Student ID: 406410035
Name: 秦紫頤
E-mail: chinjoyce30@gmail.com

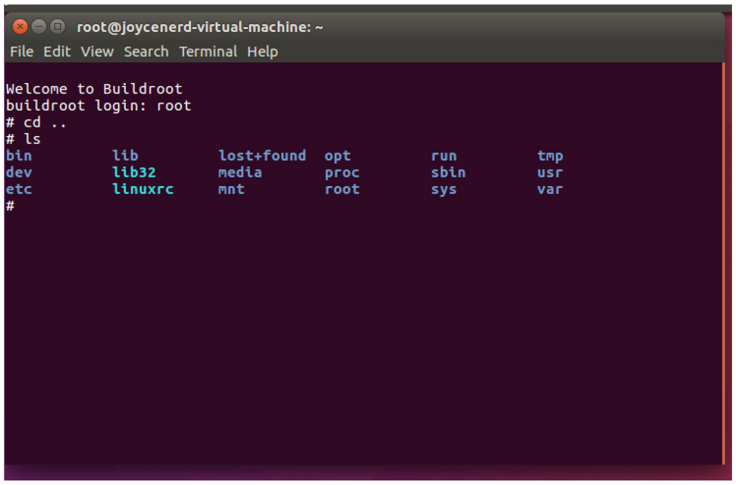
Lab Title: System Backup and Configuration

Lab Purpose:

You will learn three things from this lab. First, you will learn how to use the serial port to log in to Raspberry Pi. Second, you will learn how to extract files from the image file instead of the actual SD card. Third, you will learn how to backup your SD card by using dd and rsync command.

Lab Procedure:

1. Using serial port to log in to Raspberry Pi
    a. Connect Pi with the laptop using **USB to TTL serial transmission capable**
    b. Need to allow the connection of USB-Serial Controller in your virtual machine
    c. Using as root: sudo -s
    d. Install screen: apt install screen
    e. Connect Pi: screen /dev/ttyUSB0 115200


root@joycenerd-virtual-machine: ~
File Edit View Search Terminal Help

```
Welcome to Buildroot
buildroot login: root
# cd ..
# ls
bin         lib         lost+found  opt         run         tmp
dev         lib32       media       proc        sbin        usr
etc         linuxrc     mnt         root        sys         var
#
```

2. Extract files in the sdcard.img
    a. Go into the directory that sdcard.img reside: cd ~/buildroot/output/images/
    b. Check the partition of sdcard.img: fdisk ./sdcard.img
    c. Mount image file
        i. mkdir P1_BOOT, PR_ROOTFS
        ii. Check the partition of sdcard.img: fdisk -l sdcard.img
        iii. Mount the first partition to BOOT: sudo mount -o loop,offset=$((1*512)) sdcard.img ./P1_BOOT/
        iv. Mount the second partition to ROOTFS: sudo mount -o loop,offset=$((65537*512)) sdcard.img ./P2_ROOTFS/
3. Backup SD card
    a. Plug SD card into the computer
    b. Use as root: sudo -s

c. Backup BOOT partition: dd if=/dev/sdb1 of=boot_part.img

d. Backup filesystem partition

   i. Mount the second partition of the SD card: mount /dev/sdb2 /mnt/mmc2

   ii. Get the block count of the SD card filesystem: fdisk -l /dev/sdb -> convert Sectors to Blocks (Sectors/2)

   iii. Generate an image file the same size as the filesystem: dd if=/dev/zero of=filesystem.img bs=1024 count=15491584

   iv. Create a partition in the image file: fdisk filesystem.img -> n -> p -> w

   v. Check the partition status: fdisk filesystem.img -> p

   vi. Connect the partition in the image file via losetup and setup a pseudo device at the same time: losetup --offset $((2048*512)) --sizelimit=$((30981120*512)) /dev/loop0 filesystem.img

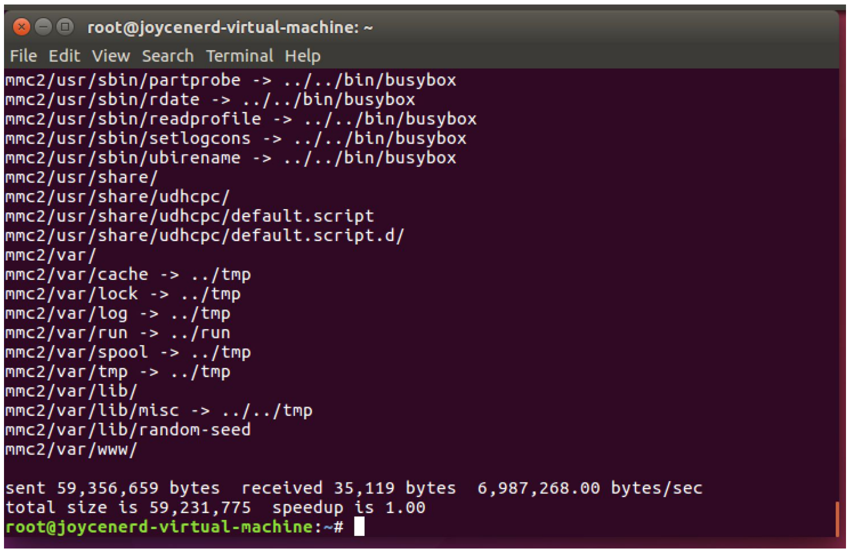
   vii. Format the partition in the image file: mkfs.ext4 /dev/loop0

   viii. Mount the partition:

      1. mkdir /mnt/sys_backup
      2. mount /dev/loop0 /mnt/sys_backup

   ix. Using **rsync** to backp the filesystem: rsync -axvH --delete /mnt/mmc2 /mnt/sys_backup/

   x. Remove the pseudo device: losetup -d /dev/loop0


root@joycenerd-virtual-machine: ~
File Edit View Search Terminal Help

```
mmc2/usr/sbin/partprobe -> ../../bin/busybox
mmc2/usr/sbin/rdate -> ../../bin/busybox
mmc2/usr/sbin/readprofile -> ../../bin/busybox
mmc2/usr/sbin/setlogcons -> ../../bin/busybox
mmc2/usr/sbin/ubirename -> ../../bin/busybox
mmc2/usr/share/
mmc2/usr/share/udhcpc/
mmc2/usr/share/udhcpc/default.script
mmc2/usr/share/udhcpc/default.script.d/
mmc2/var/
mmc2/var/cache -> ../tmp
mmc2/var/lock -> ../tmp
mmc2/var/log -> ../tmp
mmc2/var/run -> ../run
mmc2/var/spool -> ../tmp
mmc2/var/tmp -> ../tmp
mmc2/var/lib/
mmc2/var/lib/misc -> ../../tmp
mmc2/var/lib/random-seed
mmc2/var/www/

sent 59,356,659 bytes  received 35,119 bytes  6,987,268.00 bytes/sec
total size is 59,231,775  speedup is 1.00
root@joycenerd-virtual-machine:~#
```

Problems and Discussion:

- Q&A

  What is a loop device?

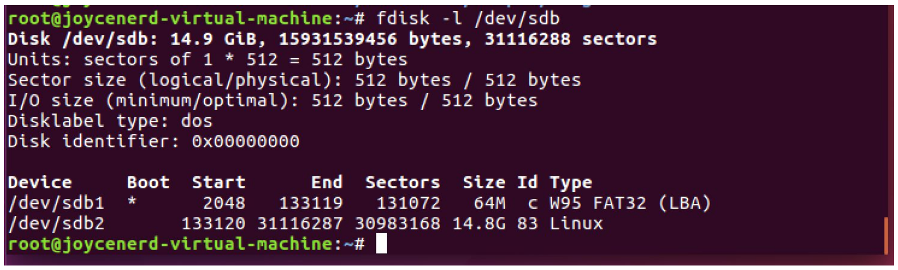
  A loop device is a pseudo-device that makes a file accessible as a block device. A loop device must be connected to an existing file in the filesystem, the file may then be mounted as if it were a disk device. After mounting a file that holds a filesystem, for example, the files within the filesystem can be accessed through the usual filesystem interface of the operating system, without any need for special functionality, such as reading and writing to ISO images. The loop device has several uses. Ex: it is a convenient method for managing and editing filesystem images offline, that are later used for normal system operation, or for filesystem backup.

Please explain the meaning of all the parameters in rsync -axvH --delete /mnt/mmc2 /mnt/sys_backup/ this command

1. -a (--archive): archive files and directory while synchronizing
2. -x (--one-file-sytem): don't cross filesystem boundaries
3. -v (--verbose): increase verbosity
4. -H (--hard-links): preserve hard links
5. --delete: delete extraneous files from the destination directories

- Discussions

When I'm trying to get the Blocks count of my filesystem I get Sectors instead. I search online and found out that if showing Blocks indicated that your Linux kernel is older., as for nowadays it shows **Sectors** instead. But to complete the task for this experiment I still need to get the Blocks. First of all, every sector is 512 bytes. And according to the experiment slides each block size is 1024 bytes. From here I get that every block is equal to 2 sectors. Then I divide Sectors by 2 to get the Blocks.



```
root@joycenerd-virtual-machine:~# fdisk -l /dev/sdb
Disk /dev/sdb: 14.9 GiB, 15931539456 bytes, 31116288 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disklabel type: dos
Disk identifier: 0x00000000

Device     Boot  Start      End  Sectors  Size Id Type
/dev/sdb1  *      2048   133119   131072   64M  c W95 FAT32 (LBA)
/dev/sdb2       133120 31116287 30983168 14.8G 83 Linux
root@joycenerd-virtual-machine:~#
```

The reason I use a different method to backup BOOT and filesystem is because:

1. BOOT: This partition is not possible to change from time to time so I use **dd** to copy all of the things inside into an image file, and also the size of the BOOT partition is quite small so when using dd, it will not take a very long time
2. Filesystem: This is where we store and manipulate the files and folders after booting up our Pi, so this will definitely change from time to time. Using rsync will not re-copy all the files again every time, it only syncs the differences. Also the filesystem is super large so using dd will take a tremendous of computation resource and time but using rsync is much faster.

I found out that when using the command sudo mount -o loop,offset=$((1*512)) sdcard.img P1_BOOT this won't work. But if I change to **sudo mount -o loop,offset=$((1*512)) sdcard.img ./P1_BOOT/** this will work. I think the reason is that when using P1_BOOT the computer doesn't recognize this is a directory. But using ./P1_BOOT/ the computer will recognize this is a directory in the current directory and eventually mount the sdcard.img to it.