# Host & Guest Tracing in Virtualization: "To sync, or not to sync?"

Stefano De Venuto SUSE, Intern Tzvetomir Stoyanov VMware Open Source Technology Center

## Who we are



#### Stefano De Venuto

Computer Science Student at University of Turin Intern at SUSE

- e-mail: stefano.devenuto99@gmail.com
- github: https://github.com/stefanodevenuto/



#### **Tzvetomir Stoyanov**

Software engineer in the Open Source Technology Center, VMware/Bulgaria working on the Linux Kernel tracing infrastructure.

- e-mail: tz.stoyanov@gmail.com
- github: https://github.com/tzstoyanov

## Ftrace

# trac	racer: function_graph								
#									
# CPU	DURATION	FUNCTION CALLS							
#									
5)		wake_up_q() {							
5)		<pre>try_to_wake_up() {</pre>							
5)	0.148 us	<pre>_raw_spin_lock_irqsave();</pre>							
5)		<pre>select_task_rq_fair() {</pre>							
5)	0.167 us	<pre>available_idle_cpu();</pre>							
5)	0.415 us	update_cfs_rq_h_load();							
5)		<pre>select_idle_sibling() {</pre>							
5)	0.216 us	available_idle_cpu();							
5)	0.519 us	}							
5)	1.931 us	}							
5)	0.151 us	_raw_spin_lock();							
5)		<pre>update_rq_clock() {</pre>							
5)	0.193 us	<pre>update_irq_load_avg();</pre>							
5)	0.529 us	}							

- The official tracer of the Linux kernel
- Developed by Steven Rostedt more than 10 years ago
- Part of the kernel, compiled by default in most popular Linux distros.
- Allows you to look inside every corner of a live running kernel.

# Host-Guest kernel tracing



### Challenges

- Fast transfer of huge tracing data between guest and host
- Time stamps synchronisation

# **Trace data transfer**

Channel	Throughput		
FIFOs	1000 MB/s		
vsockets	900 MB/s		
TCP/IP sockets	275 MB/s		

Measured on a laptop with Intel i5 CPU with 8 cores and and 16G RAM

# **Time stamps synchronisation - PTP**



Clock offset

- (T1' T1 T2' + T2) / 2
- round trip time is not symmetric
- no hardware timestamping
- Up to few hundred packets are exchanged in one clock offset measurement
- ftrace is used to get the packet times

# Time stamps synchronisation - KVM

/sys/kernel/debug/kvm:

- tsc-offset
- tsc-scaling-ratio
- tsc-scaling-ratio-frac-bits

guest\_tsc = tsc-offset + (host\_tsc \* tsc-scaling-ratio) >> tsc-scaling-ratio-frac-bits

# **Tools & Libraries**



Next Prev Graph follows

>>	#	CPU	Time Stamp	Task	PID	Latency	Event	Info	-
0	754331	22	1544715470.404868	webrtc_audio_mo	11960	S.	softirq_exit	vec=1 [action=TIMER]	
0	754332	18	1544715470.404869	CPU 4/KVM	22832	d	kvm_entry	vcpu 4	
0	754333	4	1544715470.404869	CPU	22829	S.	softirq_entry	vec=7 [action=SCHED]	
0	754334	9	1544715470.404870	CPU 2/KVM	22830	d	kvm_entry	vcpu 2	
0	754335	22	1544715470.404870	webrtc_audio_mo	11960	s.	softirq_entry	vec=7 [action=SCHED]	
0	754336	5	1544715470.404871	CPU 0/KVM	22828	S.	softirq_exit	vec=7 [action=SCHED]	-
0	754337	10	1544715470.404871	CPU 5/KVM	22833		kvm_exit	reason EXTERNAL_INTERRUPT rip 0xfffffff89a00011 info 0 800000ec	
1	754338	4	1544715470.404872	systemd	1	d	mm_lru_insertion	page=0xffffd76b88617d40 pfn=2196981 lru=1 flags=Ma b	
0	754339	10	1544715470.404873	CPU 5/KVM	22833	d	kvm_entry	vcpu 5	
0	754340	5	1544715470.404873	CPU 0/KVM	22828		kvm_exit	reason EXTERNAL_INTERRUPT rip 0xfffffff8919efe7 info 0 800000ec	
0	754341	4	1544715470.404874	CPU	22829	S.	softirq_exit	vec=7 [action=SCHED]	
0	754342	2	1544715470.404875	<idle></idle>	0	S.	softirq_exit	vec=7 [action=SCHED]	
1	75/3/3	0	1544715470 404975	trace-cmd	216/2	d	sched switch	nrau comm-trace.cmd nrau nid-21642 nrau nrin-120 nrau state-D> next comm-swanner/0 next nid-0 next nrin-120	

- trace-cmd 2.9
- KernelShark 2.0
- trace libraries
  - librtaceevent
  - libtracefs
  - libtracecmd
  - libkshark

# Verifying the results

• Given a combined (host + guest[s]) trace, how good did the merging and the timestamp synchronization algorithm of choice work?

- Two different aspects:
  - event stream validation
  - timestamp synchronization accuracy evaluation

# Validation: kvm\_entry & kvm\_exit events

Two crucial events:

- **kvm\_entry:** marks when a CPU starts executing instructions from the guest
- **kvm\_exit:** marks when the CPU stops executing instructions from the guest

What we expect:

- **only** guest events between a kvm\_entry and the next kvm\_exit
- **no** guest events between a kvm\_exit and the next kvm\_entry

# Validation: what we expect?

File	Filter I	Plots	Tools Help							
Point	Pointer: 11241.652510									
<	< + - > ++ Marker A Marker B A,B Delta:									
		11	241.652511				11241.652523 11241.65	52535		
СР	U 0									
	CPU 0KW-1977									
CP	o artini io									
Searc	h: Colum	nn >>	¢ cont	ains 🛟		Next Prev Graph	follows			
Searco >>	h: Colum:	ın >> CPU	¢ cont	ains 🛟	PID I	Next Prev ✓Graph tency Event	follows Info			
Searco >> 0	:h: Colum # 182	nn >> CPU 3	¢ cont Time Stamp 11241.652511	ains ‡ Task CPU 0/KVM	PID 1	Next Prev Graph tency Event . kvm/kvm_entry	follows Info vcpu 0, rip 0xfffffffaa079246	0		
CP Searco >> 0 1	:h: Colum # 182 184	n >> CPU 3 0	¢ cont Time Stamp 11241.652511 11241.652512	Task CPU 0/KVM cidle>	PID 1 1977 0	Next Prev         Graph           tency         Event           .         kvm/kvm_entry           n.         timer/hrtimer_expire_exit	Info       vcpu 0, rip 0xfffffffaa079246       hrtimer=0xffff90127dc21540	_0		
Searco >> 0 1 1	:h: Colum # 182 184 185	CPU 3 0 0	¢ cont Time Stamp 11241.652511 11241.652512 11241.652513	Task CPU 0/KVM cidle>	PID 1 1977 0 0 0	Next         Prev         ✓Graph           tency         Event            kvm/kvm_entry             n.         timer/hrtimer_expire_exit            n.         timer/hrtimer_start	Info         vcpu 0, rip 0xfffffffaa079246         hrtimer=0xffff90127dc21540         hrtimer=0xffff90127dc21540 function=tick_sched_timer/0x0 expires=3616998901864 softexpires=3616998901864	0		
CP Searc >> 0 1 1 1	:h: Colum # 182 184 185 186	CPU 3 0 0 0	cont Time Stamp 11241.652511 11241.652512 11241.652513 11241.652521	Task CPU 0/KVM cidle> cidle> cidle>	PID 1 1977 0 0 0 0 0	Next         Prev         ✓Graph           tency         Event            .         kvm/kvm_entry             timer/hrtimer_expire_exit            n.         timer/hrtimer_start            n.         msr/write_msr	Follows         Info         vcpu 0, rip 0xfffffffaa079246         hrtimer=0xffff90127dc21540         hrtimer=0xffff90127dc21540 function=tick_sched_timer/0x0 expires=3616998901864 softexpires=3616998901864         6e0, value 9e5fd35bc24			
CP Searco >> 0 1 1 1 1 1	h: Colum # 182 184 185 186 187	CPU 3 0 0 0 0 0 0	¢ cont Time Stamp 11241.652511 11241.652512 11241.652513 11241.652511 11241.652511 11241.652513	Task CPU 0/KVM cidle> cidle> cidle> cidle>	PID 1 1977 0 0 0 0 0 0 0 0 0 0	Next     Prev     ✓Graph       tency     Event       .     kvm/kvm_entry        timer/hrtimer_expire_exit        timer/hrtimer_start        msr/write_msr        power/cpu_idle	Follows         Info         vcpu 0, rip 0xfffffffaa079246         hrtimer=0xffff90127dc21540         hrtimer=0xffff90127dc21540 function=tick_sched_timer/0x0 expires=3616998901864 softexpires=3616998901864         6e0, value 9e5fd35bc24         state=4294967295 cpu_id=0	0		

# Validation: what we expect?

File	Filter	Plots	Tools Help								
Pointe	Pointer: 11241.652532										
<	< + - > ++ Marker B A/B Delta:										
		1'	241.652533						11241.652537 11241.652542		
CPL	0										
CPL	0/KVM-19	77						1			
Searc	n: Colum	in >>	¢ cont	ains 🛟			Next Prev	Graph f	illows		
>>	#	CPU	Time Stamp	Task	PID	Latency	Event		Info		
0	190	3	11241.652535	CPU 0/KVM	1977	d	kvm/kvm_exit		reason HLT rip 0xfffffffaaa6917d info 0 0		
0	191	3	11241.652536	CPU 0/KVM	1977	d	kvm/kvm_wait_l	apic_expire	vcpu 0: delta 280220 (late)		
0	192	3	11241.652537	CPU 0/KVM	1977		kvm/kvm_eoi		apicid 0 vector 236		
0	193	3	11241.652537	CPU 0/KVM	1977		kvm/kvm_pv_ed	di .	apicid 0 vector 236		
0	194	3	11241.652539	CPU 0/KVM	1977	d	timer/hrtimer_s	tart	hrtimer=0xfff949286151210 function=apic_timer_fn/0x0 expires=3718274710335 softexpires=3718274710335		
1	195	0	11241.652540	<idle></idle>	0	d	timer/tick_stop		success=1 dependency=NONE		
1	196	0	11241.652540	<idle></idle>	0	d	timer/hrtimer_c	ancel	hrtimer=0xfff90127dc21540		
1	197	0	11241.652541	<idle></idle>	0	d	timer/hrtimer_s	tart	hrtimer=0xffff90127dc21540 function=tick_sched_timer/0x0 expires=3617026901864 softexpires=3617026901864		
0	199	3	11241.652541	CPU 0/KVM	1977		kvm/kvm_hv_tir	ner_state	vcpu_id 0 hv_timer 0		
1	200	0	11241.652542	<idle></idle>	0	d	power/cpu_idle		state=1 cpu id=0		
	-										

# Validation: what we expect?

File	File Filter Plots Tools Help												
Point	Pointer: 11241.652487												
<	< + - > ++ Marker A Marker B A,B Delta:												
		11	241.652486					11241.652497 11241.652509					
СРІ	CPU 0 KVM-1977												
					Search: Column >>  Contains  Next Prev Graph follows								
Searc	h: Colum#	n >>	t cont.	ains ‡	PID	Latency	Next Prev Graph f	follows					
Searc	:h: Colum # 173	n >> CPU 3	¢ cont. Time Stamp 11241.652487	ains 🛟	PID 1977	Latency	Next Prev Graph f Event kvm/kvm entry	follows Info vcpu 0, rip 0xfffffffaaa6917e					
Searc >> 0	:h: Colum # 173 174	n >> CPU 3 3	¢ cont. Time Stamp 11241.652487 11241.652489	Task CPU 0/KVM CPU 0/KVM	PID 1977 1977	Latency d d	Next Prev Graph f Event kvm/kvm_entry msr/write_msr	Follows Info vcpu 0, rip 0xfffffffaaa6917e 48, value 0					
Searc >> 0 0	h: Colum # 173 174 175	n >> CPU 3 3 0	¢ cont. Time Stamp 11241.652487 11241.652489 11241.652505	ains Task CPU 0/KVM CPU 0/KVM <idle></idle>	PID 1977 1977 0	Latency d d d.h.	Next Prev Graph f Event kvm/kvm_entry msr/write_msr timer/hrtimer_cancel	follows Info vcpu 0, rip 0xfffffffaaa6917e 48, value 0 hrtimer=0xfff90127dc21540	0				
Searc >> 0 0 1 1	h: Colum # 173 174 175 175	n → CPU 3 3 0 0	¢ cont Time Stamp 11241.652487 11241.652489 11241.652505 11241.652507	ains ; Task CPU 0/KVM CPU 0/KVM <idle> <idle></idle></idle>	PID 1977 1977 0 0	Latency d d d.h. d.h.	Next Prev Graph f Event kvm/kvm_entry msr/write_msr timer/hrtimer_cancel timer/hrtimer_expire_entry	Info vcpu 0, rip 0xfffffffaaa6917e 48, value 0 hrtimer=0xffff90127dc21540 hrtimer=0xffff90127dc21540 now=3616994997849 function=tick_sched_timer/0x0					
Searc >> 0 0 1 1 1 0	+: Colum # 173 174 175 176 177	n >> CPU 3 3 0 0 0 3	time Stamp 11241.652487 11241.652489 11241.652505 11241.652507 11241.652507	Task CPU 0/KVM CPU 0/KVM <idle> <idle> CPU 0/KVM</idle></idle>	PID 1977 1977 0 0 0 1977	Latency d d d.h. d.h. d.h.	Next Prev Graph f Event kvm/kvm_entry msr/write_msr timer/hrtimer_cancel timer/hrtimer_expire_entry msr/read_msr	Info vcpu 0, rip 0xfffffffaaa6917e 48, value 0 hrtimer=0xffff90127dc21540 hrtimer=0xffff90127dc21540 now=3616994997849 function=tick_sched_timer/0x0 48, value 0	0				
Searc >> 0 0 1 1 1 0 0	+ Colum 4 173 174 175 176 177 178	n >> CPU 3 3 0 0 0 3 3 3	¢ cont Time Stamp 11241.652487 11241.652489 11241.652505 11241.652507 11241.652508	Task CPU 0/KVM CPU 0/KVM <idle> <idle> CPU 0/KVM CPU 0/KVM</idle></idle>	PID 1977 1977 0 0 0 1977 1977	Latency d d d.h. d.h. d d	Next Prev Graph f Event kvm/kvm_entry msr/write_msr timer/hrtimer_cancel timer/hrtimer_expire_entry msr/read_msr msr/write_msr	Info vcpu 0, rip 0xfffffffaaa6917e 48, value 0 hrtimer=0xfff90127dc21540 hrtimer=0xfff90127dc21540 now=3616994997849 function=tick_sched_timer/0x0 48, value 0 48, value 0 48, value 4	0				

# Time-sync accuracy evaluation: an "event-based" approach

- Set(s) of consequential events:
  - event in the guest that causes one or more events on the host
  - event on the host that causes one or more events in the guest

G: XXX - guest event	H: YYY - host event
H: YYY - host event, consequence of XXX	G: XXX - guest event, consequence of YYY

- Reproducible and stable:
  - **always** the same events!

# First set: hrtimer & MSR events

Programming a timer

• Baremetal

+ Write the timeout value in an MSR (for instance, IA32\_TSC\_DEADLINE MSR)

+ Will get an interrupt when timeout expires

### • Virtual Machine

+ Write the timeout value in an MSR... Of course, it can't be the host MSR directly!

- + VMExit with reason: msr\_write
- + Host handles the exit (and the timer, and the real and emulated MSR write, etc)

+ ... ... ...

## **First set: hrtimer & MSR events**

### • High Resolution Timers:

When in TSC Deadline mode, they interact with the Deadline MSR

# trace-cmd record -C x86-tsc -e msr:\* -e kvm:\* -e timer:\* -A tumbleweed:823 -e timer:\* -e msr:\* sleep 1

H:	CPU 0/KVM-1977	[000]	2360.150329: kvm_entry:	vcpu 0, rip 0xffffffffaa079246
G:	<idle>-0</idle>	[000]	2360.150331: hrtimer_start:	hrtimer=0xffff90127dc21540
H:	CPU 0/KVM-1977	[000]	2360.150333: kvm_exit:	vcpu 0 reason MSR_WRITE
H:	CPU 0/KVM-1977	[000]	2360.150333: kvm_hv_timer_state:	vcpu_id 0 hv_timer 1
H:	CPU 0/KVM-1977	[000]	2360.150333: kvm_msr:	msr_write 6e0 = 0x61d24e6ecd4
H:	CPU 0/KVM-1977	[000]	2360.150333: kvm_entry:	vcpu 0, rip 0xffffffffaa079246
G:	<idle>-0</idle>	[000]	2360.150334: write_msr:	6e0, value 61d24e6ecd4

# Second set: idle task & HLT events

### • Idle task:

Special task executed when there are no other runnable tasks. Runs the **idle loop**.

### • Idle loop:

Conceptually (and originally), a NOP busy loop. Nowadays, optimized with idle states (**c states**).

```
while (!need_resched()) {
    // ...
    if (cpu_idle_force_poll || tick_check_broadcast_expired()) {
        tick_nohz_idle_restart_tick();
        cpu_idle_poll();
    } else {
        cpuidle_idle_call();
    }
    arch_cpu_idle_exit();
```

```
• Idle loop inside a VM:
```

(typicall) VMExit to the host, to let other tasks/VMs run

# Second set: idle task & HLT events

# trace-cmd record -C x86-tsc -e power:\* -e kvm:\* -A tumbleweed:823 -e power:\* sleep 1

• idle=halt

G:	<idle>-0</idle>	[000]17183092310903: cpu_idle:	<pre>state=1 cpu_id=0</pre>	
H:	CPU 0/KVM-7361	[003]17183092315864: kvm_exit:	vcpu 0 <b>reason HLT</b> rip 0xffffffffabda5e07	

### • idle=*poll*

G:	<idle>-0</idle>	[000]11350708411803: <b>cpu_idle</b> :	<pre>state=0 cpu_id=0</pre>	X
н:	CPU 0/KVM-1594	[000]11350713125170: kvm_exit:	vcpu 0 reason INTR rip 0xfffffff9f8792b6	

# Infer information: we have sets, and then?

• We can use the timers and idle sets to measure the achieved time synchronization accuracy

• For each set, **subtract** the timestamps of the events in it, in order to generate many **event deltas** 

• Compute the **mean** and the **standard deviation** of the event deltas found, which combined will indicate the overall performance

## Let's automate it: final tool

- ./checker <host-file> <guest-file>... [-n event\_name]... [-s samples-file]
- Multi vCPUs and multi guests
  - -n event\_name: exclude event from validation process
  - -s filename: store all samples
- Implemented with **libkshark**, directly using the generated *.dat* files
- Might also be useful to re-think the placement of the tracepoints in the kernel.

# Single guest

#### # ./checker trace.dat trace-tumbleweed.dat

#### # ./checker trace.dat trace-tumbleweed.dat -s <filename>

<pre># Generated samples: <filename>-timer.txt</filename></pre>	<pre># Generated samples: <filename>-idle.txt</filename></pre>
# Timer samples: 1729 samples 3580 2860 3122	# Idle samples: 632 samples 1328 1057 1298

## **Multiple guests**

#### # ./checker trace.dat trace-tumbleweed.dat trace-tumbleweed2.dat

#### 

Number of events: 121480 Host events inside kvm\_entry/kvm\_exit block: 7247 Guest events outside kvm\_entry/kvm\_exit block: 0

 TIMER events:
 N\_Samples: 64,
 Mean: 2773.515625 ns,
 Variance: 924496.750000 ns^2,
 SD: 961.507568 ns

 HLT events:
 N\_Samples: 18,
 Mean: 1791.000000 ns,
 Variance: 388813.437500 ns^2,
 SD: 623.549072 ns

#### 

[+] /home/stefano/tracce/2guests/trace-tumbleweed.dat

Number of events: 71 Events outside kvm\_entry/kvm\_exit block: 0

TIMER events: N\_Samples: 10, Mean: 2912.399902 ns, Variance: 1457231.750000 ns^2, SD: 1207.158569 ns HTL events: N\_Samples: 5, Mean: 1227.800049 ns, Variance: 446477.187500 ns^2, SD: 668.189514 ns

[+] /home/stefano/tracce/2guests/trace-tumbleweed2.dat

Number of events: 1805 Events outside kvm\_entry/kvm\_exit block: 0

TIMER events: N\_Samples: 54, Mean: 2747.796387 ns, Variance: 821608.562500 ns<sup>2</sup>, SD: 906.426270 ns HTL events: N\_Samples: 13, Mean: 2007.615356 ns, Variance: 197714.312500 ns<sup>2</sup>, SD: 444.650787 ns

# **PTP vs KVM: analysis**

- Different scenarios
  - Idle system
  - Stressed system(s): stress-ng --matrix 0

- 30 sessions of tracing, 20 seconds long
  - ~8000 samples for timer sequence
  - ~3000 samples for idle sequences

# **PTP vs KVM: validation**

### Invalid guest events



# **PTP vs KVM: evaluation**

		Timer events		Halt events	
		Mean	Standard Deviation	Mean	Standard Deviation
	No stress	7.40 µs	4.36 µs	4.43 µs	2.71 µs
	Guest stress	3.59 µs	0.65 µs	N/A	N/A
PTP	Host stress	6.70 µs	7.45 µs	4.50 µs	1.91 µs
	Both stress	13.52 µs	1.66 µs	N/A	N/A
	No stress	3.33 µs	1.14 µs	2.40 µs	0.59 µs
	Guest stress	0.78 µs	0.16 µs	N/A	N/A
KVM	Host stress	0.85 µs	0.15 µs	0.67 µs	0.16 µs
	Both stress	1.28 µs	0.38 µs	N/A	N/A

# PTP vs KVM: not a fair battle...

#### Performance





# PTP vs KVM: not a fair battle...

• Stability



# Conclusion

- Host-Guest tracing is possible, if we can synchronize the traces
- trace-cmd supports multiple synchronization mechanisms:

PTP	Complex to implement	Very simple implementation	KVM
	Not accurate enough	Very accurate	
	Hypervisor agnostic	Relies on debugfs entries	

# Conclusion

- KVM relies on **debugfs entries** 
  - stable enough ABI?
  - what if debugfs is compiled out (for security reasons)?

- **Feedback wanted:** how else can we read the (per-vCPU) tsc-offset, tsc-scaling-ratio, etc values:
  - a new system call ?
  - o ...?

# Links & Acknowledgments

- Links
  - trace-cmd.org
  - kernelshark.org
  - Sync evaluation tool
- Acknowledgments
  - Yordan Karadzhov (VMware) <y.karadz@gmail.com>
  - Steven Rostedt (VMware) <rostedt@goodmis.org>
  - Dario Faggioli (SUSE) <dfaggioli@suse.com>

# **Thanks for your attention!**