VM Analysis

Execution Path Analysis of Virtualized Environments using Host Kernel Tracing

Hani Nemati

Dec 7, 2017

Polytechnique Montréal
Laboratoire DORSAL
Agenda

Introduction

• Research update and research motivation

New Investigations

• Execution Path Analysis of virtualized environments using host kernel tracing
  • State of the art
  • Proposed Algorithm
  • Demo

Conclusion and in-progress
Previously on “VM Analysis”

Available Trace-Points in different layers
Previously on “VM Analysis”

Used Trace-Points in our approaches
Previously on “VM Analysis”

States for a Process inside VM

<table>
<thead>
<tr>
<th>Idle</th>
<th>Running</th>
</tr>
</thead>
<tbody>
<tr>
<td>Wait for pCPU</td>
<td>Preempted L0</td>
</tr>
<tr>
<td>kvm_inj_irq (Timer vec)</td>
<td>sched_switch</td>
</tr>
<tr>
<td>Wait for Timer</td>
<td></td>
</tr>
<tr>
<td>kvm_inj_irq (Task vec)</td>
<td></td>
</tr>
<tr>
<td>Wait for Task</td>
<td>sched_switch</td>
</tr>
<tr>
<td>kvm_inj_irq (Net vec)</td>
<td></td>
</tr>
<tr>
<td>Wait for Network</td>
<td>kvm_exit (non-hlt)</td>
</tr>
<tr>
<td>kvm_inj_irq (Disk vec)</td>
<td></td>
</tr>
<tr>
<td>Wait for Disk</td>
<td>kvm_entry</td>
</tr>
<tr>
<td>kvm_inj_irq (other vec)</td>
<td></td>
</tr>
<tr>
<td>Wait for Other</td>
<td></td>
</tr>
<tr>
<td>kvm_exit VMRESUME</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>VMX Non-ROOT L1</td>
</tr>
<tr>
<td></td>
<td>kvm_exit</td>
</tr>
<tr>
<td></td>
<td>kms_entry</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>VMX Non-ROOT L2</td>
<td>kvm_exit (non-hlt, CR3 changes)</td>
</tr>
</tbody>
</table>
Motivation

Why is the VM waiting?

![VMM Resource and Process View]

- Hypervisor Resource View
- VM
  - external/testU1
    - vCPU
      - CPU 0
      - CPU 1
  - vProcess
    - 1994059776
    - 2078416896
    - 2077757440
    - 2056454144
    - 2023870464
    - 887529472

- Tasks
- State System Explorer
- Qemu View
- Progress
- Control Flow
- Process View

<table>
<thead>
<tr>
<th>Process</th>
<th>TID</th>
</tr>
</thead>
<tbody>
<tr>
<td>qemu-system-x86</td>
<td>61</td>
</tr>
<tr>
<td>qemu-system-x86</td>
<td>61</td>
</tr>
<tr>
<td>qemu-system-x86</td>
<td>61</td>
</tr>
<tr>
<td>qemu-system-x86</td>
<td>61</td>
</tr>
<tr>
<td>qemu-system-x86</td>
<td>61</td>
</tr>
<tr>
<td>qemu-system-x86</td>
<td>61</td>
</tr>
<tr>
<td>qemu-system-x86</td>
<td>61</td>
</tr>
<tr>
<td>qemu-system-x86</td>
<td>61</td>
</tr>
<tr>
<td>qemu-system-x86</td>
<td>61</td>
</tr>
<tr>
<td>qemu-system-x86</td>
<td>71</td>
</tr>
<tr>
<td>qemu-system-x86</td>
<td>61</td>
</tr>
<tr>
<td>vhost-6503</td>
<td>61</td>
</tr>
</tbody>
</table>
Motivation

Let's use the Critical Flow view of Trace Compass?
Investigations

Methodology

sched_out(vCPU0)
sched_out(vCPU1)
inj_virq(vec 0)

vCPU0

vCPU1

Disk

Network

Process#1

Process#2

Process#3

Process#4

Process#5

<table>
<thead>
<tr>
<th>Timer</th>
<th>Task</th>
<th>Network</th>
<th>Disk</th>
<th>IDLE</th>
<th>Running</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Investigations

Methodology

sched_in(vCPU0)
inj_virq(vec 0 )
enter_guest(CR3 P#3)

vCPU0

vCPU1

Disk

Network

Process#1

Process#2

Process#3

Process#4

Process#5
Investigations

Methodology

sched_out(vCPU0)
sched_in(vCPU1)
inj_virq(vec 1)
enter_quest(CR3 P#1)
Investigations

Control Flow View

vCPU View
Investigations

Control Flow View

vProcess View
Investigations

Execution Flow Analysis View
# Investigations

## Slow Hadoop Slave

<table>
<thead>
<tr>
<th>Processes</th>
<th>Root</th>
<th>non-Root</th>
<th>Task</th>
<th>Timer</th>
<th>Disk</th>
<th>Net</th>
</tr>
</thead>
<tbody>
<tr>
<td>2039963648</td>
<td>0.004</td>
<td>1.499</td>
<td>54.040</td>
<td>43.512</td>
<td>0.198</td>
<td>0.217</td>
</tr>
<tr>
<td>869769216</td>
<td>0.001</td>
<td>0.979</td>
<td>39.938</td>
<td>56.978</td>
<td>1.251</td>
<td>0.700</td>
</tr>
<tr>
<td>2046287872</td>
<td>0.004</td>
<td>3.125</td>
<td>57.357</td>
<td>38.022</td>
<td>0.766</td>
<td>0.283</td>
</tr>
<tr>
<td>2029756416</td>
<td>0.002</td>
<td>1.898</td>
<td>36.508</td>
<td>60.883</td>
<td>0.098</td>
<td>0.340</td>
</tr>
<tr>
<td>877412352</td>
<td>0.001</td>
<td>0.970</td>
<td>24.947</td>
<td>72.767</td>
<td>0.029</td>
<td>1.130</td>
</tr>
<tr>
<td>886243328</td>
<td>0.005</td>
<td>8.350</td>
<td>85.240</td>
<td>5.588</td>
<td>0.003</td>
<td>0.258</td>
</tr>
</tbody>
</table>
# Investigations

## Slow Hadoop Slave

<table>
<thead>
<tr>
<th>Processes</th>
<th>Root</th>
<th>non-Root</th>
<th>Task</th>
<th>Timer</th>
<th>Disk</th>
<th>Net</th>
</tr>
</thead>
<tbody>
<tr>
<td>2039963648</td>
<td>0.004</td>
<td>1.499</td>
<td>54.040</td>
<td>43.512</td>
<td>0.198</td>
<td>0.217</td>
</tr>
<tr>
<td>869769216</td>
<td>0.001</td>
<td>0.979</td>
<td>39.938</td>
<td>56.978</td>
<td>1.251</td>
<td>0.700</td>
</tr>
<tr>
<td>2046287872</td>
<td>0.004</td>
<td>3.125</td>
<td>57.357</td>
<td>38.022</td>
<td>0.766</td>
<td>0.283</td>
</tr>
<tr>
<td>2029756416</td>
<td>0.002</td>
<td>1.898</td>
<td>36.508</td>
<td>60.883</td>
<td>0.098</td>
<td>0.340</td>
</tr>
<tr>
<td>877412352</td>
<td>0.001</td>
<td>0.970</td>
<td>24.947</td>
<td>72.767</td>
<td>0.029</td>
<td>1.130</td>
</tr>
<tr>
<td>886243328</td>
<td>0.005</td>
<td>8.350</td>
<td>85.240</td>
<td>5.588</td>
<td>0.003</td>
<td>0.258</td>
</tr>
</tbody>
</table>
Investigations

Cap on Network

Without Cap on Network
Investigations

Contention on Block Disk
Investigations

VM Boot-up comparison

![Graph showing VM boot-up time comparison across different configurations.](image)
Demo
Investigations

How to try these new features?

- Access to Host only
- Run LTTng on Host with my new added tracepoint (vcpu_enter_guest)
- Clone TraceCompass from github (incubator)
  - Open vCPU block View of TraceCompass (XML view)
  - Open vProcess block View of TraceCompass (XML view)
  - Use Execution Flow Analysis of TraceCompass
Investigations

One More Thing ...
Let’s make TraceCompass Intelligent
Demo
Conclusion and in-progress

Inferences

- Wait Analysis of process inside VM
  - A process is waiting for
    - A Disk Block request to finish
    - A Network packet to receive
    - Another process
    - A Timer to fire
    - Other devices

- Critical Path Analysis of process inside VM

In progress

- VM contention avoidance based on VM classification
Outcome of this project


[2] Hani Nemati, Genevieve Bastien, Michel Dagenais “Wait analysis of Virtual Machines using host kernel tracing”, Accepted at IEEE Cloud Summit 2018
Questions?

Hani.nemati@polymtl.ca
https://github.com/Nemati