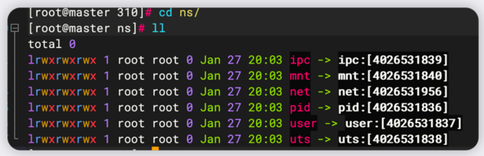
# linux namespace(kernel5.6)

- Linux namespace
  - 是linux提供一种内核级别的隔离方法,命名空间将全局系统资源隔离，是命名空间内的进程看起来拥有自己的全局资源。
- chroot
  - 可以改变当前运行进程以及其子进程的根目录，调用chroot后，后续命令针对新的根目录(/)运行。
- 支持namespace隔离
  - PID 容器中init进程(pid 1)
    - 不能被kill，因为kernel会ignore PID =1进程
    - Init进程被销毁，其ns下所有进程会收到kill信号
    - 为什么1号进程不能被杀死?
  - UTS(Unix time-sharing system) 主机和域名
  - IPC 信号量、消息队列和共享内存
  - User 隔离安全相关标识符和属性，包括用户ID, 组ID、root目录，通俗的讲一个普通用户的进程通过clone创建新的进程在新user ns中可以拥有不同的用户和用户组.(一个进程在容器外是一个没有特权的普通用户，但是在容器中可以是超级用户)
  - mount 通过隔离文件系统挂载点对隔离文件支持
  - Cgroup Control group root directory
  - network 网络资源隔离
    - 网络设备
    - IPv4和IPv6协议栈
    - IP路由表
    - iptables
    - /proc/net
    - /sys/class/net
    - 套接字
  - time 运行进程看到不同的系统时间
- 每个进程对应一个/proc/[pid]/ns
  - 里面保存每个进程的链接文件
  - 如果两个进程的namespace编号相同，则说明两个进程在同一个ns




- namespace生命周期
  - 各种类型的namespace在没有干扰的情况下，当namespace最后一个进程终止或者离开该namespace的时候，此namespace会自动销毁
  - 一些其他情况namespace会继续存在
    - 1.对应的/proc/[pid]/ns* 被打开或者挂载
    - 2.包含层级关系(PID或者user name),这两个ns比较特殊，是分层设计的
    - 3.一个user ns绑定了为销毁的NonUser ns
    - 4.PID ns对应的proc文件系统被挂载
- 3个系统API
  - clone 通过clone创建新进程同时创建namespace
  - setns 通过setns()加入一个已经存在的ns，比如docker exec使用到此方法
  - unshare 让进程脱离到新的namespace