Forensic Analysis Notes:

Case Information:

Case: USB Forensic Imaging

Analyst: Julius Niyonzima

Date: 9/20/205

Professor: Dr. Chris Lamb

Purpose: Acquire a forensic image of a 4GB disk (dev/sdb) and verify evidence integrity via hashing

Email: jnbfg@umkc.edu

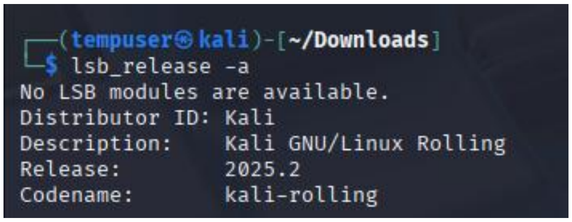
Id	name	Hash(None)	Notes
1	/dev/sdb	4GB USB disk (/dev/sdb)	Verify integrity through hashing

Tools and Environment

Host OS: Windows 11

Virtualization Platform: VMware Workstation Pro

Guest OS: Kali Linux v2025.2

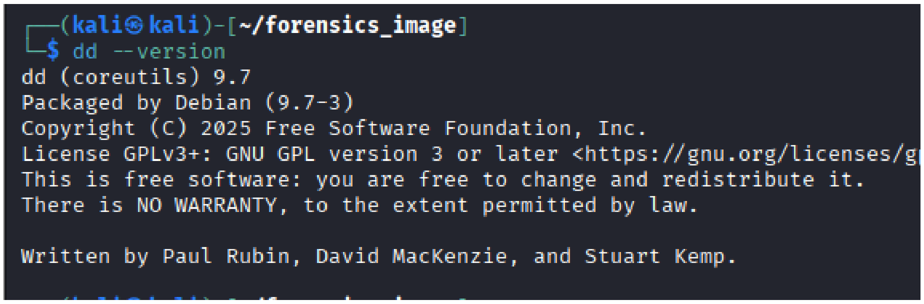


```
┌──(tempuser㉿kali)-[~/Downloads]
└─$ lsb_release -a
No LSB modules are available.
Distributor ID: Kali
Description:    Kali GNU/Linux Rolling
Release:        2025.2
Codename:       kali-rolling
```

VM state: Restored to original snapshot before starting

dd: command was used because of its ability to create a bit-for-bit copy of the entire disk

dd v9.7



```
┌──(kali㉿kali)-[~/forensics_image]
└─$ dd --version
dd (coreutils) 9.7
Packaged by Debian (9.7-3)
Copyright (C) 2025 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <https://gnu.org/licenses/g
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.

Written by Paul Rubin, David MacKenzie, and Stuart Kemp.
```

Evidence Description:

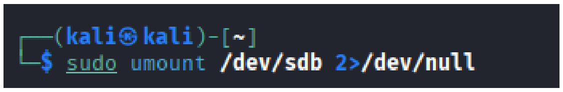
Device: simulated USB drive

Device path: /dev/sdb

Size: 4GB

Partition(s): /dev/sdb1 (not Mounted or altered)

Condition: unmounted prior to imaging



```
┌──(kali㉿kali)-[~]
└─$ sudo umount /dev/sdb 2>/dev/null
```

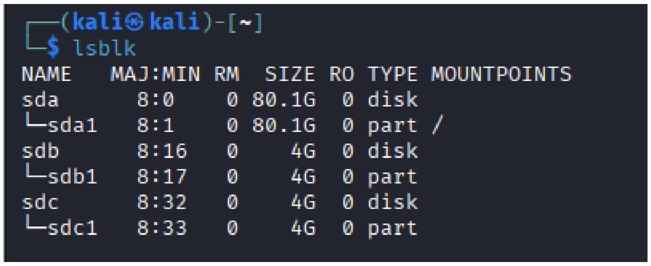
Analysis Steps :

Step1: Restored the Kali VM to the original snapshot

- ➢ Open the VMware and locate the Kali Linux, restore it to its original state

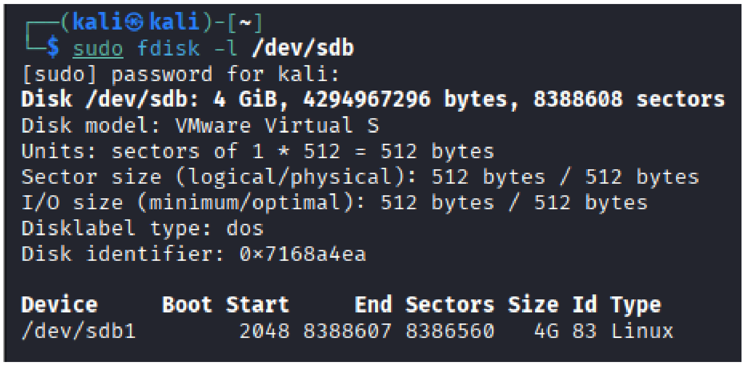
Step 2: Confirm the drive state

Used lsblk to confirm the presence of /dev/sdb



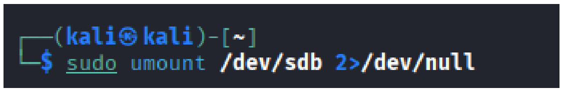
```
┌──(kali㉿kali)-[~]
└─$ lsblk
NAME   MAJ:MIN RM  SIZE RO TYPE MOUNTPOINTS
sda      8:0    0 80.1G  0 disk
└─sda1   8:1    0 80.1G  0 part /
sdb      8:16   0    4G  0 disk
└─sdb1   8:17   0    4G  0 part
sdc      8:32   0    4G  0 disk
└─sdc1   8:33   0    4G  0 part
```

Step3: verified the partitioning



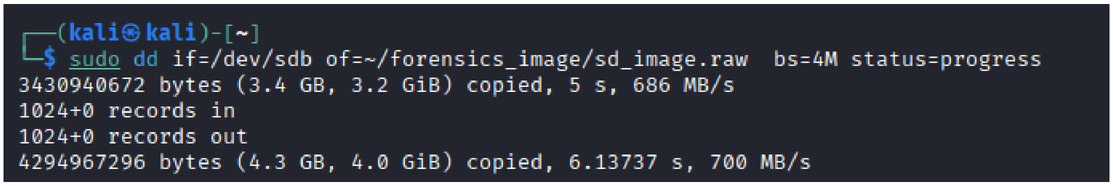
```
┌──(kali㉿kali)-[~]
└─$ sudo fdisk -l /dev/sdb
[sudo] password for kali:
Disk /dev/sdb: 4 GiB, 4294967296 bytes, 8388608 sectors
Disk model: VMware Virtual S
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disklabel type: dos
Disk identifier: 0×7168a4ea

Device     Boot Start     End Sectors Size Id Type
/dev/sdb1        2048 8388607 8386560   4G 83 Linux
```

Step 4: Unmounting just in case the drive is mounted



```
┌──(kali㉿kali)-[~]
└─$ sudo umount /dev/sdb 2>/dev/null
```

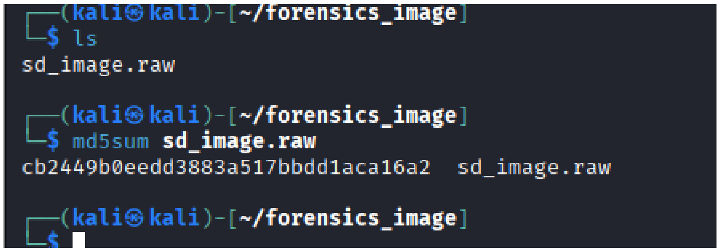
Step 5: Acquire the image



```
┌──(kali㉿kali)-[~]
└─$ sudo dd if=/dev/sdb of=~/forensics_image/sd_image.raw  bs=4M status=progress
3430940672 bytes (3.4 GB, 3.2 GiB) copied, 5 s, 686 MB/s
1024+0 records in
1024+0 records out
4294967296 bytes (4.3 GB, 4.0 GiB) copied, 6.13737 s, 700 MB/s
```

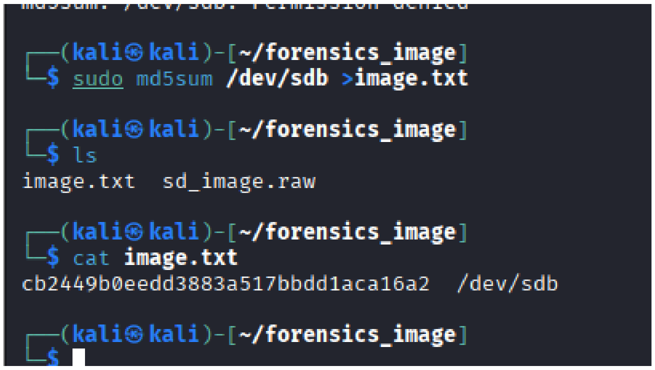
Step 6: Calculate the hash

- Once the image is created, a hash is calculated to ensure integrity of the image
- Md5um hash was used



```
┌──(kali㉿kali)-[~/forensics_image]
└─$ ls
sd_image.raw

┌──(kali㉿kali)-[~/forensics_image]
└─$ md5sum sd_image.raw
cb2449b0eedd3883a517bbdd1aca16a2  sd_image.raw

┌──(kali㉿kali)-[~/forensics_image]
└─$ 
```

Step 6: calculate the hash of the origin devices to validate if the device was not modified



```
md5sum: /dev/sdb: Permission denied

┌──(kali㉿kali)-[~/forensics_image]
└─$ sudo md5sum /dev/sdb >image.txt

┌──(kali㉿kali)-[~/forensics_image]
└─$ ls
image.txt  sd_image.raw

┌──(kali㉿kali)-[~/forensics_image]
└─$ cat image.txt
cb2449b0eedd3883a517bbdd1aca16a2  /dev/sdb

┌──(kali㉿kali)-[~/forensics_image]
└─$ 
```

Finding:

The hash of the origin device and the forensic image matched, proving authenticity and ensuring a complete, bit-by-bit copy was obtained.

Conclusions:

The forensic acquisition of the 4 GB USB drive (/dev/sdb) was completed successfully. The resulting image (sd_image.raw) accurately represents the state of the evidence at the time of acquisition. Integrity was confirmed through MD5sum hash verification by comparing the hash of the image with that of the origin device.