CIS-387: Digital Forensics (4 credits)

With Dr. Jinhua Guo

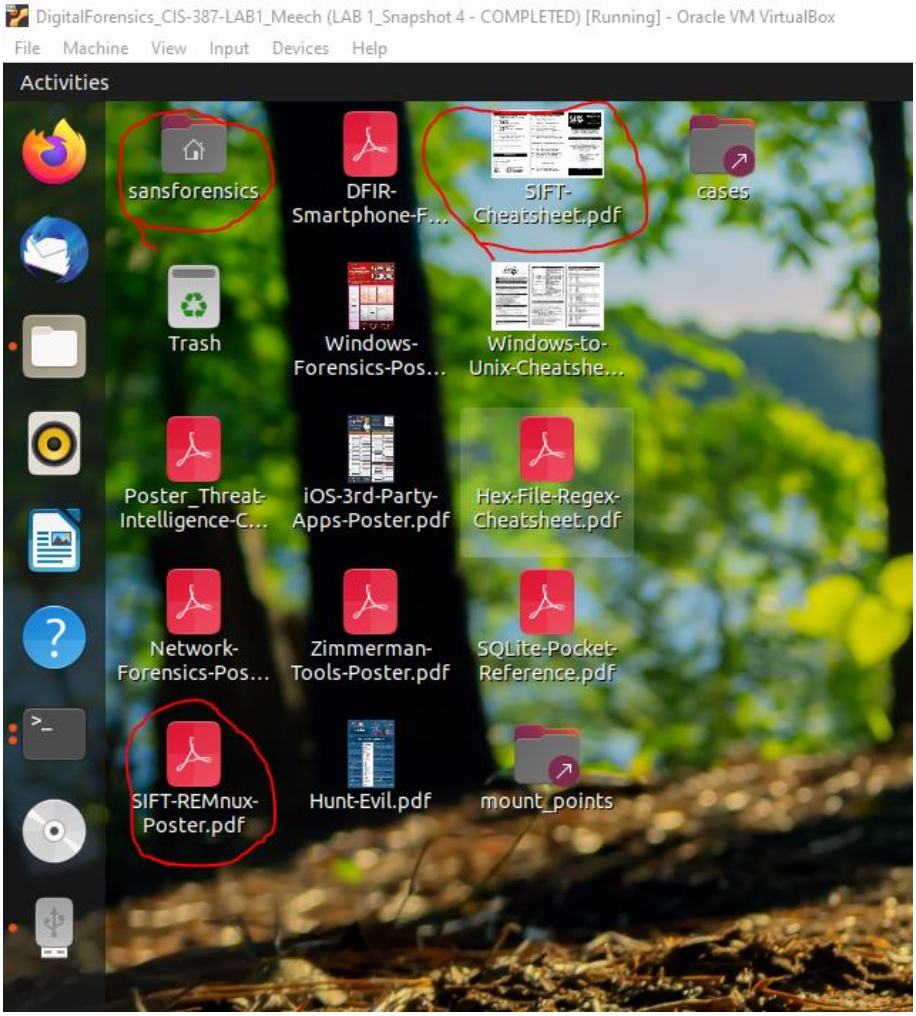
Lab 2

Demetrius Johnson

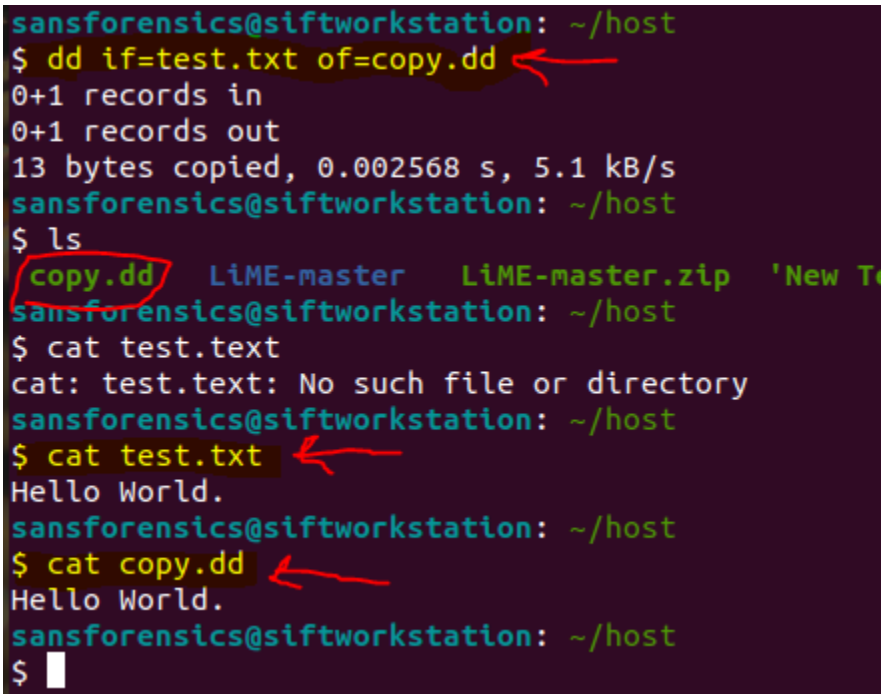
September 26, 2022

ACTIVITY 1: USING DD TO COPY AND COMPARE FILES

1. Launch SIFT Workstation 3 and open a terminal

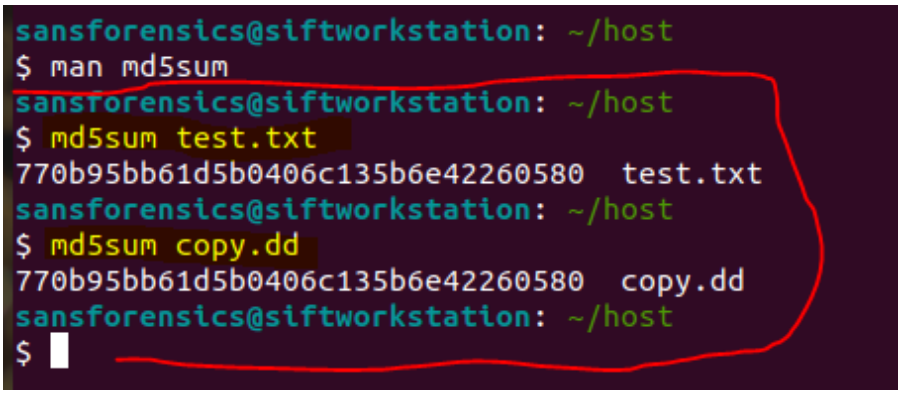


2. Use the command dd to copy an existing file on your computer. Name the new file copy.dd.

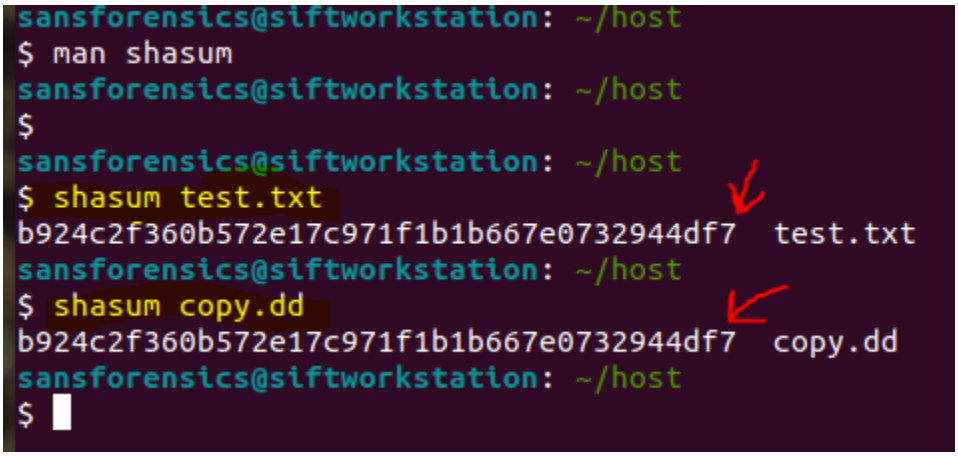


3. Using md5sum, create MD5 hashes of the original file and the copy.

4. Compare the hash of the copy to the hash of the original file; confirm that the hashes are the same.



5. Repeat Steps 3 and 4 using shasum to generate SHA1 hashes.

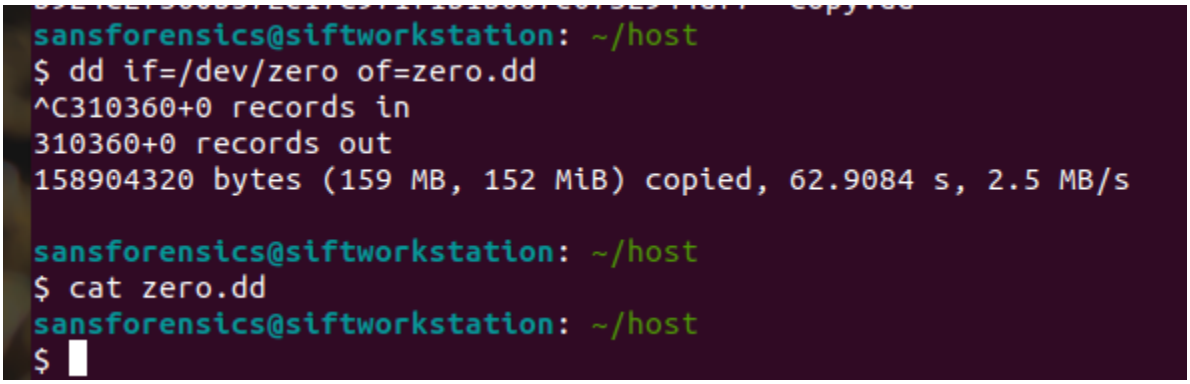


6. Use dd to copy one block of zero from /dev/zero to a file called zero.dd.

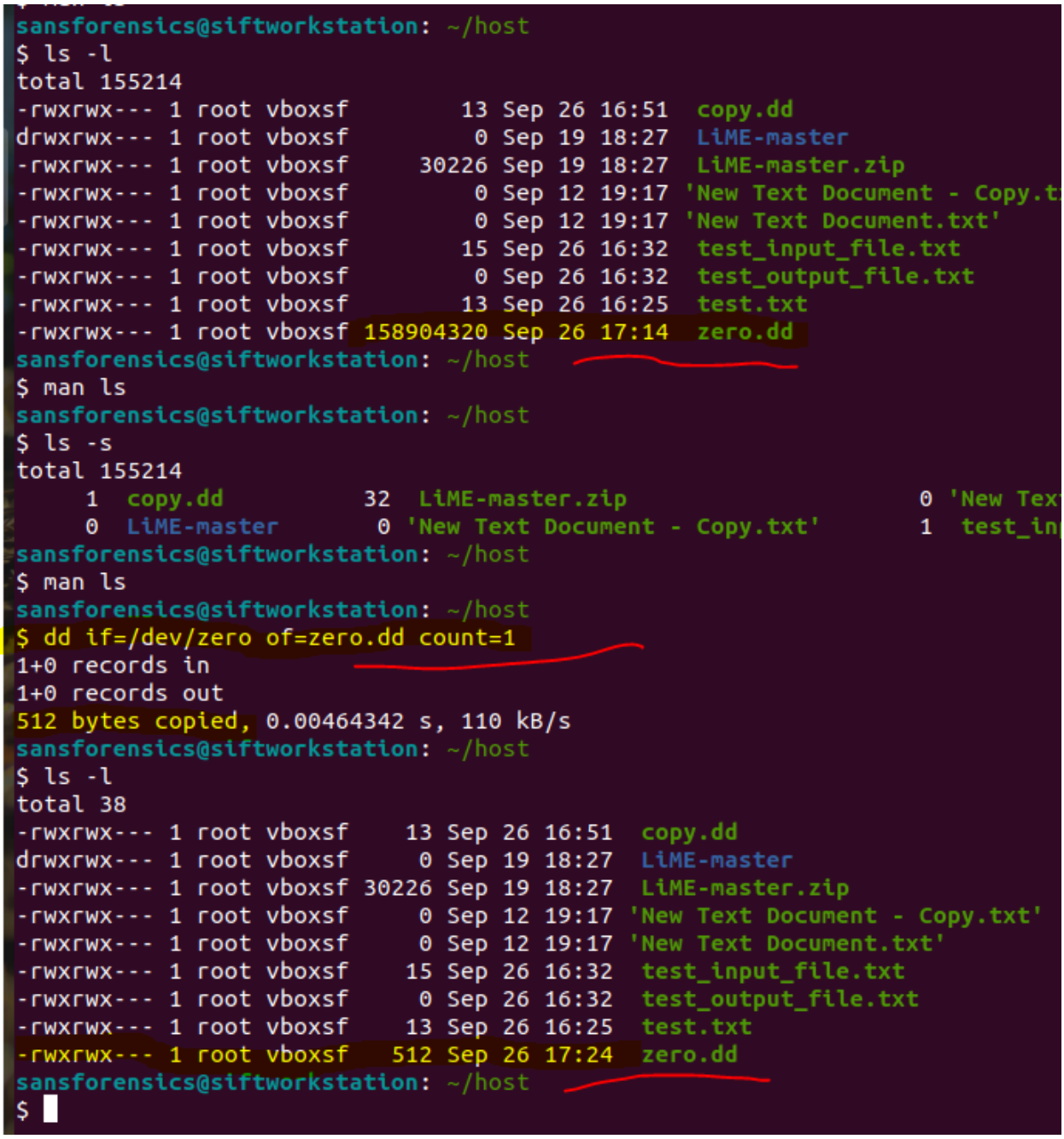
(Hint: use the dd option count).

Side note: I am guessing the /dev/zero is a large file with 0s written to it so that blocks of memory can be overwritten with zeroed data. I think this because I did the dd command but did not specify how many blocks to copy to my zero.dd file, then I realized it so I ctrl-c to stop the process, and it wrote 159MB of data to my zero.dd file. Then, When I 'cat zero.dd' to check the contents that was copied to the file, there was a long pause as it was gathering (reading) the file so it could output – but, when cat operation finished, the output was nothing (no characters or anything at all).

See:



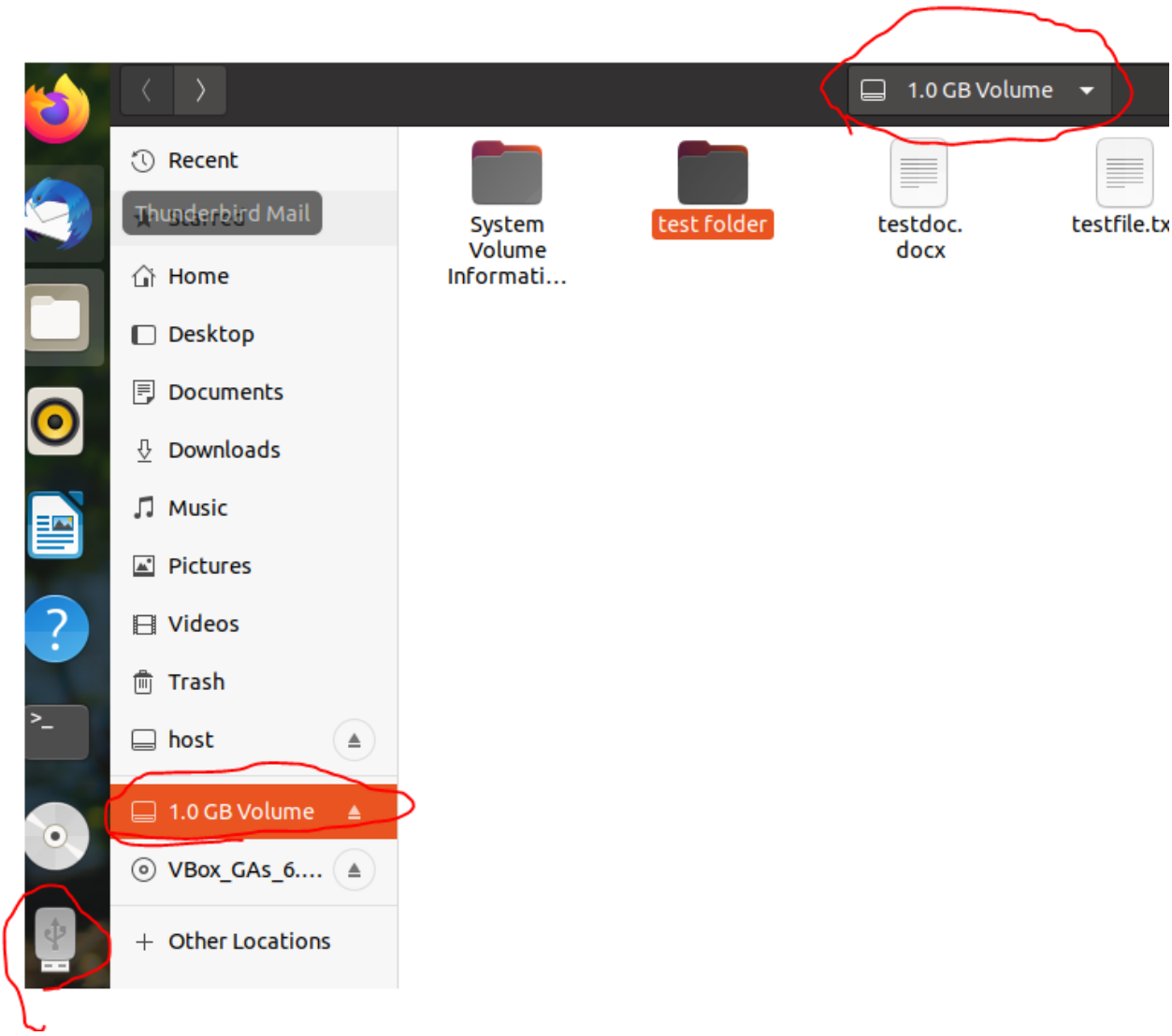
Also, notice the file size when I issue ls -l command before and after I re-ran the dd command and set count=1 block for copying:



Finally, I note what I learned in class today (9/26/22) that on many systems, the minimum write size is 512 bytes, and the minimum block size is 1 sector (512 bytes= the typical size of 1 sector for the main storage, such as HDD); so writing one block (cluster) meant writing 512 bytes.

7. Insert the USB drive and connect your USB to SIFT Workstation 3.

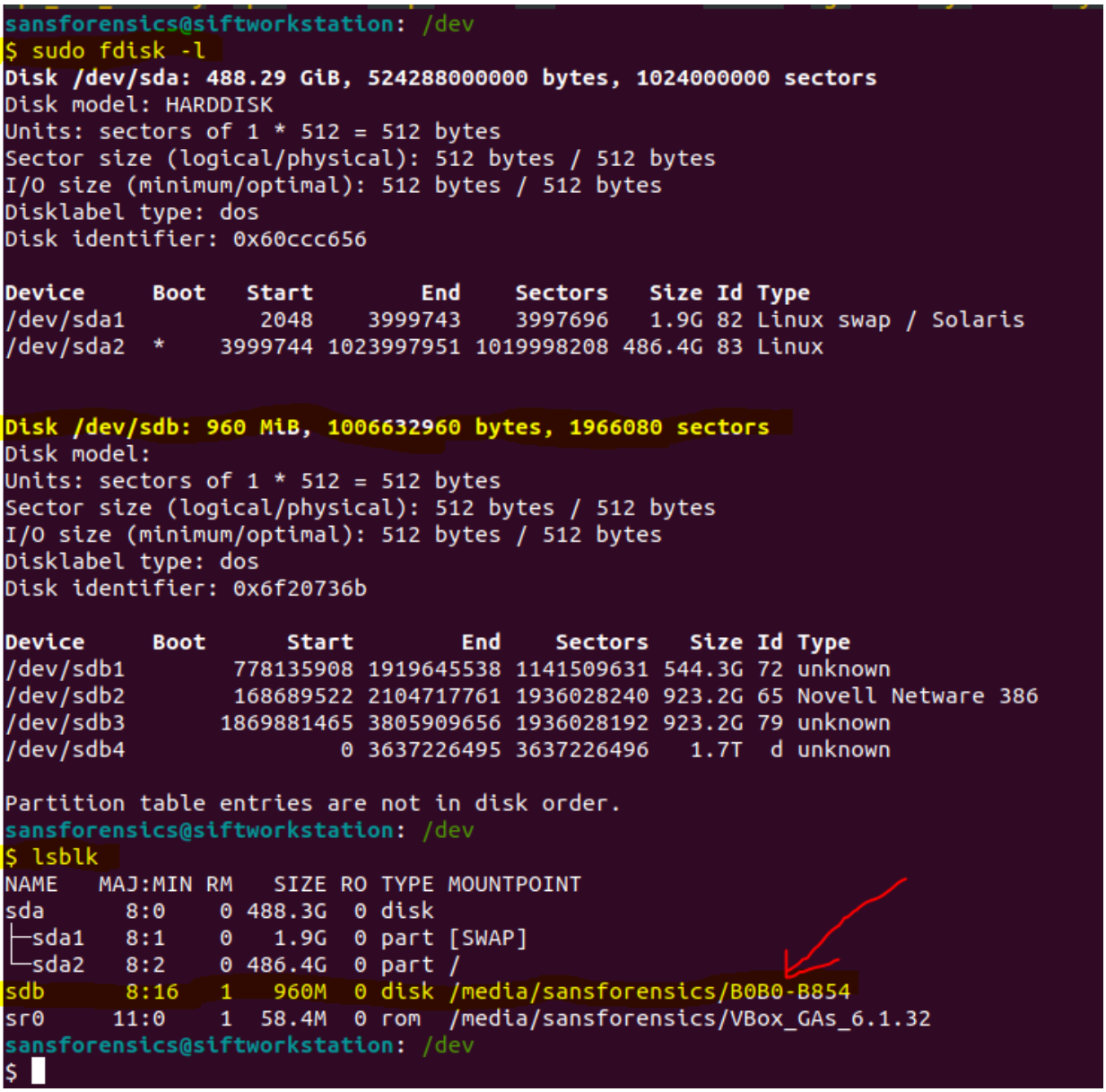
The USB drive should auto-mount. (NOTE: In a real investigation, you should use a write blocker to prevent the SIFT Workstation from modifying the USB drive.)



8. Run the command mount to find the USB device file name.

You will use the device file name in command dd to make a full image of your USB. For example, my USB's device file is /dev/sdb; it is mounted on /media/sansforensics/003B-38D3.

(Hint: using "lsblk" or "sudo fdisk -l" command to find it out)

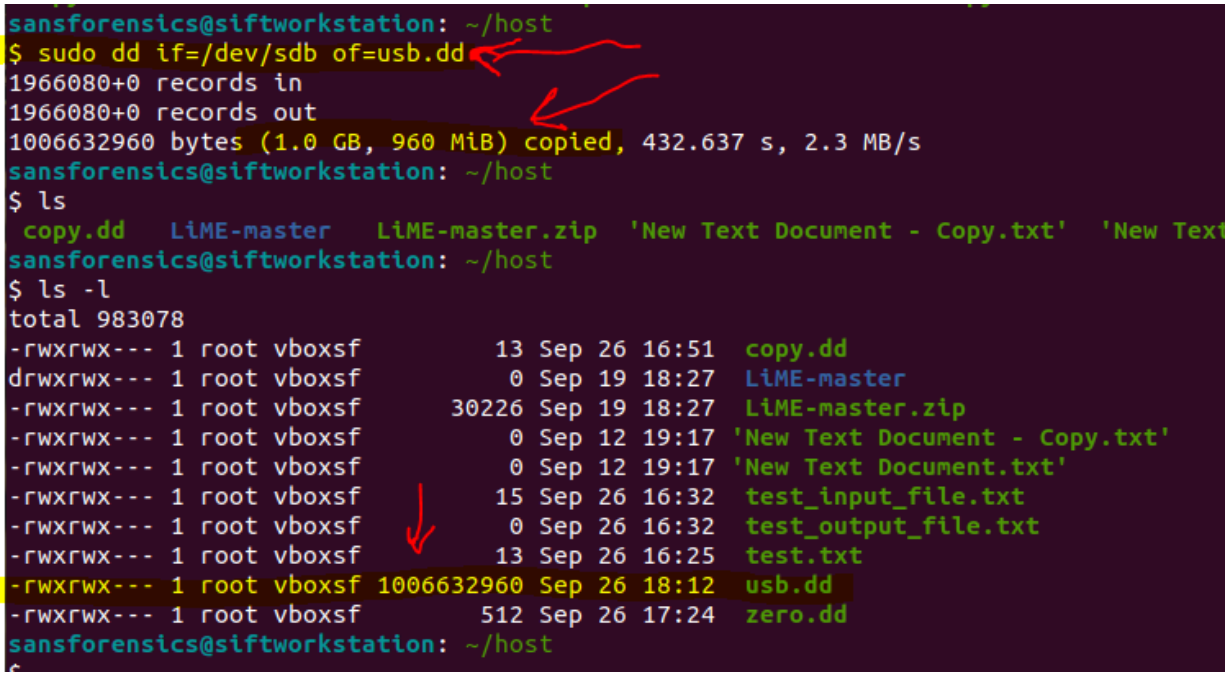


From above, I issued both the fdisk and the lsblk commands; I have found the location of my 1 GB (960MiB, where MiB means “mebibytes; prefix mebi = 2^20) USB device folder; I also have found the mount point of the drive; I realize I need device file location in order to do the image using the dd command because the mount point is not the root of the drive; I want to copy all contents of the drive bit-for-bit or byte-for-byte including file system formatting data. So, I must use /dev/sdb location.

Comment from professor Guo: It is because we are doing a Disk-to-Image acquisition instead of a logical acquisition.

9. Use dd to make a full image of your USB flash drive. Name the image usb.dd.

(Hint: dd if=/dev/sdb of=usb.dd)



10. Create both MD5 and SHA1 hashes of the USB flash.

(Hint: md5sum /dev/sdb; shasum /dev/sdb)

11. Create both MD5 and SHA1 hashes of the USB image.

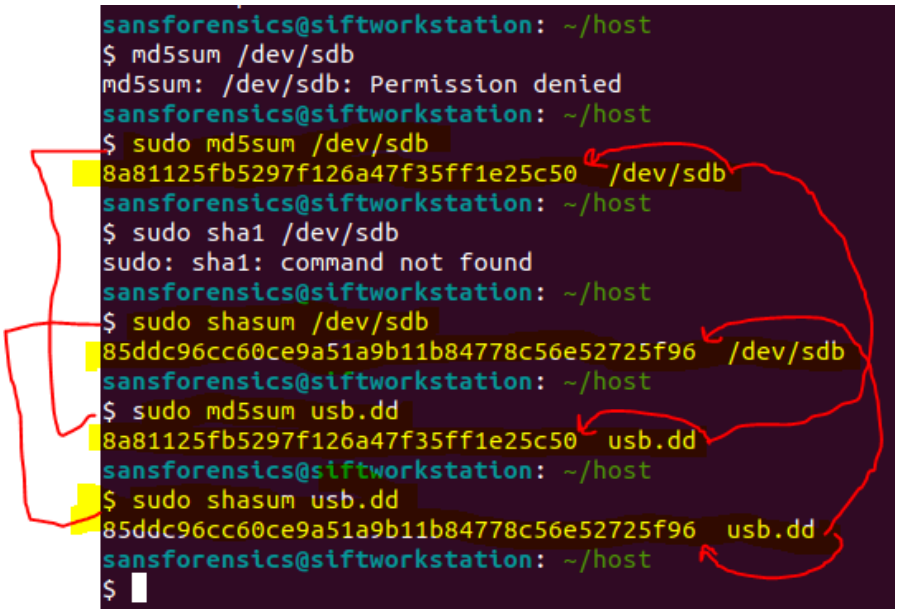
(Hint: md5sum usb.dd; shasum usb.dd)

12. Make sure that:

The md5 hash of the USB flash matches with the md5 hash of the USB image.

The sha1 hash of the USB flash matches with the sha1 hash of the USB image.

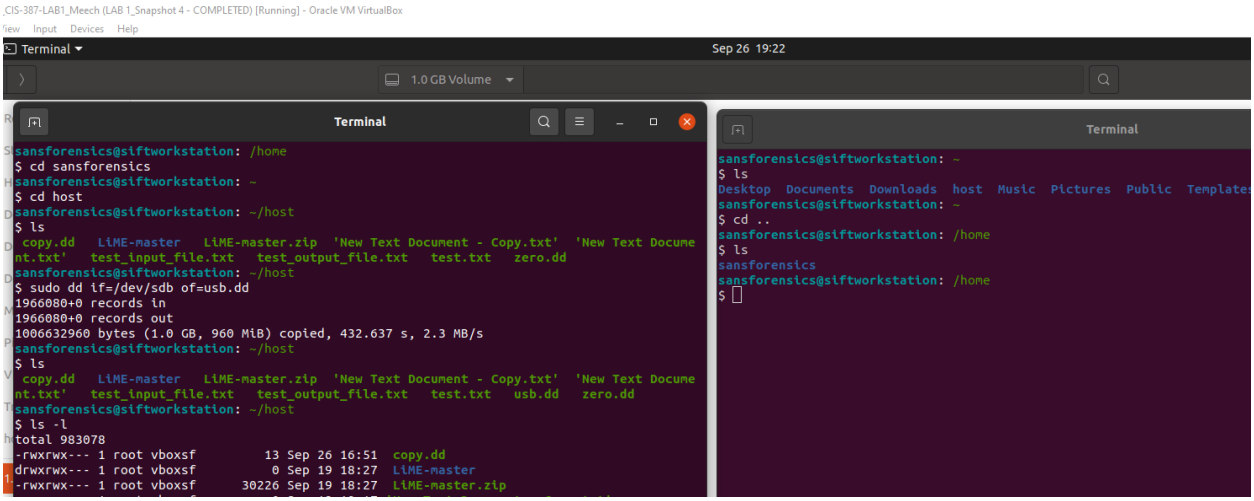
(10, 11, and 12 are all under same screenshot):



ACTIVITY 2: IMAGING WITH NETCAT OVER A NETWORK

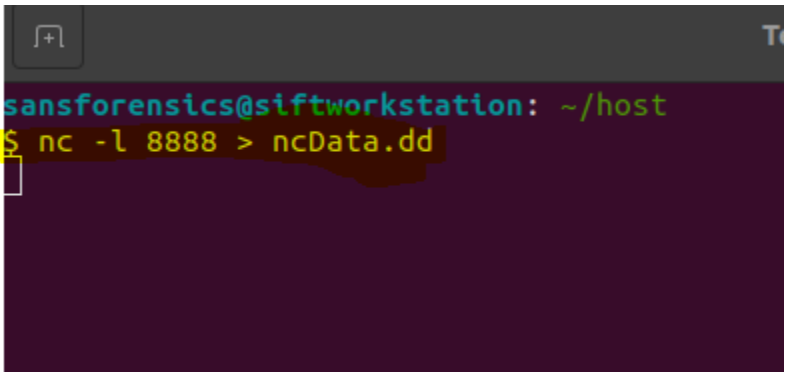
1. Launch SIFT Workstation 3.

2. Open two terminals on SIFT Workstation 3. One terminal represents a forensic machine; the other represents the suspect machine.



3. On the forensic machine terminal, use nc –l to listen on port 8888 for the incoming data. Save the received data as ncData.dd.

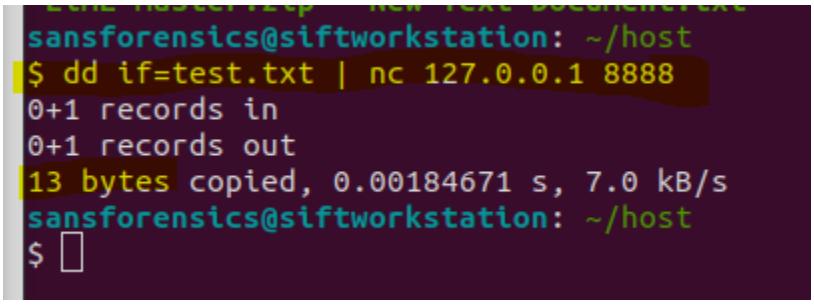
(Hint: nc –l 8888 > ncData.dd)



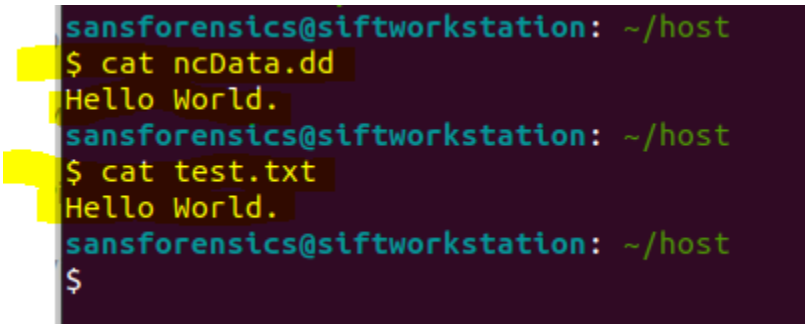
4. On the suspect machine terminal, use dd to copy an existing file and pipe (|) to netcat (nc), sending copy of the file to the forensic machine terminal.

Since we are sending data to the same machine, we use local host's loopback IP address 127.0.0.1. If you send data to a networked machine, replace 127.0.0.1 with the receiving machine's IP address.

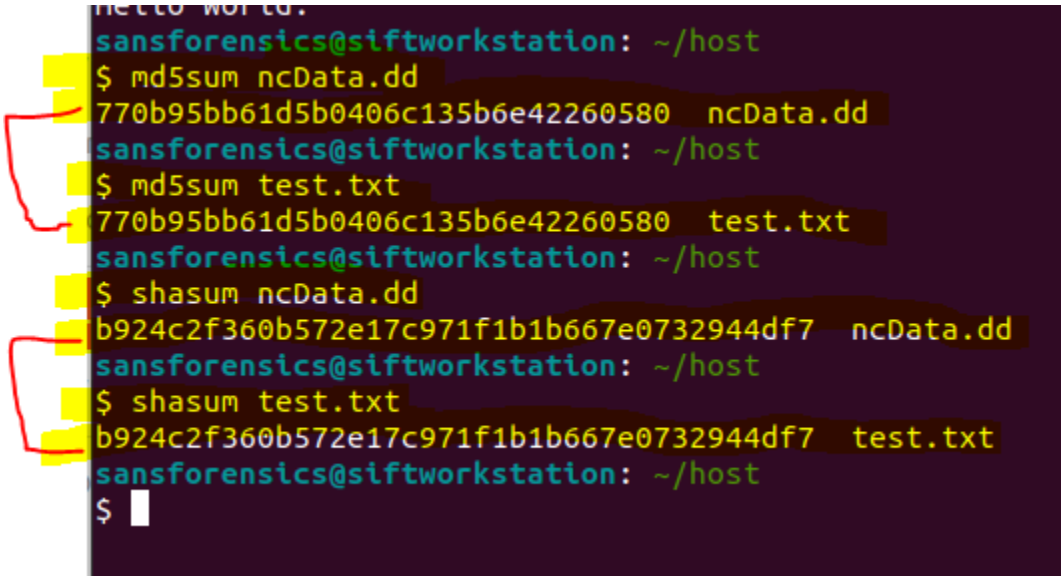
(Hint: In our case, we run dd if=the-original-file | nc 127.0.0.1 8888)



Notice file contents are the same:



5. Generate MD5 and SHA1 hashes of ncData.dd and compare them with the original file's MD5 and SHA1 hashes



Notice above, hash output is the same for both hash algorithms, signifying that we have made an exact copy of test.txt file.

Summary/Reflection

Overall, I learned the power of dd and netcat commands. I also understand better how file systems work, and why drives need to be formatted so that the OS can read and understand the connected device. Also, I learned better how to use the redirect operators > and |, where > is used to send output to a file, and | is used to send output to another function (process) to be used as in put. I also am more familiar with the practicality of using hash algorithms to check data integrity.

Now, if I ever need to make sure that some copy I made of a very important file is an exact copy, and if later I want to make sure that my file did not get corrupted, I can compute the hash of the original file, save that hash value somewhere (like write it down), then later computer the hash to check its integrity and the integrity of any other copies I have made of the file to see if even the slightest change to any of the file data contents were changed. This is a very powerful tool and can actually save a lot of time, and gives you the best possible data integrity check in even the most practical sense.