Control of major resources in cgroup v2

Tejun Heo, Facebook
Comprehensive hierarchical control of all significant resource consumptions in the system.
Resource domains
Resource domains

- A resource domain is what contains actual resource consumptions.
- All resource consumptions - be that CPU cycles, memory allocations or IOs, are accounted to and controlled by a resource domain.
- Resource domains can’t be nested. Every resource domain is terminal.
Resource domains

Roughly, leaf cgroups are resource domains which contain processes and resource consumptions while the internal cgroups organize and distribute resources across the resource domains.
Account for and control operations which span multiple resource types.
$ free -m

  total  used  free  shared  buff/cache  available
Mem:    7862  3233  401  1779  4227   2488
Swap:   8191    1  8190

$ sysctl -a |grep vm.dirty_
  vm.dirty_background_bytes = 0
  vm.dirty_background_ratio = 10
  vm.dirty_bytes = 0
  vm.dirty_expire_centisecs = 3000
  vm.dirty_ratio = 20
  vm.dirty_writeback_centisecs = 1500
Always have unambiguous resource config.
D1

Direct and anonymous consumptions, and consumptions from D2 and D3

D2

Direct and anonymous consumptions

D3

Direct and anonymous consumptions
Resource distribution config models

- **Weights**
  
  
  
  
  
  
  
  “.weight”, work-conserving proportional distribution.

- **Limits**
  
  
  
  
  
  
  
  “.max” or “.high”, upper limit specified in absolute quantity.
  
  may or may not be work-conserving.

- **Protections**
  
  
  
  
  
  
  “.low” or “.min”, the opposite of limits.
  
  Work-conserving.

- **Allocations**
  
  
  
  
  
  
  Like limits but hard allocations.
memory

- Limits and protections.
  - memory.low
  - memory.high
  - memory.max
- Covers most significant consumptions including fs caches and network buffers.
- Co-operates with io to control writeback.
- Pressure measurement in the works.
Memory pressure measurement

- Nobody really had it. Sizing always has been through trial-and-error.
- It’s hard. thikk of cp.
- Gets more painful with segmented memory domains.
- Why we had frequent OOMs and userland handlers in cgroup v1.

What we’re implementing.

- Canonical time based measurement of memory pressure.
- “Was everyone blocked on memory?”
- Also, “Was anyone blocked on memory?”
Weights implemented by cfq.
- Kinda problematic.
- May be replaced by bfq.
- Unlikely to be usable with high-iops devices.

Limits implemented by blk-throttle.
- io.max is not work-conserving.
- io.high is in the works. This will be difficult to configure but usable for high-iops devices.
- Not unusably slow but not super efficient either.

Works with memory to control writeback IOs.
CPU

Not merged yet. Quite a bit of discussions going on with the scheduler people.

- **Weights**
  - Work-conserving.
  - Not particularly low overhead. Needs to be improved.

- **Limits**
  - Bandwidth limit.

- **Do not cooperate with other controllers or manages anonymous consumptions yet.**
Questions?