System Programming for Linux Containers Control Groups (cgroups)

Michael Kerrisk, man7.org $\ensuremath{\mathbb{C}}$ 2020

mtk@man7.org

February 2020

Outline

19	Cgroups	19-1
19.1	Introduction to cgroups v1 and v2	19-3
19.2	Cgroups v1: hierarchies and controllers	19-17
19.3	Cgroups v1: populating a cgroup	19-24
19.4	Cgroups v1: release notification	19-33
19.5	Cgroups v1: a survey of the controllers	19-43
19.6	Cgroups /proc files	19-65
19.7	Cgroup namespaces	19-68

Outline

19	Cgroups	19-1
19.1	Introduction to cgroups v1 and v2	19-3
19.2	Cgroups v1: hierarchies and controllers	19-17
19.3	Cgroups v1: populating a cgroup	19-24
19.4	Cgroups v1: release notification	19-33
19.5	Cgroups v1: a survey of the controllers	19-43
19.6	Cgroups /proc files	19-65
19.7	Cgroup namespaces	19-68

Goals

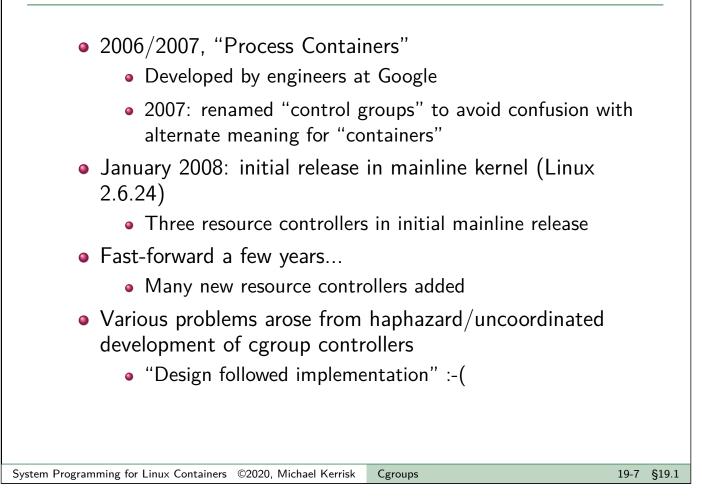
- Cgroups is a big topic
 - Many controllers
 - V1 versus V2 interfaces
- Our goal: understand fundamental semantics of cgroup filesystem and interfaces
 - Useful from a programming perspective
 - How do I build container frameworks?
 - What else can I build with cgroups?
 - And useful from a system engineering perspective
 - What's going on underneath my container's hood?

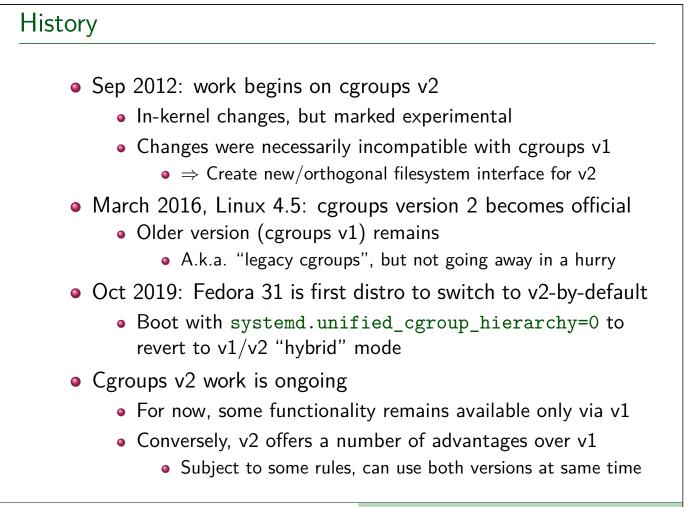
Focus

<list-item><list-item><list-item><list-item><list-item>
We'l focus on:
General principles of operation; goals of cgroups
The cgroup filesystem
Interacting with the cgroup filesystem using shell commands
Problems with cgroups v1, motivations for cgroups v2
Differences between cgroups v1 and v2
We'l look briefly at some of the controllers

Resources Kernel Documentation files Documentation/cgroup-v1/*.txt Documentation/admin-guide/cgroup-v2.rst cgroups(7) man page Neil Brown's excellent (2014) LWN.net series on Cgroups: https://lwn.net/Articles/604609/ Thought-provoking commentary on the meaning of grouping and hierarchy https://lwn.net/Articles/484254/ - Tejun Heo's initial thinking about redesigning cgroups Other articles at https://lwn.net/Kernel/Index/#Control_groups

History

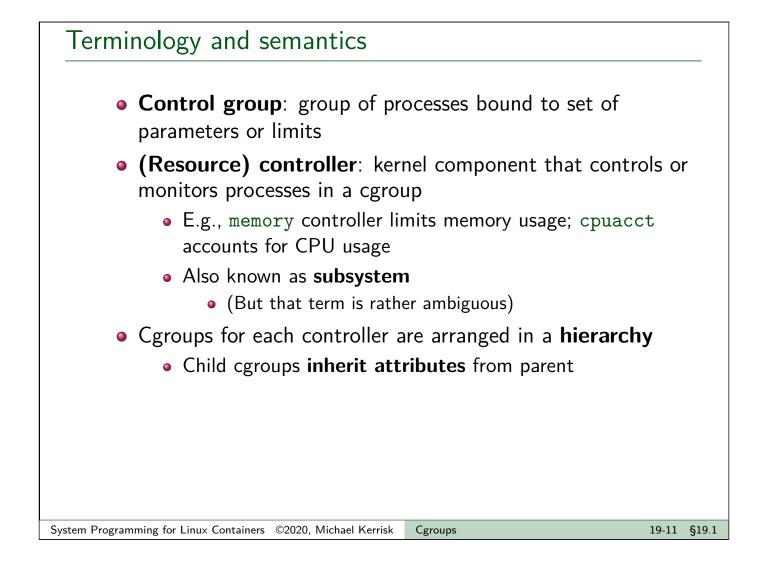


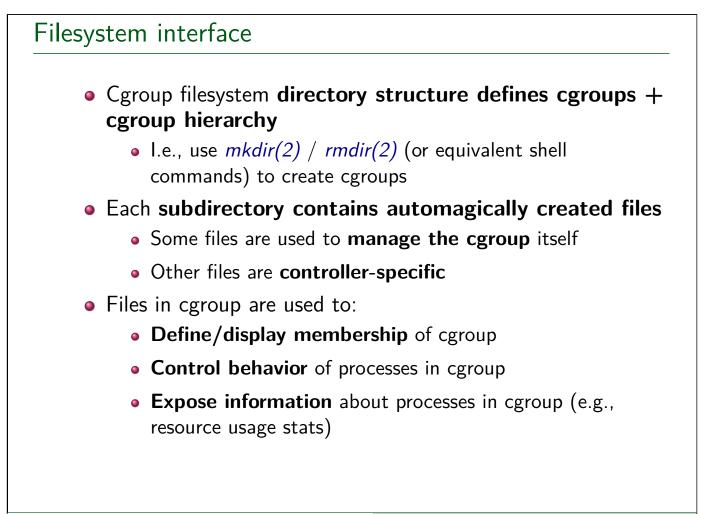


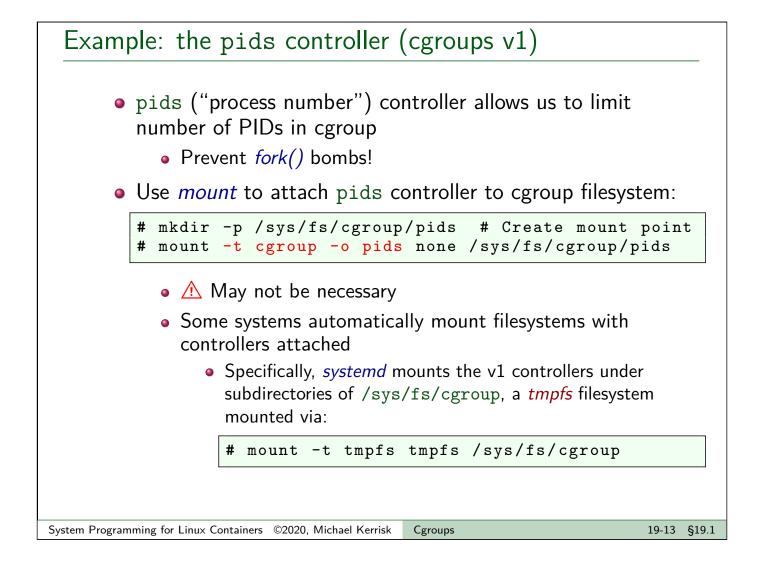
Cgroups overview • Two principle components: • A mechanism for hierarchically grouping processes • A set of **controllers** (kernel components) that manage, control, or monitor processes in cgroups • (Resources such as CPU, memory, block I/O bandwidth) • Interface is via a pseudo-filesystem • Cgroup manipulation takes form of filesystem operations, which might be done: Via shell commands Programmatically • Via management daemon (e.g., *systemd*) • Via your container framework's tools (e.g., LXC, Docker) 19-9 §19.1 System Programming for Linux Containers ©2020, Michael Kerrisk Cgroups

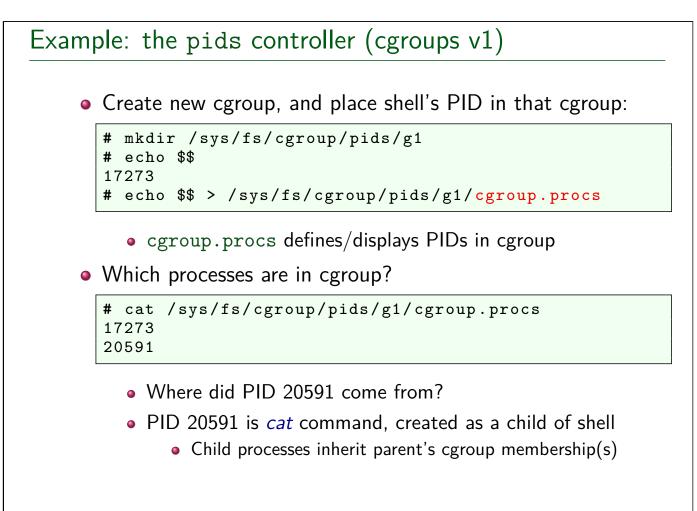
What do cgroups allow us to do?

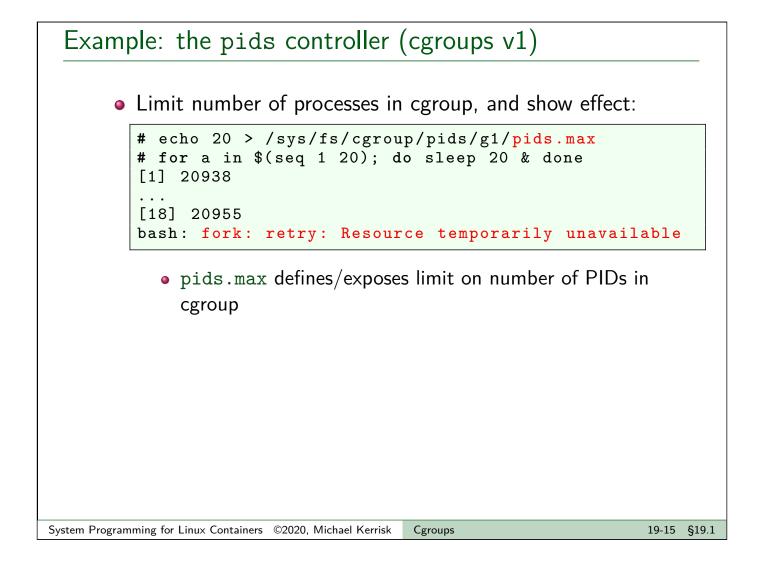
- Limit resource usage of group
 - E.g., limit percentage of CPU available to group
- Prioritize group for resource allocation
 - E.g., some group might get greater proportion of CPU
- Resource accounting
 - Measure resources used by processes
- Freeze a group
 - Freeze, restore, and checkpoint a group
- And more...

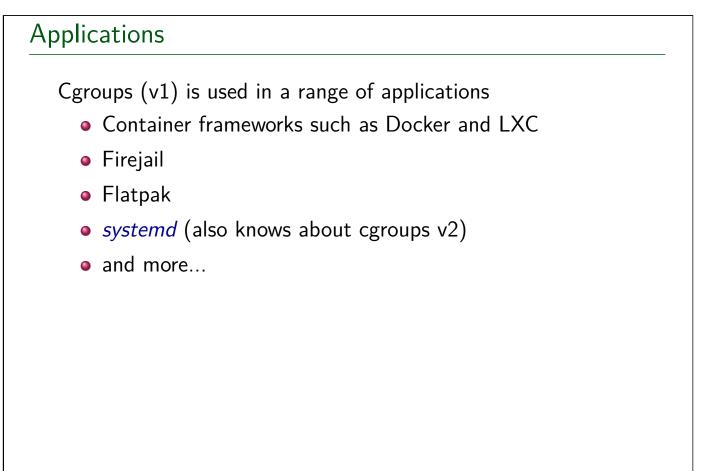










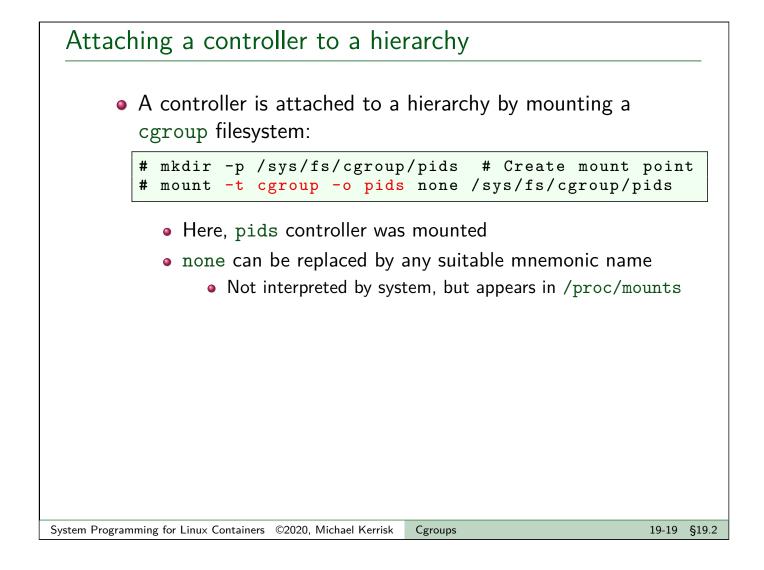


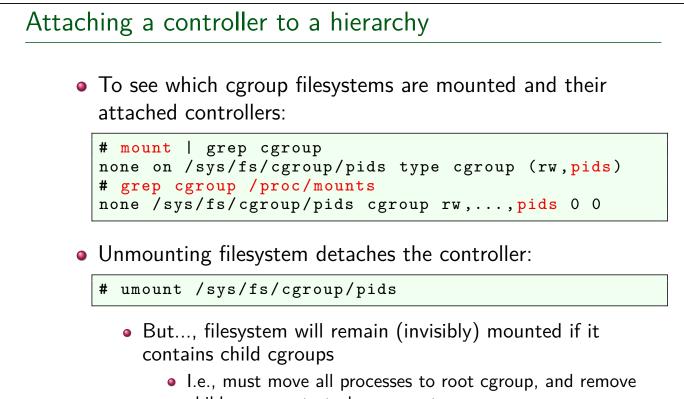
Outline

19	Cgroups	19-1
19.1	Introduction to cgroups v1 and v2	19-3
19.2	Cgroups v1: hierarchies and controllers	19-17
19.3	Cgroups v1: populating a cgroup	19-24
19.4	Cgroups v1: release notification	19-33
19.5	Cgroups v1: a survey of the controllers	19-43
19.6	Cgroups /proc files	19-65
19.7	Cgroup namespaces	19-68

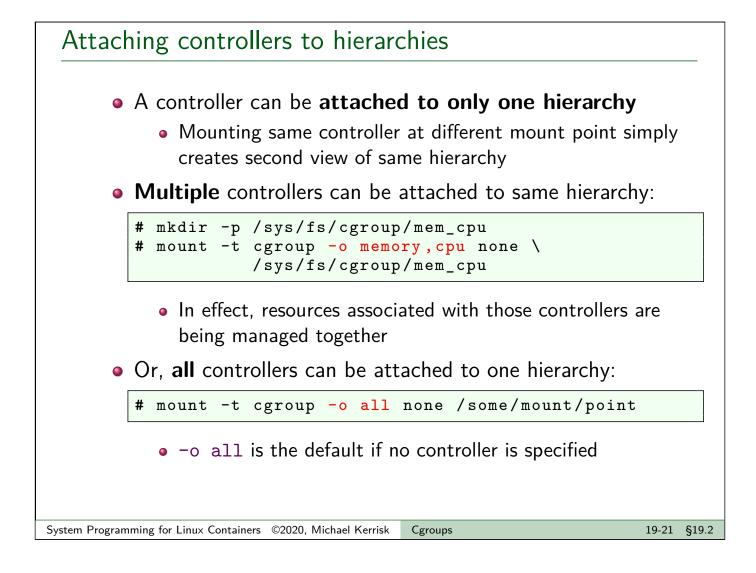
Cgroup hierarchies

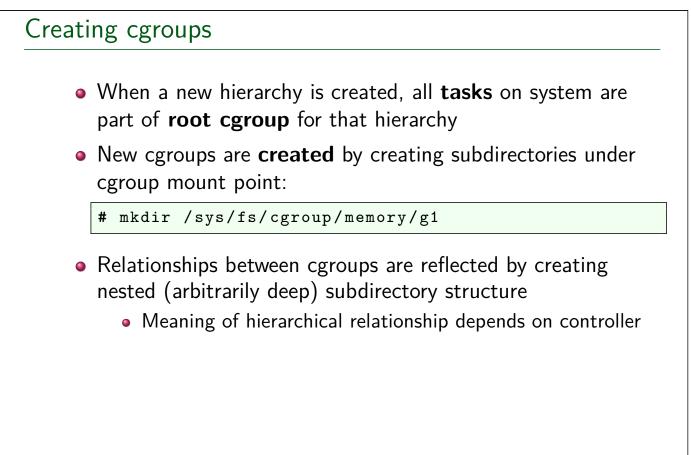
- **Cgroup** == collection of processes
- **Cgroup hierarchy** == hierarchical arrangement of cgroups
 - Implemented via a cgroup pseudo-filesystem
- Structure and membership of cgroup hierarchy is defined by:
 - Mounting a cgroup filesystem
 - Oreating a subdirectory structure that reflects desired cgroup hierarchy
 - Oving processes within hierarchy by writing their PIDs to special files in cgroup subdirectories
 - E.g., cgroup.procs

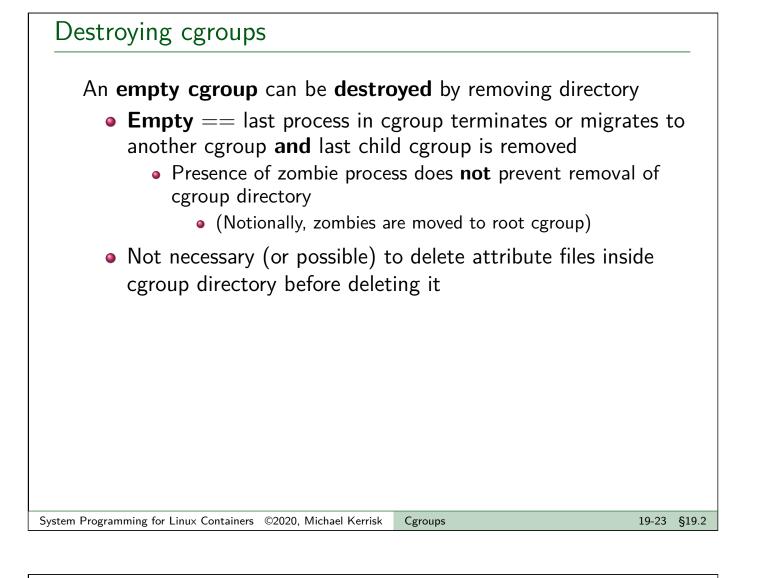




child cgroups, to truly unmount







Outline

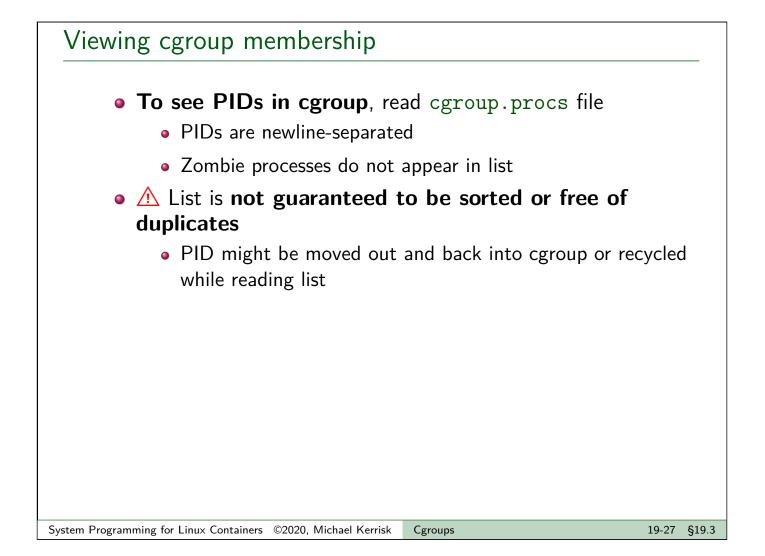
19	Cgroups	19-1
19.1	Introduction to cgroups v1 and v2	19-3
19.2	Cgroups v1: hierarchies and controllers	19-17
19.3	Cgroups v1: populating a cgroup	19-24
19.4	Cgroups v1: release notification	19-33
19.5	Cgroups v1: a survey of the controllers	19-43
19.6	Cgroups /proc files	19-65
19.7	Cgroup namespaces	19-68

Placing a process in a cgroup

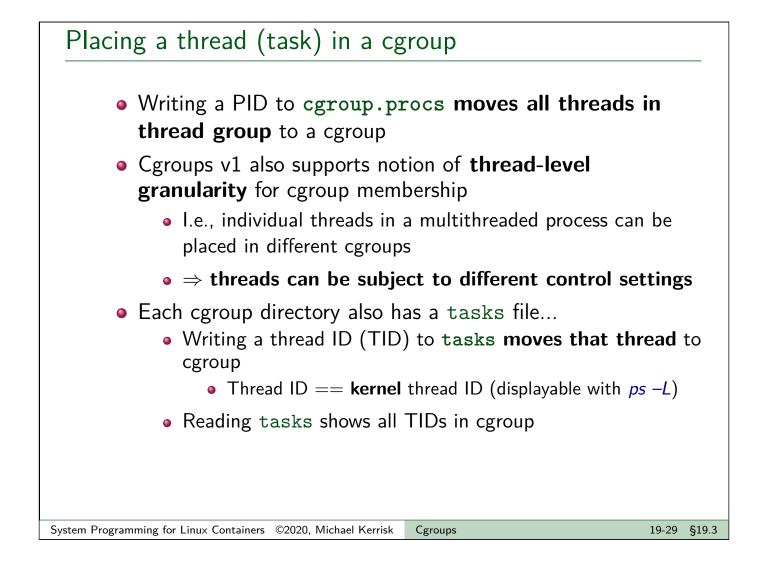
• To move a **process** to a cgroup, we write its PID to cgroup.procs file in corresponding subdirectory

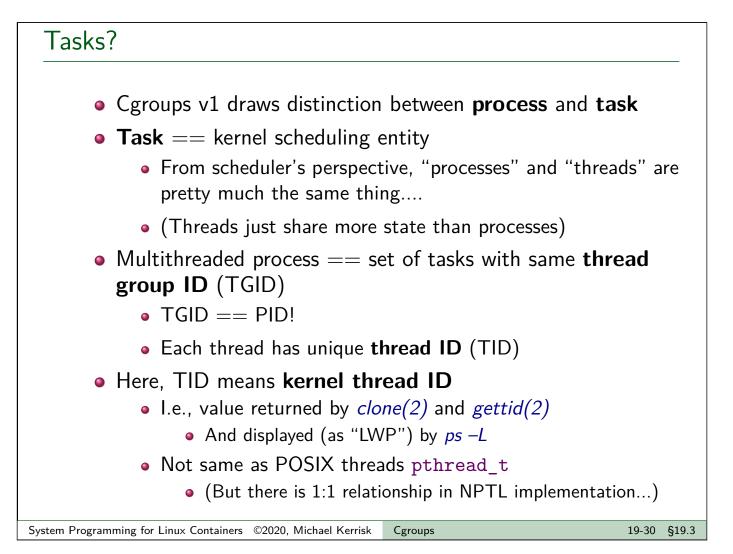
echo \$\$ > /sys/fs/cgroup/memory/g1/cgroup.procs

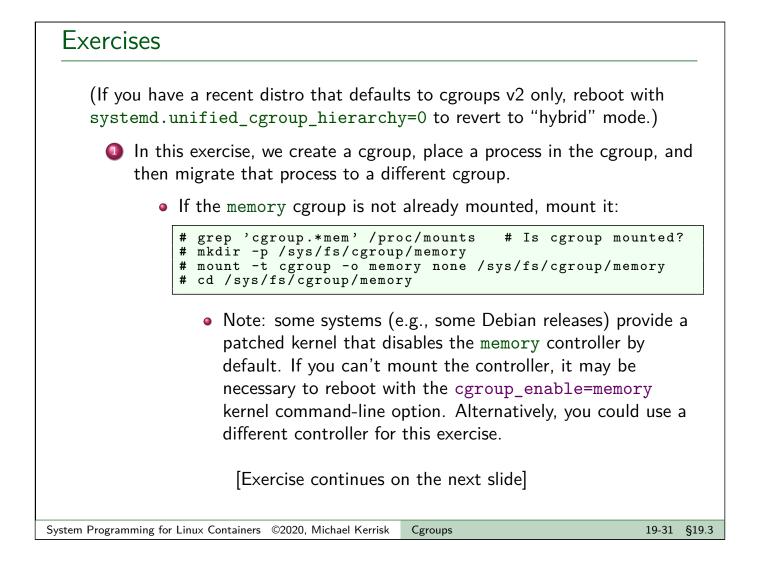
- In multithreaded process, moves all threads to cgroup...
- Can write only one PID at a time
 - write() fails with EINVAL
- Writing 0 to cgroup.procs moves writing process to cgroup



Cgroup membership details Within a hierarchy, a process can be member of just one cgroup That association defines attributes / parameters that apply to the process Adding a process to a different cgroup automatically removes it from previous cgroup A process can be a member of multiple cgroups, each of which is in a different hierarchy On fork(), child inherits cgroup memberships of parent Afterward, cgroup memberships of parent and child can be independently changed







 Create two subdirectories, m1 and m2, in the memory cgroup root directory.
 Execute the following command, and note the PID assigned to the resulting process:
sleep 300 &
 Write the PID of the process created in the previous step into the file m1/cgroup.procs, and verify by reading the file contents.
 Now write the PID of the process into the file m2/cgroup.procs
Is the PID still visible in the file m1/cgroup.procs? Explain.
 Try removing cgroup m1 using the command rm -rf m1. Why doesn't this work?
• Remove the cgroups m1 and m2 using the <i>rmdir</i> command.