Cgroups V2

Introduction

Cgroup v2 provides a unified control system with enhanced resource management capabilities. Control groups (cgroups), are a Linux kernel feature which allow processes to be organized into hierarchical groups whose usage of various types of resources can then be limited and monitored. The kernel's cgroup interface is provided through a pseudo-filesystem called cgroupfs.

Identify cgroup version

The cgroup version depends on the Linux distribution. To check which cgroup version is used, run:

```
stat -fc %T /sys/fs/cgroup/
```

For cgroup v2, the output is **cgroup2fs**. For cgroup v1, the output is **tmpfs**

Hierarchy

Current cgroup hierarchy can be seen with **systemctl status** or **systemd-cgls** command.

```
$ systemctl status

● mylinux
    State: running
     Jobs: 0 queued
   Failed: 0 units
    Since: Wed 2019-12-04 22:16:28 UTC; 1 day 4h ago
   CGroup: /
           ├─user.slice
           │ └─user-1000.slice
           │   ├─user@1000.service
           │   │ ├─gnome-shell-wayland.service
           │   │ │ ├─ 1129 /usr/bin/gnome-shell
           │   │ ├─gnome-terminal-server.service
           │   │ │ ├─33519 /usr/lib/gnome-terminal-server
           │   │ │ ├─37298 fish
           │   │ │ └─39239 systemctl status
           │   │ ├─init.scope
           │   │ │ ├─1066 /usr/lib/systemd/systemd --user
           │   │ │ └─1067 (sd-pam)
           │   └─session-2.scope
           │     ├─1053 gdm-session-worker [pam/gdm-password]
           │     ├─1078 /usr/bin/gnome-keyring-daemon --daemonize --login
```

```
           │     ├─1082 /usr/lib/gdm-wayland-session /usr/bin/gnome-session
           │     ├─1086 /usr/lib/gnome-session-binary
           │     └─3514 /usr/bin/ssh-agent -D -a /run/user/1000/keyring/.ssh
           ├─init.scope
           │ └─1 /sbin/init
           └─system.slice
             ├─systemd-udevd.service
             │ └─285 /usr/lib/systemd/systemd-udevd
             ├─systemd-journald.service
             │ └─272 /usr/lib/systemd/systemd-journald
             ├─NetworkManager.service
             │ └─656 /usr/bin/NetworkManager --no-daemon
             ├─gdm.service
             │ └─668 /usr/bin/gdm
             └─systemd-logind.service
```

cgroup resource usage

The **systemd-cgtop** command can be used to see the resource usage:

```
$ systemd-cgtop

Control Group                              Tasks   %CPU   Memory  Input/s
Output/s
user.slice                                   540  152,8     3.3G        -
-
user.slice/user-1000.slice                   540  152,8     3.3G        -
-
user.slice/u…000.slice/session-1.scope       425  149,5     3.1G        -
-
system.slice                                  37      -   215.6M        -
-
```

Cgroup Filesystem

The /sys/fs/cgroup/ directory, also called the root control group, by default, contains interface files (starting with cgroup) and controller-specific files such as cpuset.cpus.effective. In addition, there are some directories related to systemd, such as, /sys/fs/cgroup/init.scope, /sys/fs/cgroup/system.slice, and /sys/fs/cgroup/user.slice.

```
root@homeserver:/sys/fs/cgroup# ls -al

dr-xr-xr-x 14 root root 0 jan 20 17:23 .
drwxr-xr-x  7 root root 0 jan 20 17:23 ..
-r--r--r--  1 root root 0 jan 20 17:23 cgroup.controllers
-rw-r--r--  1 root root 0 jan 20 17:30 cgroup.max.depth
-rw-r--r--  1 root root 0 jan 20 17:30 cgroup.max.descendants
-rw-r--r--  1 root root 0 jan 20 17:23 cgroup.procs
-r--r--r--  1 root root 0 jan 20 17:30 cgroup.stat
```

```
-rw-r--r--  1 root root 0 jan 20 17:23 cgroup.subtree_control
-rw-r--r--  1 root root 0 jan 20 17:30 cgroup.threads
-rw-r--r--  1 root root 0 jan 20 17:30 cpu.pressure
-r--r--r--  1 root root 0 jan 20 17:30 cpuset.cpus.effective
-r--r--r--  1 root root 0 jan 20 17:30 cpuset.mems.effective
-r--r--r--  1 root root 0 jan 20 17:30 cpu.stat
drwxr-xr-x  2 root root 0 jan 20 17:23 dev-hugepages.mount
drwxr-xr-x  2 root root 0 jan 20 17:23 dev-mqueue.mount
drwxr-xr-x  2 root root 0 jan 20 17:23 init.scope
-rw-r--r--  1 root root 0 jan 20 17:30 io.cost.model
-rw-r--r--  1 root root 0 jan 20 17:30 io.cost.qos
-rw-r--r--  1 root root 0 jan 20 17:30 io.pressure
-r--r--r--  1 root root 0 jan 20 17:30 io.stat
drwxr-xr-x  2 root root 0 jan 20 17:23 lxc.payload.fileserver
drwxr-xr-x  2 root root 0 jan 20 17:23 lxc.payload.test-container
drwxr-xr-x  2 root root 0 jan 20 17:23 lxc.pivot
-r--r--r--  1 root root 0 jan 20 17:30 memory.numa_stat
-rw-r--r--  1 root root 0 jan 20 17:30 memory.pressure
-r--r--r--  1 root root 0 jan 20 17:30 memory.stat
drwxr-xr-x  2 root root 0 jan 20 17:23 sys-fs-fuse-connections.mount
drwxr-xr-x  2 root root 0 jan 20 17:23 sys-kernel-config.mount
drwxr-xr-x  2 root root 0 jan 20 17:23 sys-kernel-debug.mount
drwxr-xr-x  2 root root 0 jan 20 17:23 sys-kernel-tracing.mount
drwxr-xr-x 23 root root 0 jan 20 17:50 system.slice
drwxr-xr-x  4 root root 0 jan 20 17:33 user.slice
```

Available Controllers

Verify which controllers are supported by the kernel and available in the /sys/fs/cgroup/cgroup.controllers file:

```
# cat /sys/fs/cgroup/cgroup.controllers

cpuset cpu io memory hugetlb pids rdma
```

Enabled Controllers

Check which controller are activated / enabled:

```
# /sys/fs/cgroup/cgroup.subtree_control

cpuset cpu io memory hugetlb pids rdma
```

Enable or disable a specific controllers:

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

```

```
# echo "-cpu" >> /sys/fs/cgroup/cgroup.subtree_control
```

These commands enable controllers for the immediate children groups of the /sys/fs/cgroup/ root control group. A child group is where you can specify processes and apply control checks to each of the processes based on your criteria. Users can read the contents of the cgroup.subtree_control file at any level to get an idea of what controllers are going to be available for enablement in the immediate child group.

Child Control Group

The /sys/fs/cgroup/ listing above shows some subdirectories (Child Control Groups) like: lxc.payload.fileserver. New Child Groups can be added by creating a subdirectory in /sys/fs/cgroup.

```
# mkdir /sys/fs/cgroup/example/
```

The /sys/fs/cgroup/example/ directory defines a child group. The /sys/fs/cgroup/cgroup.subtree_control file in the root level specifies which enabled controllers apply to this child group.

When you create the /sys/fs/cgroup/example/ directory, some cgroups-v2 interface files and controller-specific files are automatically created in the directory. The /sys/fs/cgroup/example/ directory contains:

```
#ls -al example

drwxr-xr-x  2 root root 0 jan 20 18:20 .
dr-xr-xr-x 15 root root 0 jan 20 18:20 ..
-r--r--r--  1 root root 0 jan 20 18:20 cgroup.controllers
-r--r--r--  1 root root 0 jan 20 18:20 cgroup.events
-rw-r--r--  1 root root 0 jan 20 18:20 cgroup.freeze
-rw-r--r--  1 root root 0 jan 20 18:20 cgroup.max.depth
-rw-r--r--  1 root root 0 jan 20 18:20 cgroup.max.descendants
-rw-r--r--  1 root root 0 jan 20 18:20 cgroup.procs
-r--r--r--  1 root root 0 jan 20 18:20 cgroup.stat
-rw-r--r--  1 root root 0 jan 20 18:20 cgroup.subtree_control
-rw-r--r--  1 root root 0 jan 20 18:20 cgroup.threads
-rw-r--r--  1 root root 0 jan 20 18:20 cgroup.type
-rw-r--r--  1 root root 0 jan 20 18:20 cpu.max
-rw-r--r--  1 root root 0 jan 20 18:20 cpu.pressure
-rw-r--r--  1 root root 0 jan 20 18:20 cpuset.cpus
-r--r--r--  1 root root 0 jan 20 18:20 cpuset.cpus.effective
-rw-r--r--  1 root root 0 jan 20 18:20 cpuset.cpus.partition
-rw-r--r--  1 root root 0 jan 20 18:20 cpuset.mems
-r--r--r--  1 root root 0 jan 20 18:20 cpuset.mems.effective
-r--r--r--  1 root root 0 jan 20 18:20 cpu.stat
-rw-r--r--  1 root root 0 jan 20 18:20 cpu.weight
-rw-r--r--  1 root root 0 jan 20 18:20 cpu.weight.nice
-r--r--r--  1 root root 0 jan 20 18:20 hugetlb.1GB.current
-r--r--r--  1 root root 0 jan 20 18:20 hugetlb.1GB.events
```

```
-r--r--r--  1 root root 0 jan 20 18:20 hugetlb.1GB.events.local
-rw-r--r--  1 root root 0 jan 20 18:20 hugetlb.1GB.max
-r--r--r--  1 root root 0 jan 20 18:20 hugetlb.1GB.rsvd.current
-rw-r--r--  1 root root 0 jan 20 18:20 hugetlb.1GB.rsvd.max
-r--r--r--  1 root root 0 jan 20 18:20 hugetlb.2MB.current
-r--r--r--  1 root root 0 jan 20 18:20 hugetlb.2MB.events
-r--r--r--  1 root root 0 jan 20 18:20 hugetlb.2MB.events.local
-rw-r--r--  1 root root 0 jan 20 18:20 hugetlb.2MB.max
-r--r--r--  1 root root 0 jan 20 18:20 hugetlb.2MB.rsvd.current
-rw-r--r--  1 root root 0 jan 20 18:20 hugetlb.2MB.rsvd.max
-rw-r--r--  1 root root 0 jan 20 18:20 io.max
-rw-r--r--  1 root root 0 jan 20 18:20 io.pressure
-r--r--r--  1 root root 0 jan 20 18:20 io.stat
-rw-r--r--  1 root root 0 jan 20 18:20 io.weight
-r--r--r--  1 root root 0 jan 20 18:20 memory.current
-r--r--r--  1 root root 0 jan 20 18:20 memory.events
-r--r--r--  1 root root 0 jan 20 18:20 memory.events.local
-rw-r--r--  1 root root 0 jan 20 18:20 memory.high
-rw-r--r--  1 root root 0 jan 20 18:20 memory.low
-rw-r--r--  1 root root 0 jan 20 18:20 memory.max
-rw-r--r--  1 root root 0 jan 20 18:20 memory.min
-r--r--r--  1 root root 0 jan 20 18:20 memory.numa_stat
-rw-r--r--  1 root root 0 jan 20 18:20 memory.oom.group
-rw-r--r--  1 root root 0 jan 20 18:20 memory.pressure
-r--r--r--  1 root root 0 jan 20 18:20 memory.stat
-r--r--r--  1 root root 0 jan 20 18:20 memory.swap.current
-r--r--r--  1 root root 0 jan 20 18:20 memory.swap.events
-rw-r--r--  1 root root 0 jan 20 18:20 memory.swap.high
-rw-r--r--  1 root root 0 jan 20 18:20 memory.swap.max
-r--r--r--  1 root root 0 jan 20 18:20 pids.current
-r--r--r--  1 root root 0 jan 20 18:20 pids.events
-rw-r--r--  1 root root 0 jan 20 18:20 pids.max
-r--r--r--  1 root root 0 jan 20 18:20 rdma.current
-rw-r--r--  1 root root 0 jan 20 18:20 rdma.max
```

The example output shows files specific to the enabled controllers. These controllers are manually enabled for the root's (/sys/fs/cgroup/) direct child control groups using the /sys/fs/cgroup/cgroup.subtree_control file. By default, the newly created child control group inherits these resources. The directory also includes general cgroup control interface files such as cgroup.procs or cgroup.controllers, which are common to all control groups, regardless of enabled controllers. The files such as memory.high and pids.max relate to the memory and pids controllers, which are in the root control group (/sys/fs/cgroup/), and are always enabled by default. By default, the newly created child group inherits access to all of the system's CPU and memory resources, without any limits.

Enable the CPU-related controllers in /sys/fs/cgroup/example/ to obtain controllers that are relevant only to CPU:

```
# echo "+cpu" >> /sys/fs/cgroup/example/cgroup.subtree_control
# echo "+cpuset" >> /sys/fs/cgroup/example/cgroup.subtree_control
```

These commands ensure that the immediate child control group will only have controllers relevant to regulate the CPU time distribution - not to memory or pids controllers. Create the /sys/fs/cgroup/example/tasks/ directory:

mkdir /sys/fs/cgroup/example/tasks/

The /sys/fs/cgroup/example/tasks/ directory defines a child group with files that relate purely to cpu and cpuset controllers.

Optionally, inspect another child control group:

```
# ls -l /sys/fs/cgroup/example/tasks

-r–r–r--. 1 root root 0 Jun  1 11:45 cgroup.controllers
-r–r–r--. 1 root root 0 Jun  1 11:45 cgroup.events
-rw-r–r--. 1 root root 0 Jun  1 11:45 cgroup.freeze
-rw-r–r--. 1 root root 0 Jun  1 11:45 cgroup.max.depth
-rw-r–r--. 1 root root 0 Jun  1 11:45 cgroup.max.descendants
-rw-r–r--. 1 root root 0 Jun  1 11:45 cgroup.procs
-r–r–r--. 1 root root 0 Jun  1 11:45 cgroup.stat
-rw-r–r--. 1 root root 0 Jun  1 11:45 cgroup.subtree_control
-rw-r–r--. 1 root root 0 Jun  1 11:45 cgroup.threads
-rw-r–r--. 1 root root 0 Jun  1 11:45 cgroup.type
-rw-r–r--. 1 root root 0 Jun  1 11:45 cpu.max
-rw-r–r--. 1 root root 0 Jun  1 11:45 cpu.pressure
-rw-r–r--. 1 root root 0 Jun  1 11:45 cpuset.cpus
-r–r–r--. 1 root root 0 Jun  1 11:45 cpuset.cpus.effective
-rw-r–r--. 1 root root 0 Jun  1 11:45 cpuset.cpus.partition
-rw-r–r--. 1 root root 0 Jun  1 11:45 cpuset.mems
-r–r–r--. 1 root root 0 Jun  1 11:45 cpuset.mems.effective
-r–r–r--. 1 root root 0 Jun  1 11:45 cpu.stat
-rw-r–r--. 1 root root 0 Jun  1 11:45 cpu.weight
-rw-r–r--. 1 root root 0 Jun  1 11:45 cpu.weight.nice
-rw-r–r--. 1 root root 0 Jun  1 11:45 io.pressure
-rw-r–r--. 1 root root 0 Jun  1 11:45 memory.pressure
```