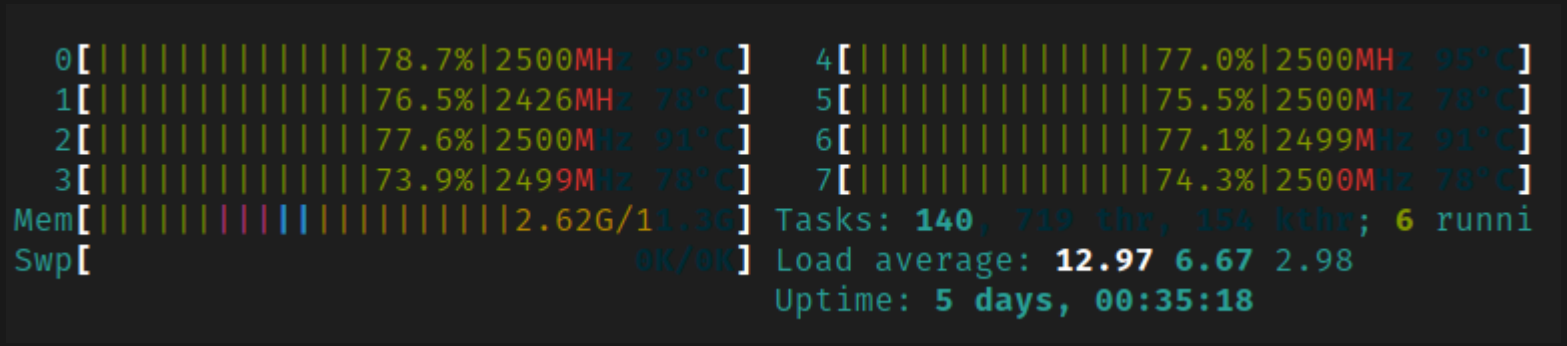
cgroups

resource constraints in linux

@tint:tint.red

has this ever happened to you????



has this ever happened to you????

```
  0[||||||||||||||78.7%|2500MHz 95°C]   4[||||||||||||||77.0%|2500MHz 95°C]
  1[||||||||||||||76.5%|2426MHz 78°C]   5[||||||||||||||75.5%|2500MHz 78°C]
  2[||||||||||||||77.6%|2500MHz 91°C]   6[||||||||||||||77.1%|2499MHz 91°C]
  3[||||||||||||||73.9%|2499MHz 78°C]   7[||||||||||||||74.3%|2500MHz 78°C]
Mem[||||||||||||||||||||2.62G/11.3G] Tasks: 140, 719 thr, 154 kthr; 6 runni
Swp[                          0K/0K] Load average: 12.97 6.67 2.98
                                     Uptime: 5 days, 00:35:18
```

has *this* ever happened to you???

```
  0[||||||||||||||97.9%|2400MHz|75°C]   4[||||||||||||||98.3%|2400MHz|75°C]
  1[||||||||||||||98.3%|2743MHz|68°C]   5[||||||||||||||97.9%|2747MHz|68°C]
  2[||||||||||||||96.6%|2505MHz|70°C]   6[||||||||||||||98.7%|2505MHz|70°C]
  3[||||||||||||||97.4%|2511MHz|67°C]   7[||||||||||||||98.3%|2525MHz|67°C]
Mem[||||||||||||||||||||10.9G/11.3G] Tasks: 170, 1166 thr, 163 kthr; 8 runn
Swp[                          0K/0K] Load average: 9.07 5.66 3.38
                                     Uptime: 5 days, 00:40:08
```

part 1: raw fs

Everything Is A File

before you do anything

make sure a `cgroup2` fs is mounted!

most linuxes should mount it at `/sys/fs/cgroup`, if not you can just `mount -t cgroup2 foldername foldername`

operations

- making a cgroup:
  `mkdir /sys/fs/cgroup/name_of_cgroup`
- removing a cgroup:
  `rmdir /sys/fs/cgroup/name_of_cgroup`
- adding a process to the cgroup:
  `echo "$pid" > /sys/fs/cgroup/name_of_cgroup/cgroup.procs`
- specifying resources to control:
  `echo "+cpu" > /sys/fs/cgroup/name_of_cgroup/cgroup.subtree_control`
- figuring out available resources for the previous command:
  `cat /sys/fs/cgroup/name_of_cgroup/cgroup.controllers`

actually doing the limitations: cpu

```
echo "+cpu" > /sys/fs/cgroup/name_of_cgroup/cgroup.subtree_control
```

- weight-based cpu distribution:
  `echo "100" > /sys/fs/cgroup/name_of_cgroup/cpu.weight`
- time-based cpu distribution:
  `echo "20000 10000" > /sys/fs/cgroup/name_of_cgroup/cpu.max`
  this tells the scheduler that processes in this cgroup can only run for 20000 (arbitrary time units) out of every 10000 (arbitrary time units) - that is, 2 cores worth of time

actually doing the limitations: memory

for reasons that i don't know you need to put processes in a nested cgroup from where you do the memory limitation

```
echo "+memory" > /sys/fs/cgroup/name_of_cgroup/cgroup.subtree_control
mkdir /sys/fs/cgroup/name_of_cgroup/name_of_nested_cgroup
echo "$pid" > /sys/fs/cgroup/name_of_cgroup/name_of_nested_cgroup/cgroup.procs
```

- set high memory, in bytes (system will start trying to reclaim memory above this point):
  `echo "1000000000" > /sys/fs/cgroup/name_of_cgroup/memory.high`
- set max memory, in bytes (system will try not to exceed this point):
  `echo "2000000000" > /sys/fs/cgroup/name_of_cgroup/memory.max`

part 2: abstraction layer

it's not really that abstract tbh

- arch: `libcgroup` (aur)
- alpine: `cgroup-tools`
- redhat: `libcgroup libcgroup-tools`

operations

- make a cgroup:
  `cgcreate -g cpu:name_of_cgroup` (replace `cpu` with comma-separated list of desired controllers)
- remove a cgroup:
  `cgdelete cpu:name_of_cgroup`
- set a parameter in a cgroup:
  `cgset -r cpu.max="20000 10000" cpu:name_of_cgroup`
- running a process in the cgroup:
  `cgexec -g cpu:name_of_cgroup sh`
- move a process to a cgroup:
  `cgclassify -g cpu:name_of_cgroup 69 420 1337`

part 3: systemd, unfortunately

or fortunately, depending

systemd manages cgroups with its concept of "slices", which creates a hierarchy of cgroup tied to systemd's services

you can make a `.slice` file in any systemd directory, and you can make slices as an unprivileged user in `~/.config/systemd/user/`

you can run things in a user slice with `systemd-run --user --slice=example.slice -t sh` (after a `systemctl --user daemon-reload`, of course)

example slice file:

```
[Slice]
MemoryHigh=2000000000
MemoryMax=2500000000
CPUQuota=50%
```

available options documented in `man systemd.resource-control`

part 4: docker

cgroups are part of a container's isolation

some of the flags in `docker run` that affect cgroup settings

- `-c, --cpu-shares [weight]`
- `--cpu-period [period]` and `--cpu-quota [quota]`
- `--cpus [number of cores]`
- `--cpuset-cpus [cpus]`
- `-m, --memory [size]`

resources

- https://www.kernel.org/doc/html/latest/admin-guide/cgroup-v2.html
- https://wiki.archlinux.org/title/Cgroups
- https://docs.redhat.com/en/documentation/red_hat_enterprise_linux/7/html/resource_management_guide/chap-introduction_to_control_groups
- https://docs.docker.com/reference/cli/docker/container/run/