.639089: ktime_get <-SyS_futex
chromium-browse-1117  [000] .... 236364.639090: ktime_add_safe <-SyS_futex
chromium-browse-1117  [000] .... 236364.639090: do_futex <-SyS_futex
chromium-browse-1117  [000] .... 236364.639091: futex_wait <-do_futex
chromium-browse-1117  [000] .... 236364.639091: hrtimer_init <-futex_wait
chromium-browse-1117  [000] .... 236364.639092: __hrtimer_init <-hrtimer_init
chromium-browse-1117  [000] .... 236364.639092: hrtimer_init_sleeper <-futex_wait
chromium-browse-1117  [000] .... 236364.639092: ktime_add_safe <-futex_wait
chromium-browse-1117  [000] .... 236364.639093: futex_wait_setup <-futex_wait
chromium-browse-1117  [000] .... 236364.639093: get_futex_key <-futex_wait_setup
chromium-browse-1117  [000] .... 236364.639093: get_futex_key_refs.isra.13 <-get_futex_key
chromium-browse-1117  [000] .... 236364.639094: hash_futex <-futex_wait_setup
chromium-browse-1117  [000] .... 236364.639095: _raw_spin_lock <-futex_wait_setup
chromium-browse-1117  [000] .... 236364.639095: get_futex_value_locked <-futex_wait_setup
chromium-browse-1117  [000] .... 236364.639095: futex_wait_queue_me <-futex_wait
chromium-browse-1117  [000] .... 236364.639096: hrtimer_start_range_ns <-futex_wait_queue_me
     Compositor-1124  [001] d... 236364.639097: do_syscall_64 <-entry_SYSCALL_64_after_hwframe
chromium-browse-1117  [000] .... 236364.639097: lock_hrtimer_base.isra.20 <-hrtimer_start_range_ns
```

Ftrace is extremely powerful but ...

```
     Compositor-1124  [001] .... 236364.639085: _cond_resched <-futex_wait_queue_me
     Compositor-1124  [001] .... 236364.639086: rcu_all_qs <-_cond_resched
chromium-browse-1117  [000] d... 236364.639086: do_syscall_64 <-entry_SYSCALL_64_after_hwframe
```

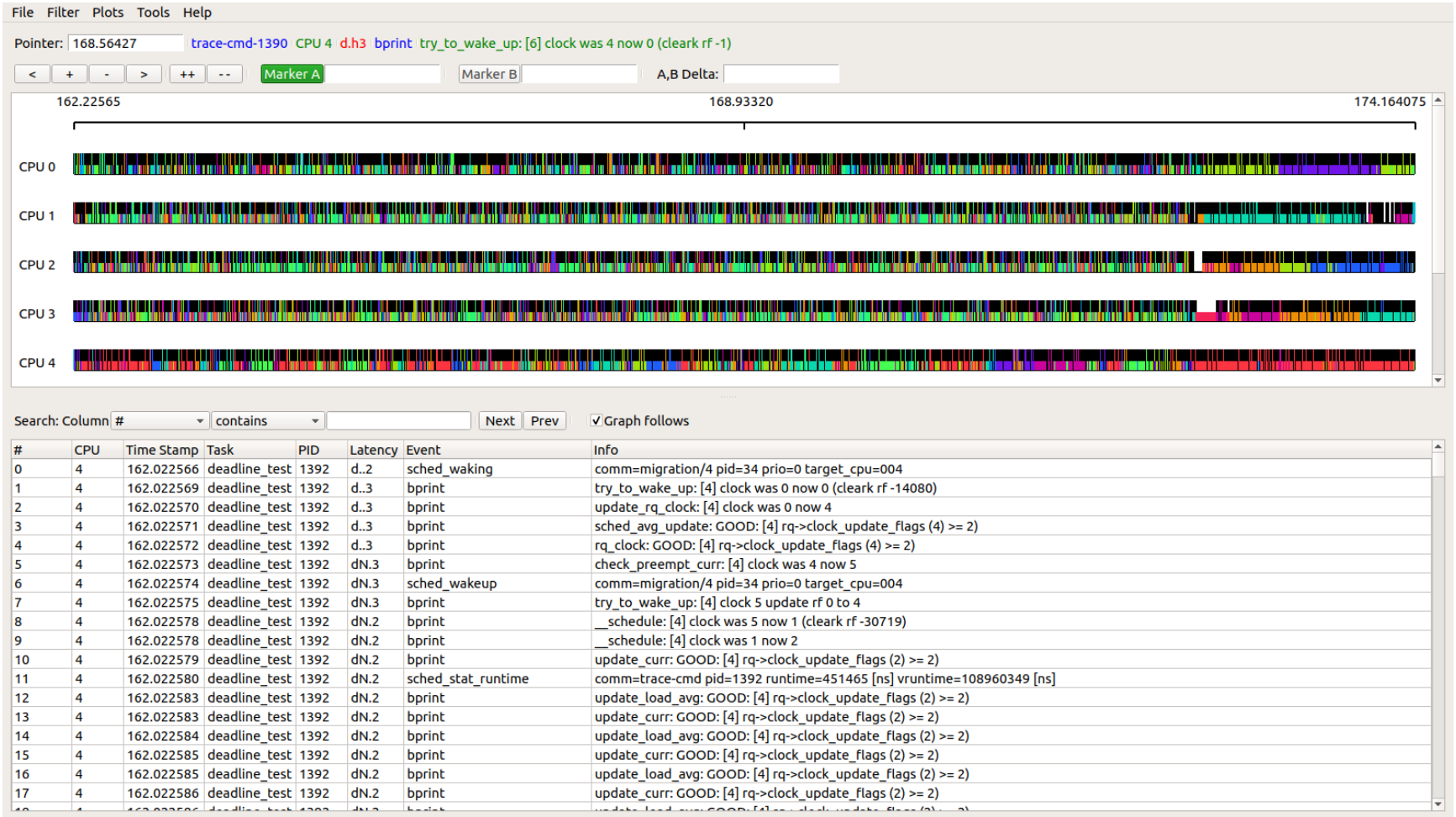
... it can be hard to find what you need. You must be **True (kernel) detective**.

```
     Compositor-1124  [001] .... 236364.639088: hrtimer_active <-hrtimer_try_to_cancel
chromium-browse-1117  [000] .... 236364.639088: seccomp_run_filters <-__seccomp_filter
     Compositor-1124  [001] .... 236364.639088: lock_hrtimer_base.isra.20 <-hrtimer_try_to_cancel
     Compositor-1124  [001] .... 236364.639088: _raw_spin_lock_irqsave <-lock_hrtimer_base.isra.20
chromium-browse-1117  [000] .... 236364.639088: SyS_futex <-do_syscall_64
     Compositor-1124  [001] d... 236364.639089: __remove_hrtimer <-hrtimer_try_to_cancel
```

Interactive visualization can be very useful.

```
chromium-browse-1117  [000] .... 236364.639092: ktime_add_safe <-futex_wait
chromium-browse-1117  [000] .... 236364.639093: futex_wait_setup <-futex_wait
chromium-browse-1117  [000] .... 236364.639093: get_futex_key <-futex_wait_setup
chromium-browse-1117  [000] .... 236364.639093: get_futex_key_refs.isra.13 <-get_futex_key
chromium-browse-1117  [000] .... 236364.639094: hash_futex <-futex_wait_setup
chromium-browse-1117  [000] .... 236364.639095: _raw_spin_lock <-futex_wait_setup
chromium-browse-1117  [000] .... 236364.639095: get_futex_value_locked <-futex_wait_setup
chromium-browse-1117  [000] .... 236364.639095: futex_wait_queue_me <-futex_wait
chromium-browse-1117  [000] .... 236364.639096: hrtimer_start_range_ns <-futex_wait_queue_me
     Compositor-1124  [001] d... 236364.639097: do_syscall_64 <-entry_SYSCALL_64_after_hwframe
chromium-browse-1117  [000] .... 236364.639097: lock_hrtimer_base.isra.20 <-hrtimer_start_range_ns
```

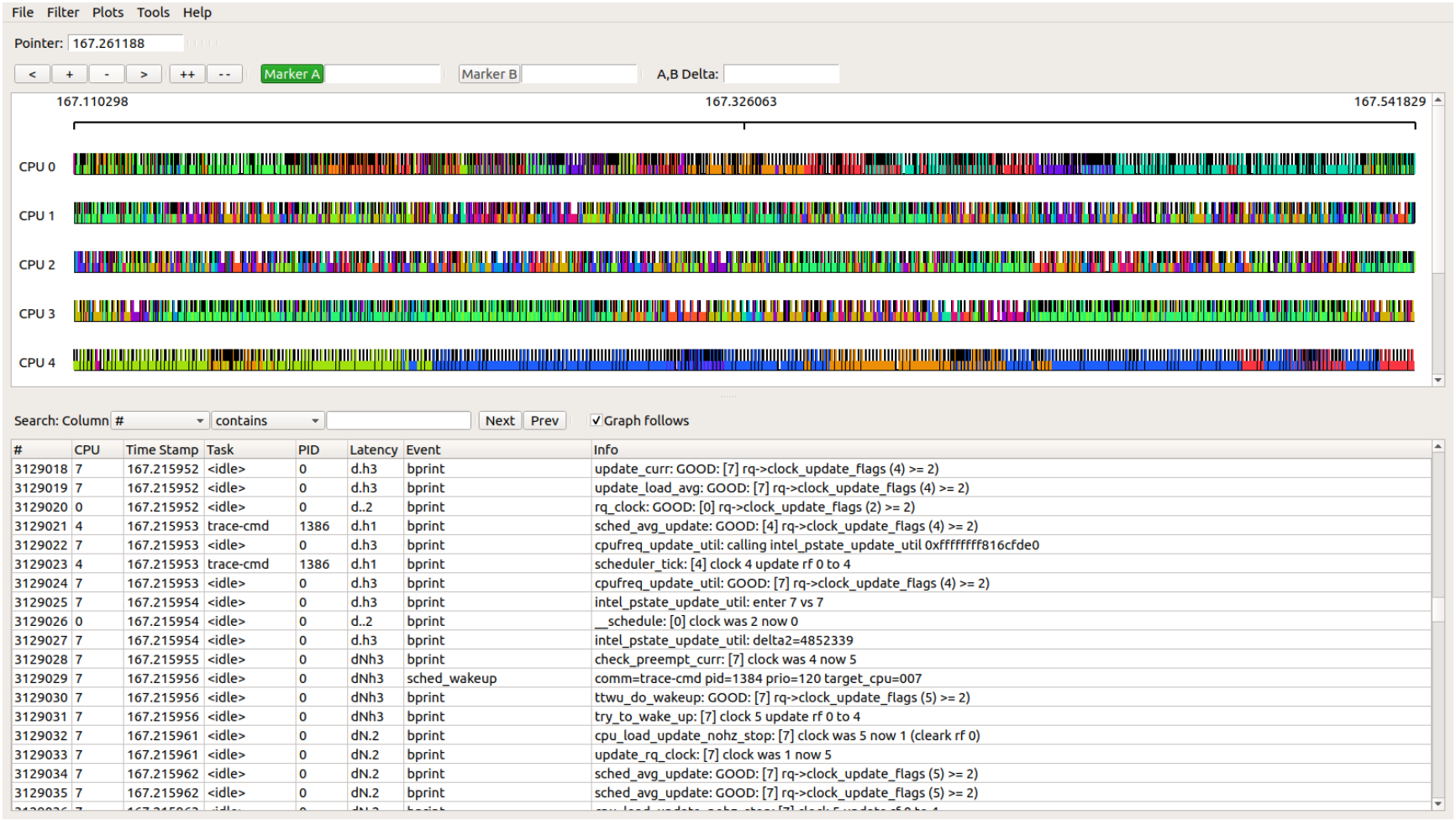
Kernel Shark



#	CPU	Time Stamp	Task	PID	Latency	Event	Info
0	4	162.022566	deadline_test	1392	d..2	sched_waking	comm=migration/4 pid=34 prio=0 target_cpu=004
1	4	162.022569	deadline_test	1392	d..3	bprint	try_to_wake_up: [4] clock was 0 now 0 (cleark rf -14080)
2	4	162.022570	deadline_test	1392	d..3	bprint	update_rq_clock: [4] clock was 0 now 4
3	4	162.022571	deadline_test	1392	d..3	bprint	sched_avg_update: GOOD: [4] rq->clock_update_flags (4) >= 2)
4	4	162.022572	deadline_test	1392	d..3	bprint	rq_clock: GOOD: [4] rq->clock_update_flags (4) >= 2)
5	4	162.022573	deadline_test	1392	dN.3	bprint	check_preempt_curr: [4] clock was 4 now 5
6	4	162.022574	deadline_test	1392	dN.3	sched_wakeup	comm=migration/4 pid=34 prio=0 target_cpu=004
7	4	162.022575	deadline_test	1392	dN.3	bprint	try_to_wake_up: [4] clock 5 update rf 0 to 4
8	4	162.022578	deadline_test	1392	dN.2	bprint	__schedule: [4] clock was 5 now 1 (cleark rf -30719)
9	4	162.022578	deadline_test	1392	dN.2	bprint	__schedule: [4] clock was 1 now 2
10	4	162.022579	deadline_test	1392	dN.2	bprint	update_curr: GOOD: [4] rq->clock_update_flags (2) >= 2)
11	4	162.022580	deadline_test	1392	dN.2	sched_stat_runtime	comm=trace-cmd pid=1392 runtime=451465 [ns] vruntime=108960349 [ns]
12	4	162.022583	deadline_test	1392	dN.2	bprint	update_load_avg: GOOD: [4] rq->clock_update_flags (2) >= 2)
13	4	162.022583	deadline_test	1392	dN.2	bprint	update_curr: GOOD: [4] rq->clock_update_flags (2) >= 2)
14	4	162.022584	deadline_test	1392	dN.2	bprint	update_load_avg: GOOD: [4] rq->clock_update_flags (2) >= 2)
15	4	162.022585	deadline_test	1392	dN.2	bprint	update_curr: GOOD: [4] rq->clock_update_flags (2) >= 2)
16	4	162.022585	deadline_test	1392	dN.2	bprint	update_load_avg: GOOD: [4] rq->clock_update_flags (2) >= 2)
17	4	162.022586	deadline_test	1392	dN.2	bprint	update_curr: GOOD: [4] rq->clock_update_flags (2) >= 2)

Hmm, why do you think that this complete mess of colors can be of any help?

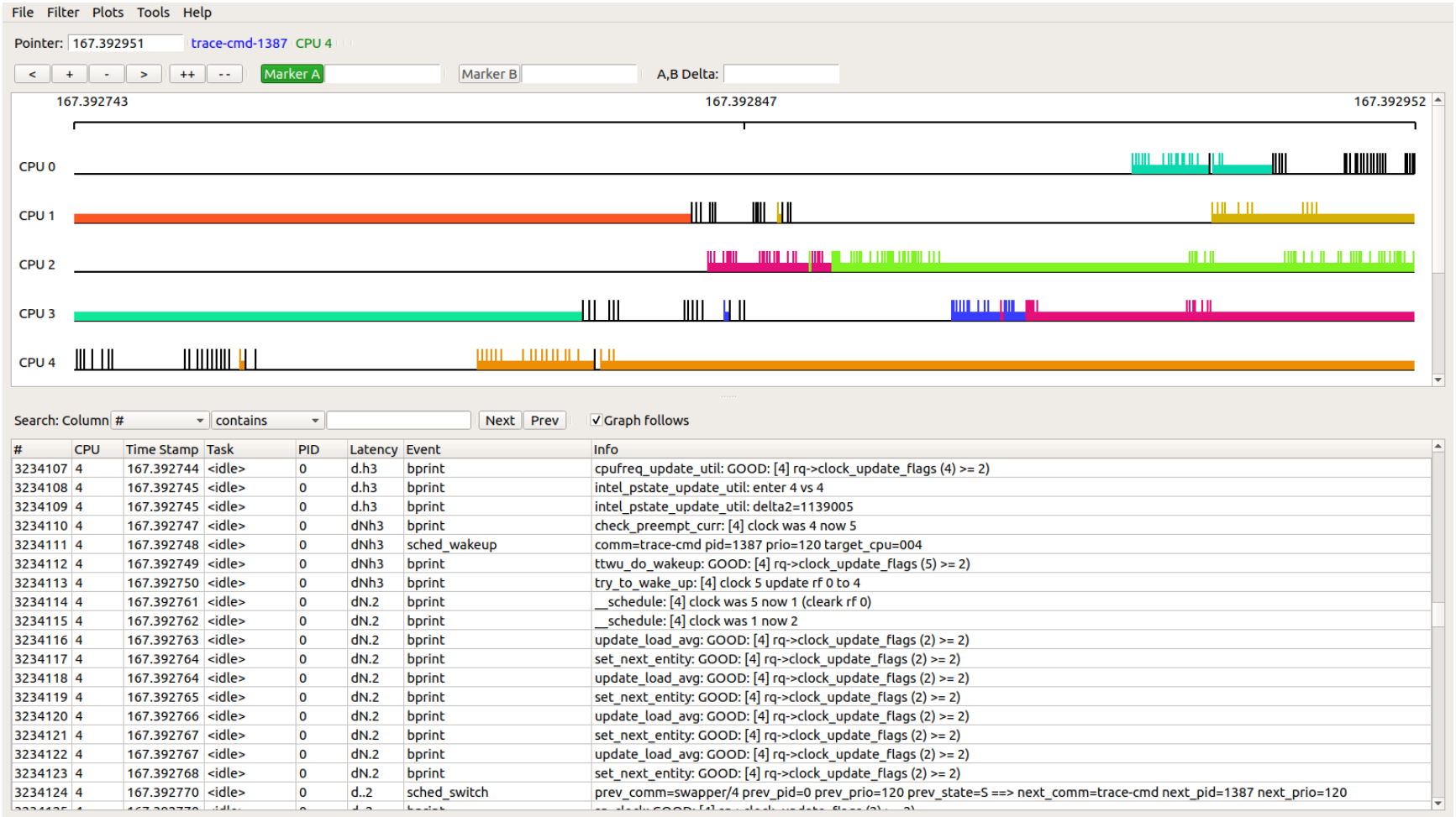
Kernel Shark



#	CPU	Time Stamp	Task	PID	Latency	Event	Info
3129018	7	167.215952	<idle>	0	d.h3	bprint	update_curr: GOOD: [7] rq->clock_update_flags (4) >= 2)
3129019	7	167.215952	<idle>	0	d.h3	bprint	update_load_avg: GOOD: [7] rq->clock_update_flags (4) >= 2)
3129020	0	167.215952	<idle>	0	d..2	bprint	rq_clock: GOOD: [0] rq->clock_update_flags (2) >= 2)
3129021	4	167.215953	trace-cmd	1386	d.h1	bprint	sched_avg_update: GOOD: [4] rq->clock_update_flags (4) >= 2)
3129022	7	167.215953	<idle>	0	d.h3	bprint	cpufreq_update_util: calling intel_pstate_update_util 0xffffffff816cfde0
3129023	4	167.215953	trace-cmd	1386	d.h1	bprint	scheduler_tick: [4] clock 4 update rf 0 to 4
3129024	7	167.215953	<idle>	0	d.h3	bprint	cpufreq_update_util: GOOD: [7] rq->clock_update_flags (4) >= 2)
3129025	7	167.215954	<idle>	0	d.h3	bprint	intel_pstate_update_util: enter 7 vs 7
3129026	0	167.215954	<idle>	0	d..2	bprint	__schedule: [0] clock was 2 now 0
3129027	7	167.215954	<idle>	0	d.h3	bprint	intel_pstate_update_util: delta2=4852339
3129028	7	167.215955	<idle>	0	dNh3	bprint	check_preempt_curr: [7] clock was 4 now 5
3129029	7	167.215956	<idle>	0	dNh3	sched_wakeup	comm=trace-cmd pid=1384 prio=120 target_cpu=007
3129030	7	167.215956	<idle>	0	dNh3	bprint	ttwu_do_wakeup: GOOD: [7] rq->clock_update_flags (5) >= 2)
3129031	7	167.215956	<idle>	0	dNh3	bprint	try_to_wake_up: [7] clock 5 update rf 0 to 4
3129032	7	167.215961	<idle>	0	dN.2	bprint	cpu_load_update_nohz_stop: [7] clock was 5 now 1 (cleark rf 0)
3129033	7	167.215961	<idle>	0	dN.2	bprint	update_rq_clock: [7] clock was 1 now 5
3129034	7	167.215962	<idle>	0	dN.2	bprint	sched_avg_update: GOOD: [7] rq->clock_update_flags (5) >= 2)
3129035	7	167.215962	<idle>	0	dN.2	bprint	sched_avg_update: GOOD: [7] rq->clock_update_flags (5) >= 2)

If we **Zoom in** a bit, we will start seeing the time structure of the trace.

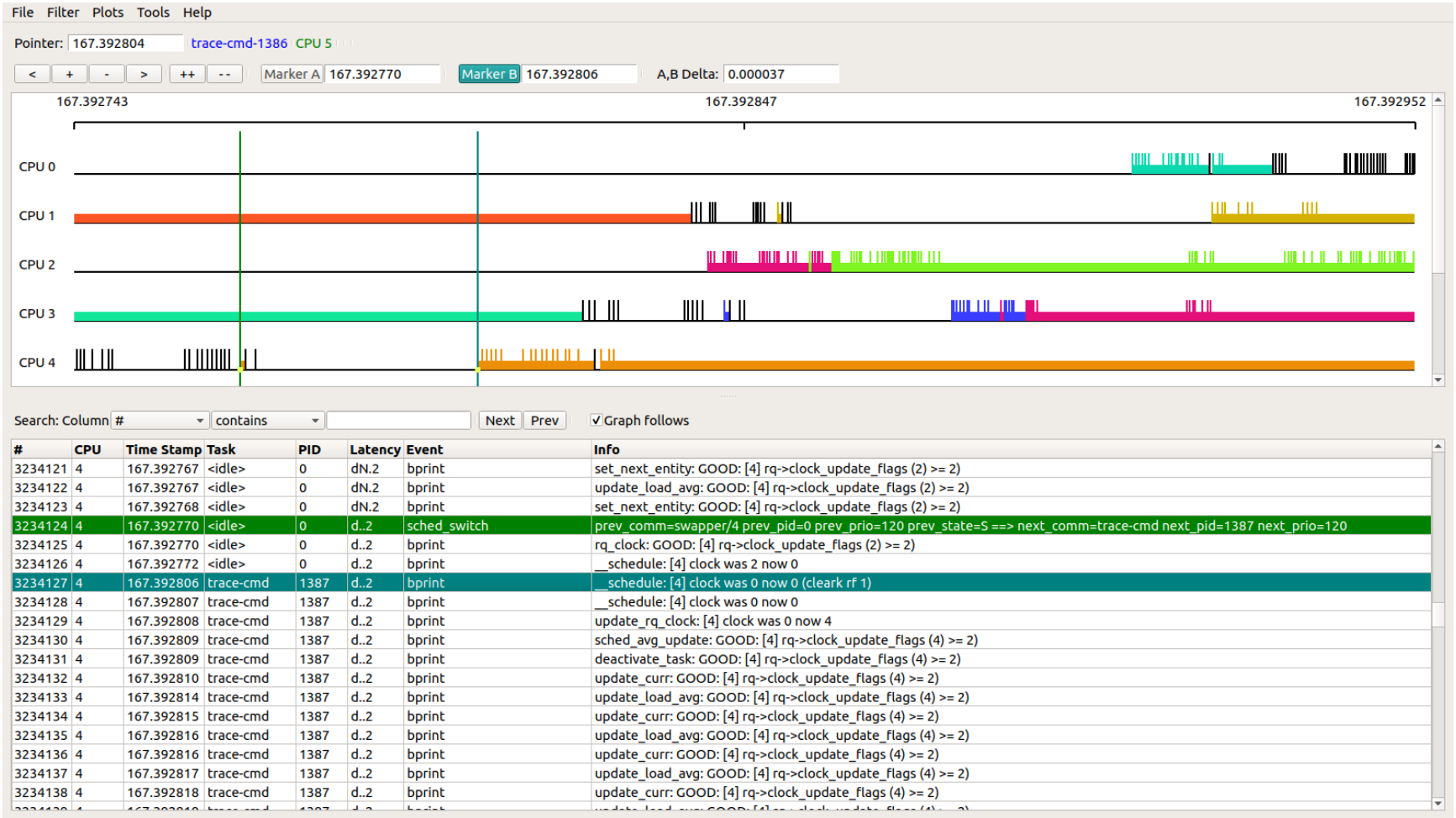
Kernel Shark



#	CPU	Time Stamp	Task	PID	Latency	Event	Info
3234107	4	167.392744	<idle>	0	d.h3	bprint	cpufreq_update_util: GOOD: [4] rq->clock_update_flags (4) >= 2)
3234108	4	167.392745	<idle>	0	d.h3	bprint	intel_pstate_update_util: enter 4 vs 4
3234109	4	167.392745	<idle>	0	d.h3	bprint	intel_pstate_update_util: delta2=1139005
3234110	4	167.392747	<idle>	0	dNh3	bprint	check_preempt_curr: [4] clock was 4 now 5
3234111	4	167.392748	<idle>	0	dNh3	sched_wakeup	comm=trace-cmd pid=1387 prio=120 target_cpu=004
3234112	4	167.392749	<idle>	0	dNh3	bprint	ttwu_do_wakeup: GOOD: [4] rq->clock_update_flags (5) >= 2)
3234113	4	167.392750	<idle>	0	dNh3	bprint	try_to_wake_up: [4] clock 5 update rf 0 to 4
3234114	4	167.392761	<idle>	0	dN.2	bprint	__schedule: [4] clock was 5 now 1 (cleark rf 0)
3234115	4	167.392762	<idle>	0	dN.2	bprint	__schedule: [4] clock was 1 now 2
3234116	4	167.392763	<idle>	0	dN.2	bprint	update_load_avg: GOOD: [4] rq->clock_update_flags (2) >= 2)
3234117	4	167.392764	<idle>	0	dN.2	bprint	set_next_entity: GOOD: [4] rq->clock_update_flags (2) >= 2)
3234118	4	167.392764	<idle>	0	dN.2	bprint	update_load_avg: GOOD: [4] rq->clock_update_flags (2) >= 2)
3234119	4	167.392765	<idle>	0	dN.2	bprint	set_next_entity: GOOD: [4] rq->clock_update_flags (2) >= 2)
3234120	4	167.392766	<idle>	0	dN.2	bprint	update_load_avg: GOOD: [4] rq->clock_update_flags (2) >= 2)
3234121	4	167.392767	<idle>	0	dN.2	bprint	set_next_entity: GOOD: [4] rq->clock_update_flags (2) >= 2)
3234122	4	167.392767	<idle>	0	dN.2	bprint	update_load_avg: GOOD: [4] rq->clock_update_flags (2) >= 2)
3234123	4	167.392768	<idle>	0	dN.2	bprint	set_next_entity: GOOD: [4] rq->clock_update_flags (2) >= 2)
3234124	4	167.392770	<idle>	0	d..2	sched_switch	prev_comm=swapper/4 prev_pid=0 prev_prio=120 prev_state=S ==> next_comm=trace-cmd next_pid=1387 next_prio=120

This is a deep zoom. Let's try to understand what is visualized here.

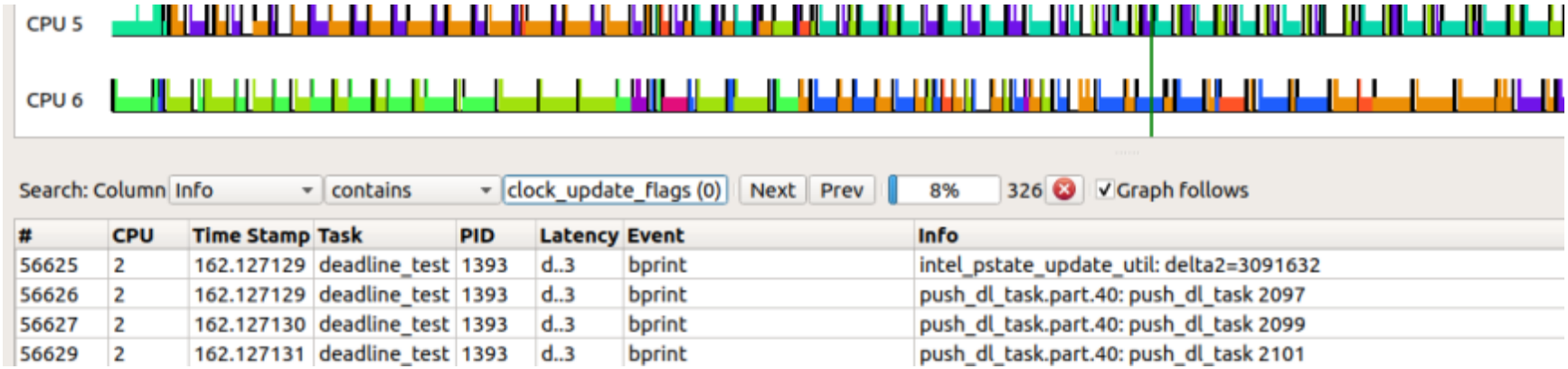
Kernel Shark



a. Double click on the graph selects a trace record.

b. The same can be done by clicking on the table row which shows the record as text.

Kernel Shark



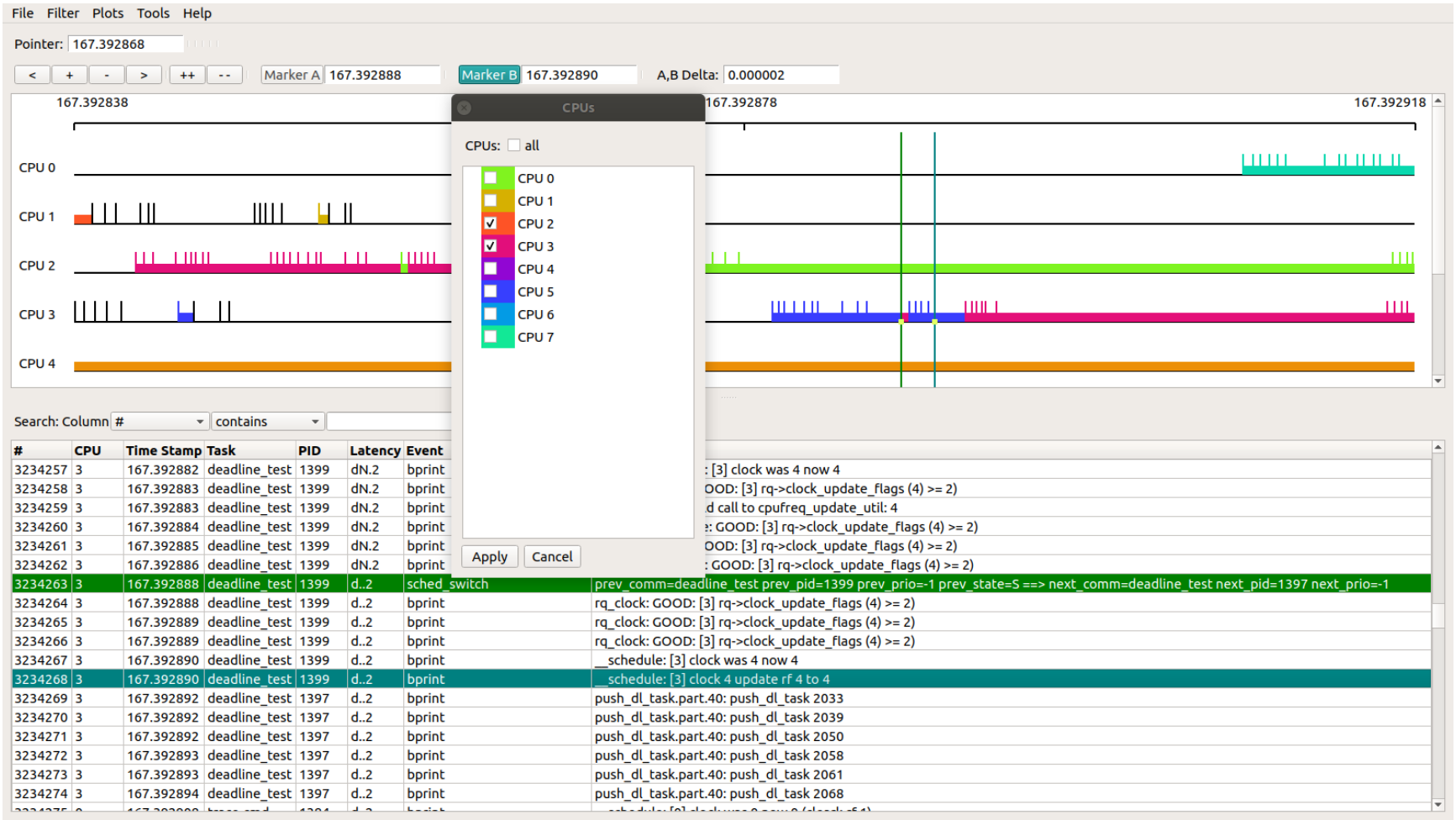
#	CPU	Time Stamp	Task	PID	Latency	Event	Info
56625	2	162.127129	deadline_test	1393	d..3	bprint	intel_pstate_update_util: delta2=3091632
56626	2	162.127129	deadline_test	1393	d..3	bprint	push_dl_task.part.40: push_dl_task 2097
56627	2	162.127130	deadline_test	1393	d..3	bprint	push_dl_task.part.40: push_dl_task 2099
56629	2	162.127131	deadline_test	1393	d..3	bprint	push_dl_task.part.40: push_dl_task 2101

Search panel

a. Provides interactive searching.

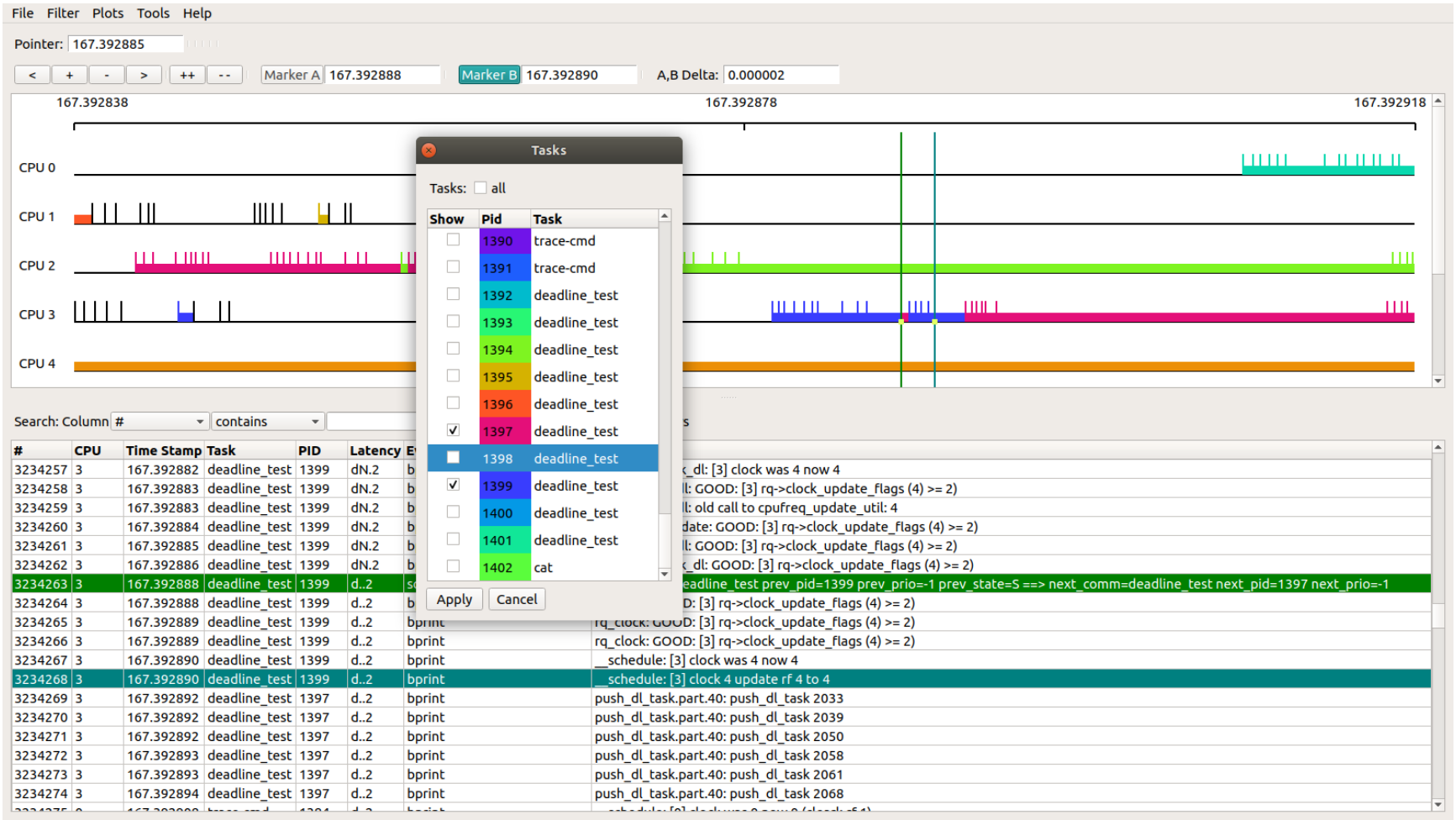
b. Very useful in combination with the Dual marker.

Kernel Shark



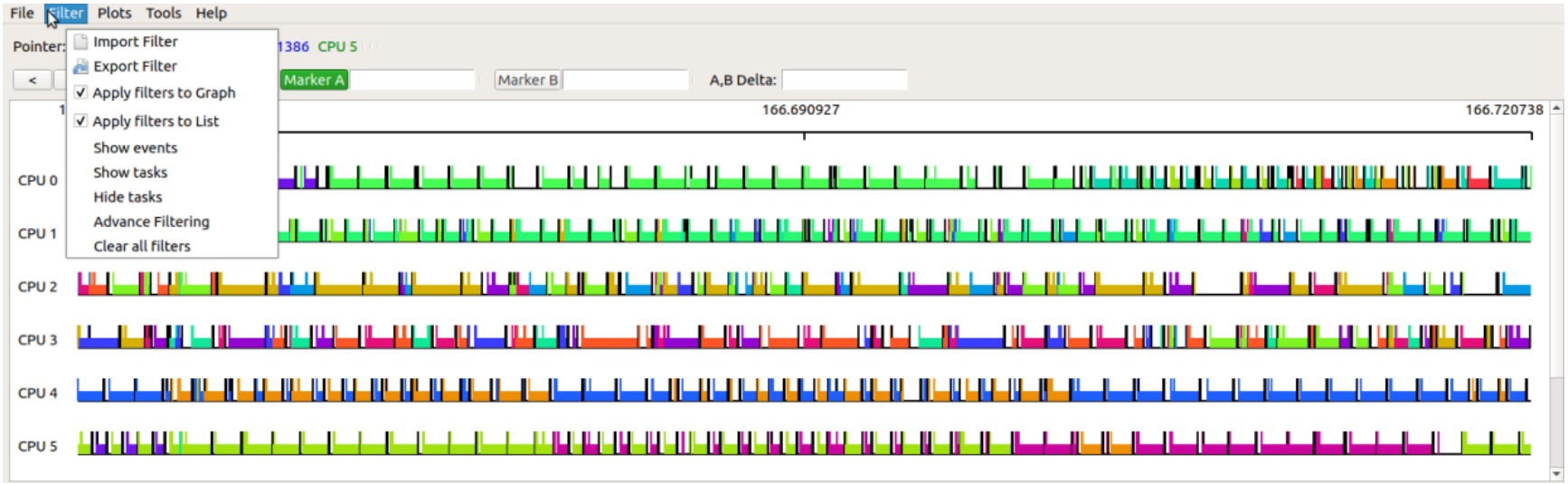
Selecting which CPU to visualize.

Kernel Shark



Selecting which Task to visualize.

Kernel Shark



Filter menu

a. Filter out tasks

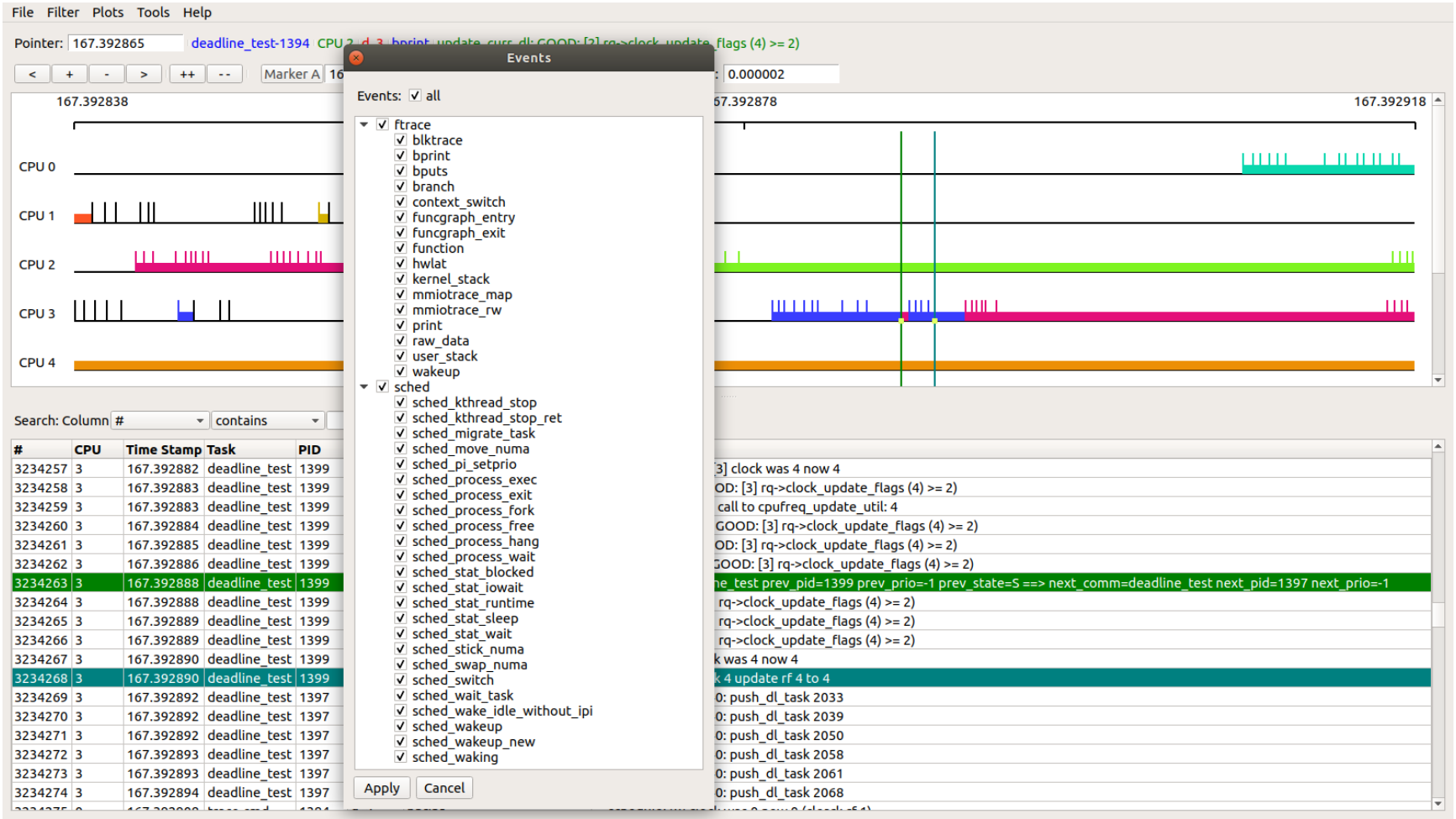
b. Filter in tasks

c. Filter events

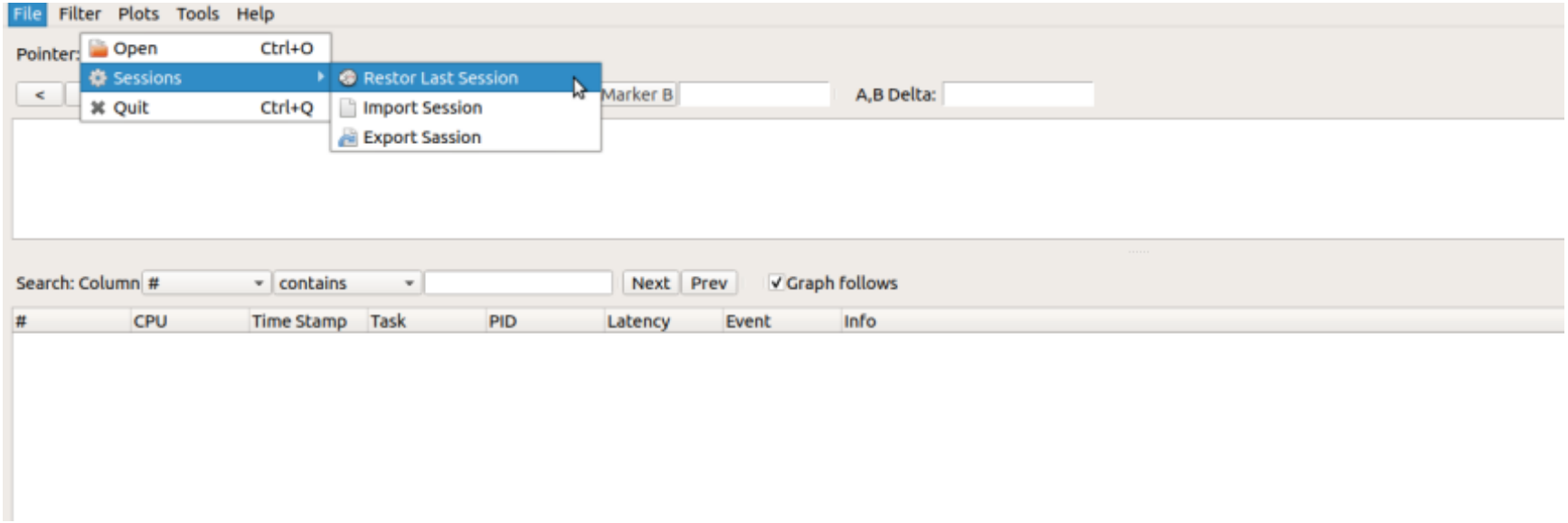
d. Advanced (content-based) filtering

Kernel Shark

Event filters



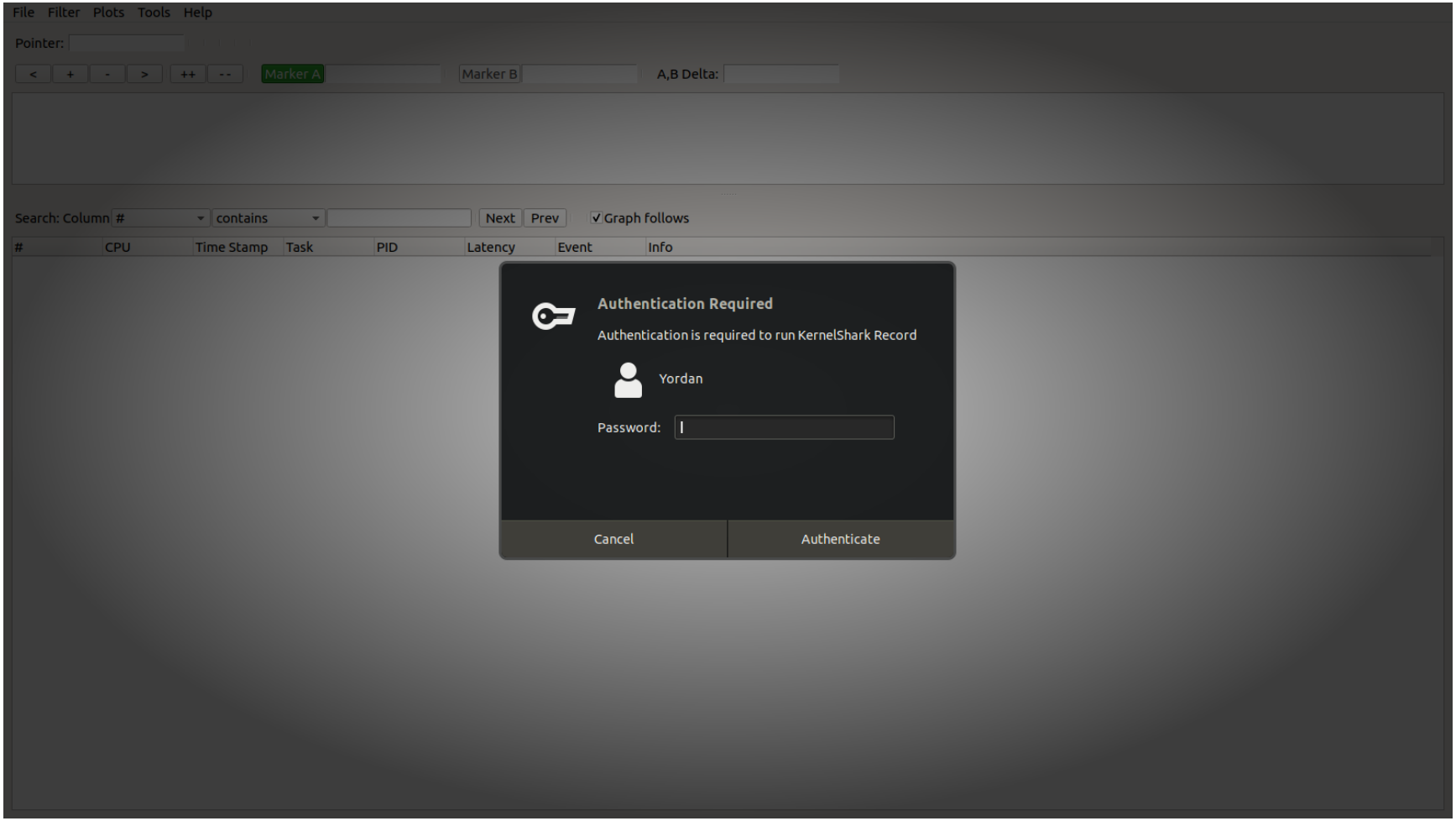
Kernel Shark - Sessions



a. Import/Export session

b. Restor last session.

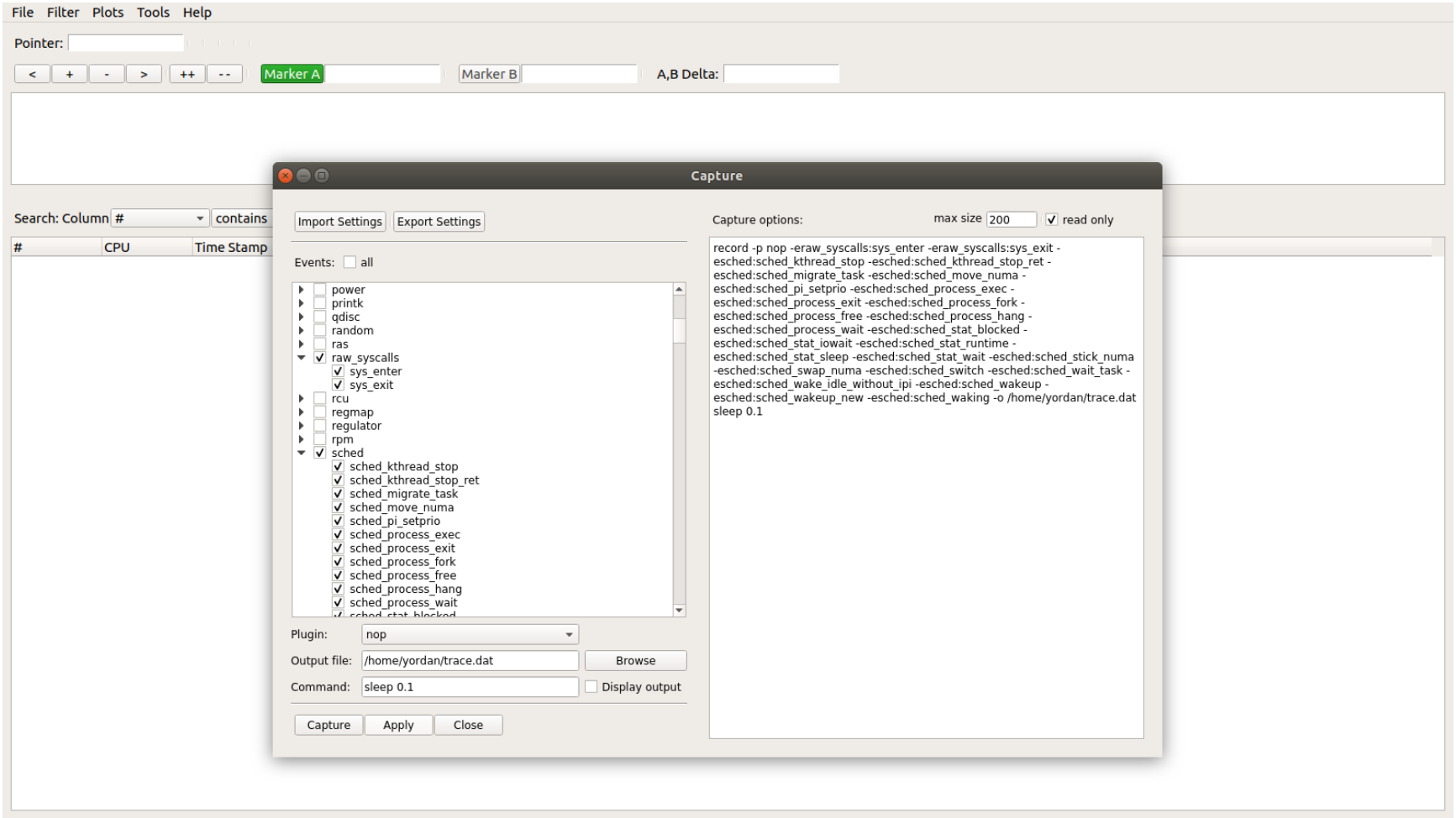
Kernel Shark - Recording



a. Trace data can be recorded directly from Kernel Shark.

b. Root password is required.

Kernel Shark - Recording



Ftrace is extremely powerful but ...

```
     Compositor-1124  [001] .... 236364.639085: _cond_resched <-futex_wait_queue_me
     Compositor-1124  [001] .... 236364.639086: rcu_all_qs <-_cond_resched
chromium-browse-1117  [000] d... 236364.639086: do_syscall_64 <-entry_SYSCALL_64_after_hwframe
     Compositor-1124  [001] .... 236364.639086: drop_futex_key_refs.isra.14 <-futex_wait
chromium-browse-1117  [000] .... 236364.639087: syscall_trace_enter <-do_syscall_64
     Compositor-1124  [001] .... 236364.639087: hrtimer_cancel <-futex_wait
chromium-browse-1117  [000] .... 236364.639087: __secure_computing <-syscall_trace_enter
     Compositor-1124  [001] .... 236364.639087: hrtimer_try_to_cancel <-hrtimer_cancel
chromium-browse-1117  [000] .... 236364.639087: __seccomp_filter <-__secure_computing
     Compositor-1124  [001] .... 236364.639088: hrtimer_active <-hrtimer_try_to_cancel
chromium-browse-1117  [000] .... 236364.639088: seccomp_run_filters <-__seccomp_filter
     Compositor-1124  [001] .... 236364.639088: lock_hrtimer_base.isra.20 <-hrtimer_try_to_cancel
     Compositor-1124  [001] .... 236364.639088: _raw_spin_lock_irqsave <-lock_hrtimer_base.isra.20
chromium-browse-1117  [000] .... 236364.639088: SyS_futex <-do_syscall_64
     Compositor-1124  [001] d... 236364.639089: __remove_hrtimer <-hrtimer_try_to_cancel
     Compositor-1124  [001] d... 236364.639089: _raw_spin_unlock_irqrestore <-hrtimer_try_to_cancel
chromium-browse-1117  [000] .... 236364.639089: ktime_get <-SyS_futex
chromium-browse-1117  [000] .... 236364.639090: ktime_add_safe <-SyS_futex
chromium-browse-1117  [000] .... 236364.639090: do_futex <-SyS_futex
chromium-browse-1117  [000] .... 236364.639091: futex_wait <-do_futex
chromium-browse-1117  [000] .... 236364.639091: hrtimer_init <-futex_wait
chromium-browse-1117  [000] .... 236364.639092: __hrtimer_init <-hrtimer_init
chromium-browse-1117  [000] .... 236364.639092: hrtimer_init_sleeper <-futex_wait
chromium-browse-1117  [000] .... 236364.639092: ktime_add_safe <-futex_wait
chromium-browse-1117  [000] .... 236364.639093: futex_wait_setup <-futex_wait
chromium-browse-1117  [000] .... 236364.639093: get_futex_key <-futex_wait_setup
chromium-browse-1117  [000] .... 236364.639093: get_futex_key_refs.isra.13 <-get_futex_key
chromium-browse-1117  [000] .... 236364.639094: hash_futex <-futex_wait_setup
chromium-browse-1117  [000] .... 236364.639095: _raw_spin_lock <-futex_wait_setup
chromium-browse-1117  [000] .... 236364.639095: get_futex_value_locked <-futex_wait_setup
chromium-browse-1117  [000] .... 236364.639095: futex_wait_queue_me <-futex_wait
chromium-browse-1117  [000] .... 236364.639096: hrtimer_start_range_ns <-futex_wait_queue_me
     Compositor-1124  [001] d... 236364.639097: do_syscall_64 <-entry_SYSCALL_64_after_hwframe
chromium-browse-1117  [000] .... 236364.639097: lock_hrtimer_base.isra.20 <-hrtimer_start_range_ns
```

Ftrace is extremely powerful but ...

```
     Compositor-1124  [001] .... 236364.639085: _cond_resched <-futex_wait_queue_me
     Compositor-1124  [001] .... 236364.639086: rcu_all_qs <-_cond_resched
chromium-browse-1117  [000] d... 236364.639086: do_syscall_64 <-entry_SYSCALL_64_after_hwframe
     Compositor-1124  [001] .... 236364.639086: drop_futex_key_refs.isra.14 <-futex_wait
chromium-browse-1117  [000] .... 236364.639087: syscall_trace_enter <-do_syscall_64
     Compositor-1124  [001] .... 236364.639087: hrtimer_cancel <-futex_wait
```

What do we do if:

a. We have a very large data-set of trace records.

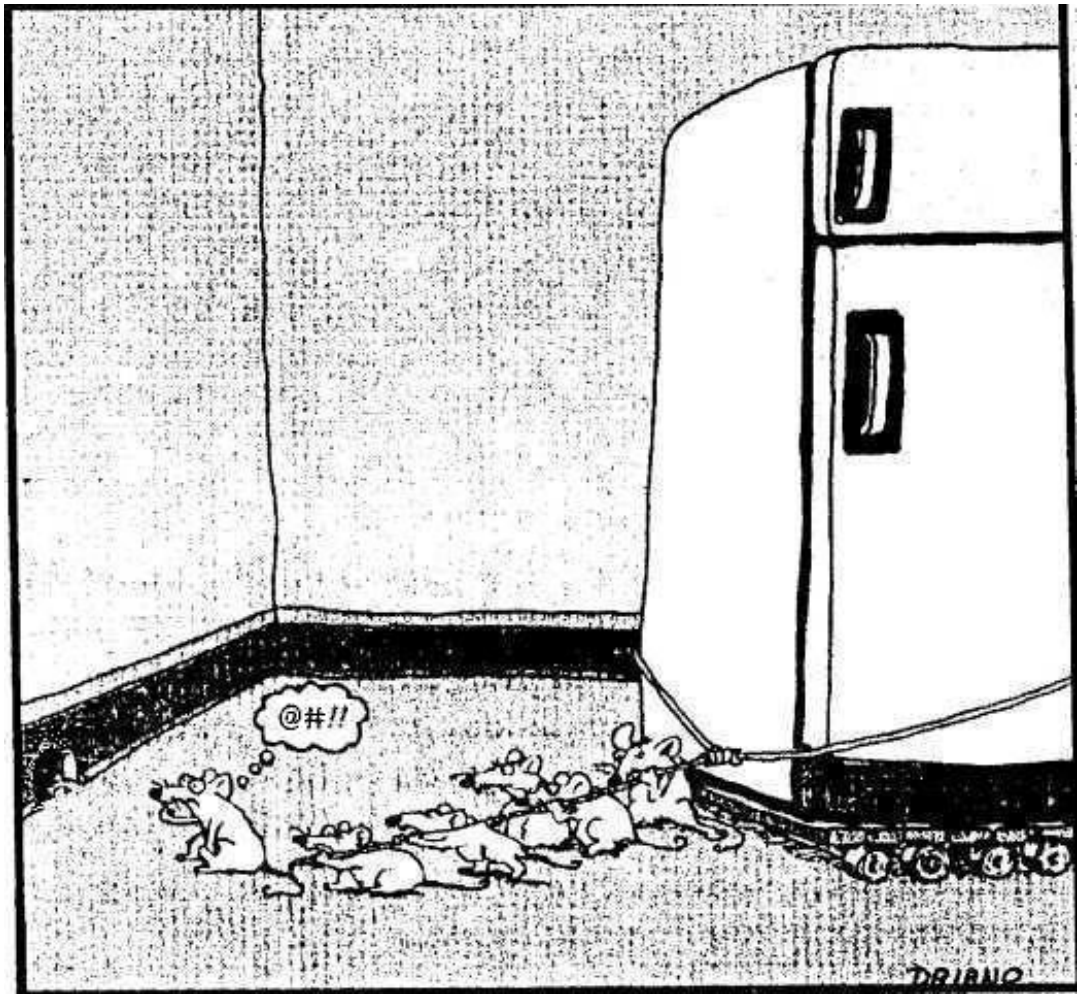
b. We are limited by the number of screen pixels available.

```
     Compositor-1124  [001] d... 236364.639089: _raw_spin_unlock_irqrestore <-hrtimer_try_to_cancel
chromium-browse-1117  [000] .... 236364.639089: ktime_get <-SyS_futex
chromium-browse-1117  [000] .... 236364.639090: ktime_add_safe <-SyS_futex
chromium-browse-1117  [000] .... 236364.639090: do_futex <-SyS_futex
chromium-browse-1117  [000] .... 236364.639091: futex_wait <-do_futex
chromium-browse-1117  [000] .... 236364.639091: hrtimer_init <-futex_wait
chromium-browse-1117  [000] .... 236364.639092: __hrtimer_init <-hrtimer_init
```

And we have to process all this in a reasonable amount of time.

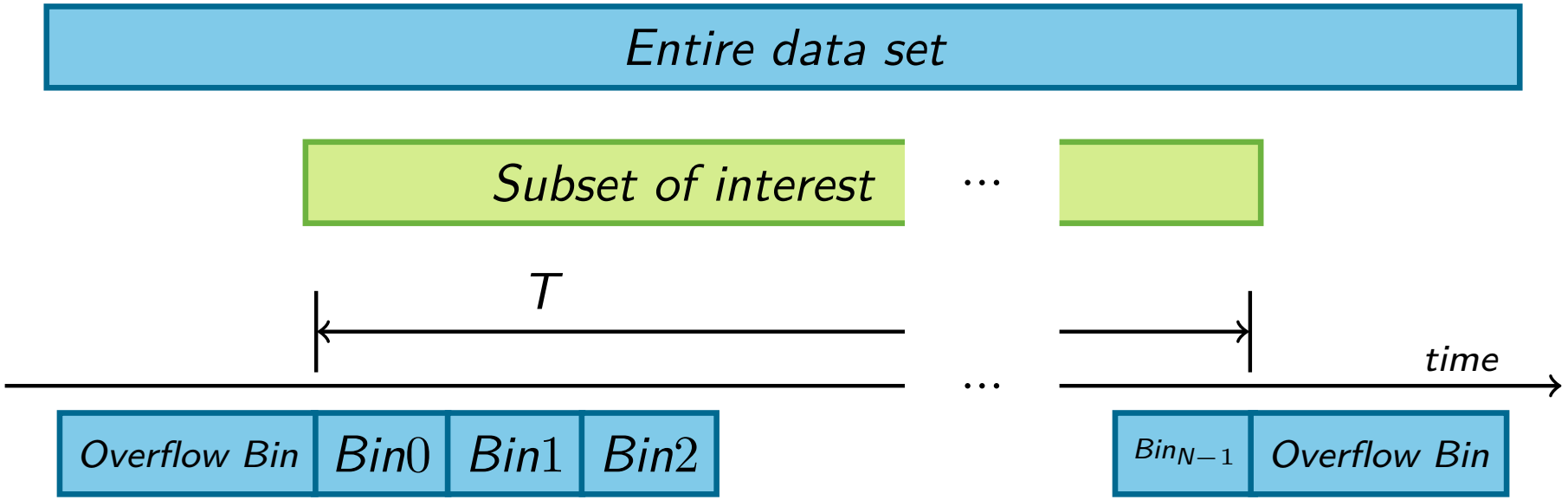
```
chromium-browse-1117  [000] .... 236364.639093: get_futex_key_refs.isra.13 <-get_futex_key
chromium-browse-1117  [000] .... 236364.639094: hash_futex <-futex_wait_setup
chromium-browse-1117  [000] .... 236364.639095: _raw_spin_lock <-futex_wait_setup
chromium-browse-1117  [000] .... 236364.639095: get_futex_value_locked <-futex_wait_setup
chromium-browse-1117  [000] .... 236364.639095: futex_wait_queue_me <-futex_wait
chromium-browse-1117  [000] .... 236364.639096: hrtimer_start_range_ns <-futex_wait_queue_me
     Compositor-1124  [001] d... 236364.639097: do_syscall_64 <-entry_SYSCALL_64_after_hwframe
chromium-browse-1117  [000] .... 236364.639097: lock_hrtimer_base.isra.20 <-hrtimer_start_range_ns
```

Fitting something large inside something small …



… and we have to do this quickly!

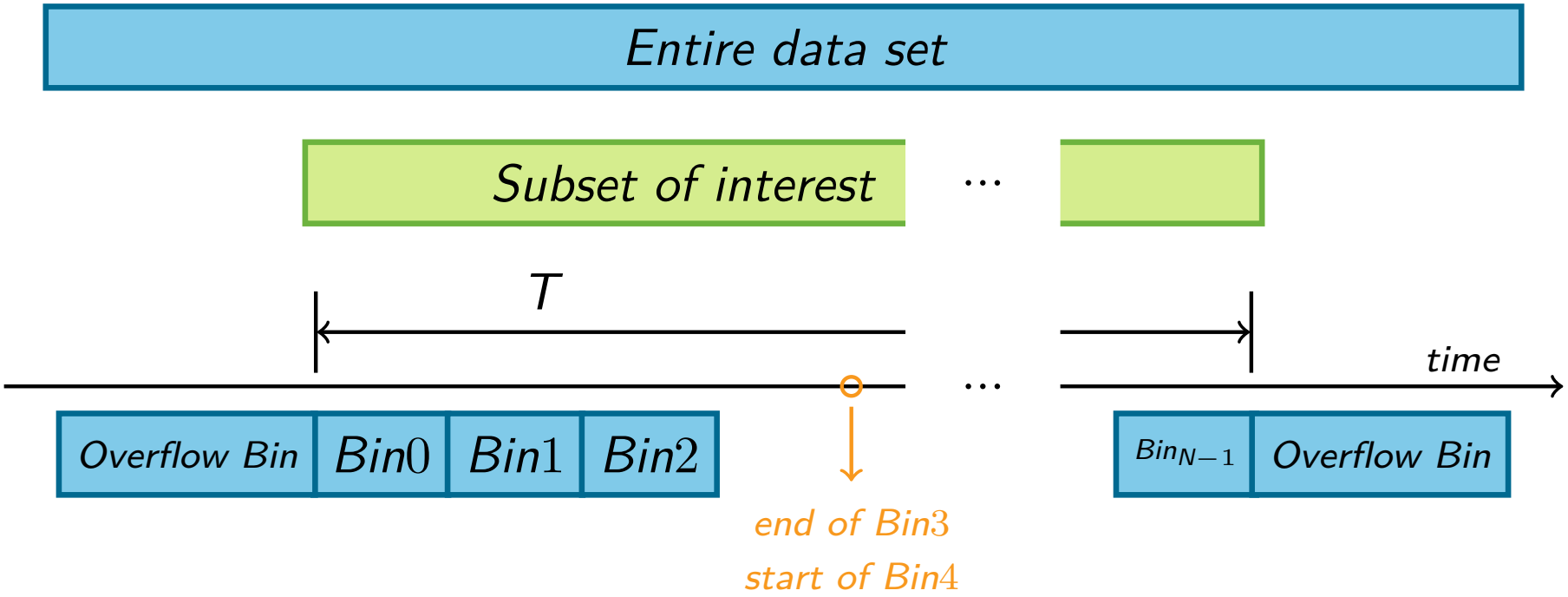
Visualization model - How does it work?



a. Break the data-set into ***time-bins*** $\mapsto$ like a histogram.

b. Check only the records at the beginning and at the end of each bin. $\mapsto$ constant time.

Visualization model - How does it work?

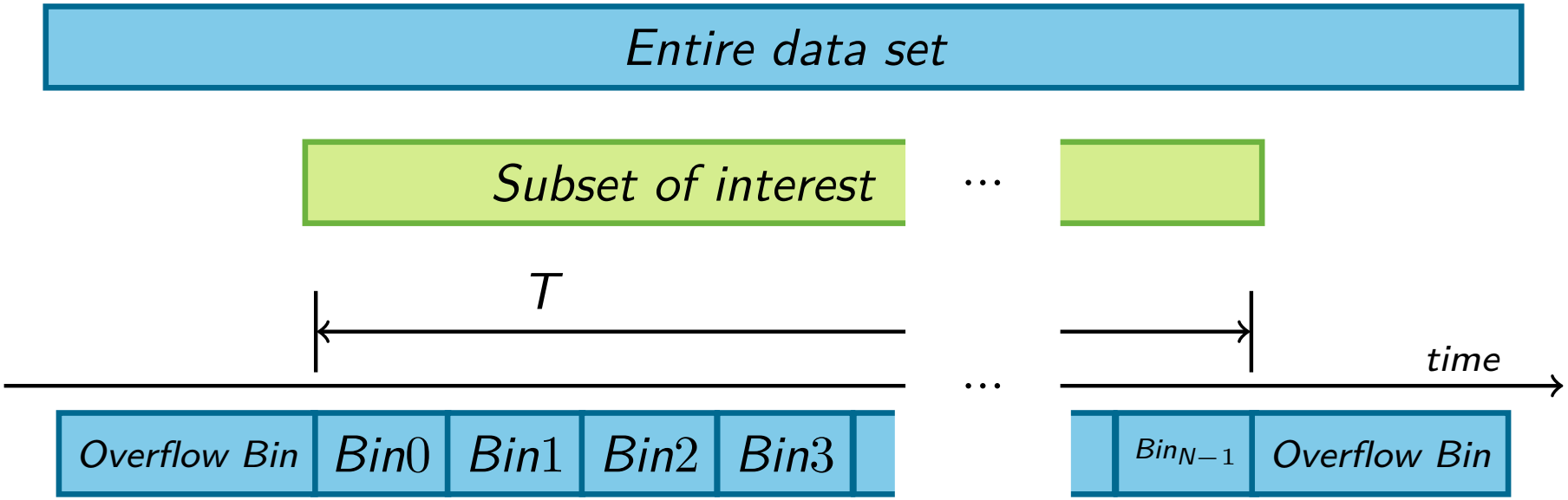


c. Have the trace records, sorted in time.

d. Knowing the index of the first record in each *Bin* determines the state of the model.

e. But the first element can be found with a binary search $\mapsto$ log(n) complexity.

Visualization model - How does it work?

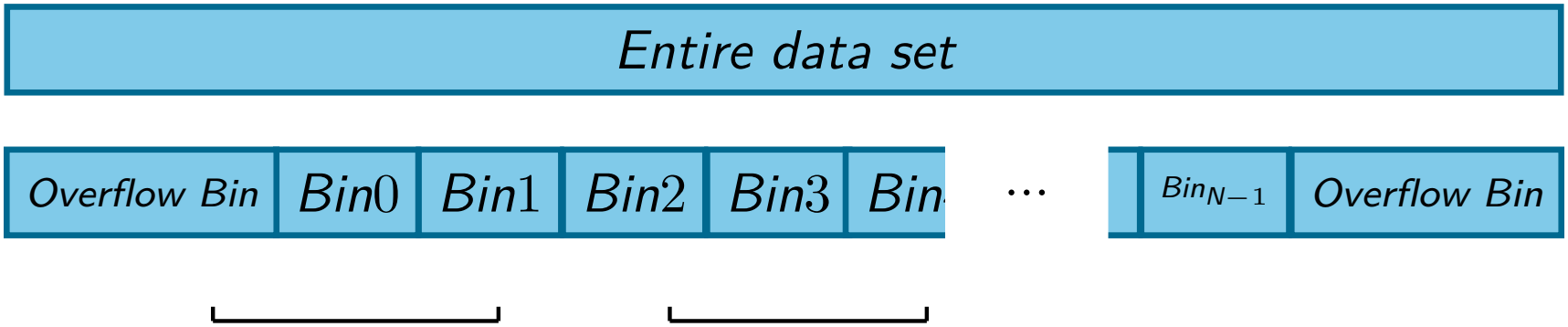


Data Binning provides $O(\log_2(n))$ average time complexity of all operations of the model.

Visualization model & tracing data formats

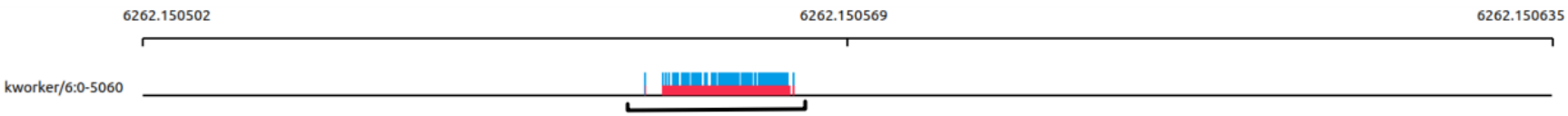
- The KernelShark Visualization model is not coupled to a particular data format.
- Uses KernelShark-specific data structure.
- Contains only the absolute minimum of information need by the model.
- The rest of the information - available on demand (can be slow)

Visualization model & tracing data



- Only one model (data structure) for all graphs.
- Worst-case complexity becomes linear.
- Solution - Data collections.

Visualization model & Data collections.



- Only one model (data structure) for all graphs.
- Worst-case complexity becomes linear.
- Solution - Data collections.

DEMO

KernelShark: current version 0.9

https://git.kernel.org/pub/scm/utils/trace-cmd/trace-cmd.git/

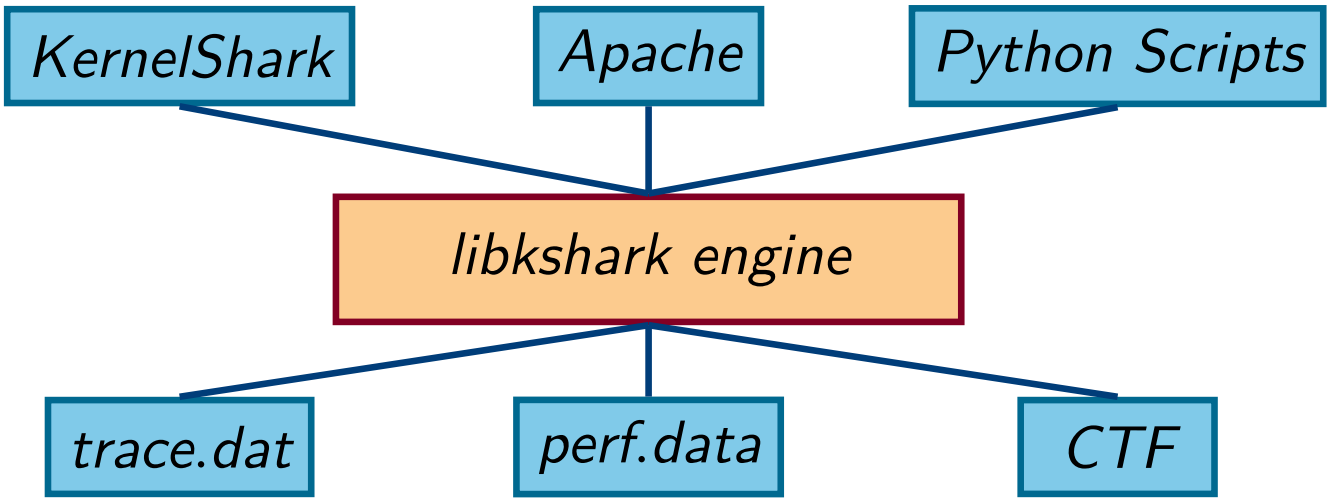
To build the code follow the instructions in

/trace-cmd/kernel-shark-qt/README

and

/trace-cmd/README

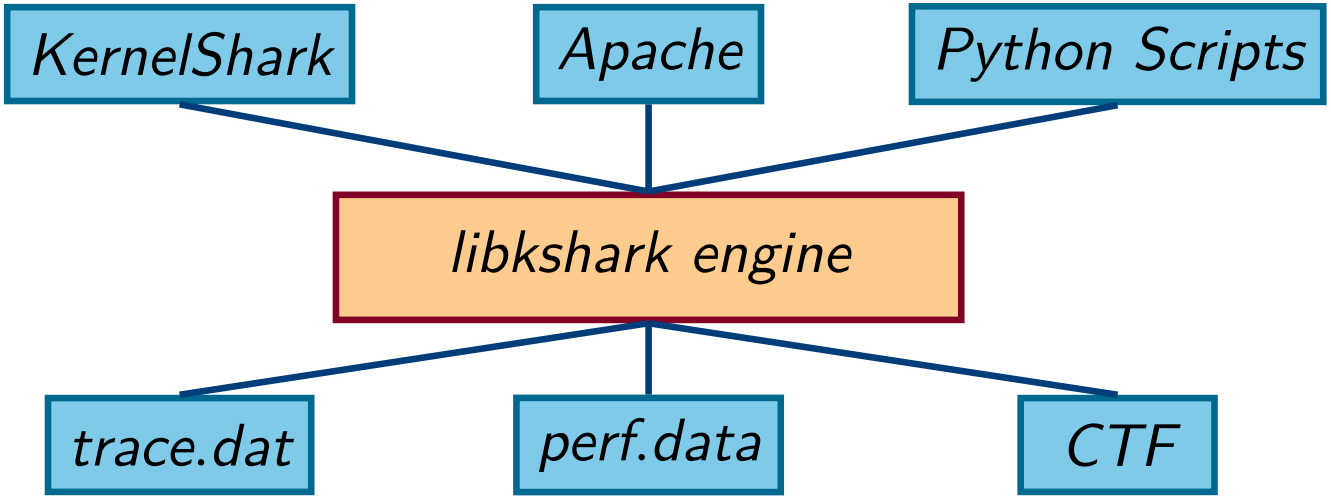
KernelShark is not a GUI. KernelShark is a toolkit.



What's next after KernelShark 1.0?

a. KernelShark engine (libkshark.so)
b. Available under GNU LGPL v2.1
c. Highly customizable (via plugins)
d. Will read multiple data formats

KernelShark is not a GUI. KernelShark is a toolkit.



What's next after KernelShark 1.0?

a. Any tool will be able to use the library
b. Available for Python applications (libkshark.py)
c. The KernelShark application is just a "shell".