Monitoring and Analyzing Virtual Machine Interference via Host Kernel Tracing

Hani Nemati

Dec 10, 2015

Polytechnique Montréal

Laboratoire **DORSAL**

Agenda

Motivation

- Why tracing virtual machines ?
- What are the challenges in tracing virtual machines?

Investigations

- Background
- Analyzing virtual machine interference via host kernel tracing
  - Resource Monitoring for virtual machines by host kernel tracing
    - Demo
  - Monitoring Processes and threads inside virtual machine with host kernel tracing
    - Demo

Conclusion and in-progress

References

Motivation

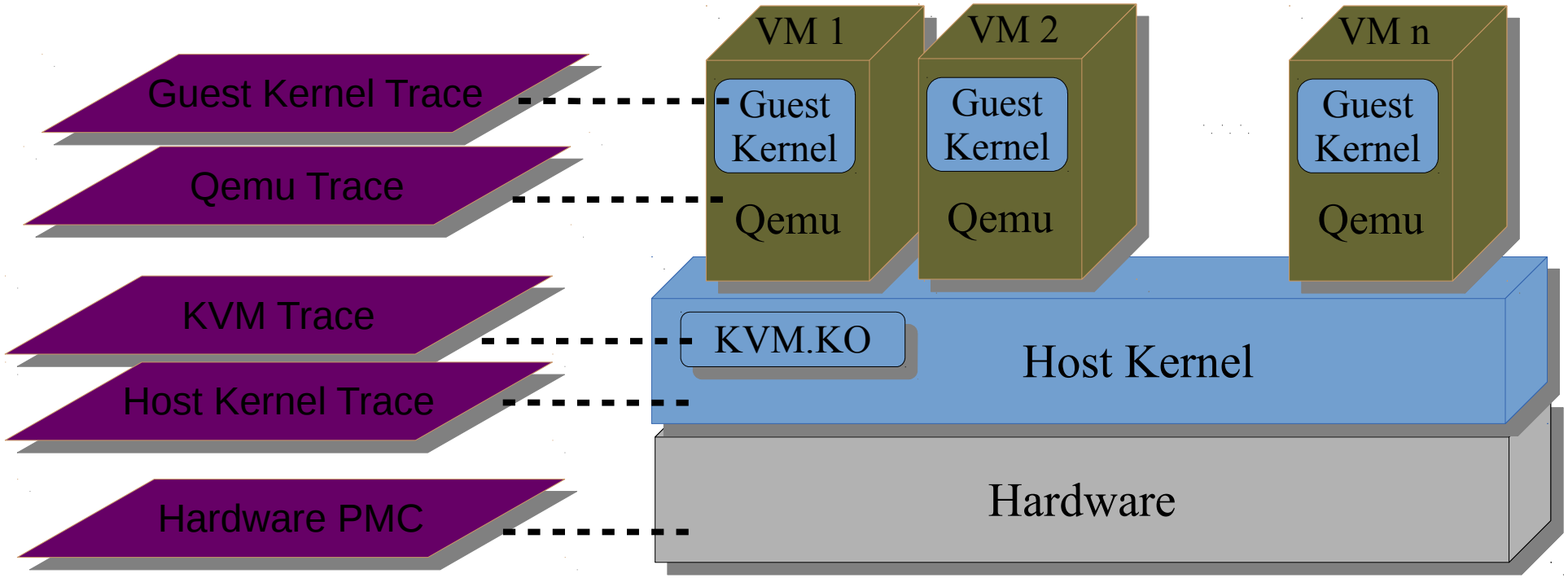
Why tracing virtual machines?

- Cloud User:
  - Identify performance anomaly
  - Find the root cause and correct it
  - What if the performance degradation is not from a process inside the VM?

- Cloud Administrator:
  - Identify performance degradation of each VM
  - Contention for shared resources
  - Find the root cause and solve the problem

Motivation

What are the main challenges in tracing virtual machines?



Investigations

Background

- **The Paper that uses kernel level Information**
  - **PerfCompass** [1]
    - Global and local anomaly detection tool
- **The Papers that uses Hardware level Information**
  - **Resource contention detection in virtualized Environments** [2]
  - **DeepDive** [3]
    - Identifies when interference occurs and what resources is causing it
    - Using Hardware Performance Counters
  - **CPI2** [4]
    - Performance interference detector in shared resources
    - Using CPI and CPU usage

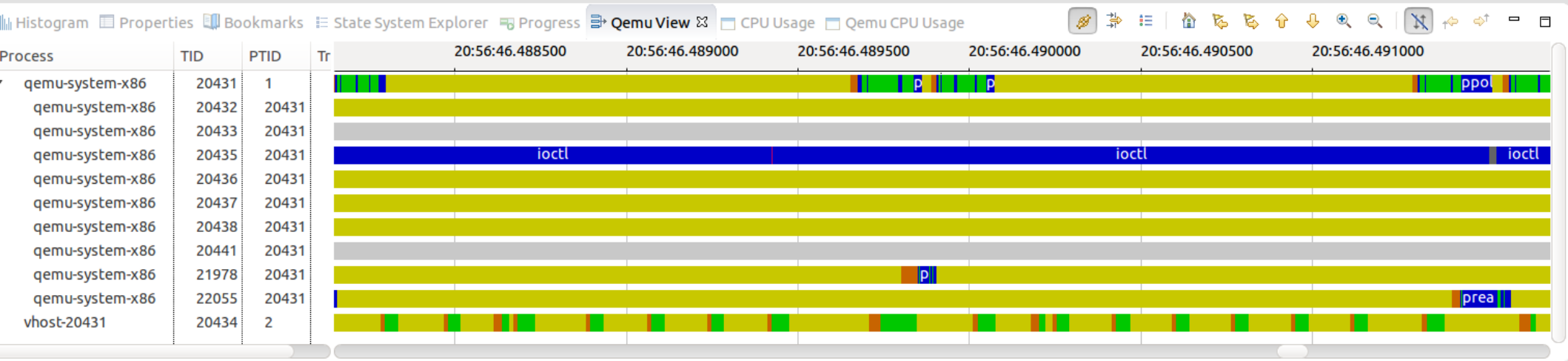
Investigations

Analyzing virtual machine interference via host kernel tracing

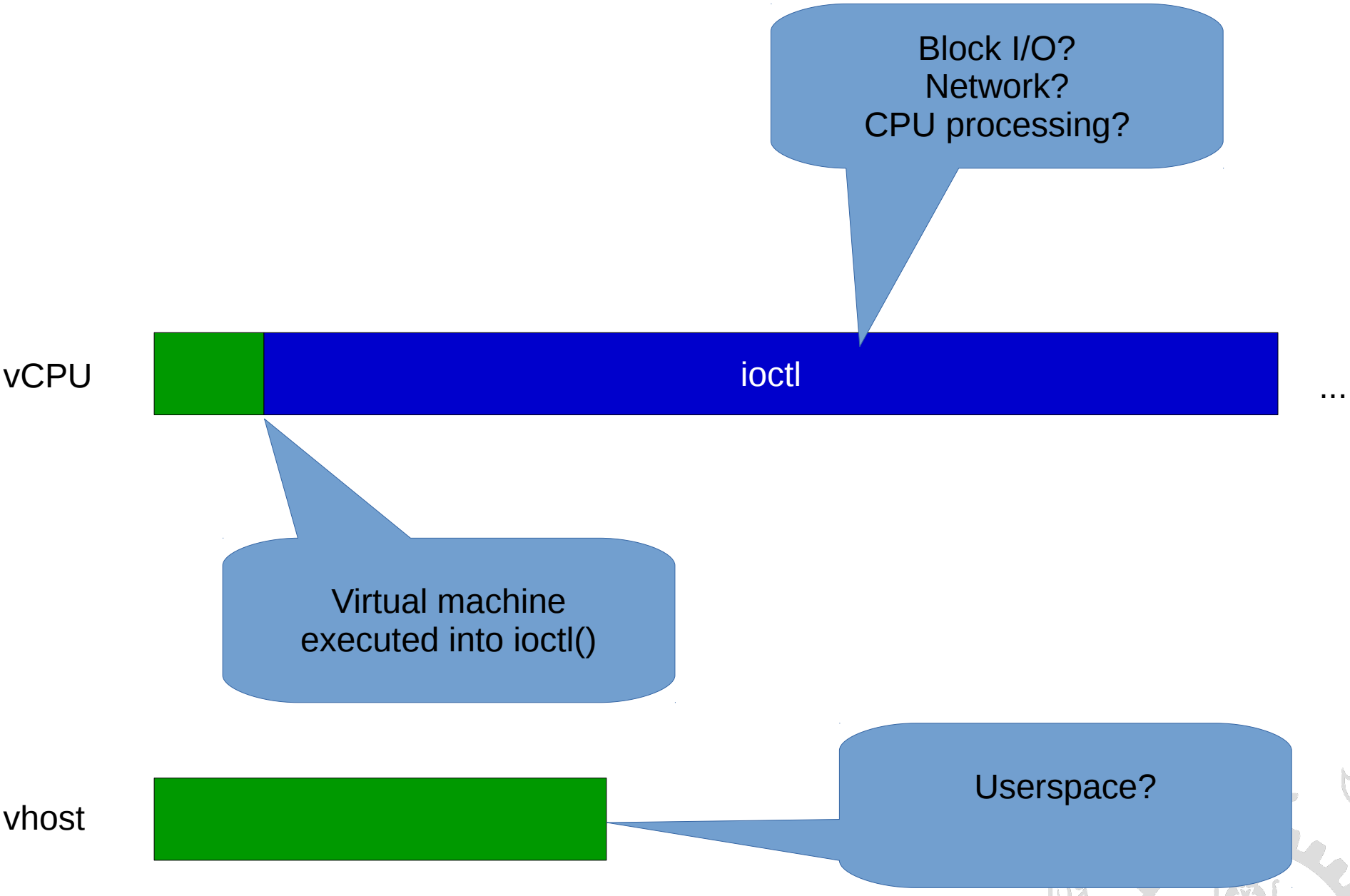
- **Interference** is what happens when VMs are concurrently competing for hardware resources
  - Increase the latency and reduce the performance
- **Goal:**
  - Identify the interference
    - By applying Machine Learning
  - Find the root cause of interference
    - By finding most frequent events during the interference period
  - Solve the interference
    - By categorizing VMs based on workload by using Machine Learning
    - Migrate the Vms , limitation on resource usage, add more resources

Investigations

Qemu threads in Control Flow View



Investigations



Investigations

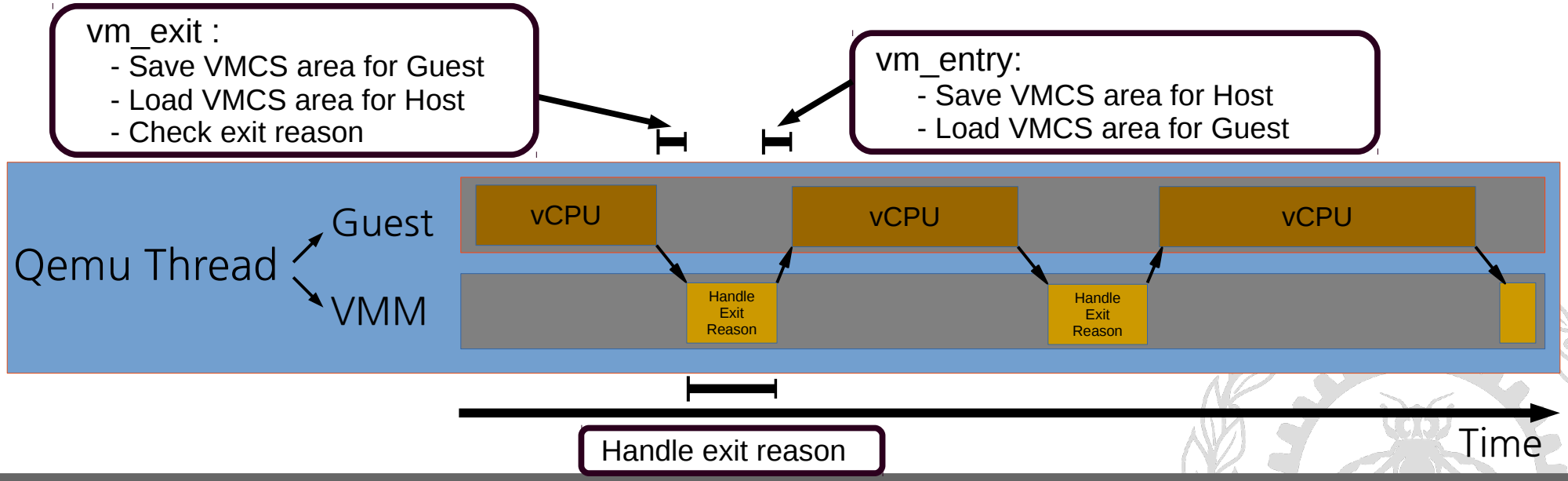
Resource Monitoring for virtual machine by host kernel tracing

- Disk I/O:
  - Using Qemu trace points
    - qemu:thread_pool_submit, qemu:thread_pool_complete, qemu:bdrv_co_io_em
- Network:
  - Using Host Kernel Trace
    - net_if_rx, net_dev_xmit, sched_switch and sched_wakup for vhost-$(VM-main-thread)
- CPU:
  - Using KVM Trace and Host Kernel Trace
    - sched_switch for qemu-system-x86, vm_entry, vm_exit, vcpu_guest_entry

Investigations

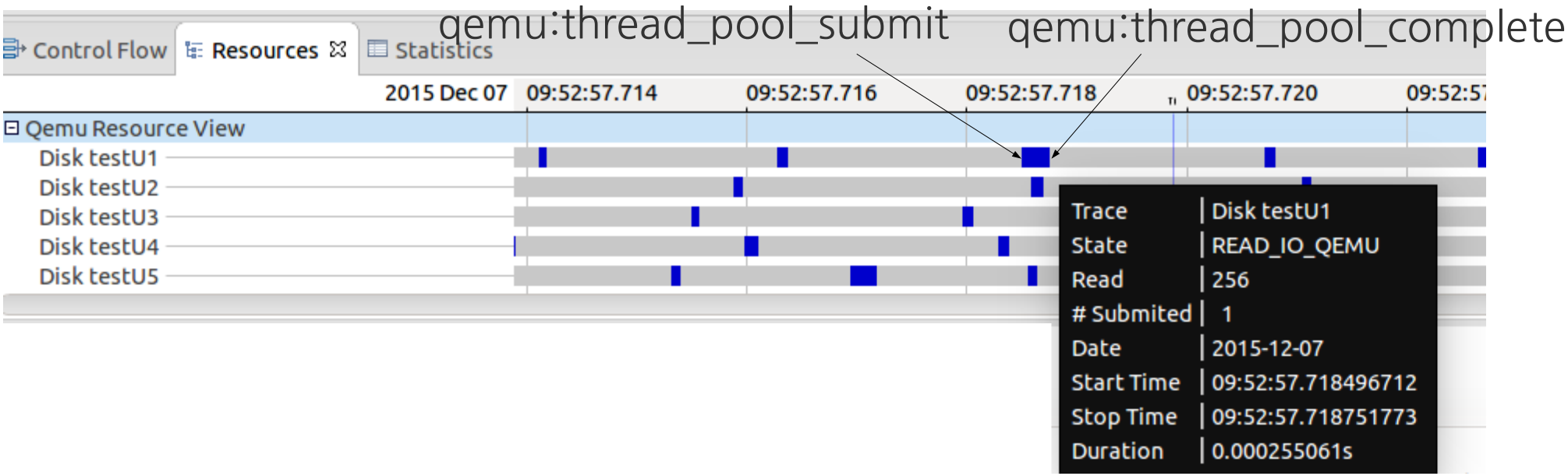
Resource Monitoring for virtual machine by host kernel tracing

- CPU Virtualization with intel-VT-x:
  - VMX transition
    - Between Guest and Virtual Machine Manager (VMM)
      - vm_entry , vm_exit
      - Virtual Machine Control Structure (VMCS)



Investigations

Resource Monitoring for virtual machine by host kernel tracing

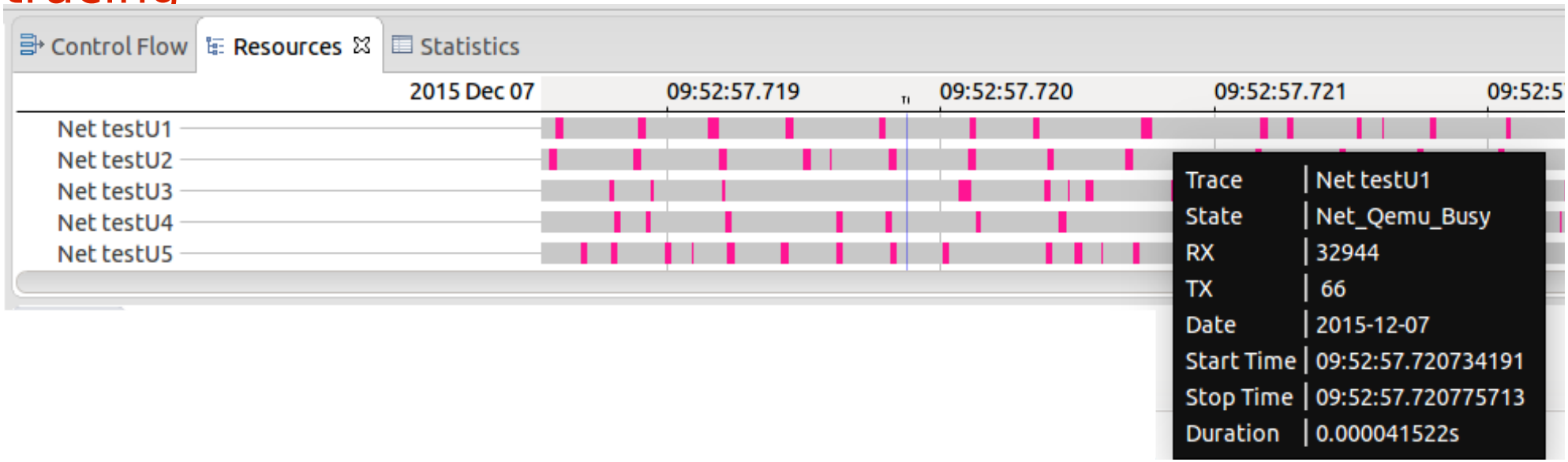


Metrics for Disk I/O:

- Number of submitted disk request(s) (read/write) for each or all VM(s)
- How much read/write is submitted by each or all VM(s)
- Duration of completing a disk request
- Variance of submitting disk requests by each VM
- Variance of completing disk requests by each VM

Investigations

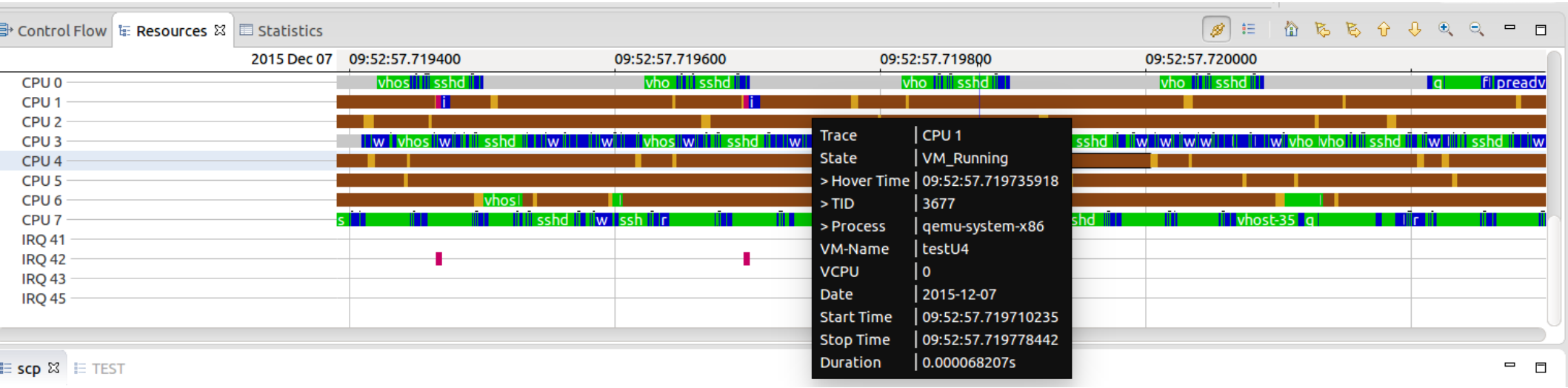
Resource Monitoring for virtual machine by host kernel tracing



- Metrics for Network:
  - Number of bytes received and transmitted
  - Variance of submitting network requests by each VM
  - Waiting time to receive/transmit packets

Investigations

Resource Monitoring for virtual machine by host kernel tracing



VM_Running

VMM_Running

Metrics for CPU:

- How many vCPUs are active
- The amount of time qemu thread is in root mode and non-root mode
- Most frequent exit reasons and their frequency
- How much pCPU is used by each VM
- Variance of exiting non-root mode
- The amount of time vCPU is preempted in each or all VM(s)
- Variance of preemption for each VM

Investigations

Demo

Investigations

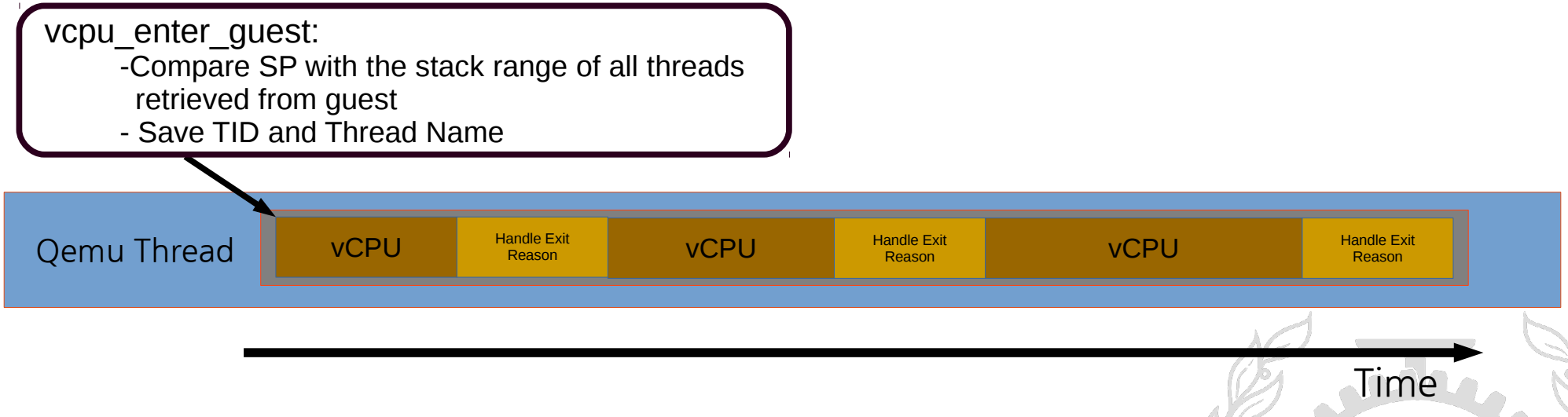
Monitoring Processes and threads inside virtual machine with host kernel tracing

- Guest Processor State is saved into VMCS [6]
  - Guest Register State
    - Control Register CR0, CR3, and CR4
    - Debug Register DR7
    - SP and IP and FLAGS and …
  - Guest non-Register State
    - Activity State
    - Interruptibility State and …

Investigations

Monitoring Processes and threads inside virtual machine with host kernel tracing

- New trace points:
  - vcpu_enter_guest to obtain unique tuple (CR3, SP, IP)
  - If you want **more** information:
    - lttng_statedump_stack to obtain thread stack range, thread ID, thread name, hostname



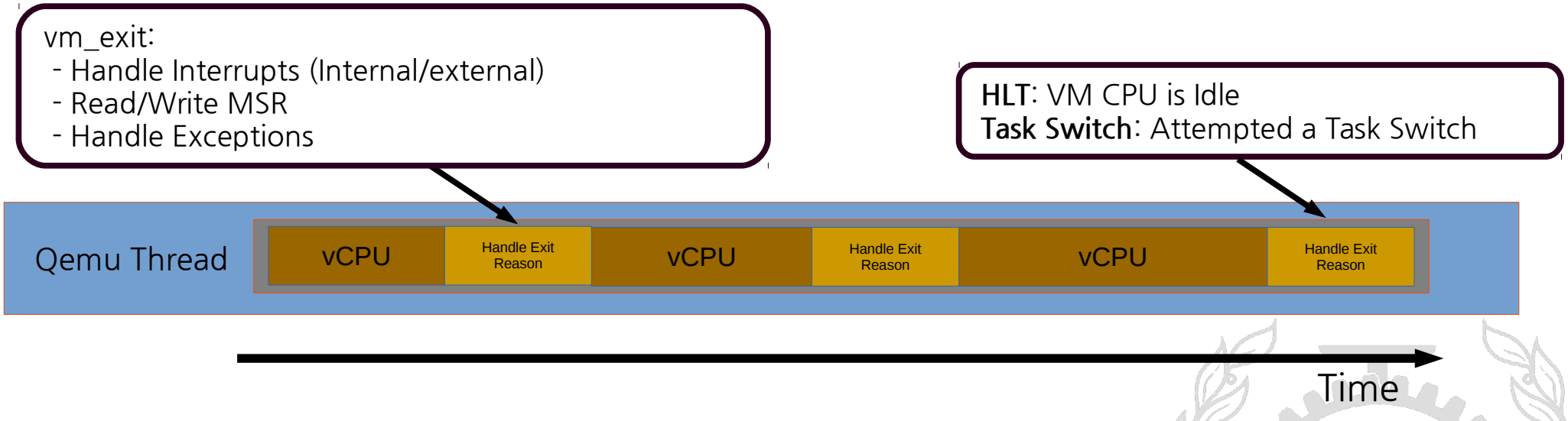
Investigations

One More Thing ...

Investigations

Monitoring Processes and threads inside virtual machine with host kernel tracing

- **What are these exit reasons?**
  - Schedule out qemu-system-x86 on Host without sending HLT or Task switch
  - VM assume that the task is still running (Preemption)



Investigations

Demo

Conclusion and in-progress

Inferences

- Fine-grained resource monitoring lets the cloud administrator identify the contention for resources.
- Monitoring processes and associated threads lets the cloud administrator find the amount of time a thread in the VM is actually running.
  - Preemption and too many switches between guest and VMM are causing latency in the VM.

Going Further

- Memory related usage view for each virtual machine by Host Kernel Tracing
- Add some performance indicators from hardware performance counter
- Identifying interference between resources automatically
- Building Control flow view of each virtual machine by Host Kernel Tracing

References

[1] Dean, D.; Nguyen, H.; Wang, P.; Gu, X.; Sailer, A.; Kochut, A., "PerfCompass: Online Performance Anomaly Fault Localization and Inference in Infrastructure-as-a-Service Clouds," in Parallel and Distributed Systems, IEEE Transactions on , vol.PP, no.99, pp.1-1

[2] Mukherjee, J.; Krishnamurthy, D.; Rolia, J., "Resource Contention Detection in Virtualized Environments," in Network and Service Management, IEEE Transactions on , vol.12, no.2, pp.217-231, June 2015

[3] D. Novakovic , N. Vasic , S. Novakovic , D. Kostic and R. Bianchini "DeepDive: Transparently identifying and managing performance interference in virtualized environments", Proc. Annu. Tech. Conf., pp.219 -230 2013

[4] X. Zhang, E. Tune, R. Hagmann, R. Jnagal, V. Gokhale, and J. Wilkes, “CPI2: CPU performance isolation for shared compute clusters,” in Proceedings of the 8th ACM European Conference on Computer Systems, ser. EuroSys ’13. New York, NY, USA: ACM, 2013, pp. 379-391.

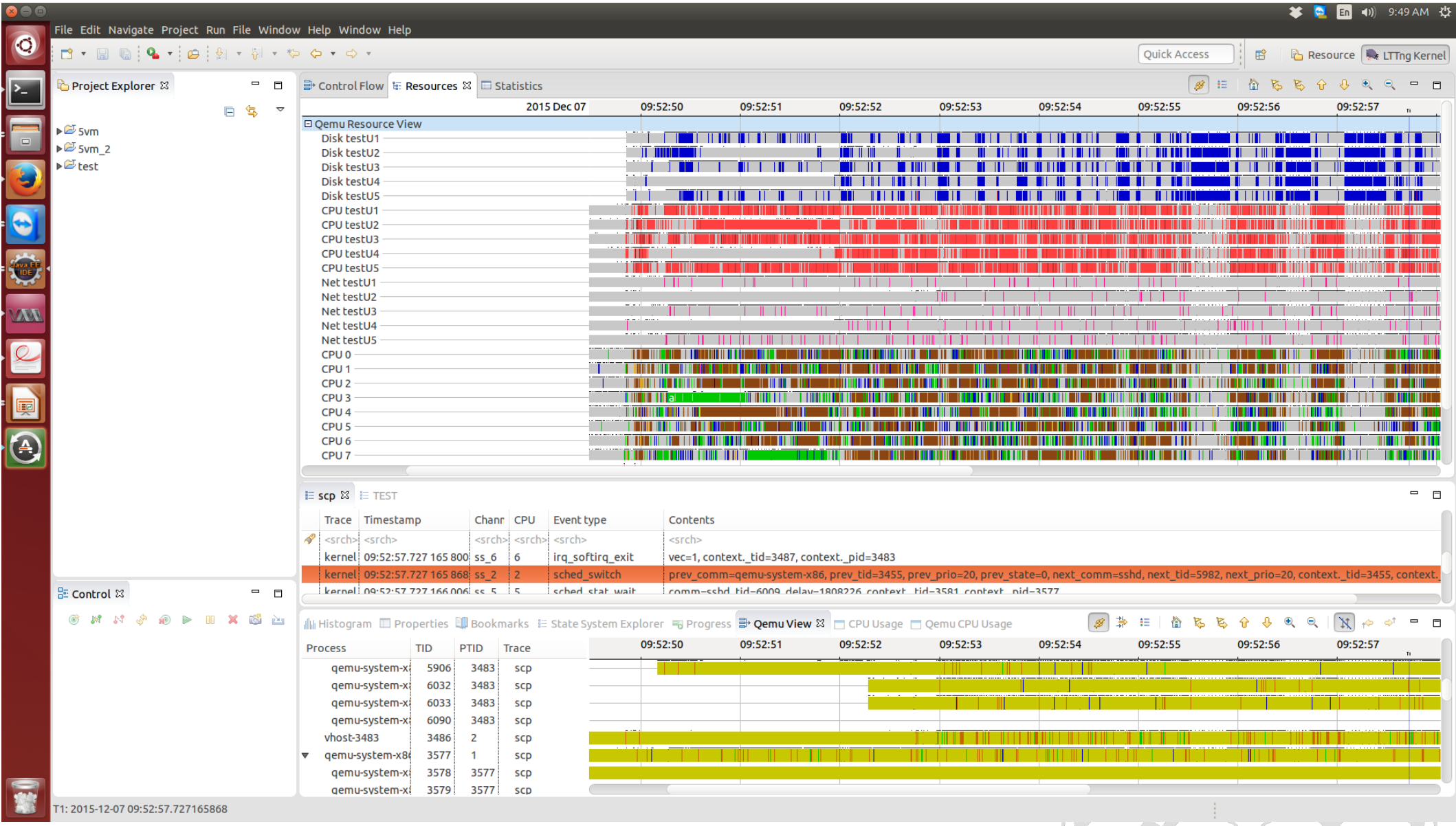
[5] Intel 64 and IA-32 Architectures Software Develope's Manual, Volume 3B, System Programming Guide, Part 2

Questions?

Hani.nemati@polymtl.ca

https://github.com/Nemati/Trace-Compass

Demo


File Edit Navigate Project Run File Window Help Window Help
Quick Access
Resource
LTTng Kernel
Project Explorer
5vm
5vm_2
test
Control Flow
Resources
Statistics
2015 Dec 07
09:52:50
09:52:51
09:52:52
09:52:53
09:52:54
09:52:55
09:52:56
09:52:57
Qemu Resource View
Disk testU1
Disk testU2
Disk testU3
Disk testU4
Disk testU5
CPU testU1
CPU testU2
CPU testU3
CPU testU4
CPU testU5
Net testU1
Net testU2
Net testU3
Net testU4
Net testU5
CPU 0
CPU 1
CPU 2
CPU 3
CPU 4
CPU 5
CPU 6
CPU 7
scp
TEST
Trace
Timestamp
Chanr
CPU
Event type
Contents
kernel 09:52:57.727 165 800 ss_6 6 irq_softirq_exit vec=1, context._tid=3487, context._pid=3483
kernel 09:52:57.727 165 868 ss_2 2 sched_switch prev_comm=qemu-system-x86, prev_tid=3455, prev_prio=20, prev_state=0, next_comm=sshd, next_tid=5982, next_prio=20, context._tid=3455, context.
Control
Histogram
Properties
Bookmarks
State System Explorer
Progress
Qemu View
CPU Usage
Qemu CPU Usage
Process
TID
PTID
Trace
qemu-system-x
5906 3483 scp
6032 3483 scp
6033 3483 scp
6090 3483 scp
vhost-3483 3486 2 scp
qemu-system-x86 3577 1 scp
3578 3577 scp
3579 3577 scp
T1: 2015-12-07 09:52:57.727165868

Demo

LTTng Kernel - 5vm_2/Experiments/scp/scp_ - Eclipse Platform

File Edit Navigate Project Run File Window Help Window Help

Project Explorer

- 5vm
- 5vm_2
- test

Control Flow | Resources | Statistics

2015 Dec 07 — 09:52:54.500 — 09:52:55.000 — 09:52:55.500 — 09:52:56.000 — 09:52:56.500 — 09:52:57.000

Qemu Resource View

Disk testU1, Disk testU2, Disk testU3, Disk testU4, Disk testU5, CPU testU1, CPU testU2, CPU testU3, CPU testU4, CPU testU5, Net testU1, Net testU2, Net testU3, Net testU4, Net testU5, CPU 0, CPU 1, CPU 2, CPU 3, CPU 4, CPU 5, CPU 6, CPU 7

Trace | CPU testU2

scp | TEST

Trace	Timestamp	Chanr	CPU	Event type	Contents
<srch>	<srch>	<srch>	<srch>	<srch>	<srch>
kernel	09:52:57.727 165 800	ss_6	6	irq_softirq_exit	vec=1, context._tid=3487, context._pid=3483
kernel	09:52:57.727 165 868	ss_2	2	sched_switch	prev_comm=qemu-system-x86, prev_tid=3455, prev_prio=20, prev_state=0, next_comm=sshd, next_tid=5982, next_prio=20, context._tid=3455, context.
kernel	09:52:57.727 166 006	ss_5	5	sched_stat_wait	comm=sshd, tid=6009, delay=1808226, context._tid=3581, context._pid=3577

Control

Histogram | Properties | Bookmarks | State System Explorer | Progress | Qemu View | CPU Usage | Qemu CPU Usage

TID	Process	%	Time
3673	qemu-system-x86	59.637 %	20909
3483	qemu-system-x86	58.652 %	20564
3768	qemu-system-x86	61.074 %	21413
3577	qemu-system-x86	60.604 %	21248
3450	qemu-system-x86	61.527 %	21572

CPU usage

% CPU: 600, 400, 200, 0

09:52:54.163 630 386 — 09:52:54.763 630 386 — 09:52:55.363 630 386 — 09:52:55.963 630 386 — 09:52:56.563 630 386 — 09:52:57.163 630 386

Time

Total, 3577, 3450, 3673, 3483, 3768

Demo

LTTng Kernel - 5vm_2/Experiments/scp/scp_ - Eclipse Platform

Project Explorer: 5vm, 5vm_2, test

Control Flow | Resources | Statistics

Qemu Resource View: Disk testU1–U5, CPU testU1–U5, Net testU1–U5, CPU 0–7

Trace	CPU 5
State	SYSCALL
> Hover Time	09:52:55.576665884
> TID	5821
> Process	lttng-consumerd
> System Call	splice
Date	2015-12-07
Start Time	09:52:55.576630056
Stop Time	09:52:55.576700045
Duration	0.000069989s

scp | TEST

Trace	Timestamp	Chanr	CPU	Event type	Contents
kernel	09:52:57.727 165 800	ss_6	6	irq_softirq_exit	vec=1, context._tid
kernel	09:52:57.727 165 868	ss_2	2	sched_switch	prev_comm=qemu …=20, prev_state=0, next_comm=sshd, next_tid=5982, next_prio=20, context._tid=3455, context.

Histogram | Properties | Bookmarks | State System Explorer | Progress | Qemu View | CPU Usage | Qemu CPU Usage

Process	TID	PTID	Trace
qemu-system-x	5906	3483	scp
qemu-system-x	6032	3483	scp
qemu-system-x	6033	3483	scp
qemu-system-x	6090	3483	scp
vhost-3483	3486	2	scp
qemu-system-x86	3577	1	scp
qemu-system-x	3578	3577	scp
qemu-system-x	3579	3577	scp

T: 2015-12-07 09:52:55.576665884 T1: 2015-12-07 09:52:57.727165868

Demo

File Edit Navigate Project Run File Window Help Window Help

Quick Access | Resource | LTTng Kernel

Project Explorer
- 5vm
- 5vm_2
- test

Control Flow | Resources | Statistics

2015 Dec 07 — 09:52:57.731000 — 09:52:57.731500 — 09:52:57.732000

Qemu Resource View
- Disk testU1
- Disk testU2
- Disk testU3
- Disk testU4
- Disk testU5
- CPU testU1
- CPU testU2
- CPU testU3
- CPU testU4
- CPU testU5
- Net testU1
- Net testU2
- Net testU3
- Net testU4
- Net testU5
- CPU 0
- CPU 1
- CPU 2
- CPU 3
- CPU 4
- CPU 5
- CPU 6
- CPU 7

scp | TEST

Trace	Timestamp	Chanr	CPU	Event type	Contents
<srch>	<srch>	<srch>	<srch>	<srch>	<srch>
kernel	09:52:57.727 165 800	ss_6	6	irq_softirq_exit	vec=1, context._tid=3487, context._pid=3483
kernel	09:52:57.727 165 868	ss_2	2	sched_switch	prev_comm=qemu-system-x86, prev_tid=3455, prev_prio=20, prev_state=0, next_comm=sshd, next_tid=5982, next_prio=20, context._tid=3455, context._
kernel	09:52:57.727 166 006	ss_5	5	sched_stat_wait	comm=sshd, tid=6009, delay=1808226, context._tid=3581, context._pid=3577

Control

Histogram | Properties | Bookmarks | State System Explorer | Progress | Qemu View | CPU Usage | Qemu CP

Process	TID	PTID	Trace
qemu-system-x	5906	3483	scp
qemu-system-x	6032	3483	scp
qemu-system-x	6033	3483	scp
qemu-system-x	6090	3483	scp
vhost-3483	3486	2	scp
qemu-system-x86	3577	1	scp
qemu-system-x	3578	3577	scp
qemu-system-x	3579	3577	scp

09:52:57.731000 — 09:52:57.732000

Process	vhost-3483
State	WAIT_FOR_CPU
Date	2015-12-07
Start Time	09:52:57.731020856
Stop Time	09:52:57.731947300
Duration	0.000926444s

T1: 2015-12-07 09:52:57.727165868

Demo

File Edit Navigate Project Run File Window Help Window Help

Quick Access | Resource | LTTng Kernel

Project Explorer
- 5vm
- 5vm_2
- test

Control Flow | Resources | Statistics

2015 Dec 03 | 20:56:42.876500 | 20:56:42.877000 | 20:56:42.877500 | 20:56:42.878000

Qemu Resource View
- Disk testU1
- CPU testU1
- Net testU1
- CPU 0
- CPU 1
- CPU 2
- CPU 3
- CPU 4
- CPU 5
- CPU 6
- CPU 7
- IRQ 42
- IRQ 43
- IRQ 45

Trace	CPU 4
State	VMX_Running
> Hover Time	20:56:42.877578295
> TID	20435
> Process	qemu-system-x86
VM-Name	testU1
VCPU	0
T-ID-IN	1583
T-Name-IN	ssh
Exit Reason	I/O instruction
Date	2015-12-03
Start Time	20:56:42.877571613
Stop Time	20:56:42.877582718
Duration	0.000011105s

scp | TEST

Trace	Timestamp	Channel	CPU	Event type	Contents
<srch>	<srch>	<srch>	<srch>	<srch>	<srch>
nemati/hani-test-20151203-205636/kernel	20:56:42.875 948 921	ss_2	2	timer_cancel	timer=18446612165067142040, context._tid=27952, context._pid=27945
nemati/hani-test-20151203-205636/kernel	20:56:42.875 949 410	ss_3	3	rcu_utilization	s=End context switch, context._tid=0, context._pid=0
nemati/hani-test-20151203-205636/kernel	20:56:42.875 949 969	ss_2	2	timer_start	timer=18446612165067142040, function=18446744071585915600, expires=4366935578, now=4366935

Control

Histogram | Properties | Bookmarks | State System Explorer | Progress | Qemu View | CPU Usage | Qemu CPU Usage

Process	TID	PTID	Trace
qemu-system-x8	2043	1	TEST
qemu-system-x	2043	2043	TEST
qemu-system-x	2043	2043	TEST
qemu-system-x	2043	2043	TEST
qemu-system-x	2043	2043	TEST
qemu-system-x	2043	2043	TEST
qemu-system-x	2043	2043	TEST
qemu-system-x	2044	2043	TEST

T: 2015-12-03 20:56:42.877578295 T1: 2015-12-03 20:56:42.875949310

Questions?

Hani.nemati@polymtl.ca

https://github.com/Nemati/Trace-Compass