Cgroup v1/v2 Abstraction Layer
Linux Plumbers Conference 2019

Tom Hromatka
Software Engineer
Oracle
September 10, 2019
Oracle Linux

• Emphasis on reliability, scalability, security, and performance for demanding enterprise workloads
• Support lifecycle of 10+ years
• Stable kABI
2011

- UEK2 kernel – 2.6.39
- System V
- Containers – up and coming
- cgroup v1
  - cpu, cpuacct, cpuset, debug, devices, freezer, mem, net_cls, ns

Today

- Kernel 5.2
- systemd
- Containers - ubiquitous
- cgroup v2
  - cpu, cpuacct, cpuset, debug, devices, freezer, huge_tlb, io, memory, net_cls, net_prio, perf_event, pids, rdma
Today

- Kernel 5.2
- systemd
- Containers - ubiquitous
- cgroup v2
  - cpu, cpuacct, cgroup, debug, devices, freezer, huge_tlb, io, memory, net_cls, net_prio, perf_event, pids, rdma

2029

?
Current “Solutions” Available

• Continue using v1 only
• Backport v2 features into v1
• Switch to v2, let apps break, then fix them
Abstraction Layer Requirements

- Hide v1 vs. v2 differences
  - Users shall no longer need to use sysfs
  - Users should no longer need to use libcgroup

- Bindings for multiple languages – C, Java, Go?, Python?

- Abstraction layer likely needs to be a separate userspace executable

- How abstract do we go?
  - `GiveMeCpus(cgname=foo, cpu_count=2, exclusive=True, numa_aligned=True, ...)`
  - `CgroupCreate(cgname=foo, secure_from_sidechannel=True, ...)`
Current Status

- Add unit testing support to libcgroup [DONE]
- Add functional testing support to libcgroup [OUT FOR REVIEW]
- Add cgroup v2 support to libcgroup [IN PROGRESS]
- Define scope of a cgroup abstraction layer [IN PROGRESS]
- Begin work on the abstraction layer [TODO]