南昌大学实验报告

姓名：Qing Liu

学号：6130116184

邮箱地址：1119637652@qq.com

专业班级：Class 164 of Computer Science and Technology

实验日期：THU. April 25th, 2019

课程名称：Cloud Computing Technology Experiments

实验项目名称

Live Migration Massively

实验目的

- **Understanding the basic techniques for VM migration**
- **Migrate containers to test your migration skill**
- **Understanding the concept of checkpoint and restore**
- **Successfully migrate multiple images either one by one or in a batch**
- **Writing a decent report**

实验基础

CRIU、Checkpoint/Restore of Docker

实验步骤

Prepare code for test and build it on docker

`looplog.go`:

```
package main

import (
  "log"
  "time"
)
```

```
func main() {
  i := 1
  for {
    log.Print(i)
    i++
    time.Sleep(time.Second)
  }
}
```

`Dockerfile` :

```
FROM golang:latest

WORKDIR /go/src/looplog
COPY . /go/src/looplog

RUN go build .

ENTRYPOINT ["./looplog"]
```

Build process:

```
$ cd ~/Code/go/app2
$ docker build -t cleo0625/looplog .
```

```
cleo@vm-ubuntu:~/Code/go/app2$ docker build -t cleo0625/looplog .
Sending build context to Docker daemon  3.072kB
Step 1/5 : FROM golang:latest
 ---> b860ab44e93e
Step 2/5 : WORKDIR /go/src/looplog
 ---> Running in 7da14bd0ceae
Removing intermediate container 7da14bd0ceae
 ---> f02dd8e1e3e6
Step 3/5 : COPY . /go/src/looplog
 ---> 45ced1794621
Step 4/5 : RUN go build .
 ---> Running in f368afa3561d
Removing intermediate container f368afa3561d
 ---> 2551296a822a
Step 5/5 : ENTRYPOINT ["./looplog"]
 ---> Running in f9e5cd31da25
Removing intermediate container f9e5cd31da25
 ---> 5f9029cb6464
Successfully built 5f9029cb6464
Successfully tagged cleo0625/looplog:latest
```

Build NFS Between two host

- Install `rpc` and `nfs` in two nodes.

```
$ sudo apt-get install rpcbind
$ sudo apt-get install nfs-kernel-server
```

- Start the rpc service.

```
$ systemctl start rpcbind.service
```

- Check the ip address for two host.

  By executing the command `ifconfig`, I get the ip address of two virtual machines. One is `192.168.124.143/24` which was called host1, and another is `192.168.124.134' called host2. I build the docker image in host1, and I will migrate the containers in host1 to host2. So host1 is the nfs server and host2 is the nfs client.

- Configure for host1 and start the nfs server.

```
$ mkdir -p /tmp/checkpoints
$ vim /etc/exports
$ sudo exportfs -a
$ systemctl start nfs-server.service
```

`/etc/exports`:

```
# /etc/exports: the access control list for filesystems which may be exporte
#               to NFS clients.  See exports(5).
#
# Example for NFSv2 and NFSv3:
# /srv/homes       hostname1(rw,sync,no_subtree_check) hostname2(ro,sync,no_
#
# Example for NFSv4:
# /srv/nfs4        gss/krb5i(rw,sync,fsid=0,crossmnt,no_subtree_check)
# /srv/nfs4/homes  gss/krb5i(rw,sync,no_subtree_check)
#

/tmp/checkpoints   192.168.124.134/24(rw,no_subtree_check,no_root_squash)
```

- Check the localhost

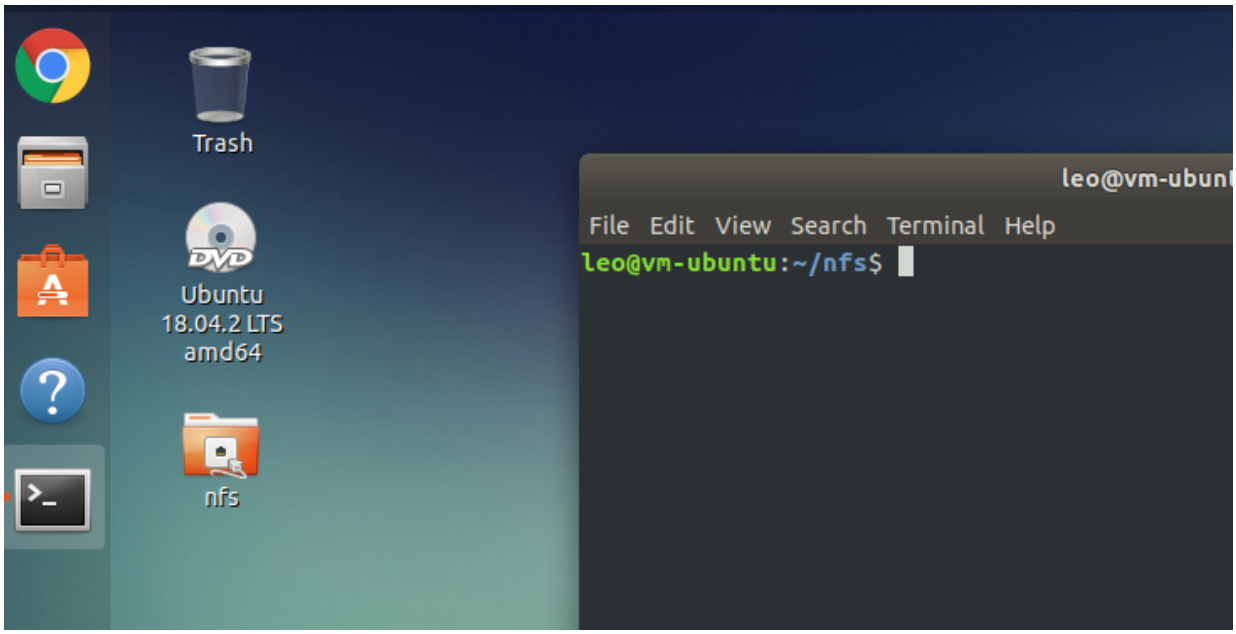
```
$ showmount -e localhost
```

```
cleo@vm-ubuntu:~$ showmount -e localhost
Export list for localhost:
/tmp/checkpoints 192.168.124.134/24
```

- Mount the target directory in client

```
$ systemctl start rpcbind
$ systemctl start nfs-client.target
$ mkdir -p ~/nfs
$ sudo mount 192.168.124.143:/tmp/checkpoints ~/nfs
```

We can enter the nfs directory in client host, and there is an new icon for it in desktop.



Write the scripts

- Write the script in host1(nfs server) to make a backup for the image, run some containers and make checkpoints for them into `/tmp/checkpoints`.

`checkpoint.sh`:

```
#!/bin/bash

date
docker save -o /tmp/checkpoints/looplog.tar cleo0625/looplog

for((i=1;i<=10;i++)); do
        echo "============================================================"
        echo "container$i:"
        docker run -d --name=qingliu$i cleo0625/looplog
        echo "run the container qingliu$i successfully."
        sleep 2
        docker container ls -all
```

```
        docker logs qingliu$i
        docker checkpoint create --checkpoint-dir=/tmp/checkpoints qingliu$i
        echo "checkpoint operation successfully."
done
date
```

- Write the script in host2(nfs client) to load the image and restore containers.

`restore.sh`:

```
#!/bin/bash

cd ~/nfs
date
docker load -i looplog.tar

for((i=1;i<=10;i++)); do
        echo "============================================================"
        echo "restore container qingliu$i"
        docker create --name qingliu-clone$i cleo0625/looplog
        sudo mv checkpoint$i /var/lib/docker/containers/$(docker ps \
         -aq --no-trunc --filter name=qingliu-clone$i)/checkpoints/
        docker start --checkpoint=checkpoint$i qingliu-clone$i
        sleep 1
        docker logs qingliu-clone$i
done
date
```

实验数据或结果

- Execute the `checkpoint.sh`, and here are the screenshots for its output:

```
cleo@vm-ubuntu:~$ ./checkpoint.sh
2019年 04月 25日 星期四 16:28:53 CST
=====================================================
container1:
b917905cc8065eeca348ed77f26d7da2a14b6f71e76d34836eeec78bd152d8bf
run the container qingliu1 successfully.
CONTAINER ID        IMAGE               COMMAND             CREATED             STATUS              PORTS               NAMES
b917905cc806        cleo0625/looplog    "./looplog"         3 seconds ago       Up 2 seconds                            qingliu1
2019/04/25 08:29:13 1
2019/04/25 08:29:14 2
2019/04/25 08:29:15 3
checkpoint1
checkpoint operation successfully.
=====================================================
container2:
d757bf6877e131d8fff51abd0e440fca7a7645dd1fc3732f622be05e45f54e0f
run the container qingliu2 successfully.
CONTAINER ID        IMAGE               COMMAND             CREATED             STATUS              PORTS               NAMES
d757bf6877e1        cleo0625/looplog    "./looplog"         3 seconds ago       Up 2 seconds                            qingliu2
2019/04/25 08:29:17 1
2019/04/25 08:29:18 2
2019/04/25 08:29:19 3
checkpoint2
checkpoint operation successfully.
=====================================================
container3:
741d9ec303dae9ac95e3e8dfc1e4728cdfcb5ecd061f42d82eaf2c0a5d71494c
run the container qingliu3 successfully.
CONTAINER ID        IMAGE               COMMAND             CREATED             STATUS              PORTS               NAMES
741d9ec303da        cleo0625/looplog    "./looplog"         3 seconds ago       Up 2 seconds                            qingliu3
2019/04/25 08:29:21 1
2019/04/25 08:29:22 2
2019/04/25 08:29:23 3
checkpoint3
checkpoint operation successfully.
=====================================================
container4:
e03f1c135dcf65d9cc1a2647c50af98b375b1106d49627de3293f79ab6aebb79
```

```
run the container qingliu4 successfully.
CONTAINER ID        IMAGE               COMMAND             CREATED             STATUS              PORTS               NAMES
e03f1c135dcf        cleo0625/looplog    "./looplog"         2 seconds ago       Up 2 seconds                            qingliu4
2019/04/25 08:29:24 1
2019/04/25 08:29:25 2
2019/04/25 08:29:26 3
checkpoint4
checkpoint operation successfully.
=====================================================
container5:
a5f9d5dac8e5763d744ccb54d88f14541e0b89c08ca188fc6d756c3b3c9c9177
run the container qingliu5 successfully.
CONTAINER ID        IMAGE               COMMAND             CREATED             STATUS              PORTS               NAMES
a5f9d5dac8e5        cleo0625/looplog    "./looplog"         3 seconds ago       Up 2 seconds                            qingliu5
2019/04/25 08:29:28 1
2019/04/25 08:29:29 2
2019/04/25 08:29:30 3
checkpoint5
checkpoint operation successfully.
=====================================================
container6:
194587650f201b1ecc76a5212c51b83137f8902ac570f25541c9647dd70949db
run the container qingliu6 successfully.
CONTAINER ID        IMAGE               COMMAND             CREATED             STATUS              PORTS               NAMES
194587650f20        cleo0625/looplog    "./looplog"         3 seconds ago       Up 2 seconds                            qingliu6
2019/04/25 08:29:31 1
2019/04/25 08:29:32 2
2019/04/25 08:29:33 3
checkpoint6
checkpoint operation successfully.
=====================================================
container7:
529931fd915636dbd8123674639ddbd54647a896cd8d71b0e5532009dd501dd1
run the container qingliu7 successfully.
CONTAINER ID        IMAGE               COMMAND             CREATED             STATUS              PORTS               NAMES
529931fd9156        cleo0625/looplog    "./looplog"         3 seconds ago       Up 2 seconds                            qingliu7
2019/04/25 08:29:35 1
2019/04/25 08:29:36 2
```

```
2019/04/25 08:29:33 3
checkpoint6
checkpoint operation successfully.
=====================================================
container7:
529931fd915636dbd8123674639ddbd54647a896cd8d71b0e5532009dd501dd1
run the container qingliu7 successfully.
CONTAINER ID        IMAGE               COMMAND             CREATED             STATUS              PORTS               NAMES
529931fd9156        cleo0625/looplog    "./looplog"         3 seconds ago       Up 2 seconds                            qingliu7
2019/04/25 08:29:35 1
2019/04/25 08:29:36 2
2019/04/25 08:29:37 3
checkpoint7
checkpoint operation successfully.
=====================================================
container8:
21e7218ac978278f3493a71e3e7060aaa2bc2a5ef9cdf3b4c3011319d8104967
run the container qingliu8 successfully.
CONTAINER ID        IMAGE               COMMAND             CREATED             STATUS              PORTS               NAMES
21e7218ac978        cleo0625/looplog    "./looplog"         3 seconds ago       Up 2 seconds                            qingliu8
2019/04/25 08:29:38 1
2019/04/25 08:29:39 2
2019/04/25 08:29:40 3
checkpoint8
checkpoint operation successfully.
=====================================================
container9:
36f7e8b0275de7f82f0916fd91e5c6dde734356065549ae1758d23b892695c36
run the container qingliu9 successfully.
CONTAINER ID        IMAGE               COMMAND             CREATED             STATUS              PORTS               NAMES
36f7e8b0275d        cleo0625/looplog    "./looplog"         3 seconds ago       Up 2 seconds                            qingliu9
2019/04/25 08:29:42 1
2019/04/25 08:29:43 2
2019/04/25 08:29:44 3
checkpoint9
checkpoint operation successfully.
=====================================================
container10:
```

```
3d94276499923f236584d7bbd313a86fd1d3742bf23ccff943cf88bab3a1cb82
run the container qingliu10 successfully.
CONTAINER ID        IMAGE               COMMAND             CREATED             STATUS              PORTS               NAMES
3d9427649992        cleo0625/looplog    "./looplog"         2 seconds ago       Up 2 seconds                            qingliu10
2019/04/25 08:29:45 1
2019/04/25 08:29:46 2
2019/04/25 08:29:47 3
checkpoint10
checkpoint operation successfully.
2019年 04月 25日 星期四 16:29:48 CST
```

From the output, we can see that it taked 55s to checkpoint 10 containers, of course, it contains the time for save the built image.

And we can show the list of file in `/tmp/checkpoints` :

```
cleo@vm-ubuntu:~$ ls /tmp/checkpoints/
checkpoint1   checkpoint2  checkpoint4  checkpoint6  checkpoint8  looplog.tar
checkpoint10  checkpoint3  checkpoint5  checkpoint7  checkpoint9
```

We can check the nfs directory in client:

```
leo@vm-ubuntu:~$ ls nfs/
checkpoint1   checkpoint2  checkpoint4  checkpoint6  checkpoint8  looplog.tar
checkpoint10  checkpoint3  checkpoint5  checkpoint7  checkpoint9
leo@vm-ubuntu:~$
```

- Execute the `restore.sh`, and here is the screenshot for its output:

```
leo@vm-ubuntu:~$ ./restore.sh
2019年 04月 25日 星期四 16:30:00 CST

049fb13e4687: Loading layer [==================================================>]  3.072kB/3.072kB
367f2f60847d: Loading layer [==================================================>]  4.608kB/4.608kB
8791669407d6: Loading layer [==================================================>]  2.253MB/2.253MB
Loaded image: cleo0625/looplog:latest
============================================================
restore container qingliu1
52c730e38de81edddef15c52f3852464bbb0c9df49c62c37d631777ff8057e9b
[sudo] password for leo:
Sorry, try again.
[sudo] password for leo:
Warn  (criu/cr-service.c:290): parse_options returns 0
2019/04/25 08:30:20 4
2019/04/25 08:30:21 5
============================================================
restore container qingliu2
52620dad7d517bfbae2986bc1f1eea1050e0980182452c7db788f213f6bca71d
Warn  (criu/cr-service.c:290): parse_options returns 0
2019/04/25 08:30:22 4
2019/04/25 08:30:23 5
============================================================
restore container qingliu3
6b7a4fa215fa1021d1ee53cbd30da982b8cdc3b4729daf6054c5bcd32b43b38d
Warn  (criu/cr-service.c:290): parse_options returns 0
2019/04/25 08:30:24 4
2019/04/25 08:30:25 5
============================================================
restore container qingliu4
42c8d8e9e5521c9df7aa9b161a574da49a034dfefe44fbeebb2031e9f1edbfab
Warn  (criu/cr-service.c:290): parse_options returns 0
2019/04/25 08:30:26 4
2019/04/25 08:30:27 5
============================================================
restore container qingliu5
18db9b080736e0bd42db0ac1a093a4a1a19c54a24ee065bd8946c285cc09a95e
Warn  (criu/cr-service.c:290): parse_options returns 0
```

```
2019/04/25 08:30:29 5
============================================================
restore container qingliu6
dbd1301d94ff53d9f2083438795b9f994cd5b9a80a419c3966569a4ebbbe6228
Warn  (criu/cr-service.c:290): parse_options returns 0
2019/04/25 08:30:30 4
2019/04/25 08:30:31 5
============================================================
restore container qingliu7
42ff1c638ec150fbdb9b779873fffaf4e7d37cbbf93feae393893fa6c4fb8a9f
Warn  (criu/cr-service.c:290): parse_options returns 0
2019/04/25 08:30:32 4
2019/04/25 08:30:33 5
============================================================
restore container qingliu8
47a29d9eaae2bcad4fe05f706806f37dddf2dd618399777d1cb14d2069de48f2
Warn  (criu/cr-service.c:290): parse_options returns 0
2019/04/25 08:30:34 4
2019/04/25 08:30:35 5
============================================================
restore container qingliu9
9e7751ad4746269a39a5b36ab1b95af611215dc9c8e9801254fe0867ed5a80c8
Warn  (criu/cr-service.c:290): parse_options returns 0
2019/04/25 08:30:36 4
2019/04/25 08:30:37 5
============================================================
restore container qingliu10
b6ab8367d129b79a880cc8716710819b8d716086251138c7a2112c1af998893d
Warn  (criu/cr-service.c:290): parse_options returns 0
2019/04/25 08:30:39 4
2019/04/25 08:30:40 5
2019年 04月 25日 星期四 16:30:40 CST
```

From the output, we can see that it taked 10s to checkpoint 10 containers. In this script.

I choosed a restored container randomly qingliu-clone1 to logs:

It is still running and logging. So, I successfully migrate 10 containers.

- Then, test 100 containers.

I canceled the saving and loading image and directly test the checkpoint and restore for 100 containers. Before it began, I have already removed all the containers existed. Of course, I make a change for both of two scripts.

`checkpoint.sh`:

```
#!/bin/bash

date

for((i=1;i<=100;i++)); do
        echo "============================================================"
        echo "container$i:"
        docker run -d --name=qingliu$i cleo0625/looplog
        echo "run the container qingliu$i successfully."
        sleep 2
        docker container ls -all
        docker logs qingliu$i
        docker checkpoint create --checkpoint-dir=/tmp/checkpoints qingliu$i
        echo "checkpoint operation successfully."
done

date
```

`restore.sh`:

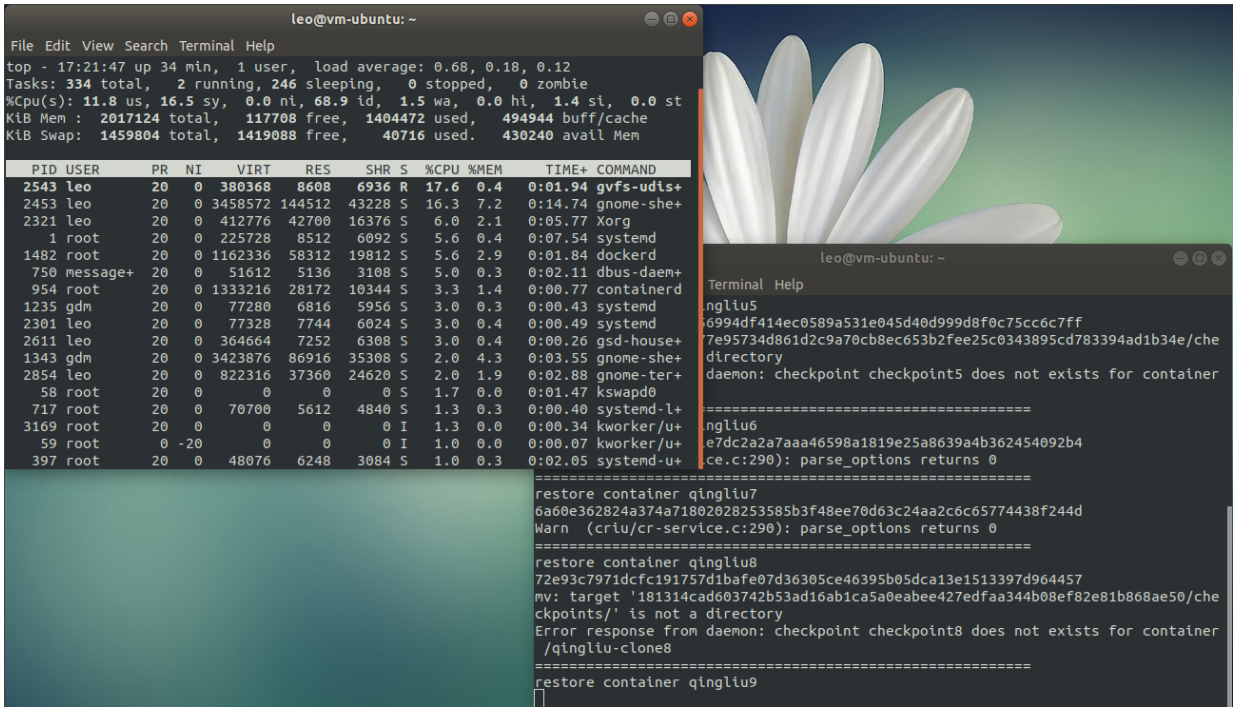
```
#!/bin/bash

cd ~/nfs
date

for((i=1;i<=100;i++)); do
        echo "============================================================"
        echo "restore container qingliu$i"
        docker create --name qingliu-clone$i cleo0625/looplog
        sudo mv checkpoint$i /var/lib/docker/containers/$(docker \
         ps -aq --no-trunc --filter name=qingliu-clone$i)/checkpoints/
        docker start --checkpoint=checkpoint$i qingliu-clone$i
        sleep 1
        docker logs qingliu-clone$i
done
date
```

Make checkpoints for 100 containers cost 5m51s including 100 times for sleeping 2 seconds.

Restoring for 100 checkpoints cost 3m59s including 100 times for sleeping 1 seconds.

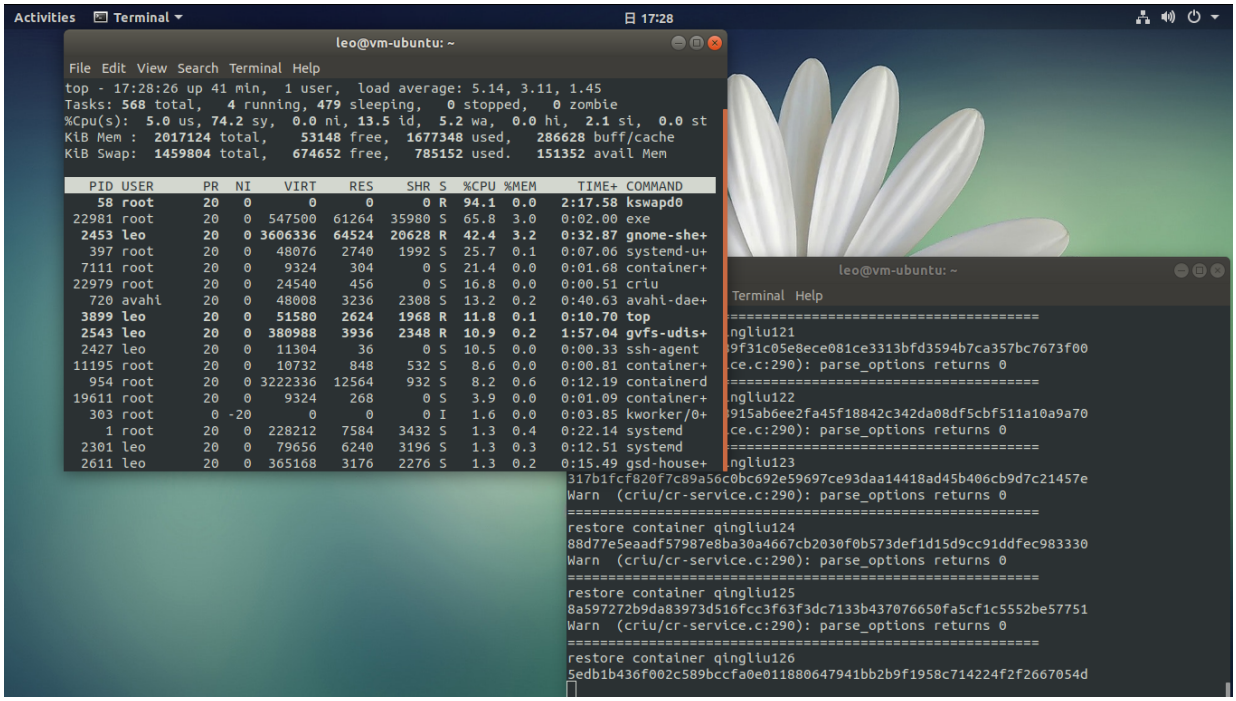
- Test 300 containers

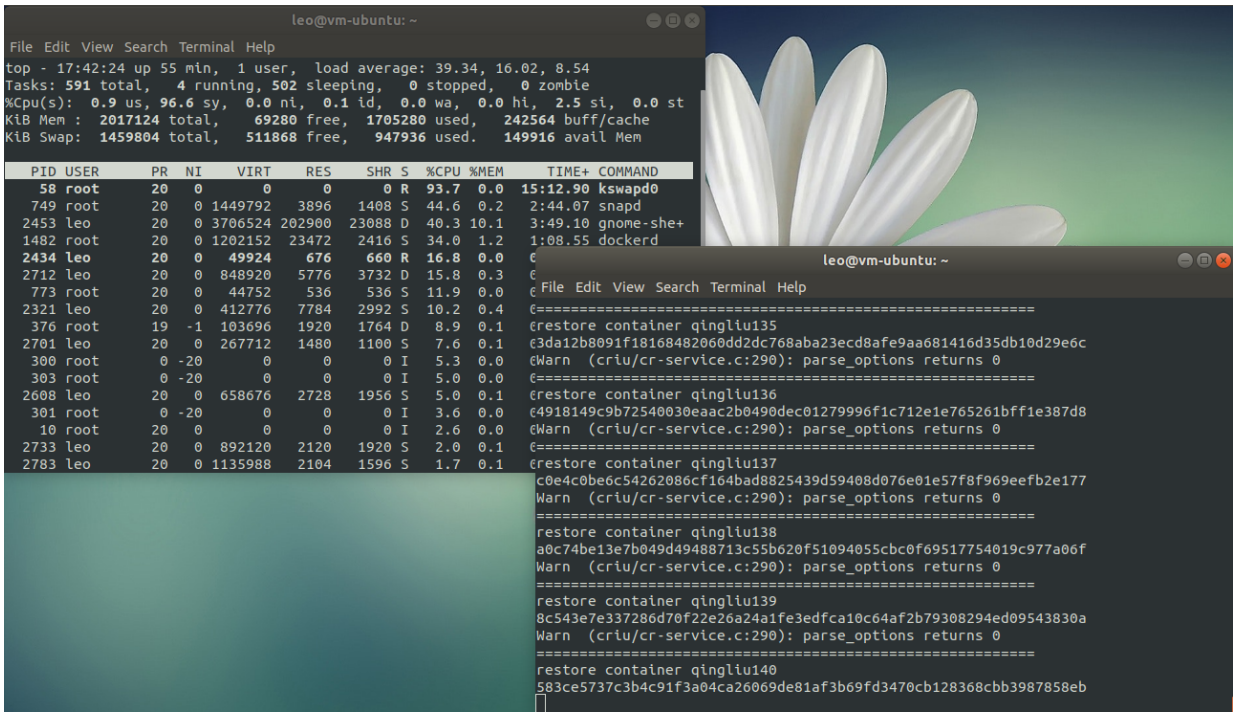
I change the variable `i` to 400. At the begining of the restoring, I used `top` to detected the status of the vm.



We can see that when the 9th continer was restored, the swap space was used was only 40716KB. The total memory the vm has is 2GB.

When it was restoring the 127th container, it got very slowly. And the system became very cartridge. When the number became 139, the system can not run anymore, and I can not even operate it. Here are 2 screen shots at the 126th and 140th.





实验思考

Instead of using concurrent mode to restore images on another machine, I restore them one by one in a loop. With regard to using NFS to pass breakpoints, the directory I share is `/tmp/checkpoints/`. First, breakpoints are operated on in Host 1, then 400 breakpoints are saved to the shared directory, and then recovered one after another from Host 2.

And maybe there is another reason for halt. If each container needs a lot of space, or there are many containers, it will lead to insufficient disk space. Or the disk speed is very slow, which will lead to slow recovery.

As shown in the figure above, 2 GB of memory plus about 1.5 GB of swap space, and 4 virtual cpus, by 140, the system was basically stuck, even the mouse click was useless. In my opinion, the bottleneck of image migration is related to the speed of image transmission and the size of machine physical memory. As the image is restored on host 2, swap space has been consumed, which indicates that it is related to memory and swap space.

参考资料

- http://cn.linux.vbird.org/linux_server/0330nfs.php
- https://criu.org/Docker
- https://docs.docker.com/docker-hub/
- https://github.com/checkpoint-restore/criu/issues/450
- https://github.com/moby/moby/issues
- https://github.com/checkpoint-restore/criu

- https://github.com/ZhuangweiKang/Docker-CRIU-Live-Migration?tdsourcetag=s_pctim_aiomsg