



# Who stole my CPU?

Leonid Podolny

[leonid@digitalocean.com](mailto:leonid@digitalocean.com)

Vineeth Ramanan Pillai

[vineeth@digitalocean.com](mailto:vineeth@digitalocean.com)

Systems Engineering @ DigitalOcean





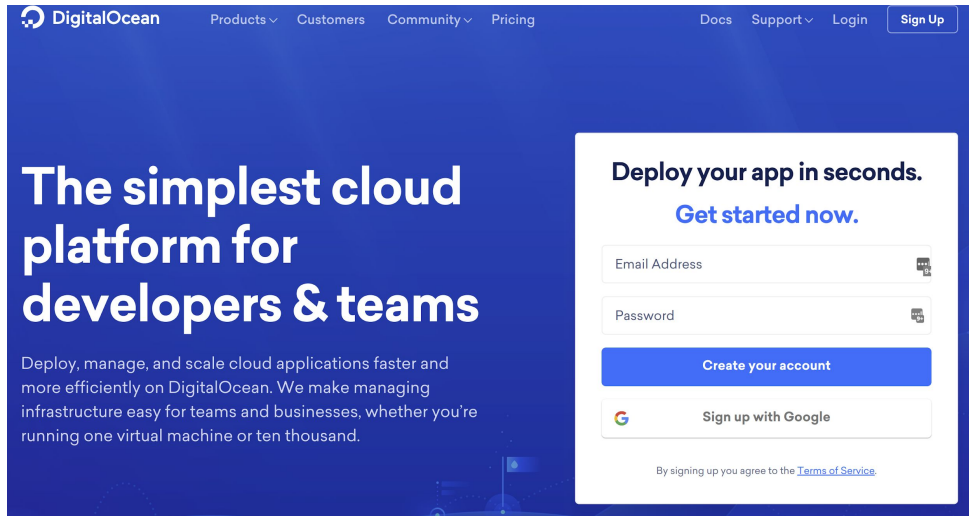
# Introduction

- DigitalOcean

Providing developers and businesses a reliable, easy-to-use cloud computing platform of virtual servers (Droplets), object storage (Spaces), and more.

- Systems Engineering

- Responsible for the Hypervisor and its software stack
- Host Operating System and kernel, KVM, qemu, libvirt and misc services to facilitate VM hosting





# Agenda

- Steal
  - Definitions
  - Causes
  - Analysis
  - Mitigation strategies
- Mitigation approach in DigitalOcean
  - Octopus: Implementation
  - Octopus: Issues and their resolutions
    - NUMA migrations problem
    - Swapoff enhancements



# What is steal?

steal, *n.* :

The fraction of time a vCPU had to wait for a physical CPU in a runqueue.

```
top - 02:09:40 up 173 days, 9:55, 2 users, load average: 14.27, 5.04, 1.82
Tasks: 381 total, 16 running, 365 sleeping, 0 stopped, 0 zombie
%Cpu(s): 69.1 us, 9.6 sy, 0.0 ni, 3.9 id, 0.1 wa, 0.0 hi, 0.0 si, 17.4 st
KiB Mem: 32946088 total, 22525292 used, 10420796 free, 921740 buffers
KiB Swap: 0 total, 0 used, 0 free. 18607000 cached Mem
```

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
29496	root	20	0	102284	77852	18256	R	21.6	0.2	0:00.65	cc1
29596	root	20	0	84536	62332	18256	R	12.6	0.2	0:00.38	cc1
29609	root	20	0	84668	62544	18032	R	12.3	0.2	0:00.37	cc1
29636	root	20	0	73828	46388	12872	R	12.3	0.1	0:00.37	cc1
29679	root	20	0	74208	46908	14160	R	9.0	0.1	0:00.27	cc1
29676	root	20	0	72196	44344	13884	R	8.6	0.1	0:00.26	cc1



## What is steal? (*cntd.*)

steal, *n.* :

- a vCPU is just another host task
- for a guest, the time stops
- exists only within the VM
- reported by the hypervisor



# Why Steal?

- Obvious reason: More runnable vCPUs than physical CPUs
- How can we explain steal when the hypervisor has enough CPU resources to satisfy the runnable but waiting vCPUs?



# Steal analysis

Steal can be analysed from two different perspectives

- VMs
  - Steal as observed from within the VM
  - Useful for determining if the steal is impacting the VM.
- Hypervisor
  - Sum of steal of individual VMs
  - Useful for determining mitigation approaches





# Steal as seen from the VM

- **Busy Steal**
  - CPU utilization + steal is close to 100%
  - VM could have made use of the stolen time had it not been stolen
- **Idle Steal**
  - Idle VM experiencing steal: utilization + steal is significantly below 100%
  - VM could not have used the stolen time even if available.



# Steal as seen from Hypervisor

Total Steal experienced by all VMs in a Hypervisor

- **Busy Steal**
  - More runnable vCPUs than physical CPUs
  - Not mitigatable in software
  - Migrating VMs out of the busy HV is the probable solution
- **Idle Steal**
  - Caused due to scheduler limitations, config issues etc
  - Mitigatable in software



# Hypervisor “idle” steal

```
top - 16:29:39 up 29 min, 2 users, load average: 1.99, 1.95, 1.64
Tasks: 105 total, 3 running, 102 sleeping, 0 stopped, 0 zombie
%Cpu(s): 49.8 us, 50.2 sy, 0.0 ni, 0.0 id, 0.0 wa, 0.0 hi, 0.0 si, 0.0 st
KiB Mem : 2048192 total, 1840772 free, 48760 used, 158660 buff/cache
KiB Swap: 0 total, 0 free, 0 used. 1840752 avail Mem
```

1vCPU VM

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
1912	root	20	0	46820	5600	3596	R	50.2	0.3	0:12.91	stress-ng-cpu
1913	root	20	0	45720	284	52	R	49.8	0.0	0:12.90	stress-ng-iosyn

```
top - 16:29:39 up 28 min, 2 users, load average: 1.97, 1.97, 1.65
Tasks: 105 total, 3 running, 102 sleeping, 0 stopped, 0 zombie
%Cpu(s): 46.8 us, 48.8 sy, 0.0 ni, 0.0 id, 0.0 wa, 0.0 hi, 0.0 si, 4.3 st
KiB Mem : 2048192 total, 1840896 free, 48548 used, 158748 buff/cache
KiB Swap: 0 total, 0 free, 0 used. 1840944 avail Mem
```

1vCPU VM

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
1928	root	20	0	46816	5608	3604	R	49.8	0.3	0:04.08	stress-ng-cpu

```
top - 16:29:39 up 28 min, 2 users, load average: 1.85, 1.99, 1.74
Tasks: 114 total, 3 running, 111 sleeping, 0 stopped, 0 zombie
%Cpu(s): 87.9 us, 11.8 sy, 0.0 ni, 0.0 id, 0.0 wa, 0.0 hi, 0.0 si, 0.3 st
KiB Mem : 2048056 total, 1693708 free, 49692 used, 304656 buff/cache
KiB Swap: 0 total, 0 free, 0 used. 1733924 avail Mem
```

2vCPU VM

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
1968	root	20	0	46816	5576	3572	R	100.0	0.3	0:10.64	stress-ng-cpu

```
top - 16:29:39 up 28 min, 2 users, load average: 0.00, 0.19, 0.52
Tasks: 134 total, 1 running, 133 sleeping, 0 stopped, 0 zombie
%Cpu(s): 0.2 us, 0.2 sy, 0.0 ni, 99.3 id, 0.0 wa, 0.0 hi, 0.0 si, 0.2 st
KiB Mem : 8174880 total, 7933596 free, 67252 used, 174032 buff/cache
KiB Swap: 0 total, 0 free, 0 used. 7872032 avail Mem
```

4vCPU VM

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
1382	root	20	0	65508	6144	5432	S	0.7	0.1	0:00.76	sshd

```
top - 16:29:39 up 28 min, 2 users, load average: 6.03, 5.83, 4.75
Tasks: 148 total, 7 running, 141 sleeping, 0 stopped, 0 zombie
%Cpu(s): 46.6 us, 31.3 sy, 0.0 ni, 0.4 id, 0.0 wa, 0.0 hi, 0.0 si, 21.7 st
KiB Mem : 16432144 total, 15567060 free, 148808 used, 716276 buff/cache
KiB Swap: 0 total, 0 free, 0 used. 15479948 avail Mem
```

6vCPU VM

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
2273	root	20	0	46952	5616	3612	R	100.0	0.0	0:16.88	stress-ng-cpu

1:bash# 2:bash#

```
top - 16:29:39 up 28 min, 2 users, load average: 1.95, 1.95, 1.65
Tasks: 105 total, 3 running, 102 sleeping, 0 stopped, 0 zombie
%Cpu(s): 38.7 us, 39.0 sy, 0.0 ni, 22.0 id, 0.0 wa, 0.0 hi, 0.0 si, 0.3 st
KiB Mem : 2048192 total, 1839932 free, 48904 used, 159356 buff/cache
KiB Swap: 0 total, 0 free, 0 used. 1840488 avail Mem
```

1vCPU VM

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
1935	root	20	0	46816	5568	3568	R	38.5	0.3	0:01.16	stress-ng-cpu
1936	root	20	0	45720	280	52	R	38.5	0.0	0:01.16	stress-ng-iosyn

```
top - 16:29:39 up 28 min, 2 users, load average: 1.86, 1.93, 1.64
Tasks: 106 total, 3 running, 103 sleeping, 0 stopped, 0 zombie
%Cpu(s): 50.2 us, 49.8 sy, 0.0 ni, 0.0 id, 0.0 wa, 0.0 hi, 0.0 si, 0.0 st
KiB Mem : 2048192 total, 1839568 free, 49312 used, 159312 buff/cache
KiB Swap: 0 total, 0 free, 0 used. 1840056 avail Mem
```

1vCPU VM

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
1928	root	20	0	45720	284	52	R	50.2	0.0	0:07.50	stress-ng-iosyn

```
top - 16:29:39 up 28 min, 2 users, load average: 1.98, 2.01, 1.71
Tasks: 114 total, 3 running, 111 sleeping, 0 stopped, 0 zombie
%Cpu(s): 61.3 us, 4.5 sy, 0.0 ni, 34.1 id, 0.0 wa, 0.0 hi, 0.2 si, 0.0 st
KiB Mem : 2048056 total, 1570108 free, 50296 used, 427652 buff/cache
KiB Swap: 0 total, 0 free, 0 used. 1575244 avail Mem
```

2vCPU VM

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
1928	root	20	0	46816	5624	3624	R	53.0	0.3	0:01.59	stress-ng-cpu

```
top - 16:29:39 up 28 min, 2 users, load average: 4.86, 4.67, 3.64
Tasks: 132 total, 6 running, 126 sleeping, 0 stopped, 0 zombie
%Cpu(s): 63.8 us, 24.6 sy, 0.0 ni, 4.0 id, 0.0 wa, 0.0 hi, 0.0 si, 7.6 st
KiB Mem : 8174880 total, 7410564 free, 64556 used, 699760 buff/cache
KiB Swap: 0 total, 0 free, 0 used. 7348728 avail Mem
```

4vCPU VM

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
2132	root	20	0	307996	263728	263492	R	96.3	3.2	0:02.90	stress-ng-vm

```
top - 16:29:39 up 28 min, 2 users, load average: 5.46, 5.71, 4.71
Tasks: 147 total, 7 running, 140 sleeping, 0 stopped, 0 zombie
%Cpu(s): 47.7 us, 24.1 sy, 0.0 ni, 5.7 id, 0.0 wa, 0.0 hi, 0.0 si, 22.5 st
KiB Mem : 16432144 total, 15640104 free, 81884 used, 710156 buff/cache
KiB Swap: 0 total, 0 free, 0 used. 15549920 avail Mem
```

6vCPU VM

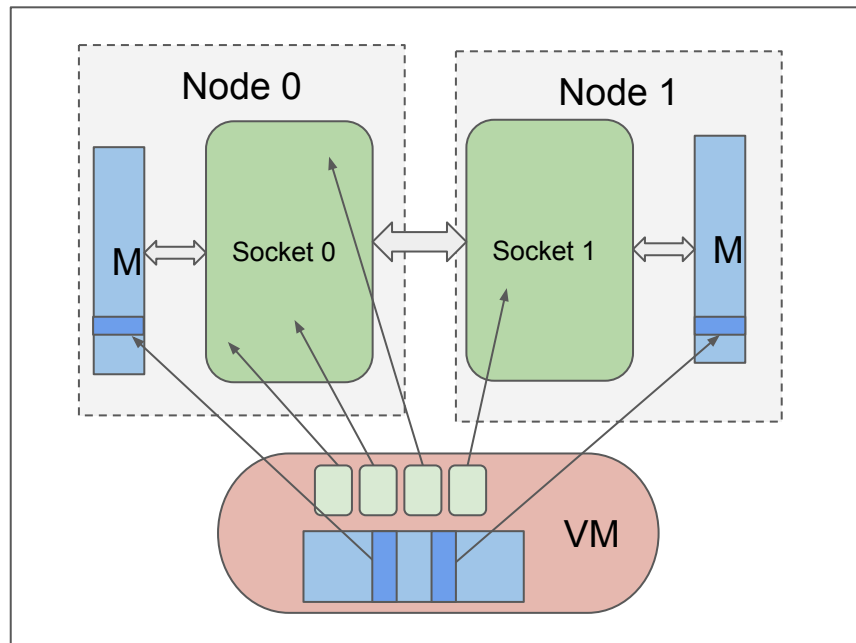
PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
2252	root	20	0	46952	5888	3908	R	94.0	0.0	0:02.83	stress-ng-cpu

16054/128891MB [||||||| ] 81.6% 24.32 25.48 22.17 95 days



# Hypervisor “idle” steal: NUMA balancing

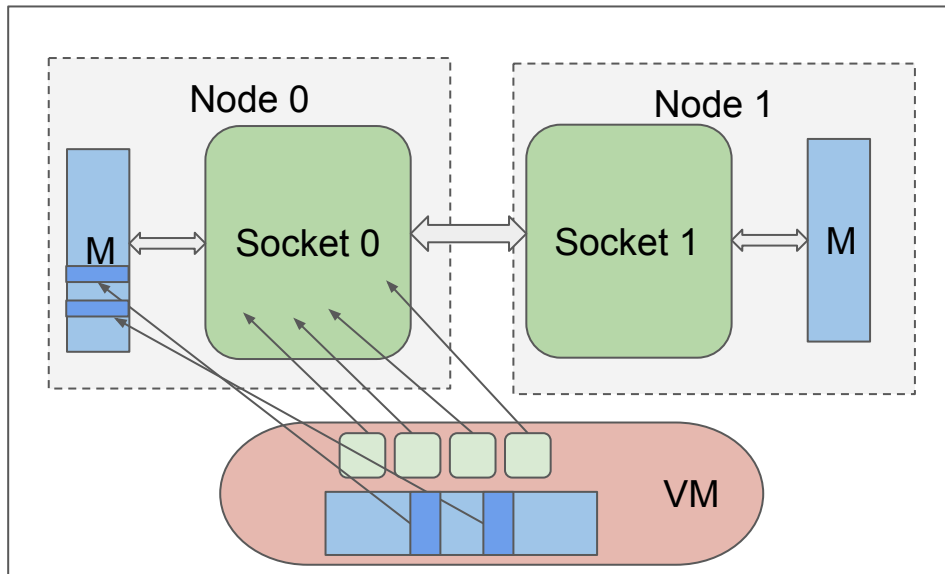
- VMs span all the NUMA nodes by default.
- Linux automatic NUMA balancing: keep tasks closer to their memory
  - Memory follow CPU model
  - CPU follow memory model
- Migration threads takes up cpu, resulting in steal





# Hypervisor “idle” steal: NUMA balancing

- Mitigation
  - Pin VMs to NUMA nodes and
  - Disable NUMA balancing





# Hypervisor “idle” steal

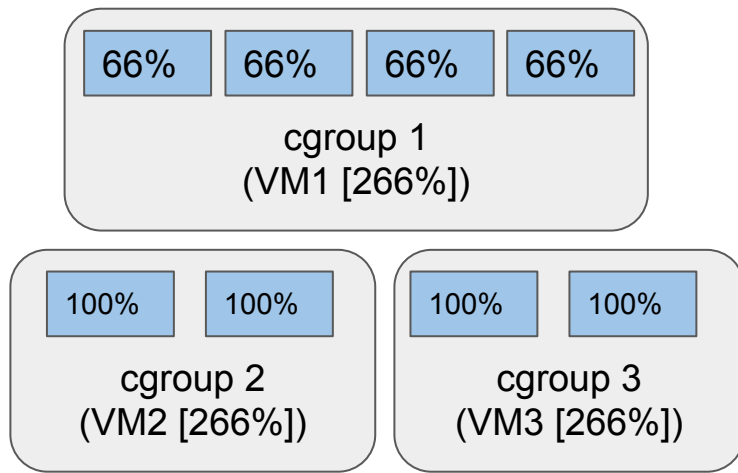
## (After disabling automatic NUMA balancing)

<pre>top - 16:42:02 up 41 min, 2 users, load average: 1.98, 1.96, 1.83 Tasks: 105 total, 3 running, 102 sleeping, 0 stopped, 0 zombie %Cpu(s): 48.5 us, 50.2 sy, 0.0 ni, 0.0 id, 0.0 wa, 0.0 hi, 0.0 si, 1.3 st KiB Mem : 2048192 total, 1840720 free, 48704 used, 158768 buff/cache KiB Swap: 0 total, 0 free, 0 used, 1840780 avail Mem  PID USER      PR  NI  VIRT  RES  SHR  S %CPU  %MEM    TIME+  COMMAND 2036 root        20   0 46820 5624 3624  R 49.8   0.3   0:12.10 stress-ng-cpu 2037 root        20   0 45720 284    52  R 49.8   0.0   0:12.10 stress-ng-iosyn</pre>	<pre>top - 16:42:02 up 41 min, 2 users, load average: 1.94, 1.95, 1.83 Tasks: 105 total, 3 running, 102 sleeping, 0 stopped, 0 zombie %Cpu(s): 32.5 us, 32.8 sy, 0.0 ni, 33.1 id, 0.0 wa, 0.0 hi, 0.0 si, 1.7 st KiB Mem : 2048192 total, 1841648 free, 46748 used, 159796 buff/cache KiB Swap: 0 total, 0 free, 0 used, 1842592 avail Mem  PID USER      PR  NI  VIRT  RES  SHR  S %CPU  %MEM    TIME+  COMMAND 2107 root        20   0 45736 2004 1700  R 4.3   0.1   0:00.13 stress-ng-cpu 2108 root        20   0 45720 284    52  R 4.3   0.0   0:00.13 stress-ng-iosyn</pre>
<pre>top - 16:42:02 up 41 min, 2 users, load average: 1.94, 1.96, 1.83 Tasks: 105 total, 3 running, 102 sleeping, 0 stopped, 0 zombie %Cpu(s): 49.7 us, 50.3 sy, 0.0 ni, 0.0 id, 0.0 wa, 0.0 hi, 0.0 si, 0.0 st KiB Mem : 2048192 total, 1840836 free, 48536 used, 158820 buff/cache KiB Swap: 0 total, 0 free, 0 used, 1840956 avail Mem  PID USER      PR  NI  VIRT  RES  SHR  S %CPU  %MEM    TIME+  COMMAND 2051 root        20   0 46820 5688 3692  R 49.8   0.3   0:02.89 stress-ng-cpu</pre>	<pre>top - 16:42:02 up 41 min, 2 users, load average: 1.98, 1.97, 1.84 Tasks: 105 total, 3 running, 102 sleeping, 0 stopped, 0 zombie %Cpu(s): 49.8 us, 49.5 sy, 0.0 ni, 0.0 id, 0.0 wa, 0.0 hi, 0.0 si, 0.7 st KiB Mem : 2048192 total, 1840124 free, 48772 used, 159296 buff/cache KiB Swap: 0 total, 0 free, 0 used, 1840664 avail Mem  PID USER      PR  NI  VIRT  RES  SHR  S %CPU  %MEM    TIME+  COMMAND 2049 root        20   0 46816 5596 3596  R 50.0   0.3   0:06.61 stress-ng-cpu</pre>
<pre>top - 16:42:02 up 41 min, 2 users, load average: 1.84, 1.94, 1.87 Tasks: 114 total, 3 running, 111 sleeping, 0 stopped, 0 zombie %Cpu(s): 91.0 us, 8.5 sy, 0.0 ni, 0.0 id, 0.0 wa, 0.0 hi, 0.0 si, 0.5 st KiB Mem : 2048056 total, 1534428 free, 50136 used, 1463492 buff/cache KiB Swap: 0 total, 0 free, 0 used, 1575148 avail Mem  PID USER      PR  NI  VIRT  RES  SHR  S %CPU  %MEM    TIME+  COMMAND 2118 root        20   0 46816 5620 3620  R 100.0   0.3   0:08.19 stress-ng-cpu</pre>	<pre>top - 16:42:02 up 41 min, 2 users, load average: 1.98, 1.96, 1.85 Tasks: 111 total, 1 running, 110 sleeping, 0 stopped, 0 zombie %Cpu(s): 94.9 us, 0.7 sy, 0.0 ni, 4.3 id, 0.0 wa, 0.0 hi, 0.0 si, 0.2 st KiB Mem : 2048056 total, 1836900 free, 48188 used, 162968 buff/cache KiB Swap: 0 total, 0 free, 0 used, 1841904 avail Mem  PID USER      PR  NI  VIRT  RES  SHR  S %CPU  %MEM    TIME+  COMMAND 1 root        20   0 37952 6032 4044  S 0.0   0.3   0:02.47 systemd</pre>
<pre>top - 16:42:02 up 40 min, 2 users, load average: 0.00, 0.01, 0.22 Tasks: 128 total, 1 running, 127 sleeping, 0 stopped, 0 zombie %Cpu(s): 0.2 us, 0.2 sy, 0.0 ni, 99.3 id, 0.0 wa, 0.0 hi, 0.0 si, 0.3 st KiB Mem : 8174880 total, 7936800 free, 63644 used, 174436 buff/cache KiB Swap: 0 total, 0 free, 0 used, 7875508 avail Mem  PID USER      PR  NI  VIRT  RES  SHR  S %CPU  %MEM    TIME+  COMMAND 1 root        20   0 37996 6064 4028  S 0.0   0.1   0:02.60 systemd</pre>	<pre>top - 16:42:02 up 40 min, 2 users, load average: 4.91, 4.84, 4.32 Tasks: 132 total, 16 running, 126 sleeping, 0 stopped, 0 zombie %Cpu(s): 41.2 us, 17.2 sy, 0.0 ni, 37.2 id, 0.0 wa, 0.0 hi, 0.1 si, 4.3 st KiB Mem : 8174880 total, 7861040 free, 62832 used, 251008 buff/cache KiB Swap: 0 total, 0 free, 0 used, 7798960 avail Mem  PID USER      PR  NI  VIRT  RES  SHR  S %CPU  %MEM    TIME+  COMMAND 2375 root        20   0 45852 1860 1548  R 2.7   0.0   0:00.08 stress-ng-cpu</pre>
<pre>top - 16:42:02 up 40 min, 2 users, load average: 5.57, 5.77, 5.33 Tasks: 140 total, 6 running, 142 sleeping, 0 stopped, 0 zombie %Cpu(s): 58.7 us, 37.1 sy, 0.0 ni, 0.2 id, 0.0 wa, 0.0 hi, 0.0 si, 4.0 st KiB Mem : 16432144 total, 15902624 free, 82868 used, 446652 buff/cache KiB Swap: 0 total, 0 free, 0 used, 15815736 avail Mem  PID USER      PR  NI  VIRT  RES  SHR  S %CPU  %MEM    TIME+  COMMAND 2543 root        20   0 46952 5604 3600  R 100.0   0.0   0:13.63 stress-ng-cpu</pre>	<pre>top - 16:42:02 up 40 min, 2 users, load average: 5.48, 5.77, 5.33 Tasks: 139 total, 1 running, 138 sleeping, 0 stopped, 0 zombie %Cpu(s): 43.6 us, 21.3 sy, 0.0 ni, 31.9 id, 0.0 wa, 0.0 hi, 0.0 si, 3.2 st KiB Mem : 16432144 total, 16170096 free, 79304 used, 182744 buff/cache KiB Swap: 0 total, 0 free, 0 used, 16079524 avail Mem  PID USER      PR  NI  VIRT  RES  SHR  S %CPU  %MEM    TIME+  COMMAND 18 root        20   0 0 0 0  S 0.3   0.0   0:00.08 ksoftirqd/2</pre>
<pre>1: bash* 2: bash#</pre>	<pre>16026/128891MB [        ] 83.5% 24.28 24.76 23.49 95 days</pre>



# Hypervisor “idle” steal: process grouping

- Cgroups
  - Default cgroups created by libvirt is per-VM
  - Bigger VMs are considered equal in weight to smaller VMs due to per-VM cgroups
- Example: 8 cpu Hypervisor
  - 8 cpu (800%)
  - 3 VMs
    - 1x 4-vCPU
    - 2x 2-vCPU





# Hypervisor “idle” steal: process grouping (*cntd.*)

- Mitigation
  - Disable CPU cgroups for VMs
- Side Effects
  - Loses the capability to control VM cpu utilization
  - Autogroup feature kicks in
  - Disabling autogroup feature works fine for newly launched VMs, but running VMs are still managed by the autogroup





# Hypervisor “idle” steal: process grouping (Contd...)

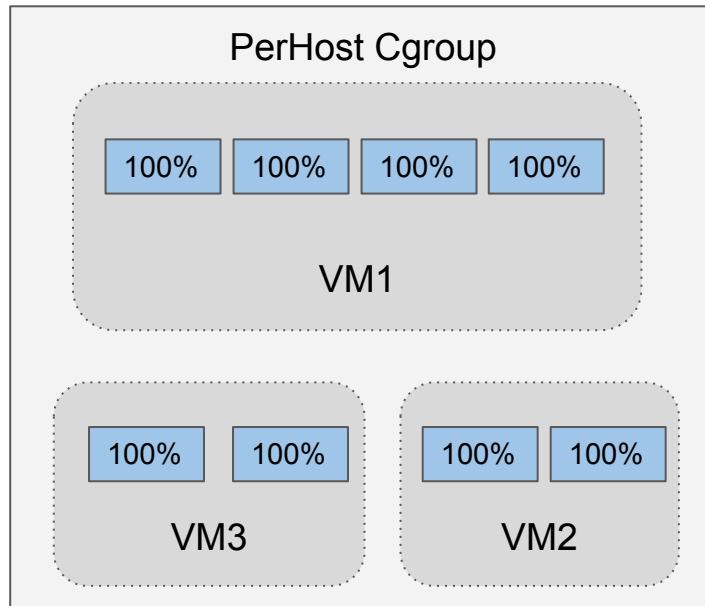
- Working solution: per-host VM Cgroup
  - Consolidate all vCPUs in one cgroup.
  - CFS allocates CPU proportionally to the number of vCPUs.
- Modified libvirt
  - A tunable to consolidate all vCPUS to one cpu cgroup and all cgroups parameters are same(default) for all VMs
    - `cfs.cfs_period_us, cfs.cfs_quota_us`
  - Moves a VM to its own cgroup if any cgroup parameters modified for a VM
  - Moves the VM back to perhost cgroup if the cgroup parameters are reverted to default



# Hypervisor “idle” steal: process grouping (Contd...)

## per-host Cgroup Example

- 8 cpu Hypervisor
  - 8 cpu (800%)
  - 3 VMs
    - 1x 4-vCPU
    - 2x 2-vCPU





# Hypervisor "idle" steal: per-host cgroup

```
top - 20:03:55 up 4:03, 2 users, load average: 1.96, 1.95, 2.00 1vCPU VM
Tasks: 107 total, 3 running, 104 sleeping, 0 stopped, 0 zombie
%Cpu(s): 49.8 us, 49.8 sy, 0.0 ni, 0.0 id, 0.0 wa, 0.0 hi, 0.0 si, 0.3 st
KiB Mem : 2048192 total, 1837600 free, 50032 used, 160560 buff/cache
KiB Swap: 0 total, 0 free, 0 used, 1839380 avail Mem
```

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
4113	root	20	0	46816	5748	3756	R	49.5	0.3	0:11.89	stress-ng-cpu
4114	root	20	0	45720	284	52	R	49.5	0.0	0:11.88	stress-ng-iosyn

```
top - 20:03:55 up 4:03, 2 users, load average: 1.94, 1.96, 2.00 1vCPU VM
Tasks: 105 total, 3 running, 102 sleeping, 0 stopped, 0 zombie
%Cpu(s): 48.2 us, 48.2 sy, 0.0 ni, 0.0 id, 0.0 wa, 0.0 hi, 0.0 si, 3.7 st
KiB Mem : 2048192 total, 1839544 free, 48664 used, 159984 buff/cache
KiB Swap: 0 total, 0 free, 0 used, 1840836 avail Mem
```

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
4123	root	20	0	45720	288	52	R	50.3	0.0	0:02.01	stress-ng-iosyn

```
top - 20:03:55 up 4:02, 2 users, load average: 1.83, 1.94, 1.99 2vCPU VM
Tasks: 114 total, 3 running, 111 sleeping, 0 stopped, 0 zombie
%Cpu(s): 61.6 us, 3.5 sy, 0.0 ni, 34.6 id, 0.0 wa, 0.0 hi, 0.0 si, 0.3 st
KiB Mem : 2048056 total, 1533872 free, 50076 used, 464108 buff/cache
KiB Swap: 0 total, 0 free, 0 used, 1575232 avail Mem
```

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
4561	root	20	0	46816	5612	3612	R	39.9	0.3	0:01.20	stress-ng-cpu

```
top - 20:03:55 up 4:02, 1 user, load average: 0.00, 0.00, 0.00 4vCPU VM
Tasks: 124 total, 1 running, 123 sleeping, 0 stopped, 0 zombie
%Cpu(s): 0.0 us, 0.0 sy, 0.0 ni, 100.0 id, 0.0 wa, 0.0 hi, 0.0 si, 0.0 st
KiB Mem : 8174880 total, 7931996 free, 59820 used, 183064 buff/cache
KiB Swap: 0 total, 0 free, 0 used, 7875020 avail Mem
```

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
1	root	20	0	37996	6064	4028	S	0.0	0.1	0:03.48	systemd

```
top - 20:03:55 up 4:02, 2 users, load average: 5.81, 5.80, 5.82 6vCPU VM
Tasks: 147 total, 6 running, 141 sleeping, 0 stopped, 0 zombie
%Cpu(s): 41.3 us, 21.1 sy, 0.0 ni, 36.3 id, 0.0 wa, 0.0 hi, 0.0 si, 1.2 st
KiB Mem : 16432144 total, 16046668 free, 78660 used, 306816 buff/cache
KiB Swap: 0 total, 0 free, 0 used, 15959396 avail Mem
```

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
6940	root	20	0	45860	1988	1692	R	5.6	0.0	0:00.17	stress-ng-cpu

1: bash\* 2: bash#

```
top - 20:03:55 up 4:03, 2 users, load average: 1.94, 1.95, 2.00 1vCPU VM
Tasks: 105 total, 3 running, 102 sleeping, 0 stopped, 0 zombie
%Cpu(s): 49.2 us, 49.5 sy, 0.0 ni, 0.0 id, 0.0 wa, 0.0 hi, 0.0 si, 1.3 st
KiB Mem : 2048192 total, 1837000 free, 48824 used, 162368 buff/cache
KiB Swap: 0 total, 0 free, 0 used, 1840472 avail Mem
```

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
4299	root	20	0	46816	5596	3596	R	50.0	0.3	0:14.91	stress-ng-cpu
4300	root	20	0	45720	280	52	R	49.7	0.0	0:14.91	stress-ng-iosyn

```
top - 20:03:55 up 4:03, 2 users, load average: 2.00, 1.97, 2.00 1vCPU VM
Tasks: 105 total, 3 running, 102 sleeping, 0 stopped, 0 zombie
%Cpu(s): 50.3 us, 49.3 sy, 0.0 ni, 0.0 id, 0.0 wa, 0.0 hi, 0.0 si, 0.3 st
KiB Mem : 2048192 total, 1838948 free, 48872 used, 160372 buff/cache
KiB Swap: 0 total, 0 free, 0 used, 1840548 avail Mem
```

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
4114	root	20	0	46820	5608	3608	R	49.7	0.3	0:05.74	stress-ng-cpu

```
top - 20:03:55 up 4:02, 2 users, load average: 1.99, 1.97, 2.00 2vCPU VM
Tasks: 114 total, 3 running, 111 sleeping, 0 stopped, 0 zombie
%Cpu(s): 99.7 us, 0.0 sy, 0.0 ni, 0.0 id, 0.0 wa, 0.0 hi, 0.0 si, 0.3 st
KiB Mem : 2048056 total, 1569256 free, 50500 used, 428300 buff/cache
KiB Swap: 0 total, 0 free, 0 used, 1575012 avail Mem
```

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
4507	root	20	0	46820	5648	3644	R	99.7	0.3	0:22.78	stress-ng-cpu

```
top - 20:03:55 up 4:02, 2 users, load average: 4.96, 4.89, 4.91 4vCPU VM
Tasks: 133 total, 6 running, 127 sleeping, 0 stopped, 0 zombie
%Cpu(s): 63.2 us, 33.3 sy, 0.0 ni, 0.7 id, 0.0 wa, 0.0 hi, 0.0 si, 2.8 st
KiB Mem : 8174880 total, 7408128 free, 64028 used, 702724 buff/cache
KiB Swap: 0 total, 0 free, 0 used, 7347768 avail Mem
```

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
6521	root	20	0	46952	5636	3632	R	97.0	0.1	0:21.06	stress-ng-cpu

```
top - 20:03:55 up 4:02, 2 users, load average: 5.92, 5.83, 5.83 6vCPU VM
Tasks: 147 total, 7 running, 140 sleeping, 0 stopped, 0 zombie
%Cpu(s): 61.9 us, 37.3 sy, 0.0 ni, 0.1 id, 0.0 wa, 0.0 hi, 0.0 si, 0.7 st
KiB Mem : 16432144 total, 15698980 free, 81472 used, 651692 buff/cache
KiB Swap: 0 total, 0 free, 0 used, 15627792 avail Mem
```

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
7018	root	20	0	46952	5692	3692	R	100.0	0.0	0:20.47	stress-ng-cpu

16603/128891MB [|||||||] 1 97.7% 26.14 25.28 25.39 95 days



## Sidenote: CFS & Idle VMs

- On a fully utilized hypervisor, idle VMs might experience steal because they are penalized by CFS.
- When idle VMs share a hypervisor with busy VMs, they will be assigned a lower weight as their utilization is low.
  - Results in scheduler latency, when competing with busy VM vCPUs.



# Octopus

- Userspace VM-placement daemon
  - Pins VMs to resources (CPUs, NUMA nodes)
  - As a result, occasionally migrates VMs across sockets
- VM awareness in vCPUs placement
- jiffy-level resolution not needed
- Functionality:
  - NUMA-partitioning
  - utilization tracking on overcommitted fleet
  - CPU-to-vCPU mapping on optimized HVs



# Octopus

- Issues
  - swap usage during NUMA migration
  - OOMs when VM aggressively allocates RAM:
    - when VM allocates RAM faster than RAM is swapped to disk
    - at swappoff phase, when swap is disabled



# Swapoff optimizations

- Swapoff implementation in kernel is not efficient
  - Goes through all process address space for each swap entry
  - For considerably large swap and a heavily loaded hypervisor with thousands of processes, might take hours or days.
- Usually swapoff happens during system shutdown
- We use it to migrate VMs between NUMA nodes and need it to be quick.
- Revived a dormant patch and initiated discussions upstream.
  - <https://lkml.org/lkml/2018/10/3/638>
  - Basic approach is to give a single pass on all process address space and page in the swapped out pages.



## Failing NUMA migrations

- The process of migrating the RAM of VM between NUMA nodes might fail, even with swap to back it.
  - A dangerous side effect is OOM kill of VMs
- The placement service makes its best effort to avoid OOM by tuning the swappiness and pre-calculating available memory and swap before the migration.
  - Dynamic nature of the hypervisor breaks this approach
  - VM launches and destroys causes all calculations to go wrong
- Efforts started inhouse to have a kernel level mechanism to do a safe memory migration without OOM and fail gracefully.
  - `migrate_pages(2)`



Questions?

Thank you!