

รายงานสรุป

Project 2 :: Overlay Disk Renew Mechanism

จัดทำโดย

นางรุจรรดา เย็นเยือก รหัส 5109035427

นายพิทักษ์ แทนแก้ว รหัส 5209035194

นายวสันต์ กุลติลภ รหัส 5209035053

นำเสนอ

อาจารย์ ดร.กษิติศ ชาญเขียว

รายงานนี้เป็นส่วนหนึ่งของวิชา

CS797 Introduction to Computer Virtualizations เทอม 1/2553

ปริญญาโท ภาควิชาวิทยาการคอมพิวเตอร์

คณะวิทยาศาสตร์ มหาวิทยาลัยธรรมศาสตร์

Table of Contents

แสดงการทำงานด้วย Flow Chart ดังนี้	4
ทำการแก้ไข Source-Code	5
แสดงลำดับการทำงานดังนี้	9
Conclusion and Future Work.....	14
แบบที่ 1	14
แบบที่ 2	15

Project 2 :: Overlay Disk Renew Mechanism

1. To automatically use overlay disk image in KVM, users can invoke KVM

with the “-snapshot” option. In this project, you will add new capabilities to KVM disk emulation mechanisms to manage the above overlay disk.

In the current implementation of KVM, the overlay disk remain the same after users issue the “commit” command on qemu monitor to merge its contents to the base disk image. Therefore, as VM computation proceeds, the size of the overlay would grow bigger.

This project intends to modify the mechanism of KVM commit operation by

renewing the overlay disk after the merging of contents in the original commit

operation completes. You are required to create a new function namely the

bdrv_renew() to do the task. You are free to define any parameter for it.

This function would be called after the original commit operation is conducted.

To renew a disk, the **bdrv_renew()** function would create an empty overlay disk

on top of the base image that you have just committed contents to, and replace the

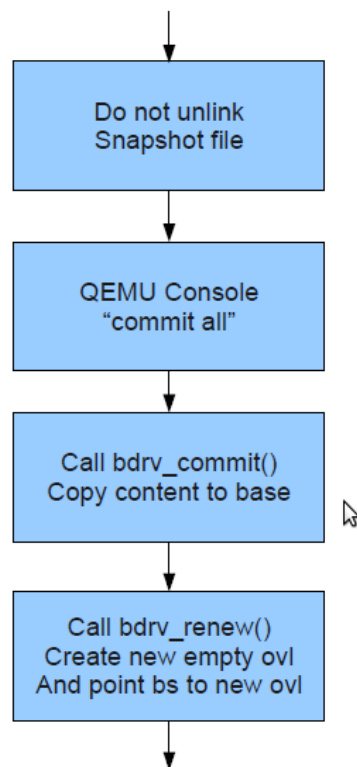
current overlay disk with the new one.

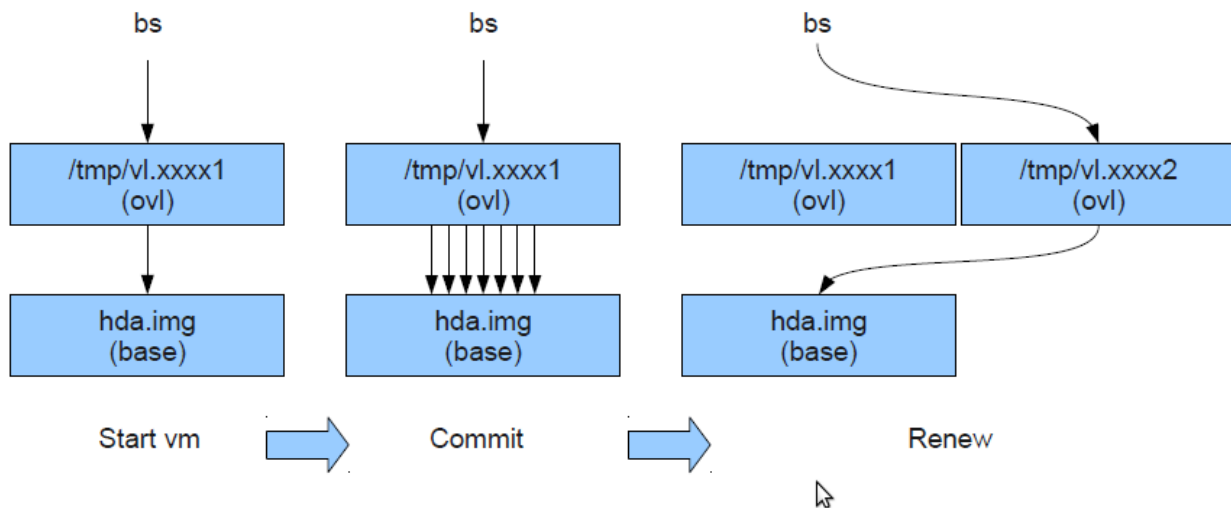
(The previous overlay disk must be deleted from host computers' file system.)

จากโจทย์ที่ได้รับ ได้ทำการแก้ไข Source-Code ในไฟล์

- 1.block.h
- 2.block.c

แสดงการทำงานด้วย **Flow Chart** ดังนี้





ทำการแก้ไข Source-Code

1. ทำการแก้ไข code ใน block.c เพื่อให้สามารถมองเห็นไฟล์ ที่ถูกสร้างขึ้นมา ภายใต path file /tmp/.... จำนวน 3 จุดดั่งนี้ ที่บรรทัด 480 , 488 , 533 ตามภาพที่จะแสดงดั่งนี้

Line no : 480

```

478 unlink_and_fail:
479     if (bs->is_temporary)
480         //unlink(filename);
481     return ret;

```

Line no : 488

```

486 #ifndef _WIN32
487     if (bs->is_temporary) {
488         //unlink(filename);
489     }

```

Line no : 533

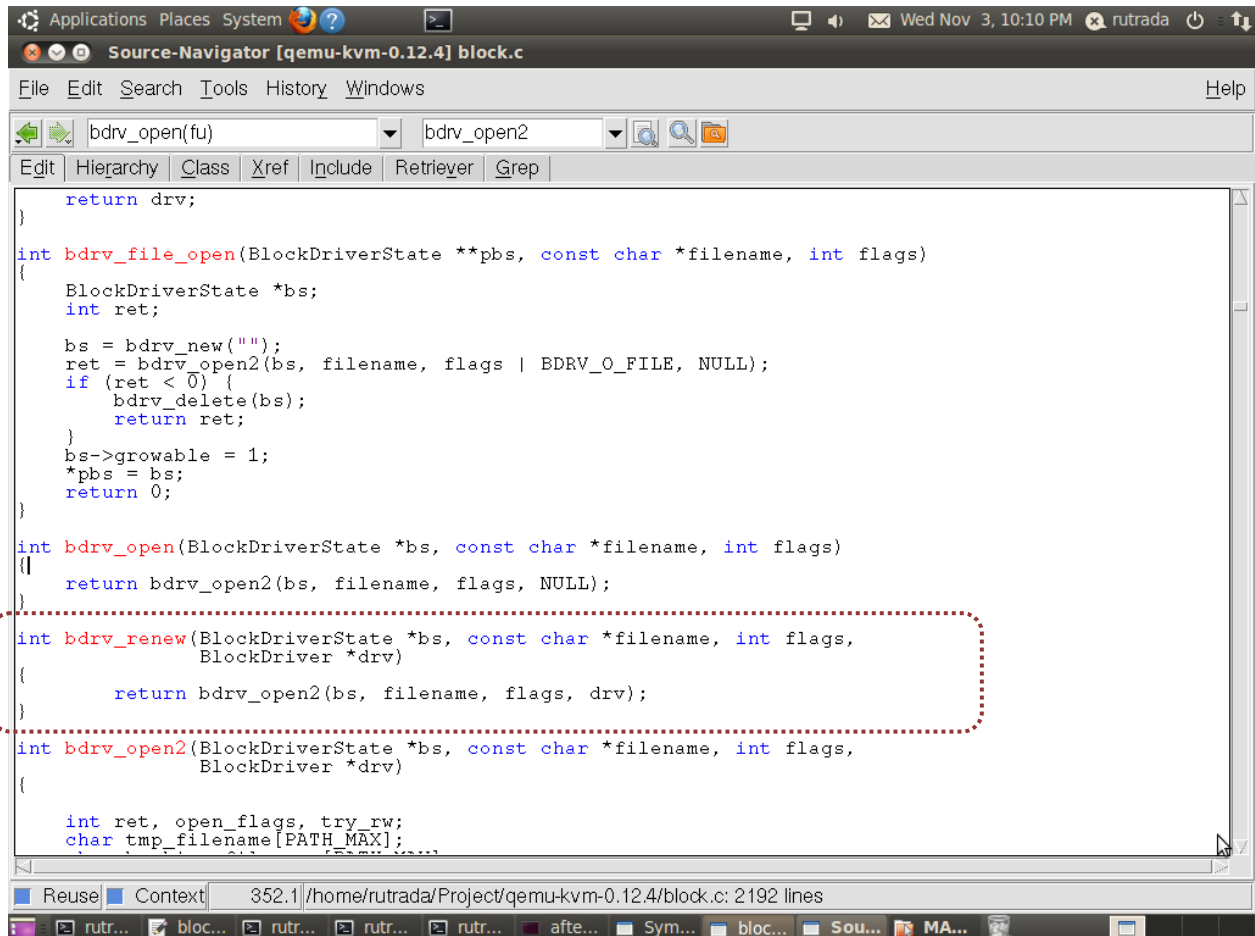
```

531 #ifdef _WIN32
532     if (bs->is_temporary) {
533         unlink(bs->filename);
534     }
535 #endif

```

2. ในฟังก์ชัน bdrv_commit()

ทำการสร้าง ฟังก์ชันใหม่ชื่อ bdrv_renew() แสดง source-code ตามรูปนี้



```
return drv;
}

int bdrv_file_open(BlockDriverState **pbs, const char *filename, int flags)
{
    BlockDriverState *bs;
    int ret;

    bs = bdrv_new("");
    ret = bdrv_open2(bs, filename, flags | BDRV_O_FILE, NULL);
    if (ret < 0) {
        bdrv_delete(bs);
        return ret;
    }
    bs->growable = 1;
    *pbs = bs;
    return 0;
}

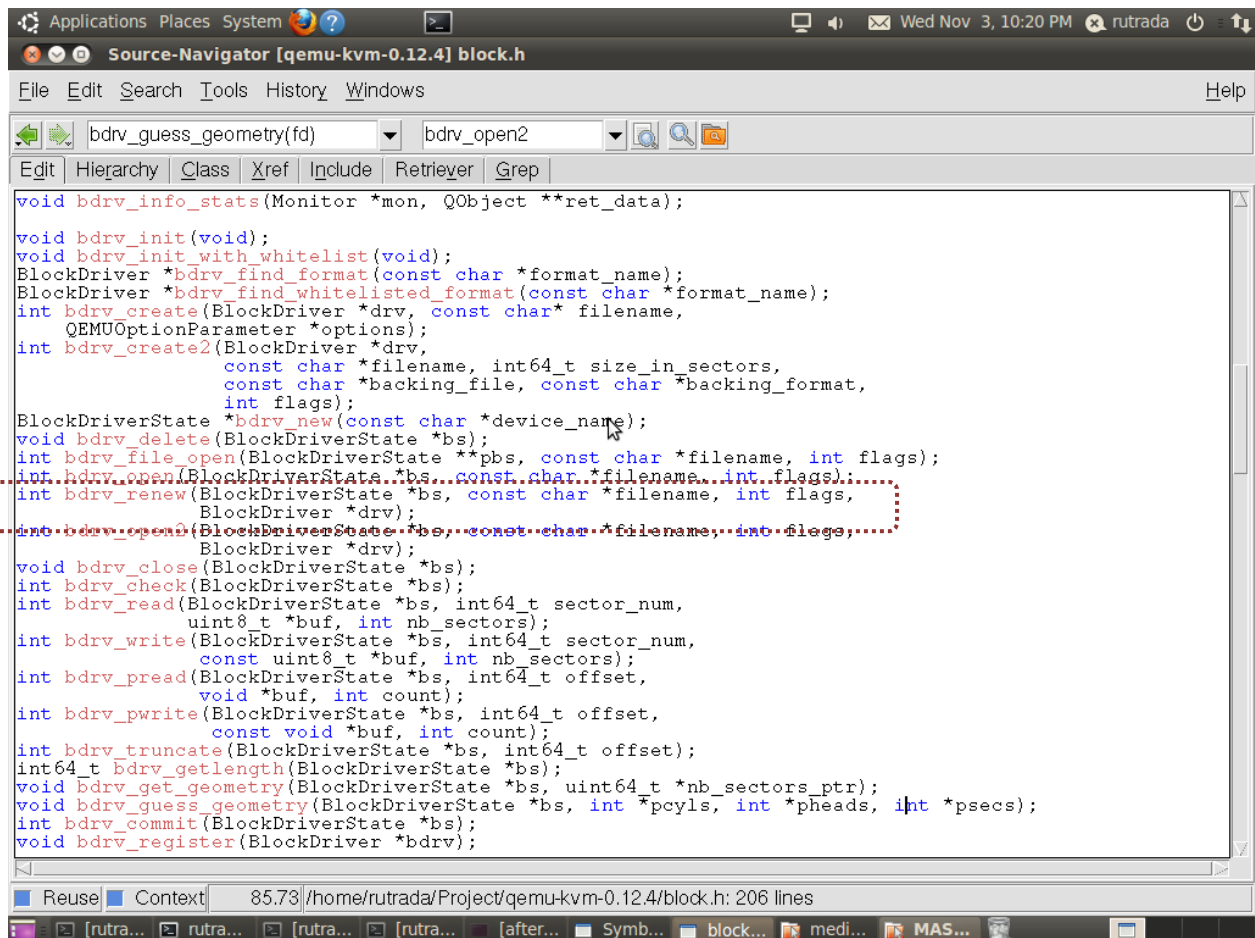
int bdrv_open(BlockDriverState *bs, const char *filename, int flags)
{
    return bdrv_open2(bs, filename, flags, NULL);
}

int bdrv_renew(BlockDriverState *bs, const char *filename, int flags,
              BlockDriver *drv)
{
    return bdrv_open2(bs, filename, flags, drv);
}

int bdrv_open2(BlockDriverState *bs, const char *filename, int flags,
              BlockDriver *drv)
{
    int ret, open_flags, try_rw;
    char tmp_filename[PATH_MAX];
```

เพื่อให้ทำงานภายหลังจากการ Commit ให้ทำการเปิด Overlay Image ใหม่ได้

3. ใน block.h ทำการเขียนประกาศโครงสร้างของฟังก์ชันใหม่ ตามรูปดังนี้



```
void bdrv_guess_geometry(fd);
void bdrv_init(void);
void bdrv_init_with_whitelist(void);
BlockDriver *bdrv_find_format(const char *format_name);
BlockDriver *bdrv_find_whitelisted_format(const char *format_name);
int bdrv_create(BlockDriver *drv, const char *filename,
               QEMUOptionParameter *options);
int bdrv_create2(BlockDriver *drv,
                const char *filename, int64_t size_in_sectors,
                const char *backing_file, const char *backing_format,
                int flags);
BlockDriverState *bdrv_new(const char *device_name);
void bdrv_delete(BlockDriverState *bs);
int bdrv_file_open(BlockDriverState **pbs, const char *filename, int flags);
int bdrv_open(BlockDriverState *bs, const char *filename, int flags);
int bdrv_renew(BlockDriverState *bs, const char *filename, int flags,
              BlockDriver *drv);
int bdrv_open2(BlockDriverState *bs, const char *filename, int flags,
              BlockDriver *drv);
void bdrv_close(BlockDriverState *bs);
int bdrv_check(BlockDriverState *bs);
int bdrv_read(BlockDriverState *bs, int64_t sector_num,
              uint8_t *buf, int nb_sectors);
int bdrv_write(BlockDriverState *bs, int64_t sector_num,
               const uint8_t *buf, int nb_sectors);
int bdrv_pread(BlockDriverState *bs, int64_t offset,
               void *buf, int count);
int bdrv_pwrite(BlockDriverState *bs, int64_t offset,
                const void *buf, int count);
int bdrv_truncate(BlockDriverState *bs, int64_t offset);
int64_t bdrv_getlength(BlockDriverState *bs);
void bdrv_get_geometry(BlockDriverState *bs, uint64_t *nb_sectors_ptr);
void bdrv_guess_geometry(BlockDriverState *bs, int *pcyls, int *pheads, int *psecs);
int bdrv_commit(BlockDriverState *bs);
void bdrv_register(BlockDriver *bdrv);
```

4. ใน bdrv_commit() ทำการเพิ่มฟังก์ชันการทำงานของ bdrv_renew()

```
574 /* commit COW file into the raw image */
575 int bdrv_commit(BlockDriverState *bs)
576 {
577     printf("commit bs = %s \n",bs);
578     BlockDriver *drv = bs->drv;
579     int64_t i, total_sectors;
580     int n, j;
581     unsigned char sector[512];
582
583     if (!drv)
584         return -ENOMEDIUM;
585
586     if (bs->read_only) {
587         return -EACCES;
588     }
589
590     if (!bs->backing_hd) {
591         return -ENOTSUP;
592     }
593
594     total_sectors = bdrv_getlength(bs) >> BDRV_SECTOR_BITS;
595     for (i = 0; i < total_sectors; i) {
596         if (drv->bdrv_is_allocated(bs, i, 65536, &n)) {
597             for(j = 0; j < n; j++) {
598                 if (bdrv_read(bs, i, sector, 1) != 0) {
599                     return -EIO;
600                 }
601
602                 if (bdrv_write(bs->backing_hd, i, sector, 1) != 0) {
603                     return -EIO;
604                 }
605                 i++;
606             }
607         } else {
608             i += n;
609         }
610     }
611
612     bdrv_renew(bs, "dsk-base.img", 72, drv); /*Add New Function*/
613
614     if (drv->bdrv_make_empty)
615         return drv->bdrv_make_empty(bs);
616     return 0;
617 }
```


สร้าง ovl ขึ้นมาใหม่ โดยอาศัย bdrv_renew() (ไปเรียกใช้ bdrv_open2()) ช่วยสร้าง snapshot ด้วยการกำหนดให้ flags มีค่า 72 ซึ่งได้จากการดำเนินการระดับ bit แบบ or ของค่าคงที่ BDRV_O_SNAPSHOT (0x0008) และ BDRV_O_CACHE_WB (0x0040) คำนวณแล้วได้ 0x0048 นั่นคือ 72 ฐาน 10 นั่นเอง

ทำการ Compile โปรแกรม ด้วยคำสั่ง

```
./configure
```

```
make
```

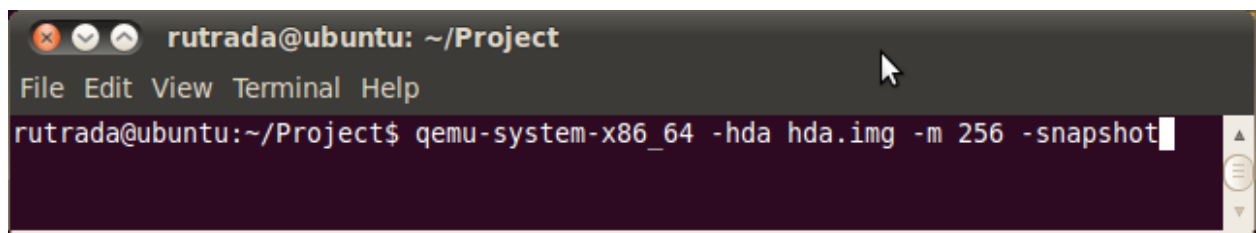
```
sudo make install
```

แสดงลำดับการทำงานดังนี้

1. Start Overlay Image

ด้วยคำสั่ง

```
Qemu-system-x86_64 -hda had.img -m 256 -boot c -snapshot
```



```
rutrada@ubuntu: ~/Project
File Edit View Terminal Help
rutrada@ubuntu:~/Project$ qemu-system-x86_64 -hda hda.img -m 256 -snapshot
```

2. จากคำสั่งนี้จะได้ไฟล์ vl.* มา ภายใต้ path file ./tmp/....

ตรวจสอบด้วยคำสั่ง

```
$ls -alh /tmp/vl.* (มี 1 ไฟล์)
```

```
rutrada@ubuntu: ~/Project
File Edit View Terminal Help
rutrada@ubuntu:~/Project$ ls -alh /tmp/vl.*
-rw----- 1 rutrada rutrada 4.2M 2010-11-03 22:42 /tmp/vl.374NcT
rutrada@ubuntu:~/Project$ ls -alh /tmp/vl.*
-rw----- 1 rutrada rutrada 9.4M 2010-11-03 22:49 /tmp/vl.374NcT
rutrada@ubuntu:~/Project$
```

3. บน Guest OS ทดลองสร้างไฟล์ ขนาด 5 MB

ด้วยคำสั่ง

```
$dd if=/dev/zero of=test.img bs=1024 count=5
```

4. ทำการตรวจสอบขนาดไฟล์

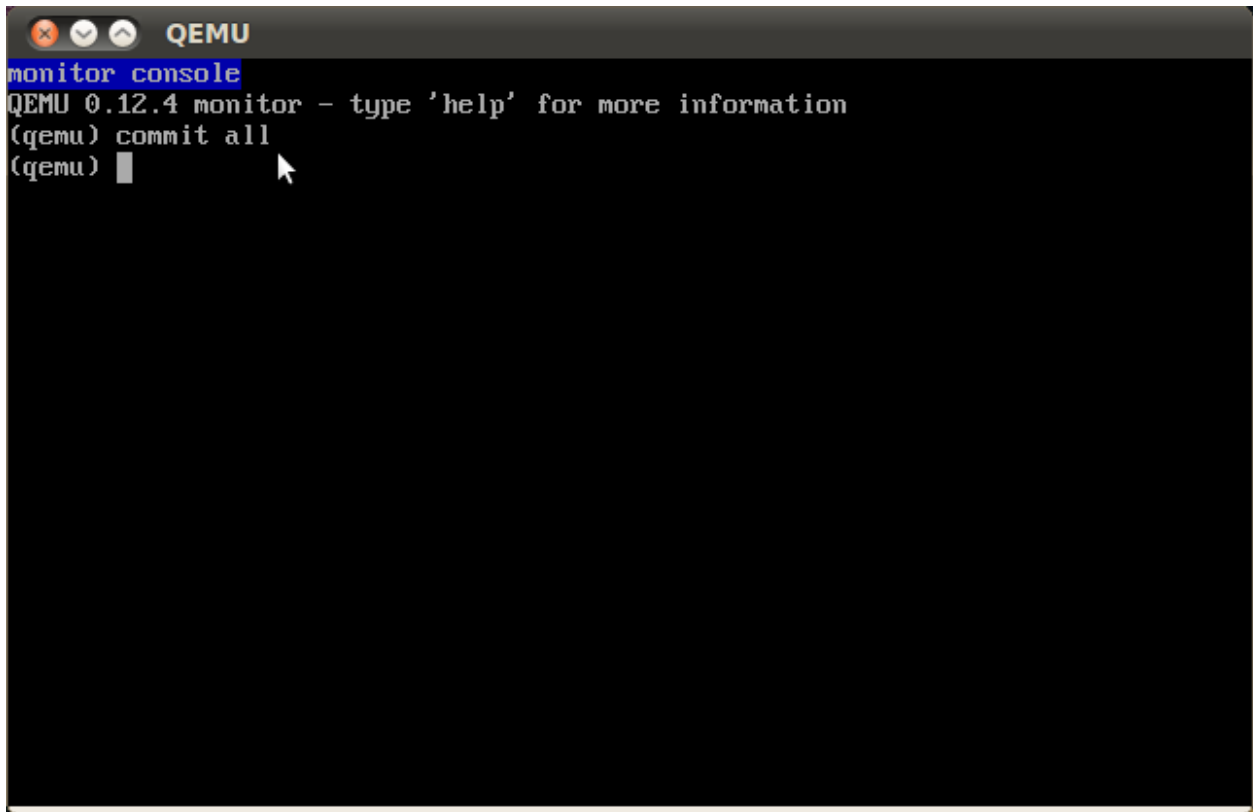
ด้วยคำสั่ง `$ls -alh /tmp/vl.*` พบว่าไฟล์มีขนาดใหญ่ขึ้น

```
rutrada@ubuntu: ~/Project
File Edit View Terminal Help
rutrada@ubuntu:~/Project$ ls -alh /tmp/vl.*
-rw----- 1 rutrada rutrada 4.2M 2010-11-03 22:42 /tmp/vl.374NcT
rutrada@ubuntu:~/Project$ ls -alh /tmp/vl.*
-rw----- 1 rutrada rutrada 9.4M 2010-11-03 22:49 /tmp/vl.374NcT
rutrada@ubuntu:~/Project$
```

5. ทำการสลับ mode การทำงาน ใช้คำสั่ง

Ctrl + Alt + 2

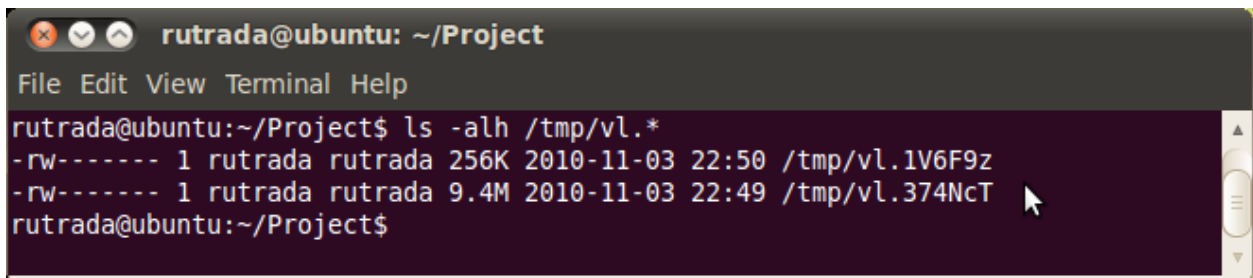
แล้วพิมพ์คำสั่ง `commit all`



```
monitor console
QEMU 0.12.4 monitor - type 'help' for more information
(qemu) commit all
(qemu) █
```

6. ทำการตรวจสอบไฟล์ vl.* ว่ามี 2 ไฟล์

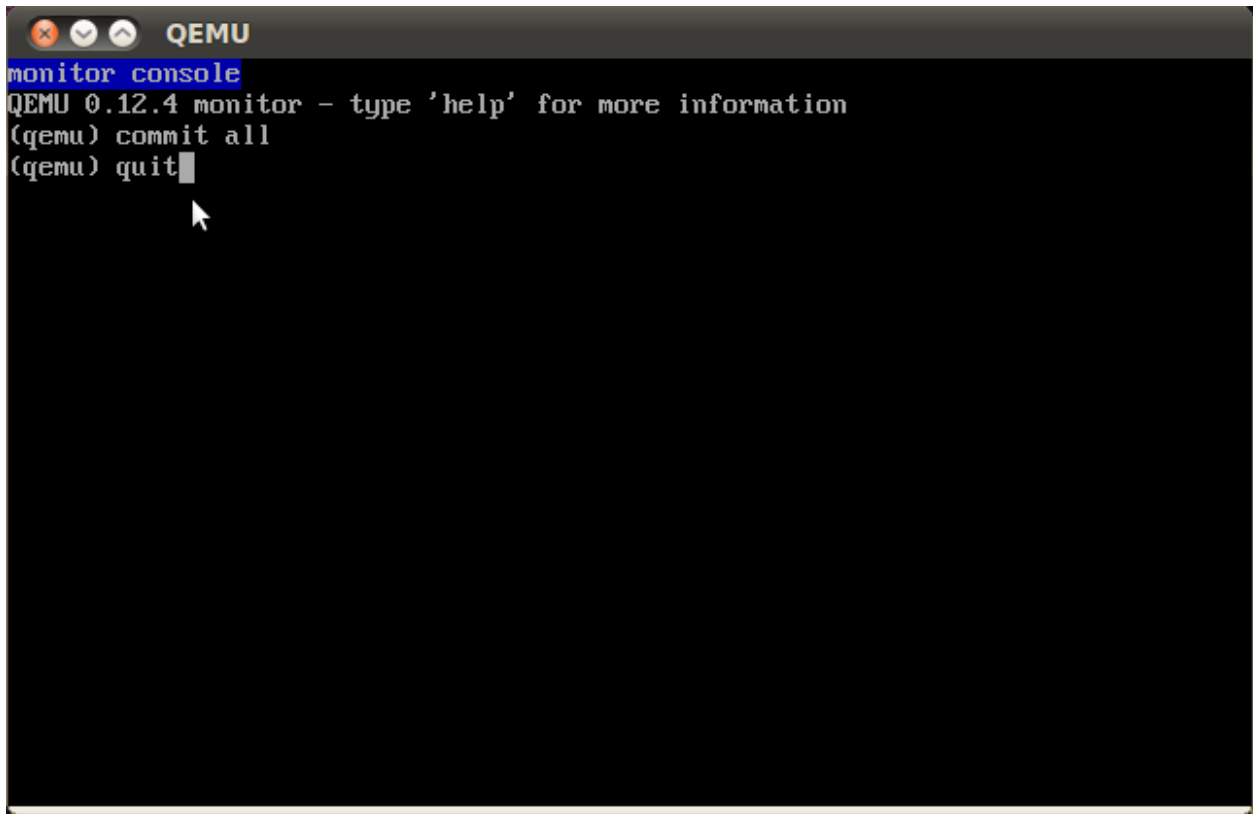
ด้วยคำสั่ง `$ls -alh /tmp/vl.*`



```
rutrada@ubuntu: ~/Project
File Edit View Terminal Help
rutrada@ubuntu:~/Project$ ls -alh /tmp/vl.*
-rw----- 1 rutrada rutrada 256K 2010-11-03 22:50 /tmp/vl.1V6F9z
-rw----- 1 rutrada rutrada 9.4M 2010-11-03 22:49 /tmp/vl.374NcT
rutrada@ubuntu:~/Project$
```

7. ทำการตรวจสอบ base image ว่ามีขนาดใหญ่ขึ้น

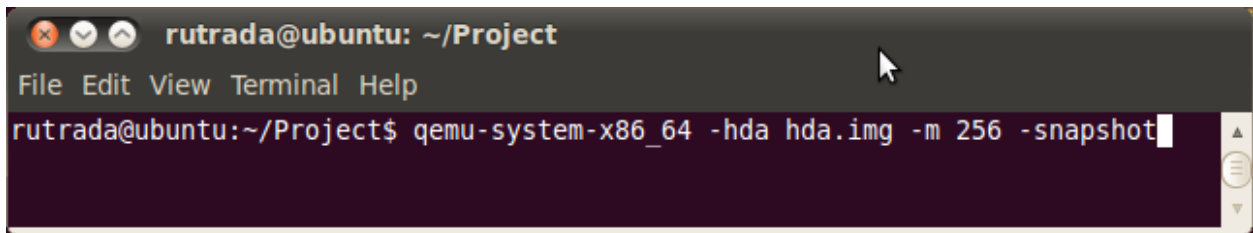
ด้วยคำสั่ง `$ls -alh hda.img` (ไฟล์มีขนาดใหญ่ขึ้น 5 MB)



A screenshot of a QEMU monitor console window. The window title is "QEMU". The text inside the console reads: "monitor console", "QEMU 0.12.4 monitor - type 'help' for more information", "(qemu) commit all", and "(qemu) quit". A mouse cursor is visible over the text.

```
monitor console
QEMU 0.12.4 monitor - type 'help' for more information
(qemu) commit all
(qemu) quit
```

9. พิมพ์คำสั่งที่ 1 อีกครั้ง (Start การทำงานของ VM อีกครั้ง) พบว่า test.img ยังอยู่



A screenshot of a terminal window titled "rutrada@ubuntu: ~/Project". The terminal shows the command: "qemu-system-x86_64 -hda hda.img -m 256 -snapshot". A mouse cursor is visible over the terminal text.

```
rutrada@ubuntu:~/Project$ qemu-system-x86_64 -hda hda.img -m 256 -snapshot
```


แบบที่ 2

1. สำเนา base image ขึ้นมา 1 ชุด (เก็บไว้ใน directory backup)
2. ขณะ commit จะมี empty ovl เกิดขึ้นมา
3. จากนั้นทำการย้าย ovl ตัวเก่าไปเก็บไว้ที่ backup
4. สร้างไฟล์ log เก็บลำดับของ ovl ที่ได้ทำการ commit แล้ว (อาจมีการเก็บ timestamp)
5. การ commit ครั้งต่อ ๆ ไป ให้ดำเนินการตามข้อ 2. – 4.
6. เขียน script รับ input จากผู้ใช้ ระบุ เวลา/ลำดับ ของ base image ที่ต้องการ
7. วง loop เพื่อ ทอย commit ทีละไฟล์ (เรียกใช้คำสั่ง qemu-img commit) ตามจำนวน input ของผู้ใช้
8. ผลที่ได้คือ base image ณ เวลา/ลำดับ ที่ผู้ใช้ต้องการ