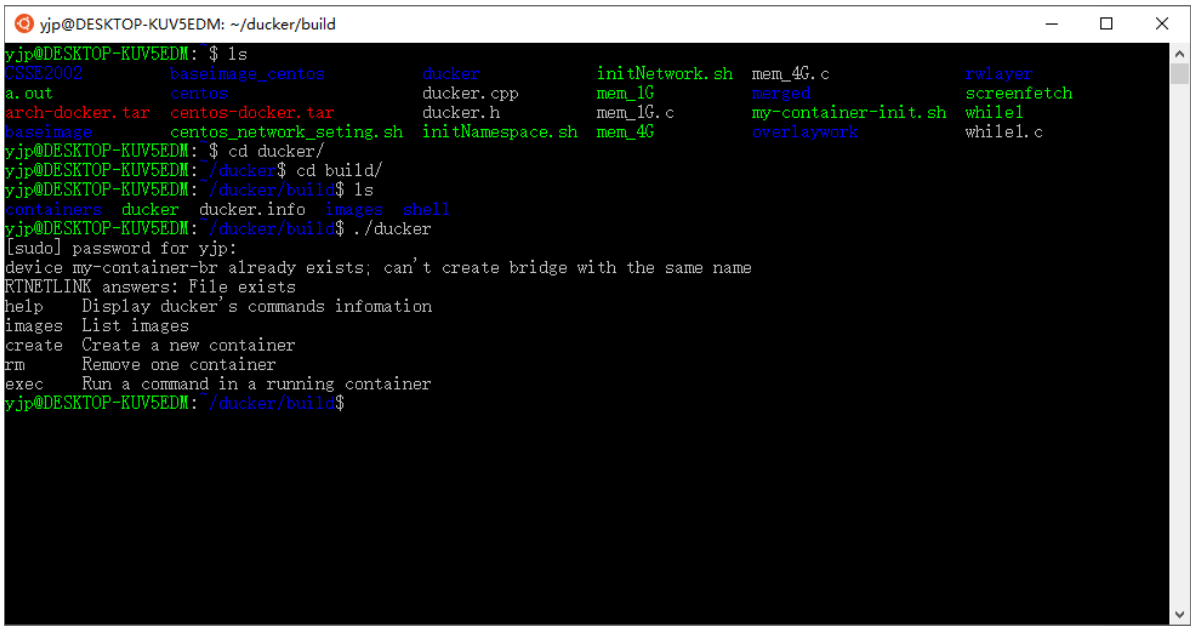
实验报告

ducker

ducker是一个模仿docker的容器demo，实现了一小部分docker的命令
目前已经实现了部分命令，如create创建容器，rm删除容器，exec在容器内执行命令等。接下来展示各个命令的效果，并说明其效果，以及达到的实验目标。

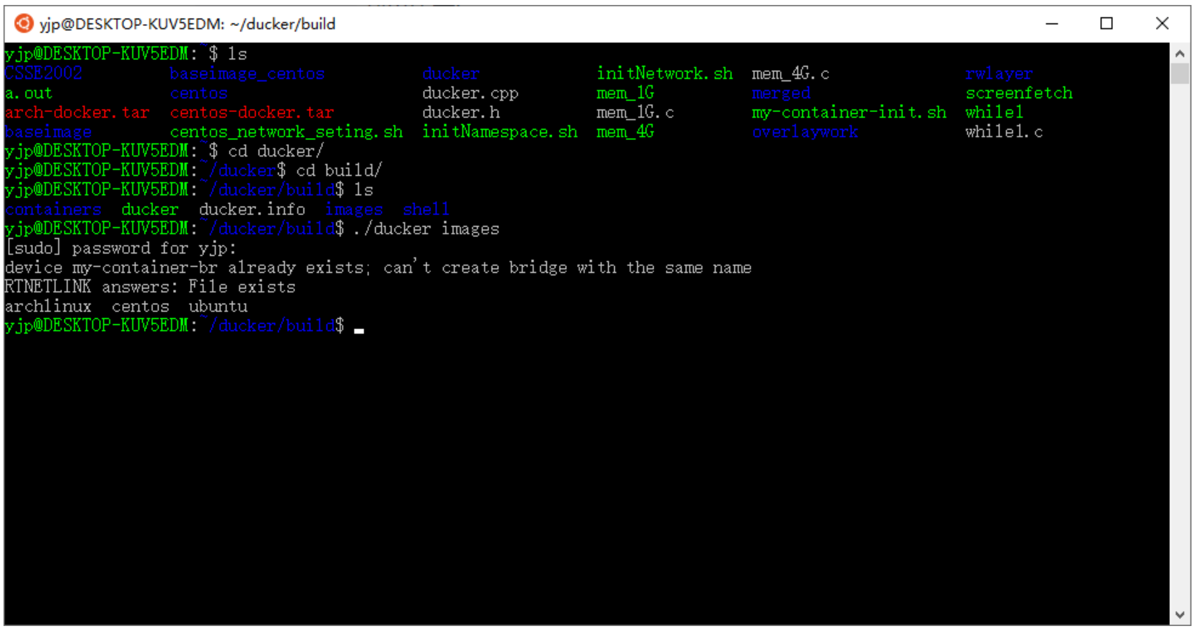


yjp@DESKTOP-KUV5EDM: ~/ducker/build

```
yjp@DESKTOP-KUV5EDM:~$ ls
CSSE2002          baseimage_centos          ducker            initNetwork.sh  mem_4G.c               rwlayer
a.out             centos                    ducker.cpp        mem_1G          merged                 screenfetch
arch-docker.tar   centos-docker.tar         ducker.h          mem_1G.c        my-container-init.sh   while1
baseimage         centos_network_seting.sh  initNamespace.sh  mem_4G          overlaywork            while1.c
yjp@DESKTOP-KUV5EDM:~$ cd ducker/
yjp@DESKTOP-KUV5EDM:~/ducker$ cd build/
yjp@DESKTOP-KUV5EDM:~/ducker/build$ ls
containers  ducker  ducker.info  images  shell
yjp@DESKTOP-KUV5EDM:~/ducker/build$ ./ducker
[sudo] password for yjp:
device my-container-br already exists; can't create bridge with the same name
RTNETLINK answers: File exists
help    Display ducker's commands infomation
images  List images
create  Create a new container
rm      Remove one container
exec    Run a command in a running container
yjp@DESKTOP-KUV5EDM:~/ducker/build$
```

images

`ducker images` 列出所有image



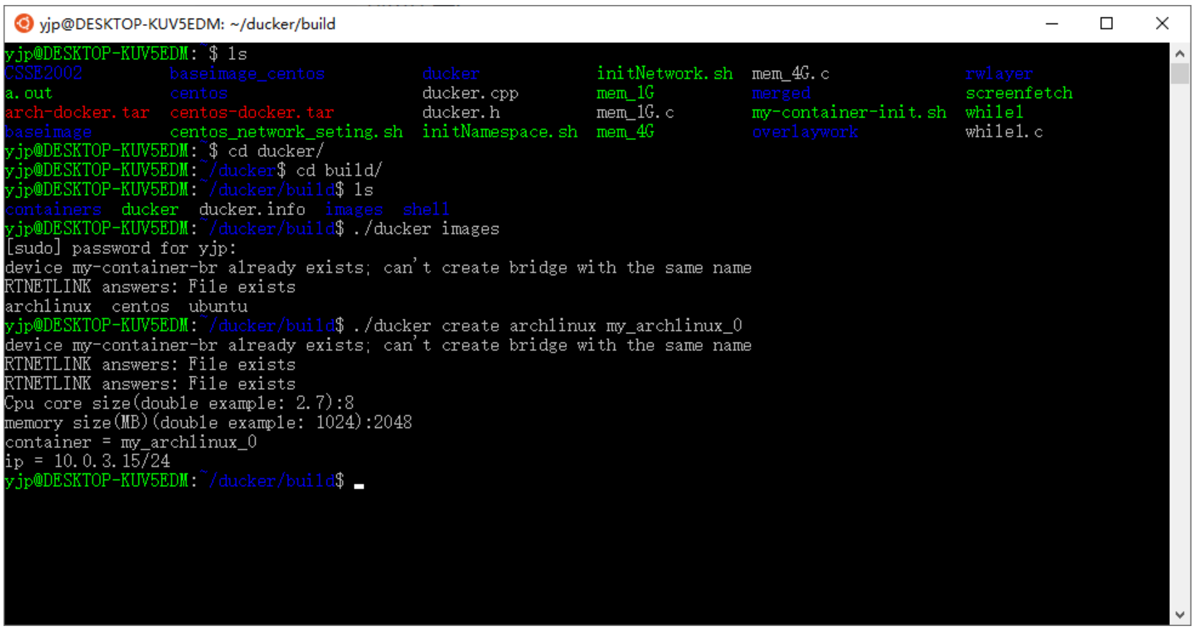
yjp@DESKTOP-KUV5EDM: ~/ducker/build

```
yjp@DESKTOP-KUV5EDM:~$ ls
CSSE2002          baseimage_centos          ducker            initNetwork.sh  mem_4G.c               rwlayer
a.out             centos                    ducker.cpp        mem_1G          merged                 screenfetch
arch-docker.tar   centos-docker.tar         ducker.h          mem_1G.c        my-container-init.sh   while1
baseimage         centos_network_seting.sh  initNamespace.sh  mem_4G          overlaywork            while1.c
yjp@DESKTOP-KUV5EDM:~$ cd ducker/
yjp@DESKTOP-KUV5EDM:~/ducker$ cd build/
yjp@DESKTOP-KUV5EDM:~/ducker/build$ ls
containers  ducker  ducker.info  images  shell
yjp@DESKTOP-KUV5EDM:~/ducker/build$ ./ducker images
[sudo] password for yjp:
device my-container-br already exists; can't create bridge with the same name
RTNETLINK answers: File exists
archlinux  centos  ubuntu
yjp@DESKTOP-KUV5EDM:~/ducker/build$ _
```

可以看到拥有archlinux，centos，ubuntu三个image

create

`ducker create [image] [container]` 生成容器

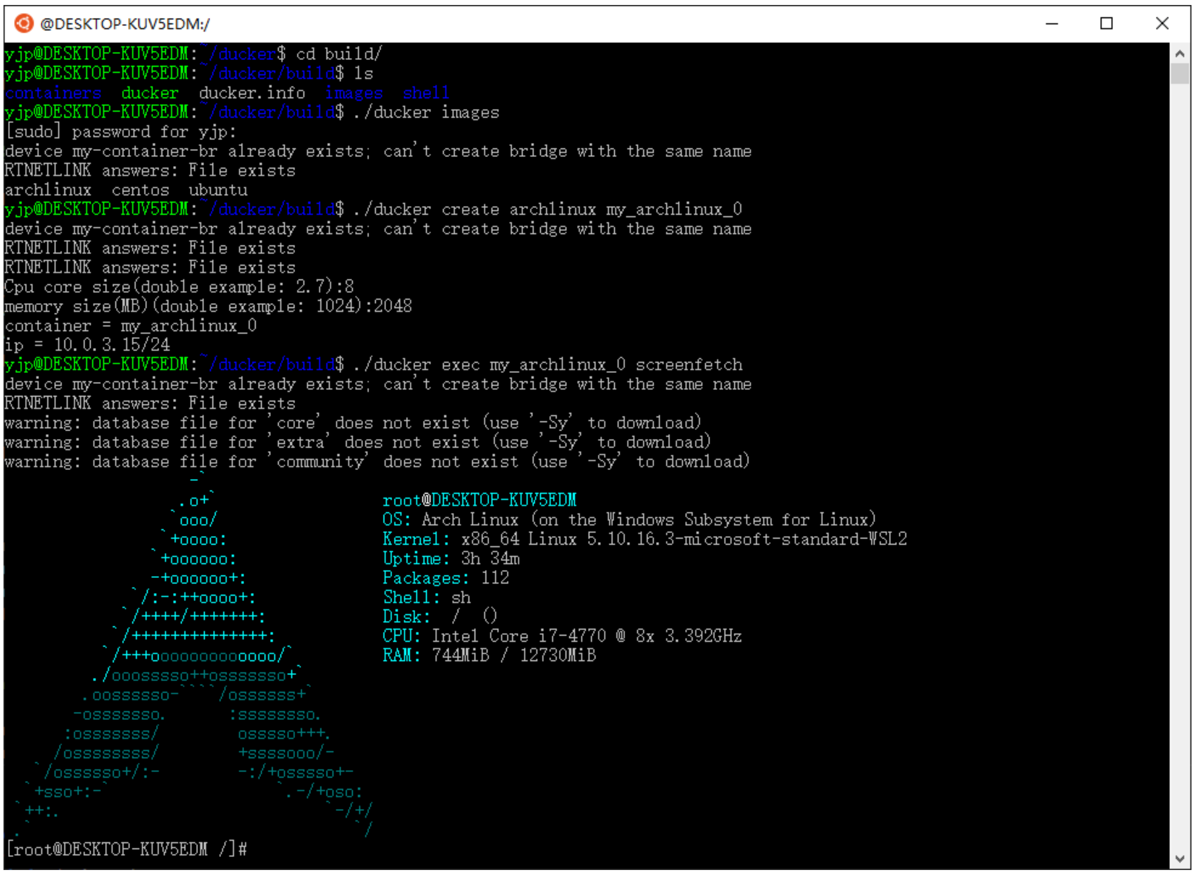


```
yjp@DESKTOP-KUV5EDM:~$ ls
CSSE2002         baseimage_centos          ducker           initNetwork.sh  mem_4G.c             rwlayer
a.out            centos                    ducker.cpp       mem_1G          merged               screenfetch
arch-docker.tar  centos-docker.tar         ducker.h         mem_1G.c        my-container-init.sh while1
baseimage        centos_network_seting.sh  initNamespace.sh mem_4G          overlaywork          while1.c
yjp@DESKTOP-KUV5EDM:~$ cd ducker/
yjp@DESKTOP-KUV5EDM:~/ducker$ cd build/
yjp@DESKTOP-KUV5EDM:~/ducker/build$ ls
containers  ducker  ducker.info  images  shell
yjp@DESKTOP-KUV5EDM:~/ducker/build$ ./ducker images
[sudo] password for yjp:
device my-container-br already exists; can't create bridge with the same name
RTNETLINK answers: File exists
archlinux  centos  ubuntu
yjp@DESKTOP-KUV5EDM:~/ducker/build$ ./ducker create archlinux my_archlinux_0
device my-container-br already exists; can't create bridge with the same name
RTNETLINK answers: File exists
RTNETLINK answers: File exists
Cpu core size(double example: 2.7):8
memory size(MB)(double example: 1024):2048
container = my_archlinux_0
ip = 10.0.3.15/24
yjp@DESKTOP-KUV5EDM:~/ducker/build$ _
```

生成容器时，ducker会提示用户输入分配的cpu数量以及内存数量，ducker会为这个container生成对应的cgroup并设置相关信息，以限制container使用的资源。

exec

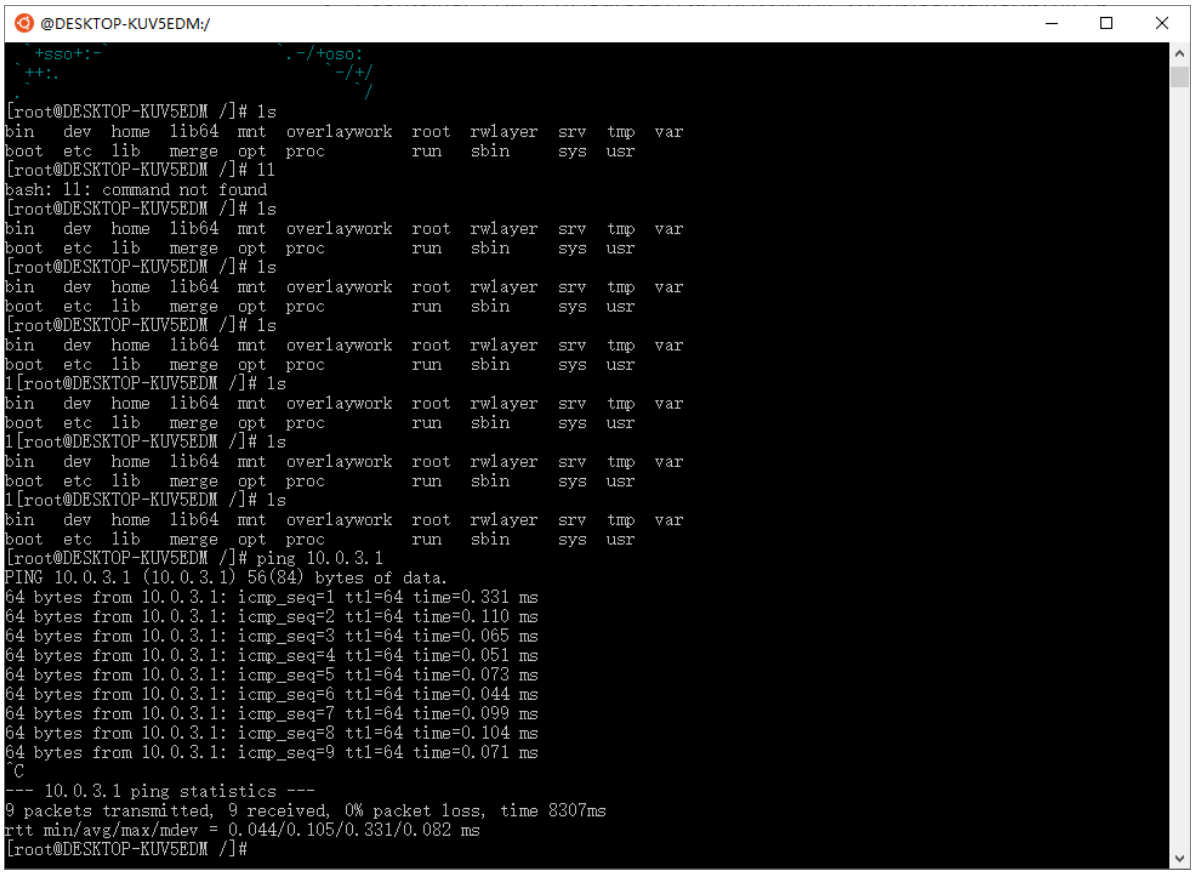
`ducker exec [image] [commands...]` 在容器中执行命令



```
yjp@DESKTOP-KUV5EDM:~/ducker$ cd build/
yjp@DESKTOP-KUV5EDM:~/ducker/build$ ls
containers  ducker  ducker.info  images  shell
yjp@DESKTOP-KUV5EDM:~/ducker/build$ ./ducker images
[sudo] password for yjp:
device my-container-br already exists; can't create bridge with the same name
RTNETLINK answers: File exists
archlinux  centos  ubuntu
yjp@DESKTOP-KUV5EDM:~/ducker/build$ ./ducker create archlinux my_archlinux_0
device my-container-br already exists; can't create bridge with the same name
RTNETLINK answers: File exists
RTNETLINK answers: File exists
Cpu core size(double example: 2.7):8
memory size(MB)(double example: 1024):2048
container = my_archlinux_0
ip = 10.0.3.15/24
yjp@DESKTOP-KUV5EDM:~/ducker/build$ ./ducker exec my_archlinux_0 screenfetch
device my-container-br already exists; can't create bridge with the same name
RTNETLINK answers: File exists
warning: database file for 'core' does not exist (use '-Sy' to download)
warning: database file for 'extra' does not exist (use '-Sy' to download)
warning: database file for 'community' does not exist (use '-Sy' to download)
                   -`
                  .o+`                 root@DESKTOP-KUV5EDM
                 `ooo/                 OS: Arch Linux (on the Windows Subsystem for Linux)
                `+oooo:                Kernel: x86_64 Linux 5.10.16.3-microsoft-standard-WSL2
               `+oooooo:               Uptime: 3h 34m
               -+oooooo+:              Packages: 112
             `/:-:++oooo+:             Shell: sh
            `/++++/+++++++:            Disk:  /  ()
           `/++++++++++++++:           CPU: Intel Core i7-4770 @ 8x 3.392GHz
          `/+++ooooooooooooo/`         RAM: 744MiB / 12730MiB
         ./ooosssso++osssssso+`
        .oossssso-````/ossssss+`
       -osssssso.      :ssssssso.
      :osssssss/        osssso+++.
     /ossssssss/        +ssssooo/-
   `/ossssso+/:-        -:/+osssso+-
  `+sso+:-`                 `.-/+oso:
 `++:.                           `-/+/
.`                                 `/
[root@DESKTOP-KUV5EDM /]#
```

可以发现我们已经成功在ubuntu上运行了archlinux，并执行screenfetch，打印了archlinux的logo。执行完成后，停留在了container的bash，我们接着测试网络以及进程、文件系统的隔离情况。

网络连接测试

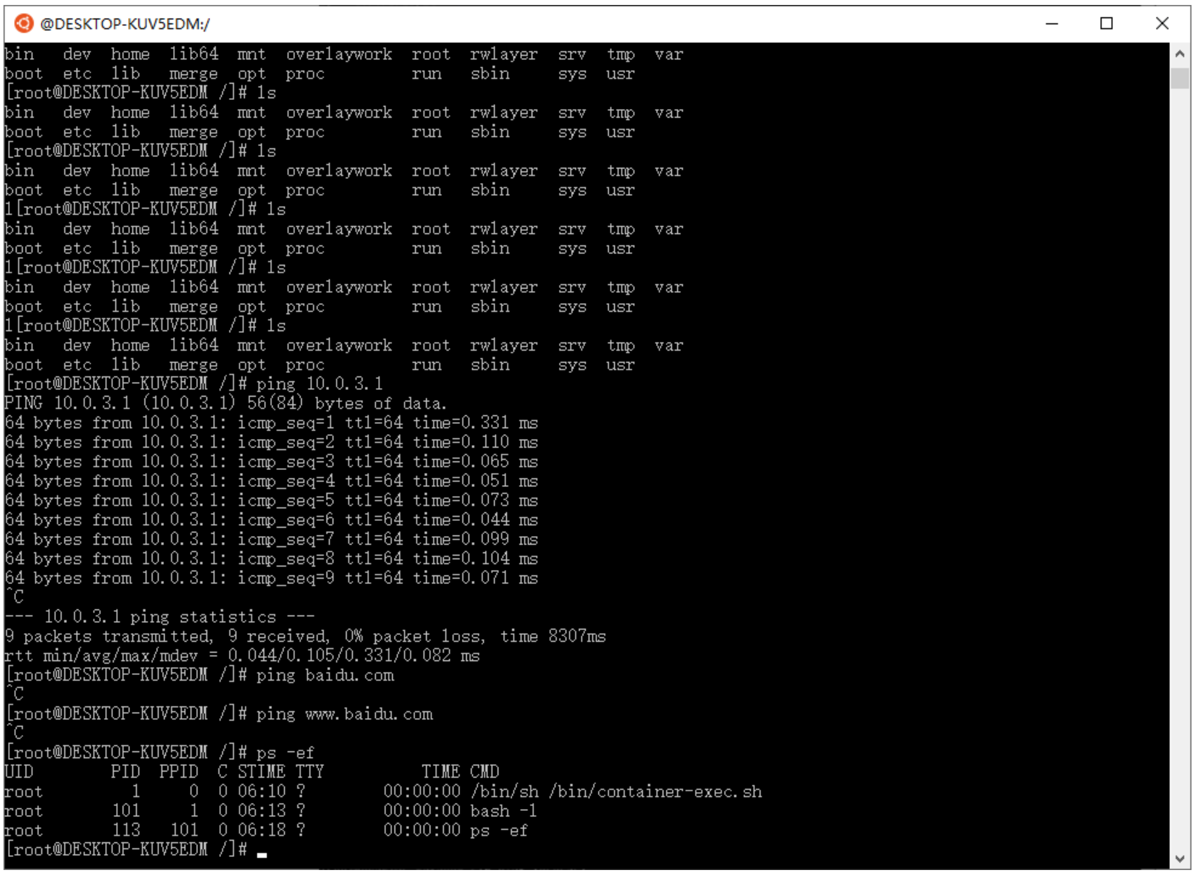


```
@DESKTOP-KUV5EDM:/
  `+sso+:-`                       `.-/+oso:
 `++:.                                 `-/+/
  `                                       `/
 .
[root@DESKTOP-KUV5EDM /]# ls
bin   dev  home  lib64  mnt  overlaywork  root  rwlayer  srv  tmp  var
boot  etc  lib   merge  opt  proc         run   sbin     sys  usr
[root@DESKTOP-KUV5EDM /]# ll
bash: ll: command not found
[root@DESKTOP-KUV5EDM /]# ls
bin   dev  home  lib64  mnt  overlaywork  root  rwlayer  srv  tmp  var
boot  etc  lib   merge  opt  proc         run   sbin     sys  usr
[root@DESKTOP-KUV5EDM /]# ls
bin   dev  home  lib64  mnt  overlaywork  root  rwlayer  srv  tmp  var
boot  etc  lib   merge  opt  proc         run   sbin     sys  usr
[root@DESKTOP-KUV5EDM /]# ls
bin   dev  home  lib64  mnt  overlaywork  root  rwlayer  srv  tmp  var
boot  etc  lib   merge  opt  proc         run   sbin     sys  usr
l[root@DESKTOP-KUV5EDM /]# ls
bin   dev  home  lib64  mnt  overlaywork  root  rwlayer  srv  tmp  var
boot  etc  lib   merge  opt  proc         run   sbin     sys  usr
l[root@DESKTOP-KUV5EDM /]# ls
bin   dev  home  lib64  mnt  overlaywork  root  rwlayer  srv  tmp  var
boot  etc  lib   merge  opt  proc         run   sbin     sys  usr
l[root@DESKTOP-KUV5EDM /]# ls
bin   dev  home  lib64  mnt  overlaywork  root  rwlayer  srv  tmp  var
boot  etc  lib   merge  opt  proc         run   sbin     sys  usr
[root@DESKTOP-KUV5EDM /]# ping 10.0.3.1
PING 10.0.3.1 (10.0.3.1) 56(84) bytes of data.
64 bytes from 10.0.3.1: icmp_seq=1 ttl=64 time=0.331 ms
64 bytes from 10.0.3.1: icmp_seq=2 ttl=64 time=0.110 ms
64 bytes from 10.0.3.1: icmp_seq=3 ttl=64 time=0.065 ms
64 bytes from 10.0.3.1: icmp_seq=4 ttl=64 time=0.051 ms
64 bytes from 10.0.3.1: icmp_seq=5 ttl=64 time=0.073 ms
64 bytes from 10.0.3.1: icmp_seq=6 ttl=64 time=0.044 ms
64 bytes from 10.0.3.1: icmp_seq=7 ttl=64 time=0.099 ms
64 bytes from 10.0.3.1: icmp_seq=8 ttl=64 time=0.104 ms
64 bytes from 10.0.3.1: icmp_seq=9 ttl=64 time=0.071 ms
^C
--- 10.0.3.1 ping statistics ---
9 packets transmitted, 9 received, 0% packet loss, time 8307ms
rtt min/avg/max/mdev = 0.044/0.105/0.331/0.082 ms
[root@DESKTOP-KUV5EDM /]#
```

可以看到我们是能ping通host机的。

进程隔离测试

接着我们输入`ps -ef`

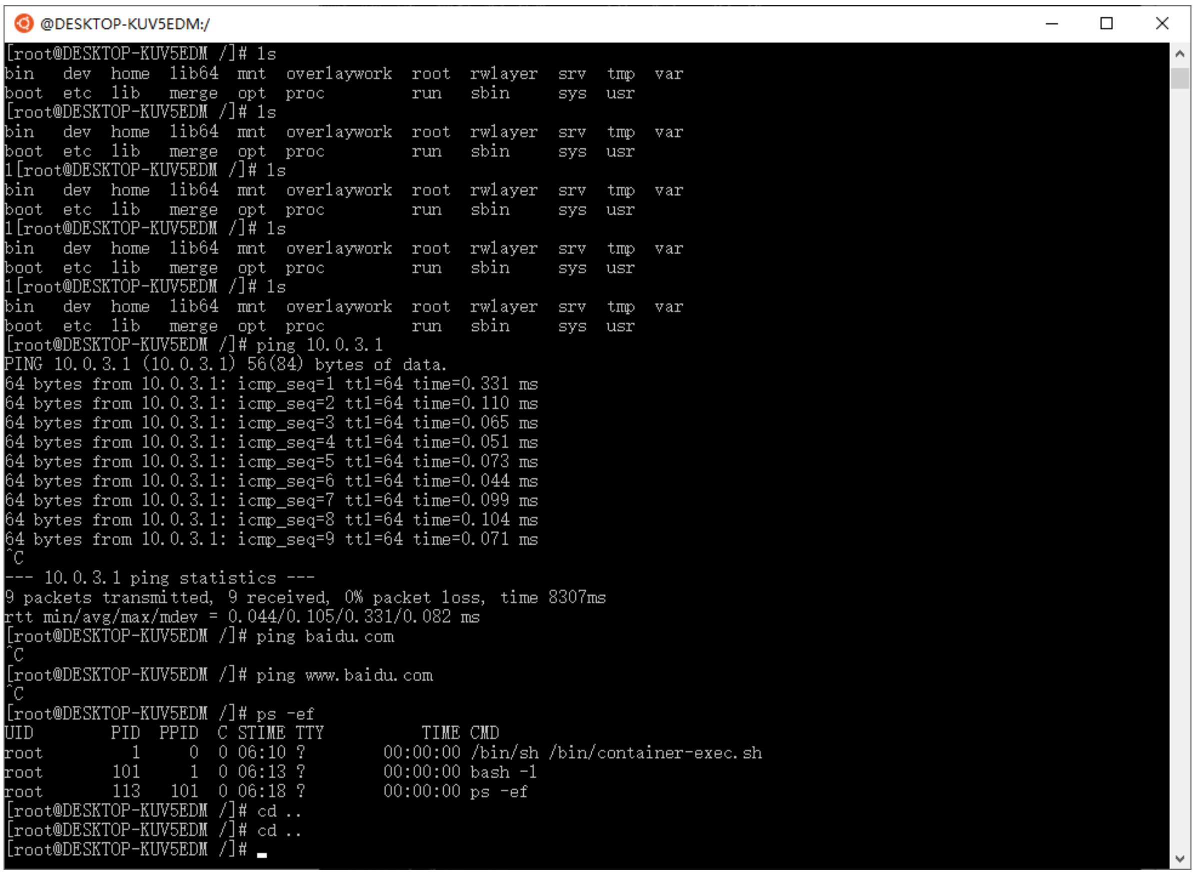


```
@DESKTOP-KUV5EDM:/
bin   dev  home  lib64  mnt  overlaywork  root  rwlayer  srv  tmp  var
boot  etc  lib   merge  opt  proc         run   sbin     sys  usr
[root@DESKTOP-KUV5EDM /]# ls
bin   dev  home  lib64  mnt  overlaywork  root  rwlayer  srv  tmp  var
boot  etc  lib   merge  opt  proc         run   sbin     sys  usr
[root@DESKTOP-KUV5EDM /]# ls
bin   dev  home  lib64  mnt  overlaywork  root  rwlayer  srv  tmp  var
boot  etc  lib   merge  opt  proc         run   sbin     sys  usr
l[root@DESKTOP-KUV5EDM /]# ls
bin   dev  home  lib64  mnt  overlaywork  root  rwlayer  srv  tmp  var
boot  etc  lib   merge  opt  proc         run   sbin     sys  usr
l[root@DESKTOP-KUV5EDM /]# ls
bin   dev  home  lib64  mnt  overlaywork  root  rwlayer  srv  tmp  var
boot  etc  lib   merge  opt  proc         run   sbin     sys  usr
l[root@DESKTOP-KUV5EDM /]# ls
bin   dev  home  lib64  mnt  overlaywork  root  rwlayer  srv  tmp  var
boot  etc  lib   merge  opt  proc         run   sbin     sys  usr
[root@DESKTOP-KUV5EDM /]# ping 10.0.3.1
PING 10.0.3.1 (10.0.3.1) 56(84) bytes of data.
64 bytes from 10.0.3.1: icmp_seq=1 ttl=64 time=0.331 ms
64 bytes from 10.0.3.1: icmp_seq=2 ttl=64 time=0.110 ms
64 bytes from 10.0.3.1: icmp_seq=3 ttl=64 time=0.065 ms
64 bytes from 10.0.3.1: icmp_seq=4 ttl=64 time=0.051 ms
64 bytes from 10.0.3.1: icmp_seq=5 ttl=64 time=0.073 ms
64 bytes from 10.0.3.1: icmp_seq=6 ttl=64 time=0.044 ms
64 bytes from 10.0.3.1: icmp_seq=7 ttl=64 time=0.099 ms
64 bytes from 10.0.3.1: icmp_seq=8 ttl=64 time=0.104 ms
64 bytes from 10.0.3.1: icmp_seq=9 ttl=64 time=0.071 ms
^C
--- 10.0.3.1 ping statistics ---
9 packets transmitted, 9 received, 0% packet loss, time 8307ms
rtt min/avg/max/mdev = 0.044/0.105/0.331/0.082 ms
[root@DESKTOP-KUV5EDM /]# ping baidu.com
^C
[root@DESKTOP-KUV5EDM /]# ping www.baidu.com
^C
[root@DESKTOP-KUV5EDM /]# ps -ef
UID        PID  PPID  C STIME TTY          TIME CMD
root         1     0  0 06:10 ?        00:00:00 /bin/sh /bin/container-exec.sh
root       101     1  0 06:13 ?        00:00:00 bash -l
root       113   101  0 06:18 ?        00:00:00 ps -ef
[root@DESKTOP-KUV5EDM /]# _
```

可以见的，我们已经将进程进行了隔离。

文件系统隔离测试

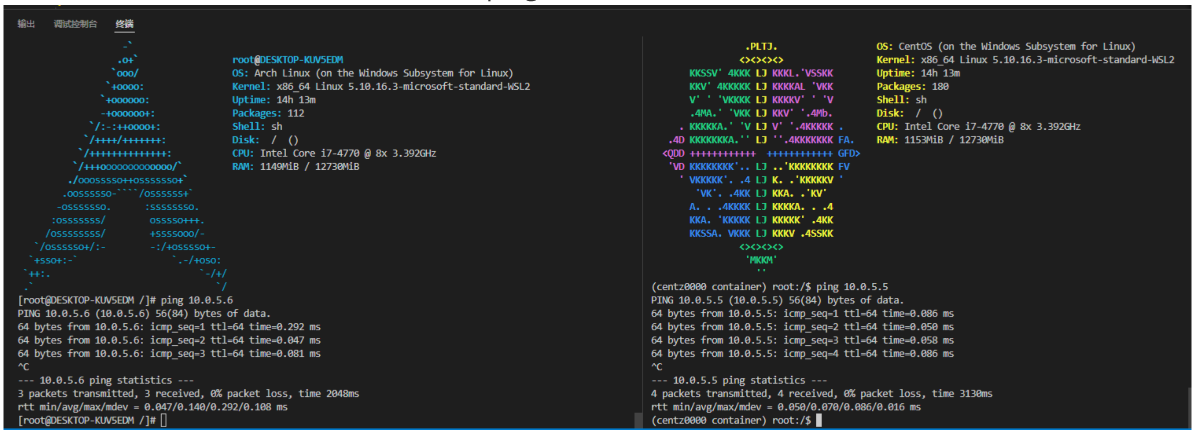
执行`cd ..`仍然在根目录，我们完成了文件系统的隔离



```
@DESKTOP-KUV5EDM:/
[root@DESKTOP-KUV5EDM /]# ls
bin   dev  home  lib64  mnt  overlaywork  root  rwlayer  srv  tmp  var
boot  etc  lib   merge  opt  proc         run   sbin     sys  usr
[root@DESKTOP-KUV5EDM /]# ls
bin   dev  home  lib64  mnt  overlaywork  root  rwlayer  srv  tmp  var
boot  etc  lib   merge  opt  proc         run   sbin     sys  usr
1[root@DESKTOP-KUV5EDM /]# ls
bin   dev  home  lib64  mnt  overlaywork  root  rwlayer  srv  tmp  var
boot  etc  lib   merge  opt  proc         run   sbin     sys  usr
1[root@DESKTOP-KUV5EDM /]# ls
bin   dev  home  lib64  mnt  overlaywork  root  rwlayer  srv  tmp  var
boot  etc  lib   merge  opt  proc         run   sbin     sys  usr
1[root@DESKTOP-KUV5EDM /]# ls
bin   dev  home  lib64  mnt  overlaywork  root  rwlayer  srv  tmp  var
boot  etc  lib   merge  opt  proc         run   sbin     sys  usr
[root@DESKTOP-KUV5EDM /]# ping 10.0.3.1
PING 10.0.3.1 (10.0.3.1) 56(84) bytes of data.
64 bytes from 10.0.3.1: icmp_seq=1 ttl=64 time=0.331 ms
64 bytes from 10.0.3.1: icmp_seq=2 ttl=64 time=0.110 ms
64 bytes from 10.0.3.1: icmp_seq=3 ttl=64 time=0.065 ms
64 bytes from 10.0.3.1: icmp_seq=4 ttl=64 time=0.051 ms
64 bytes from 10.0.3.1: icmp_seq=5 ttl=64 time=0.073 ms
64 bytes from 10.0.3.1: icmp_seq=6 ttl=64 time=0.044 ms
64 bytes from 10.0.3.1: icmp_seq=7 ttl=64 time=0.099 ms
64 bytes from 10.0.3.1: icmp_seq=8 ttl=64 time=0.104 ms
64 bytes from 10.0.3.1: icmp_seq=9 ttl=64 time=0.071 ms
^C
--- 10.0.3.1 ping statistics ---
9 packets transmitted, 9 received, 0% packet loss, time 8307ms
rtt min/avg/max/mdev = 0.044/0.105/0.331/0.082 ms
[root@DESKTOP-KUV5EDM /]# ping baidu.com
^C
[root@DESKTOP-KUV5EDM /]# ping www.baidu.com
^C
[root@DESKTOP-KUV5EDM /]# ps -ef
UID        PID  PPID  C STIME TTY          TIME CMD
root         1     0  0 06:10 ?        00:00:00 /bin/sh /bin/container-exec.sh
root       101     1  0 06:13 ?        00:00:00 bash -l
root       113   101  0 06:18 ?        00:00:00 ps -ef
[root@DESKTOP-KUV5EDM /]# cd ..
[root@DESKTOP-KUV5EDM /]# cd ..
[root@DESKTOP-KUV5EDM /]# 
```

容器间网络互连测试

接着我们开启centos的container，并尝试ping通两个container。

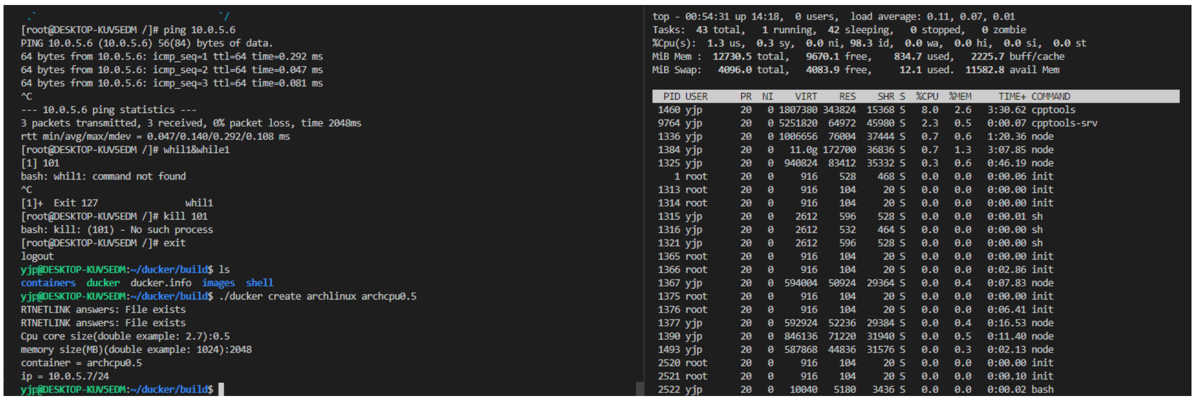


```
root@DESKTOP-KUV5EDM
OS: Arch Linux (on the Windows Subsystem for Linux)
Kernel: x86_64 Linux 5.10.16.3-microsoft-standard-WSL2
Uptime: 14h 13m
Packages: 112
Shell: sh
Disk:  /  ()
CPU: Intel Core i7-4770 @ 8x 3.392GHz
RAM: 1149MiB / 12730MiB

[root@DESKTOP-KUV5EDM /]# ping 10.0.5.6
PING 10.0.5.6 (10.0.5.6) 56(84) bytes of data.
64 bytes from 10.0.5.6: icmp_seq=1 ttl=64 time=0.292 ms
64 bytes from 10.0.5.6: icmp_seq=2 ttl=64 time=0.047 ms
64 bytes from 10.0.5.6: icmp_seq=3 ttl=64 time=0.081 ms
^C
--- 10.0.5.6 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2048ms
rtt min/avg/max/mdev = 0.047/0.140/0.292/0.108 ms
[root@DESKTOP-KUV5EDM /]# 
```

```
OS: CentOS (on the Windows Subsystem for Linux)
Kernel: x86_64 Linux 5.10.16.3-microsoft-standard-WSL2
Uptime: 14h 13m
Packages: 180
Shell: sh
Disk:  /  ()
CPU: Intel Core i7-4770 @ 8x 3.392GHz
RAM: 1153MiB / 12730MiB

(centz0000 container) root:/$ ping 10.0.5.5
PING 10.0.5.5 (10.0.5.5) 56(84) bytes of data.
64 bytes from 10.0.5.5: icmp_seq=1 ttl=64 time=0.086 ms
64 bytes from 10.0.5.5: icmp_seq=2 ttl=64 time=0.050 ms
64 bytes from 10.0.5.5: icmp_seq=3 ttl=64 time=0.058 ms
64 bytes from 10.0.5.5: icmp_seq=4 ttl=64 time=0.086 ms
^C
--- 10.0.5.5 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3130ms
rtt min/avg/max/mdev = 0.050/0.070/0.086/0.016 ms
(centz0000 container) root:/$ 
```

可以见的，两个container已经能够相互ping通。

资源定量分配测试

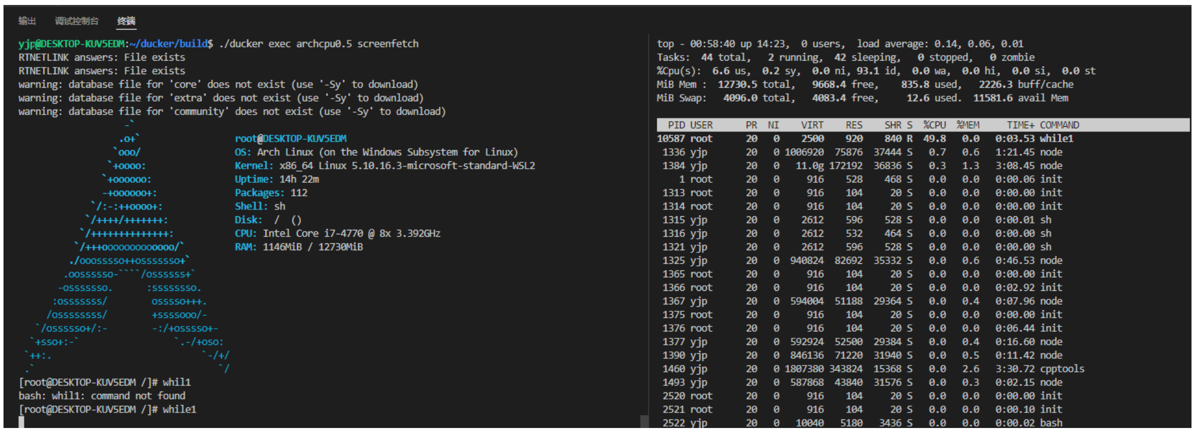


```
 .`                              `/
[root@DESKTOP-KUV5EDM /]# ping 10.0.5.6
PING 10.0.5.6 (10.0.5.6) 56(84) bytes of data.
64 bytes from 10.0.5.6: icmp_seq=1 ttl=64 time=0.292 ms
64 bytes from 10.0.5.6: icmp_seq=2 ttl=64 time=0.047 ms
64 bytes from 10.0.5.6: icmp_seq=3 ttl=64 time=0.081 ms
^C
--- 10.0.5.6 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2048ms
rtt min/avg/max/mdev = 0.047/0.140/0.292/0.108 ms
[root@DESKTOP-KUV5EDM /]# whil1&while1
[1] 101
bash: whil1: command not found
^C
[1]+  Exit 127                whil1
[root@DESKTOP-KUV5EDM /]# kill 101
bash: kill: (101) - No such process
[root@DESKTOP-KUV5EDM /]# exit
logout
yjp@DESKTOP-KUV5EDM:~/ducker/build$ ls
containers  ducker  ducker.info  images  shell
yjp@DESKTOP-KUV5EDM:~/ducker/build$ ./ducker create archlinux archcpu0.5
RTNETLINK answers: File exists
RTNETLINK answers: File exists
Cpu core size(double example: 2.7):0.5
memory size(MB)(double example: 1024):2048
container = archcpu0.5
ip = 10.0.5.7/24
yjp@DESKTOP-KUV5EDM:~/ducker/build$ 
```

```
top - 00:54:31 up 14:18,  0 users,  load average: 0.11, 0.07, 0.01
Tasks:  43 total,   1 running,  42 sleeping,   0 stopped,   0 zombie
%Cpu(s):  1.3 us,  0.3 sy,  0.0 ni, 98.3 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
MiB Mem :  12730.5 total,   9670.1 free,    834.7 used,   2225.7 buff/cache
MiB Swap:   4096.0 total,   4083.9 free,     12.1 used.  11582.8 avail Mem

  PID USER      PR  NI    VIRT    RES    SHR S  %CPU  %MEM     TIME+ COMMAND
 1460 yjp       20   0 1807380 343824  15368 S   8.0   2.6   3:30.62 cpptools
 9764 yjp       20   0 5251820  64972  45980 S   2.3   0.5   0:00.07 cpptools-srv
 1336 yjp       20   0 1006656  76004  37444 S   0.7   0.6   1:20.36 node
 1384 yjp       20   0   11.0g 172700  36836 S   0.7   1.3   3:07.85 node
 1325 yjp       20   0  940824  83412  35332 S   0.3   0.6   0:46.19 node
    1 root      20   0     916    528    468 S   0.0   0.0   0:00.06 init
 1313 root      20   0     916    104     20 S   0.0   0.0   0:00.00 init
 1314 root      20   0     916    104     20 S   0.0   0.0   0:00.00 init
 1315 yjp       20   0    2612    596    528 S   0.0   0.0   0:00.01 sh
 1316 yjp       20   0    2612    532    464 S   0.0   0.0   0:00.00 sh
 1321 yjp       20   0    2612    596    528 S   0.0   0.0   0:00.00 sh
 1365 root      20   0     916    104     20 S   0.0   0.0   0:00.00 init
 1366 root      20   0     916    104     20 S   0.0   0.0   0:02.86 init
 1367 yjp       20   0  594004  50924  29364 S   0.0   0.4   0:07.83 node
 1375 root      20   0     916    104     20 S   0.0   0.0   0:00.00 init
 1376 root      20   0     916    104     20 S   0.0   0.0   0:06.41 init
 1377 yjp       20   0  592924  52236  29384 S   0.0   0.4   0:16.53 node
 1390 yjp       20   0  846136  71220  31940 S   0.0   0.5   0:11.40 node
 1493 yjp       20   0  587868  44836  31576 S   0.0   0.3   0:02.13 node
 2520 root      20   0     916    104     20 S   0.0   0.0   0:00.00 init
 2521 root      20   0     916    104     20 S   0.0   0.0   0:00.10 init
 2522 yjp       20   0   10040   5180   3436 S   0.0   0.0   0:00.02 bash
```

我们给`archcpu0.5`分配了0.5个cpu资源以及2GB的内存，接下来进行资源测试

执行`while1`程序，可以见的cpu资源保持在50%说明只使用了50%的cpu资源

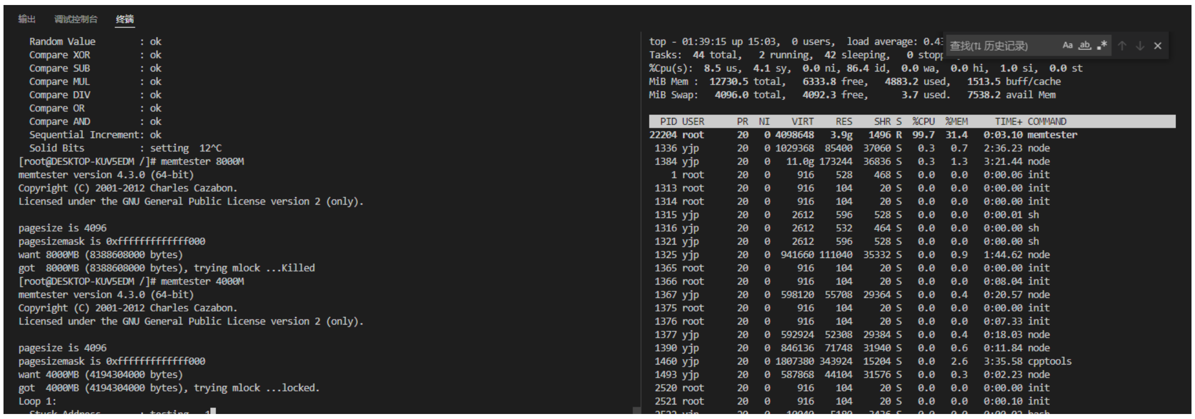


```
yjp@DESKTOP-KUV5EDM:~/ducker/build$ ./ducker exec archcpu0.5 screenfetch
RTNETLINK answers: File exists
RTNETLINK answers: File exists
warning: database file for 'core' does not exist (use '-Sy' to download)
warning: database file for 'extra' does not exist (use '-Sy' to download)
warning: database file for 'community' does not exist (use '-Sy' to download)
                   -`
                  .o+`                 root@DESKTOP-KUV5EDM
                 `ooo/                 OS: Arch Linux (on the Windows Subsystem for Linux)
                `+oooo:                Kernel: x86_64 Linux 5.10.16.3-microsoft-standard-WSL2
               `+oooooo:               Uptime: 14h 22m
               -+oooooo+:              Packages: 112
             `/:-:++oooo+:             Shell: sh
            `/++++/+++++++:            Disk:  /  ()
           `/++++++++++++++:           CPU: Intel Core i7-4770 @ 8x 3.392GHz
          `/+++ooooooooooooo/`         RAM: 1146MiB / 12730MiB
         ./ooosssso++osssssso+`
        .oossssso-````/ossssss+`
       -osssssso.      :ssssssso.
      :osssssss/        osssso+++.
     /ossssssss/        +ssssooo/-
   `/ossssso+/:-        -:/+osssso+-
  `+sso+:-`                 `.-/+oso:
 `++:.                           `-/+/
 .`                                 `/
[root@DESKTOP-KUV5EDM /]# whil1
bash: whil1: command not found
[root@DESKTOP-KUV5EDM /]# while1
```

```
top - 00:58:40 up 14:23,  0 users,  load average: 0.14, 0.06, 0.01
Tasks:  44 total,   2 running,  42 sleeping,   0 stopped,   0 zombie
%Cpu(s):  6.6 us,  0.2 sy,  0.0 ni, 93.1 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
MiB Mem :  12730.5 total,   9668.4 free,    835.8 used,   2226.3 buff/cache
MiB Swap:   4096.0 total,   4083.4 free,     12.6 used.  11581.6 avail Mem

  PID USER      PR  NI    VIRT    RES    SHR S  %CPU  %MEM     TIME+ COMMAND
10587 root      20   0    2500    920    840 R  49.8   0.0   0:03.53 while1
 1336 yjp       20   0 1006920  75876  37444 S   0.7   0.6   1:21.45 node
 1384 yjp       20   0   11.0g 172192  36836 S   0.3   1.3   3:08.45 node
    1 root      20   0     916    528    468 S   0.0   0.0   0:00.06 init
 1313 root      20   0     916    104     20 S   0.0   0.0   0:00.00 init
 1314 root      20   0     916    104     20 S   0.0   0.0   0:00.00 init
 1315 yjp       20   0    2612    596    528 S   0.0   0.0   0:00.01 sh
 1316 yjp       20   0    2612    532    464 S   0.0   0.0   0:00.00 sh
 1321 yjp       20   0    2612    596    528 S   0.0   0.0   0:00.00 sh
 1325 yjp       20   0  940824  82692  35332 S   0.0   0.6   0:46.53 node
 1365 root      20   0     916    104     20 S   0.0   0.0   0:00.00 init
 1366 root      20   0     916    104     20 S   0.0   0.0   0:02.92 init
 1367 yjp       20   0  594004  51188  29364 S   0.0   0.4   0:07.96 node
 1375 root      20   0     916    104     20 S   0.0   0.0   0:00.00 init
 1376 root      20   0     916    104     20 S   0.0   0.0   0:06.44 init
 1377 yjp       20   0  592924  52500  29384 S   0.0   0.4   0:16.60 node
 1390 yjp       20   0  846136  71220  31940 S   0.0   0.5   0:11.42 node
 1460 yjp       20   0 1807380 343824  15368 S   0.0   2.6   3:30.72 cpptools
 1493 yjp       20   0  587868  43840  31576 S   0.0   0.3   0:02.15 node
 2520 root      20   0     916    104     20 S   0.0   0.0   0:00.00 init
 2521 root      20   0     916    104     20 S   0.0   0.0   0:00.10 init
 2522 yjp       20   0   10040   5180   3436 S   0.0   0.0   0:00.02 bash
```

我们给`archmem4G`分配了8个cpu资源以及4GB的内存，接下来进行内存资源测试

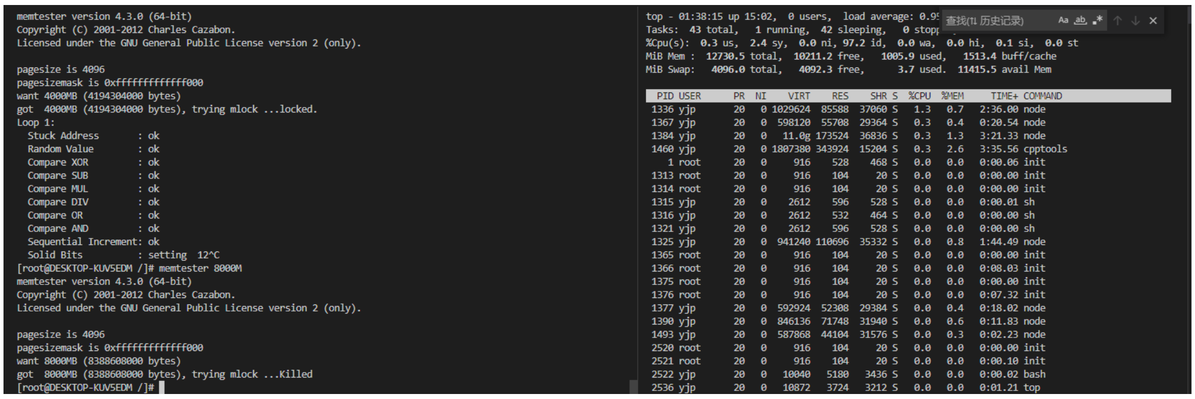
执行`memtester 4000M`程序，可以见得程序占据了31.4%的内存，说明已经成功分配了4000M的内存



```
  Random Value       : ok
  Compare XOR        : ok
  Compare SUB        : ok
  Compare MUL        : ok
  Compare DIV        : ok
  Compare OR         : ok
  Compare AND        : ok
  Sequential Increment: ok
  Solid Bits         : setting  12^C
[root@DESKTOP-KUV5EDM /]# memtester 8000M
memtester version 4.3.0 (64-bit)
Copyright (C) 2001-2012 Charles Cazabon.
Licensed under the GNU General Public License version 2 (only).

pagesize is 4096
pagesizemask is 0xfffffffffffff000
want 8000MB (8388608000 bytes)
got  8000MB (8388608000 bytes), trying mlock ...Killed
[root@DESKTOP-KUV5EDM /]# memtester 4000M
memtester version 4.3.0 (64-bit)
Copyright (C) 2001-2012 Charles Cazabon.
Licensed under the GNU General Public License version 2 (only).

pagesize is 4096
pagesizemask is 0xfffffffffffff000
want 4000MB (4194304000 bytes)
got  4000MB (4194304000 bytes), trying mlock ...locked.
Loop 1:
  Stuck Address      : testing   1
```

```
top - 01:39:15 up 15:03,  0 users,  load average: 0.4
Tasks:  44 total,   2 running,  42 sleeping,   0 stopp
%Cpu(s):  8.5 us,  4.1 sy,  0.0 ni, 86.4 id,  0.0 wa,  0.0 hi,  1.0 si,  0.0 st
MiB Mem :  12730.5 total,   6333.8 free,   4883.2 used,   1513.5 buff/cache
MiB Swap:   4096.0 total,   4092.3 free,      3.7 used.   7538.2 avail Mem

  PID USER      PR  NI    VIRT    RES    SHR S  %CPU  %MEM     TIME+ COMMAND
22204 root      20   0 4098648   3.9g   1496 R  99.7  31.4   0:03.10 memtester
 1336 yjp       20   0 1029368  85400  37060 S   0.3   0.7   2:36.23 node
 1384 yjp       20   0   11.0g 173244  36836 S   0.3   1.3   3:21.44 node
    1 root      20   0     916    528    468 S   0.0   0.0   0:00.06 init
 1313 root      20   0     916    104     20 S   0.0   0.0   0:00.00 init
 1314 root      20   0     916    104     20 S   0.0   0.0   0:00.00 init
 1315 yjp       20   0    2612    596    528 S   0.0   0.0   0:00.01 sh
 1316 yjp       20   0    2612    532    464 S   0.0   0.0   0:00.00 sh
 1321 yjp       20   0    2612    596    528 S   0.0   0.0   0:00.00 sh
 1325 yjp       20   0  941660 111040  35332 S   0.0   0.9   1:44.62 node
 1365 root      20   0     916    104     20 S   0.0   0.0   0:00.00 init
 1366 root      20   0     916    104     20 S   0.0   0.0   0:08.04 init
 1367 yjp       20   0  598120  55708  29364 S   0.0   0.4   0:20.57 node
 1375 root      20   0     916    104     20 S   0.0   0.0   0:00.00 init
 1376 root      20   0     916    104     20 S   0.0   0.0   0:07.33 init
 1377 yjp       20   0  592924  52308  29384 S   0.0   0.4   0:18.03 node
 1390 yjp       20   0  846136  71748  31940 S   0.0   0.6   0:11.84 node
 1460 yjp       20   0 1807380 343924  15204 S   0.0   2.6   3:35.58 cpptools
 1493 yjp       20   0  587868  44104  31576 S   0.0   0.3   0:02.23 node
 2520 root      20   0     916    104     20 S   0.0   0.0   0:00.00 init
 2521 root      20   0     916    104     20 S   0.0   0.0   0:00.10 init
```

当我们执行`memtester 8000M`程序时，发现程序已经被杀死了，无法分配这么多内存，说明我们的内存资源控制也是有效的。



```
memtester version 4.3.0 (64-bit)
Copyright (C) 2001-2012 Charles Cazabon.
Licensed under the GNU General Public License version 2 (only).

pagesize is 4096
pagesizemask is 0xfffffffffffff000
want 4000MB (4194304000 bytes)
got  4000MB (4194304000 bytes), trying mlock ...locked.
Loop 1:
  Stuck Address      : ok
  Random Value       : ok
  Compare XOR        : ok
  Compare SUB        : ok
  Compare MUL        : ok
  Compare DIV        : ok
  Compare OR         : ok
  Compare AND        : ok
  Sequential Increment: ok
  Solid Bits         : setting  12^C
[root@DESKTOP-KUV5EDM /]# memtester 8000M
memtester version 4.3.0 (64-bit)
Copyright (C) 2001-2012 Charles Cazabon.
Licensed under the GNU General Public License version 2 (only).

pagesize is 4096
pagesizemask is 0xfffffffffffff000
want 8000MB (8388608000 bytes)
got  8000MB (8388608000 bytes), trying mlock ...Killed
[root@DESKTOP-KUV5EDM /]# 
```

```
top - 01:38:15 up 15:02,  0 users,  load average: 0.9
Tasks:  43 total,   1 running,  42 sleeping,   0 stopp
%Cpu(s):  0.3 us,  2.4 sy,  0.0 ni, 97.2 id,  0.0 wa,  0.0 hi,  0.1 si,  0.0 st
MiB Mem :  12730.5 total,  10211.2 free,   1005.9 used,   1513.4 buff/cache
MiB Swap:   4096.0 total,   4092.3 free,      3.7 used.  11415.5 avail Mem

  PID USER      PR  NI    VIRT    RES    SHR S  %CPU  %MEM     TIME+ COMMAND
 1336 yjp       20   0 1029624  85588  37060 S   1.3   0.7   2:36.00 node
 1367 yjp       20   0  598120  55708  29364 S   0.3   0.4   0:20.54 node
 1384 yjp       20   0   11.0g 173524  36836 S   0.3   1.3   3:21.33 node
 1460 yjp       20   0 1807380 343924  15204 S   0.3   2.6   3:35.56 cpptools
    1 root      20   0     916    528    468 S   0.0   0.0   0:00.06 init
 1313 root      20   0     916    104     20 S   0.0   0.0   0:00.00 init
 1314 root      20   0     916    104     20 S   0.0   0.0   0:00.00 init
 1315 yjp       20   0    2612    596    528 S   0.0   0.0   0:00.01 sh
 1316 yjp       20   0    2612    532    464 S   0.0   0.0   0:00.00 sh
 1321 yjp       20   0    2612    596    528 S   0.0   0.0   0:00.00 sh
 1325 yjp       20   0  941240 110696  35332 S   0.0   0.8   1:44.49 node
 1365 root      20   0     916    104     20 S   0.0   0.0   0:00.00 init
 1366 root      20   0     916    104     20 S   0.0   0.0   0:08.03 init
 1375 root      20   0     916    104     20 S   0.0   0.0   0:00.00 init
 1376 root      20   0     916    104     20 S   0.0   0.0   0:07.32 init
 1377 yjp       20   0  592924  52308  29384 S   0.0   0.4   0:18.02 node
 1390 yjp       20   0  846136  71748  31940 S   0.0   0.6   0:11.83 node
 1493 yjp       20   0  587868  44104  31576 S   0.0   0.3   0:02.23 node
 2520 root      20   0     916    104     20 S   0.0   0.0   0:00.00 init
 2521 root      20   0     916    104     20 S   0.0   0.0   0:00.10 init
 2522 yjp       20   0   10040   5180   3436 S   0.0   0.0   0:00.02 bash
 2536 yjp       20   0   10872   3724   3212 S   0.0   0.0   0:01.21 top
```