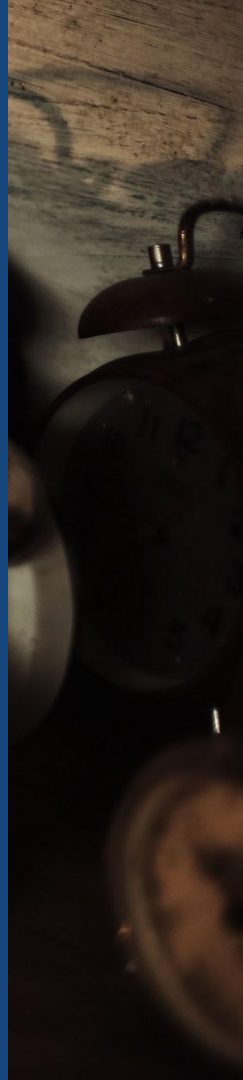
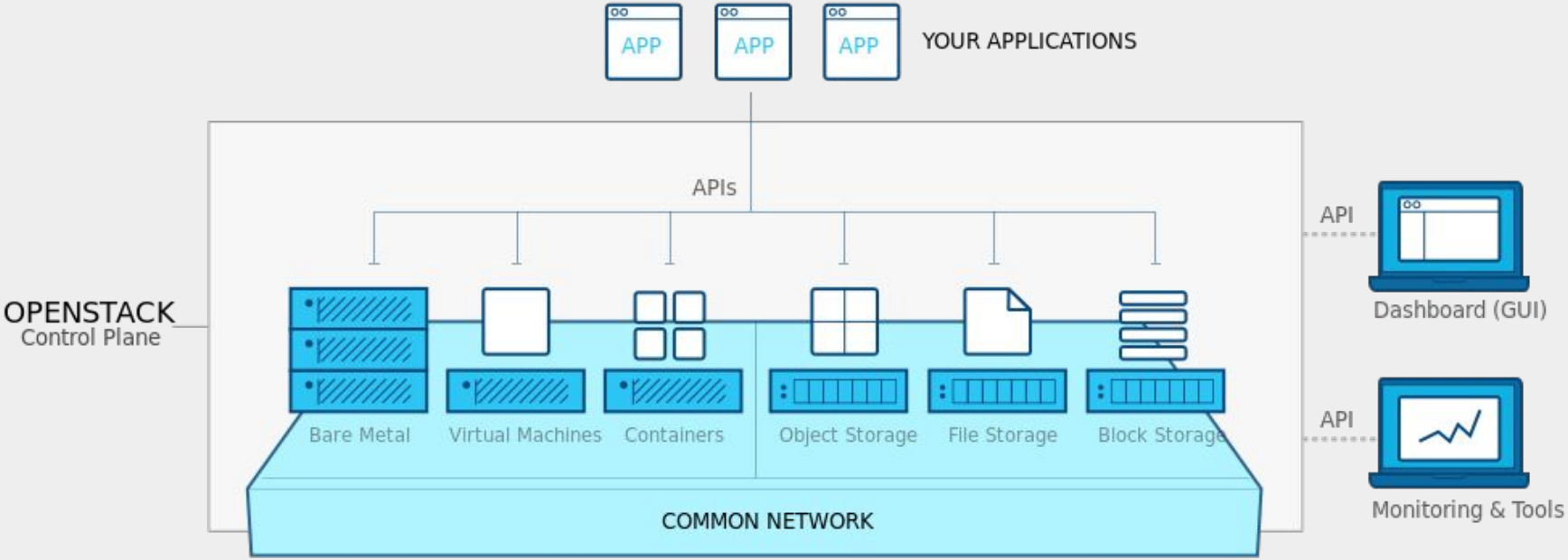
Keeping It Real (Time)

Enabling real-time compute in OpenStack

@stephenfin



A little bit of stage setting...


APP
APP
APP
YOUR APPLICATIONS
APIs
OPENSTACK
Control Plane
Bare Metal
Virtual Machines
Containers
Object Storage
File Storage
Block Storage
COMMON NETWORK
API
Dashboard (GUI)
API
Monitoring & Tools

```
$ openstack server (create|delete|list|...)
$ openstack network (create|delete|list|...)
$ openstack image (create|delete|list|...)
$ openstack volume (create|delete|list|...)
  ...
```

Um, what about NFV?

NFV History in OpenStack

Before the OpenStack “Ocata” release, we already supported:

- NUMA policies
- CPU (thread) pinning policies
- Hugepages
- SR-IOV*

NFV History in OpenStack

The OpenStack “Pike” and “Ocata” releases added two feature respectively:

- Real time policy
- Emulator threads policy

NFV History in OpenStack

The OpenStack “Pike” and “Ocata” releases added two feature respectively:

- **Real time policy**
- Emulator threads policy

Prerequisites

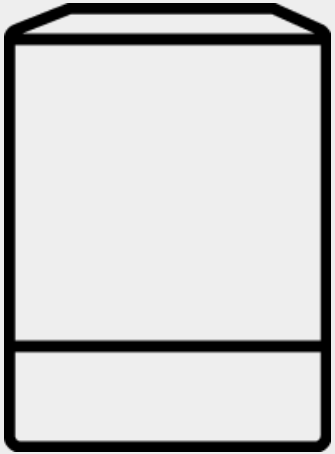
Requirements

Most configuration is done on the machine, but there are a few strict requirements.

- Suitable hardware
- OpenStack Pike or newer
- Libvirt 1.2.13 or newer
- Real-time kernel

CentOS 7.4 was used for the demo

Host Configuration (Hardware)

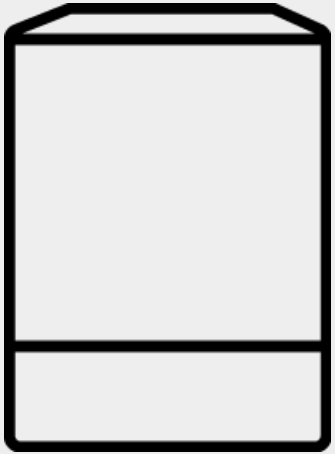


Disable the *funkier* CPU features

- Hyper Threading (SMT)
- Power management
- Turbo Boost

Essentially all the things you would do if benchmarking a system

Host Configuration (Software)



Install dependencies

- Real-time kernel
- Real-time KVM module
- Real-time tuned host profiles

Enable hugepages to prevent page faults

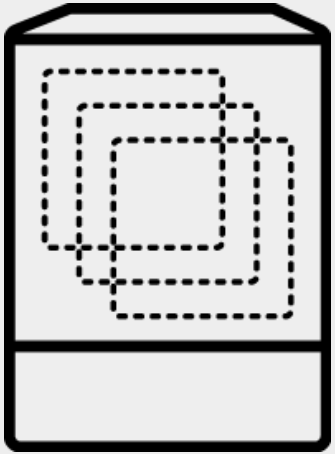
Isolate some cores

```
$ yum install -y kernel-rt.x86_64 kernel-rt-kvm.x86_64
$ yum install -y tuned-profiles-realtime tuned-profiles-nfv
```

```
$ yum install -y kernel-rt.x86_64 kernel-rt-kvm.x86_64
$ yum install -y tuned-profiles-realtime tuned-profiles-nfv

# configure tuned profile, hugepages
$ tuned-adm profile realtime-virtual-host
$ cat /etc/default/grub | grep default_hugepagesz
GRUB_CMDLINE_LINUX+="default_hugepagesz=1G"
```

```
$ yum install -y kernel-rt.x86_64 kernel-rt-kvm.x86_64
$ yum install -y tuned-profiles-realtime tuned-profiles-nfv

# configure tuned profile, hugepages
$ tuned-adm profile realtime-virtual-host
$ cat /etc/default/grub | grep default_hugepagesz
GRUB_CMDLINE_LINUX+="default_hugepagesz=1G"

# configure nova
$ cat /etc/nova/nova-cpu.conf | grep vcpu_pin_set
vcpu_pin_set = <isolated CPUs>
```

Guest Configuration (Image)



Requires many of the same dependencies

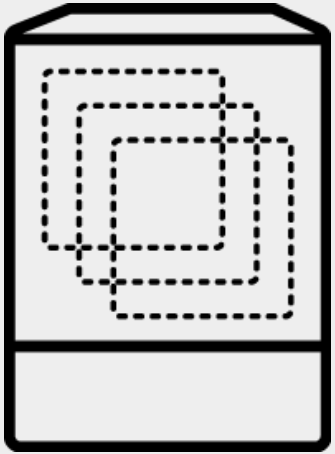
- Real-time kernel
- Real-time tuned guest profiles

If you already have an application, use that

```
$ yum install -y kernel-rt.x86_64
$ yum install -y tuned-profiles-realtime tuned-profiles-nfv
```

```
$ yum install -y kernel-rt.x86_64
$ yum install -y tuned-profiles-realtime tuned-profiles-nfv

# configure tuned profile, huge pages
$ tuned-adm profile realtime-virtual-guest
$ cat /etc/default/grub | grep default_hugepagesz
GRUB_CMDLINE_LINUX+="default_hugepagesz=1G"
```

Guest Configuration (Flavor)



Requires the following configuration options

- CPU policy
- CPU realtime policy
- Mempages

Optionally, you can also configure

- Emulator thread policy
- CPU thread policy

```
$ openstack flavor create --vcpus 4 --ram 4096 --disk 20 \
    rt1.small
```

```
$ openstack flavor create --vcpus 4 --ram 4096 --disk 20 \
    rt1.small

$ openstack flavor set rt1.small \
   --property 'hw:cpu_policy=dedicated' \
   --property 'hw:cpu_realtime=yes' \
   --property 'hw:cpu_realtime_mask=^0-1' \
   --property 'hw:mem_page_size=1GB'
```

```
$ openstack flavor create --vcpus 4 --ram 4096 --disk 20 \
    rt1.small

$ openstack flavor set rt1.small \
   --property 'hw:cpu_policy=dedicated' \
   --property 'hw:cpu_realtime=yes' \
   --property 'hw:cpu_realtime_mask=^0-1' \
   --property 'hw:mem_page_size=1GB'
```

```
$ openstack flavor create --vcpus 4 --ram 4096 --disk 20 \
    rt1.small

$ openstack flavor set rt1.small \
   --property 'hw:cpu_policy=dedicated' \
   --property 'hw:cpu_realtime=yes' \
   --property 'hw:cpu_realtime_mask=^0-1' \
   --property 'hw:mem_page_size=1GB'
```

```
$ openstack flavor create --vcpus 4 --ram 4096 --disk 20 \
    rt1.small

$ openstack flavor set rt1.small \
   --property 'hw:cpu_policy=dedicated' \
   --property 'hw:cpu_realtime=yes' \
   --property 'hw:cpu_realtime_mask=^0-1' \
   --property 'hw:mem_page_size=1GB'
```

```
$ openstack flavor create --vcpus 4 --ram 4096 --disk 20 \
    rt1.small

$ openstack flavor set rt1.small \
   --property 'hw:cpu_policy=dedicated' \
   --property 'hw:cpu_realtime=yes' \
   --property 'hw:cpu_realtime_mask=^0-1' \
   --property 'hw:mem_page_size=1GB'
```

```
$ openstack flavor create --vcpus 4 --ram 4096 --disk 20 \
    rt1.small

$ openstack flavor set rt1.small \
   --property 'hw:cpu_policy=dedicated' \
   --property 'hw:cpu_realtime=yes' \
   --property 'hw:cpu_realtime_mask=^0-1' \
   --property 'hw:mem_page_size=1GB'

$ openstack server create --flavor rt1.small --image
centos-rt
```

Under the hood

```
$ virsh dumpxml 1 | xpath /dev/stdin /domain/cputune
```

```
$ virsh dumpxml 1 | xpath /dev/stdin /domain/cputune
<cputune>
  <shares>4096</shares>
  <vcpupin vcpu="0" cpuset="2" />
  <vcpupin vcpu="1" cpuset="3" />
  <vcpupin vcpu="2" cpuset="4" />
  <vcpupin vcpu="3" cpuset="5" />
  <emulatorpin cpuset="2-3" />
  <vcpusched vcpus="2" scheduler="fifo" priority="1" />
  <vcpusched vcpus="3" scheduler="fifo" priority="1" />
</cputune>
```

```
$ virsh dumpxml 1 | xpath /dev/stdin /domain/cputune
<cputune>
  <shares>4096</shares>
  <vcpupin vcpu="0" cpuset="2" />
  <vcpupin vcpu="1" cpuset="3" />
  <vcpupin vcpu="2" cpuset="4" />
  <vcpupin vcpu="3" cpuset="5" />
  <emulatorpin cpuset="2-3" />
  <vcpusched vcpus="2" scheduler="fifo" priority="1" />
  <vcpusched vcpus="3" scheduler="fifo" priority="1" />
</cputune>
```

```
$ virsh dumpxml 1 | xpath /dev/stdin /domain/cputune
<cputune>
  <shares>4096</shares>
  <vcpupin vcpu="0" cpuset="2" />
  <vcpupin vcpu="1" cpuset="3" />
  <vcpupin vcpu="2" cpuset="4" />
  <vcpupin vcpu="3" cpuset="5" />
  <emulatorpin cpuset="2-3" />
  <vcpusched vcpus="2" scheduler="fifo" priority="1" />
  <vcpusched vcpus="3" scheduler="fifo" priority="1" />
</cputune>
```

`vcpupin`

> The optional `vcpupin` element specifies which of host's physical CPUs the domain VCPU will be pinned to. If this is omitted, and attribute `cpuset` of element `vcpu` is not specified, the vCPU is pinned to all the physical CPUs by default. It contains two required attributes, the attribute `vcpu` specifies vcpu id, and the attribute `cpuset` is same as attribute `cpuset` of element `vcpu`.
>
> Since 0.9.0

Source: [libvirt domain XML format (CPU Tuning)]

`vcpupin`

> The optional `vcpupin` element specifies which of host's physical CPUs the domain VCPU will be pinned to. If this is omitted, and attribute `cpuset` of element `vcpu` is not specified, the vCPU is pinned to all the physical CPUs by default. It contains two required attributes, the attribute `vcpu` specifies vcpu id, and the attribute `cpuset` is same as attribute `cpuset` of element `vcpu`.
>
> Since 0.9.0

Source: libvirt domain XML format (CPU Tuning)

libvirt/src/util/virprocess.c

```
int virProcessSetAffinity(pid_t pid, virBitmapPtr map)
{
    ...
    if (sched_setaffinity(pid, masklen, mask) < 0) {
        ...
    }
    ...
}
```

```
$ ps -e | grep qemu
27720 ?        00:00:04 qemu-kvm

$ ps -Tp 27720
  PID  SPID TTY          TIME CMD
27720 27720 ?        00:00:00 qemu-kvm
27720 27736 ?        00:00:00 qemu-kvm
27720 27774 ?        00:00:01 CPU 0/KVM
27720 27775 ?        00:00:00 CPU 1/KVM
27720 27776 ?        00:00:00 CPU 2/KVM
27720 27777 ?        00:00:00 CPU 3/KVM
27720 27803 ?        00:00:00 vnc_worker
```

```
$ taskset -p 27774  # CPU 0/KVM
pid 27774's current affinity mask: 4

$ taskset -p 27775  # CPU 1/KVM
pid 27775's current affinity mask: 8

$ taskset -p 27776  # CPU 2/KVM
pid 27776's current affinity mask: 10

$ taskset -p 27777  # CPU 3/KVM
pid 27777's current affinity mask: 20
```

```
$ virsh dumpxml 1 | xpath /dev/stdin /domain/cputune
<cputune>
  <shares>4096</shares>
  <vcpupin vcpu="0" cpuset="2" />
  <vcpupin vcpu="1" cpuset="3" />
  <vcpupin vcpu="2" cpuset="4" />
  <vcpupin vcpu="3" cpuset="5" />
  <emulatorpin cpuset="2-3" />
  <vcpusched vcpus="2" scheduler="fifo" priority="1" />
  <vcpusched vcpus="3" scheduler="fifo" priority="1" />
</cputune>
```

```
$ virsh dumpxml 1 | xpath /dev/stdin /domain/cputune
<cputune>
  <shares>4096</shares>
  <vcpupin vcpu="0" cpuset="2" />
  <vcpupin vcpu="1" cpuset="3" />
  <vcpupin vcpu="2" cpuset="4" />
  <vcpupin vcpu="3" cpuset="5" />
  <emulatorpin cpuset="2-3" />
  <vcpusched vcpus="2" scheduler="fifo" priority="1" />
  <vcpusched vcpus="3" scheduler="fifo" priority="1" />
</cputune>
```

`vcpusched`

> The optional `vcpusched` element specifies the scheduler type (values: `batch`, `idle`, `fifo`, `rr`) for particular vCPU threads (based on `vcpus`; leaving out `vcpus` sets the default). Valid vcpus values start at `0` through one less than the number of vCPU's defined for the domain.
>
> For real-time schedulers (`fifo`, `rr`), `priority` must be specified as well (and is ignored for non-real-time ones). The value range for the `priority` depends on the host kernel (usually 1-99).
>
> Since 1.2.13

Source: [libvirt domain XML format (CPU Tuning)]

vcpusched

> The optional `vcpusched` element specifies the scheduler type (values: `batch`, `idle`, `fifo`, `rr`) for particular vCPU threads (based on `vcpus`; leaving out `vcpus` sets the default). Valid vcpus values start at `0` through one less than the number of vCPU's defined for the domain.
>
> For real-time schedulers (`fifo`, `rr`), `priority` must be specified as well (and is ignored for non-real-time ones). The value range for the `priority` depends on the host kernel (usually 1-99).
>
> Since 1.2.13

Source: libvirt domain XML format (CPU Tuning)

libvirt/src/util/virprocess.c

```
int virProcessSetScheduler(pid_t pid,
                           virProcessSchedPolicy policy,
                           int priority)
{
    ...
    if (sched_setscheduler(pid, pol, &param) < 0) {
        ...
    }
    ...
}
```

```
$ chrt -p 27774  # CPU 0/KVM
pid 27774's current scheduling policy: SCHED_OTHER
pid 27774's current scheduling priority: 0
$ chrt -p 27775  # CPU 1/KVM
pid 27775's current scheduling policy: SCHED_OTHER
pid 27775's current scheduling priority: 0
$ chrt -p 27776  # CPU 2/KVM
pid 27776's current scheduling policy: SCHED_FIFO
pid 27776's current scheduling priority: 1
$ chrt -p 27777  # CPU 3/KVM
pid 27777's current scheduling policy: SCHED_FIFO
pid 27777's current scheduling priority: 1
```

```
$ virsh dumpxml 1 | xpath /dev/stdin /domain/memoryBacking
```

```
$ virsh dumpxml 1 | xpath /dev/stdin /domain/memoryBacking
<memoryBacking>
  <hugepages>
    <page size="1048576" unit="KiB" nodeset="0" />
  </hugepages>
  <nosharepages />
  <locked />
</memoryBacking>
```

```
$ ps -e | grep qemu
27720 ?        00:00:04 qemu-kvm

$ grep huge /proc/*/numa_maps
/proc/27720/numa_maps:7f3dc0000000 bind:0 ...
```

```
$ openstack server ssh rt-server --login centos
```

```
$ openstack server ssh rt-server --login centos

# within the guest
$ taskset -c 2 stress --cpu 4 &
$ taskset -c 2 cyclictest -m -n -q -p95 -D 1h -h100 -i 200 \
    > cyclictest.out
$ cat cyclictest.out | tail -7 | head -3
# Min Latencies: 00006
# Avg Latencies: 00007
# Max Latencies: 00020
```

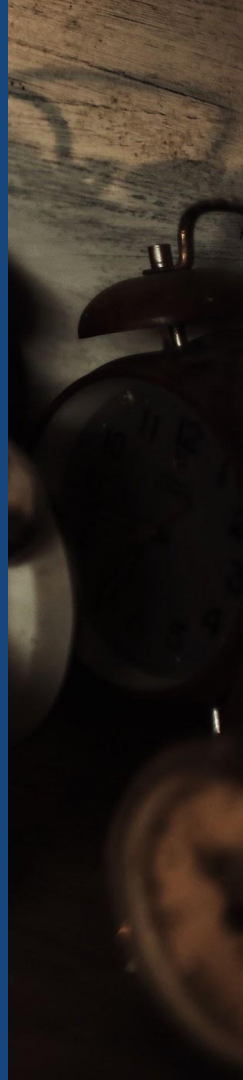
Wrap up

```
$ openstack flavor create --vcpus 4 --ram 4096 --disk 20 \
    rt1.small

$ openstack flavor set rt1.small \
   --property 'hw:cpu_policy=dedicated' \
   --property 'hw:cpu_realtime=yes' \
   --property 'hw:cpu_realtime_mask=^0-1' \
   --property 'hw:mem_page_size=1GB'

$ openstack server create --flavor rt1.small --image
centos-rt
```

Keeping It Real (Time)

Enabling real-time compute in OpenStack

@stephenfin



References

- libvirt domain XML format (CPU Tuning) — libvirt.org
- taskset(1) — man7.org
- sched_setaffinity(2) — man7.org
- chrt(1) — man7.org
- sched_setscheduler(2) — man7.org
- Completely Fair Scheduler — doc.opensuse.org
- Using and Understanding the Real-Time Cyclictest Benchmark — linuxfound.org
- Deploying Real Time Openstack — that.guru

Credits

Clocks photo by Ahmad Ossayli on Unsplash

Clouds photo by Jason Wong on Unsplash