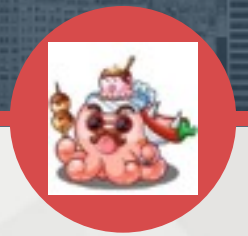
Docker背后的内核知识

Linux Cgroups资源限制



概念

- CGroups（Control Groups）是Linux内核的一个功能，用来限制，控制与分离一个进程组群的资源

- 最佳实践：系统管理员利用CGroups做下面这些事
  - 隔离一个进程集合，并限制它们消费的资源，比如绑定CPU的核
  - 为进程分配足够其使用的内存
  - 为进程分配相应的网络带宽和磁盘存储限制
  - 限制访问某些设备

概念

- CGroups的API以一个伪文件系统实现，用户通过操作文件的方式进行管理

```
cyli@R410:/sys/fs/cgroup$ ls -l
总用量 0
dr-xr-xr-x 6 root root  0 9月  17 19:41 blkio
drwxr-xr-x 2 root root 60 9月  11 12:55 cgmanager
lrwxrwxrwx 1 root root 11 9月  11 12:55 cpu -> cpu,cpuacct
lrwxrwxrwx 1 root root 11 9月  11 12:55 cpuacct -> cpu,cpuacct
dr-xr-xr-x 7 root root  0 9月  20 16:41 cpu,cpuacct
dr-xr-xr-x 4 root root  0 9月  17 20:36 cpuset
dr-xr-xr-x 6 root root  0 9月  17 19:41 devices
dr-xr-xr-x 3 root root  0 9月  17 19:41 freezer
dr-xr-xr-x 3 root root  0 9月  17 19:41 hugetlb
dr-xr-xr-x 8 root root  0 9月  18 09:19 memory
lrwxrwxrwx 1 root root 16 9月  11 12:55 net_cls -> net_cls,net_prio
dr-xr-xr-x 3 root root  0 9月  17 19:41 net_cls,net_prio
lrwxrwxrwx 1 root root 16 9月  11 12:55 net_prio -> net_cls,net_prio
dr-xr-xr-x 3 root root  0 9月  17 19:41 perf_event
dr-xr-xr-x 6 root root  0 9月  18 17:40 pids
dr-xr-xr-x 6 root root  0 9月  17 19:41 systemd
```

概念

- **task(任务)** – task表示系统的一个进程
- **cgroup(控制组)** – cgroups 中的资源控制都以cgroup为单位实现。
- **subsystem(子系统)** – cgroups中的subsystem就是一个资源调度控制器
- **hierarchy(层级树)** – hierarchy由一系列cgroup以一个树状结构排列而成

CPU限制

- deadloop.c

```
#include <stdio.h>

int main(void) {
    int i = 0;
    for(;;) {
        i++;
    }
    return 0;
}
```

- 执行 deadloop

```
  PID USER      PR  NI    VIRT    RES    SHR S  %CPU %MEM     TIME+ COMMAND
15330 cyli      20   0    4224    788    712 R 100.0  0.0   0:23.71 deadloop
```

CPU限制

- 创建cgroup

```
root@R410:/sys/fs/cgroup/cpu# mkdir tinylcy
root@R410:/sys/fs/cgroup/cpu# cd tinylcy/
root@R410:/sys/fs/cgroup/cpu/tinylcy# ls
cgroup.clone_children  cgroup.procs  cpuacct.stat  cpuacct.usage  cpuacct.usage_percpu
```

- CPU 资源限制

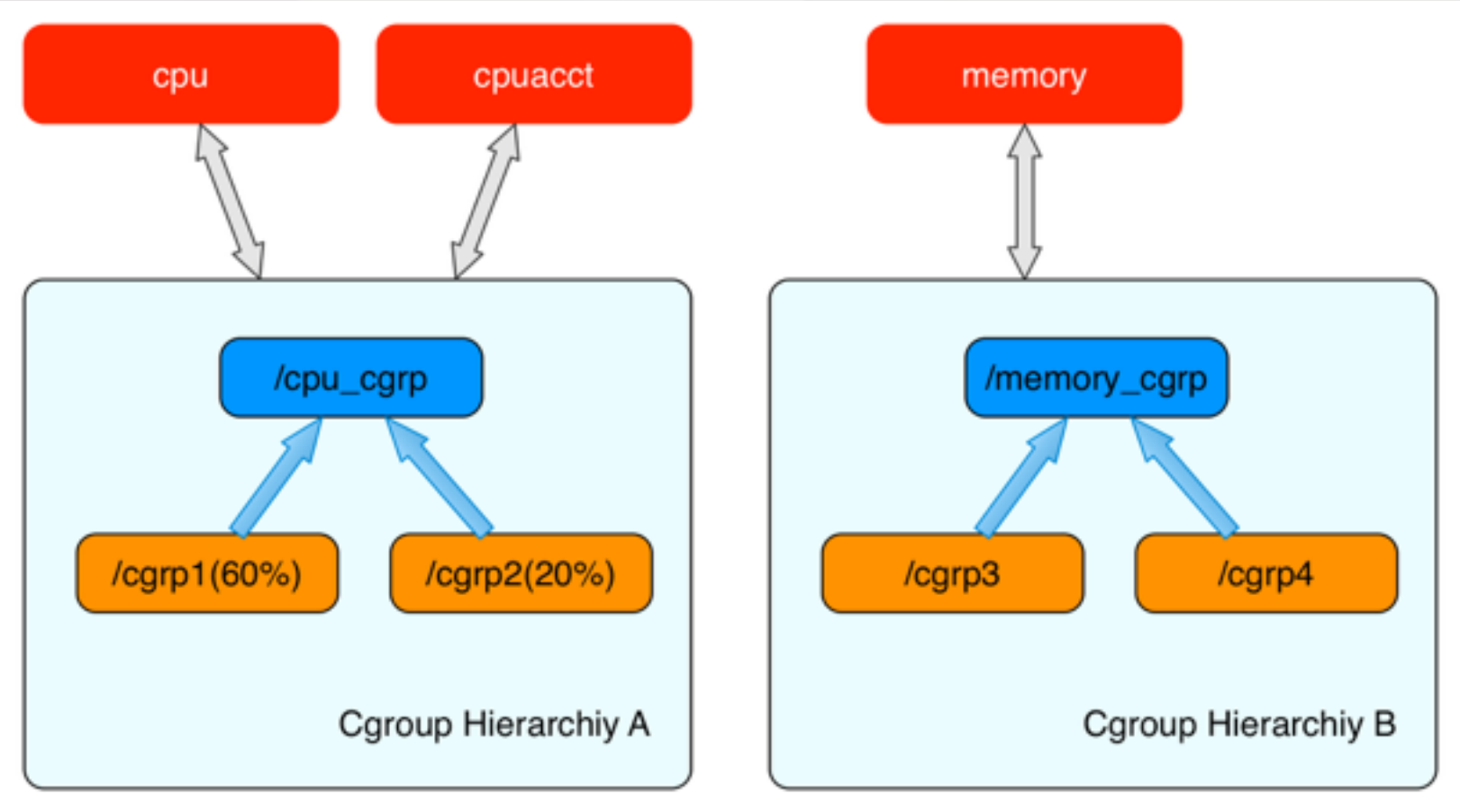
```
root@R410:/sys/fs/cgroup/cpu/tinylcy# cat cpu.cfs_quota_us
-1
root@R410:/sys/fs/cgroup/cpu/tinylcy# echo 20000 > cpu.cfs_quota_us
root@R410:/sys/fs/cgroup/cpu/tinylcy# echo 15330 >> tasks
```

- 查看deaploop资源占用

```
  PID USER      PR  NI    VIRT    RES    SHR S  %CPU %MEM     TIME+ COMMAND
15330 cyli      20   0    4224    788    712 R  19.9  0.0   8:04.85 deadloop
```

CGroups实现原理

- CGroups层级结构



CGroups实现原理

- task_struct

```
1568  #ifdef CONFIG_CGROUPS
1569          /* Control Group info protected by css_set_lock */
1570          struct css_set __rcu *cgroups;
1571          /* cg_list protected by css_set_lock and tsk->alloc_lock */
1572          struct list_head cg_list;
1573  #endif
```

- css_set

```
355           /*
356            * Set of subsystem states, one for each subsystem. This array is
357            * immutable after creation apart from the init_css_set during
358            * subsystem registration (at boot time).
359            */
360           struct cgroup_subsys_state *subsys[CGROUP_SUBSYS_COUNT];
```

- cgroup_subsys_state

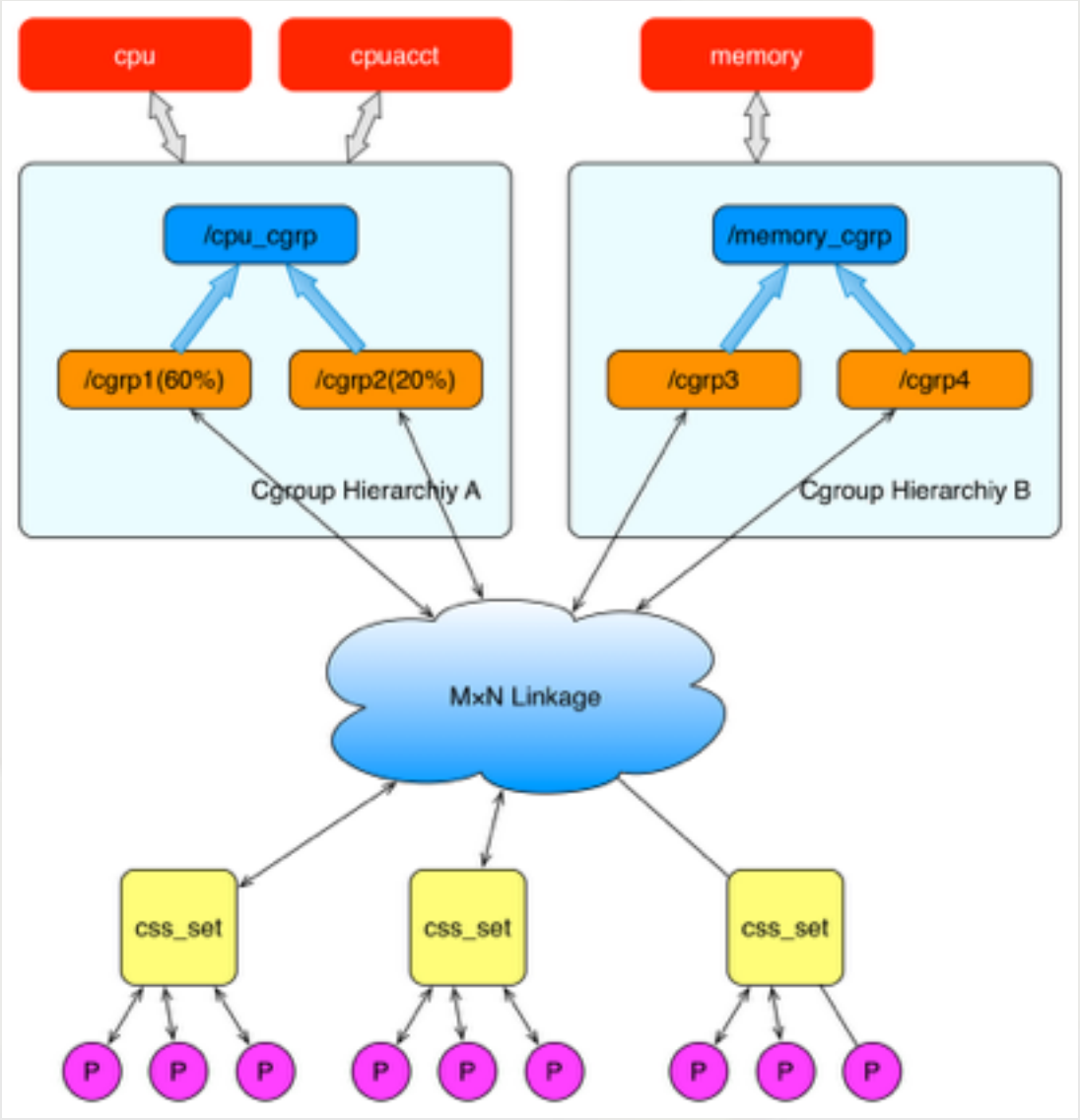
```
58  struct cgroup_subsys_state {
59          /* PI: the cgroup that this css is attached to */
60          struct cgroup *cgroup;
```

CGroups实现原理

- CGroups与进程

- css_set

```
336         /*
337          * Lists running through all tasks using this cgroup group.
338          * mg_tasks lists tasks which belong to this cset but are in the
339          * process of being migrated out or in.  Protected by
340          * css_set_rwsem, but, during migration, once tasks are moved to
341          * mg_tasks, it can be read safely while holding cgroup_mutex.
342          */
343         struct list_head tasks;
344         struct list_head mg_tasks;
```



CGroups实现原理

- Docker如何使用CGroups

```
cyli@R410:/sys/fs/cgroup/memory$ docker run -itd -m 128m nginx
WARNING: Your kernel does not support swap limit capabilities or the cgroup is not mounted. Memory limited without swap.
8c0278724e87e615436cc9222dbc1768540e8543f8254f5755647884d3dddc46
cyli@R410:/sys/fs/cgroup/memory$ cd docker/8c0278724e87e615436cc9222dbc1768540e8543f8254f5755647884d3dddc46/
cyli@R410:/sys/fs/cgroup/memory/docker/8c0278724e87e615436cc9222dbc1768540e8543f8254f5755647884d3dddc46$ cat memory.limit_in_bytes
134217728
```

THANKS