直播应用中的高并发场景实战

2-7 Docker容器
的底层原理

Produced By Idea讲师

Docker底层原理

Docker底层原理

- Namespaces Cgroups
- Cgroups

NameSpace的介绍

NameSpace的介绍

NameSpace其实是一种实现不同进程间资源隔离的机制，不同NameSpace的程序，可以享有一份独立的系统资源

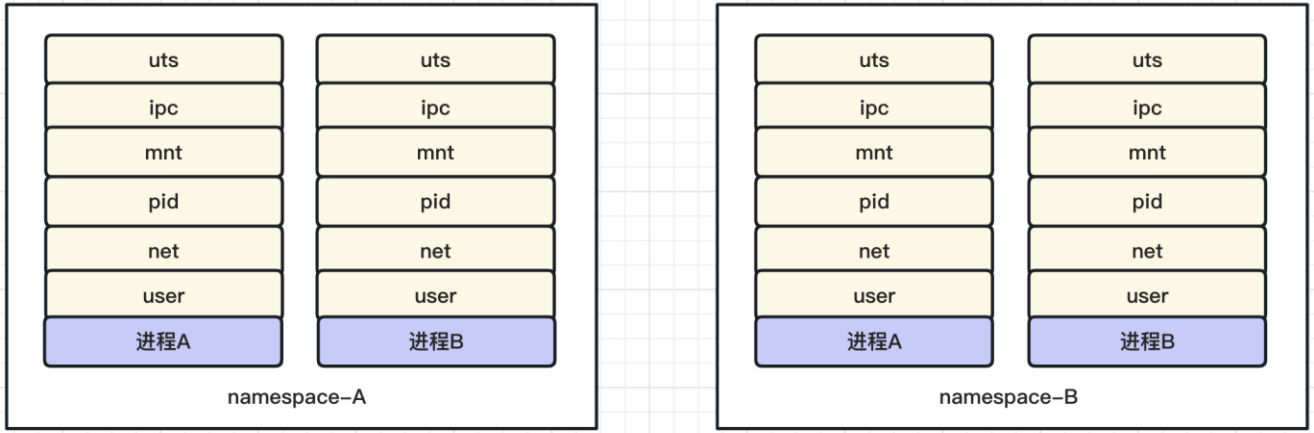
两个不同的namespaces下部署了两个进程，这两个进程的pid可能是相同的

NameSpace的源码

可以理解为
这样的一段
数据结构

```
struct task_struct {
        //...省略部分字段
        struct nsproxy *nsproxy;
    }
    struct nsproxy {
        //...省略部分字段
        atomic_t count;
        struct uts_namespace *uts_ns;
        struct ipc_namespace *ipc_ns;
        struct mnt_namespace *mnt_ns;
        struct pid_namespace *pid_ns_for_children;
        struct net_namespace *net_ns;
    }
```

NameSpace的原理图



NameSpace的组成

- Mount: 隔离文件系统挂载点
- UTS: 隔离主机名和域名信息
- IPC: 隔离进程间通信
- PID: 隔离进程的ID
- Network: 隔离网络资源
- User: 隔离用户和用户组的ID

```
[root@instance-j4y6c6os ns]# ll
total 0
lrwxrwxrwx 1 root root 0 Apr  4 16:52 cgroup -> 'cgroup:[4026531835]'
lrwxrwxrwx 1 root root 0 Apr  4 16:52 ipc -> 'ipc:[4026531839]'
lrwxrwxrwx 1 root root 0 Apr  4 16:52 mnt -> 'mnt:[4026531840]'
lrwxrwxrwx 1 root root 0 Apr  4 16:52 net -> 'net:[4026531992]'
lrwxrwxrwx 1 root root 0 Apr  4 16:52 pid -> 'pid:[4026531836]'
lrwxrwxrwx 1 root root 0 Apr  5 10:05 pid_for_children -> 'pid:[4026531836]'
lrwxrwxrwx 1 root root 0 Apr  5 10:05 time -> 'time:[4026531834]'
lrwxrwxrwx 1 root root 0 Apr  5 10:05 time_for_children -> 'time:[4026531834]'
lrwxrwxrwx 1 root root 0 Apr  4 16:52 user -> 'user:[4026531837]'
lrwxrwxrwx 1 root root 0 Apr  4 16:52 uts -> 'uts:[4026531838]'
```

NameSpace的设置

- clone
- setns
- setons

NameSpace的实战

- 进入到/proc/$pid/ns 目录下去查看指定进程的NameSpaces信息
- 比对下非Docker进程下的服务是否NameSpace ID相同
- 比对下Docker进程的NameSpace ID是否不同

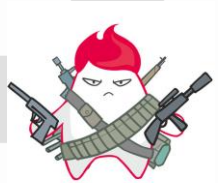
Cgroups的介绍

Cgroups的含义

Cgroups全称Control Groups，是Linux内核提供的物理资源隔离机制，通过这种机制，可以实现对Linux进程或者进程组的资源限制、隔离和统计功能

Cgroups的核心组成

- cpu: 限制进程的 cpu 使用率
- memory: 限制进程的memory使用量
- ns: 控制cgroups中的进程使用不同的namespace



CPU子系统的控制

- cpu.shares: cgroup对时间的分配。比如cgroup A设置的是1，cgroup B设置的是2，那么B中的任务获取cpu的时间，是A中任务的2倍
- cpu.cfs_period_us: 完全公平调度器的调整时间配额的周期
- cpu.cfs_quota_us: 完全公平调度器的周期当中可以占用的时间

CPU子系统的控制

查看Linux下Cgroups相关文件

```
[root@instance-j4y6c6os cgroup]# ll
total 0
dr-xr-xr-x 6 root root  0 Apr  4 09:57 blkio
lrwxrwxrwx 1 root root 11 Apr  4 09:57 cpu -> cpu,cpuacct
lrwxrwxrwx 1 root root 11 Apr  4 09:57 cpuacct -> cpu,cpuacct
dr-xr-xr-x 7 root root  0 Apr  4 09:57 cpu,cpuacct
dr-xr-xr-x 3 root root  0 Apr  4 09:57 cpuset
dr-xr-xr-x 6 root root  0 Apr  4 09:57 devices
dr-xr-xr-x 3 root root  0 Apr  4 09:57 freezer
dr-xr-xr-x 3 root root  0 Apr  4 09:57 hugetlb
dr-xr-xr-x 7 root root  0 Apr  4 09:57 memory
dr-xr-xr-x 3 root root  0 Apr  4 09:57 net_cls
dr-xr-xr-x 3 root root  0 Apr  4 09:57 perf_event
dr-xr-xr-x 6 root root  0 Apr  4 09:57 pids
dr-xr-xr-x 6 root root  0 Apr  4 09:57 systemd
dr-xr-xr-x 2 root root  0 Apr  4 09:57 tcp_throt
dr-xr-xr-x 2 root root  0 Apr  4 09:57 tos_cgroup
[root@instance-j4y6c6os cgroup]#
```

CPU子系统的控制

部署一个耗CPU的程序，然后控制它的CPU占用比例

```
cd /sys/fs/cgroup/cpu/[new file]

echo 10000 > ./cpu.cfs_quota_us

echo 100000 > ./cpu.cfs_period_us

echo $pid > ./cgroup.procs
```

撸起袖子，干~