Data Acquisition using `dd`

In digital forensics, data acquisition is very important in the process of investigation of digital evidence. One of the most widely used tools for acquiring raw data from storage devices, partitions and RAM is the `dd` command. This tool is very effective in creating exact forensic images of storage devices

Introduction to `dd`

`dd` is a low-level command line utility that is available on all UNIX-like machines that allows for copying and conversion of data from one location to another. It also allows us to directly write to files from the hardware because UNIX treats all the storage devices and RAM as a file itself, so it's like working with a *special* file. In digital forensics, it is used to create bit-by-bit copies of storage media and ensuring the integrity of the evidence

Here's the basic syntax of `dd`

```
dd if=<input_file> of=<output_file> [options]
```

- `if` is the storage device that we will be making an image of
- `of` is the output file where we will be writing the forensic image to

The output files are usually in `.img` or `.dd` formats

Creating a Forensic Image

A forensic image is a bit-by-bit copy of the source media including everything like files, file systems and unallocated space. This image is in a `.img` or a `.dd` file.

To make a forensic image of Flash Drive

WARNING: Make sure that you double check the device that you are reading from and writing to. There is no GUI for this, so it is

possible that you overwrite your existing Operating system. If you are not sure, then *do not run* the commands

Step 1: First we have to identify the device we will be making a forensic image of. This can be done with tools like `lsblk` and `fdisk` for Linux users, and `diskutil` for Mac users. First note down all of the disks you see after running these commands. Plug in the Flash Drive, then run these commands again. The new disk that appears is the Flash drive that we will be making a forensic image of. It will be labeled as `/dev/sda` or `dev/sdb`

Step 2: Use `dd` to make the forensic image of the device and store it in a file. Run the following command:

```
sudo dd if=dev/sda of=/forensic_image.dd bs=64K conv=noerror,sync
status=progress
```

- `sudo`: We have to run this command with superuser privileges because normal users do not have access to hardware devices
- `if=/dev/sda`: We are assuming that the Flash drive we have inputted is at `/dev/sda`
- `of=forensic_image.dd` : Telling where the image file has to be saved
- `bs=64K` : This sets the block size to 64 Kilobytes. It can be used to optimize the transfer speed
- `conv=noerror,sync`: This option tells `dd` to continue even if there are errors and to make sure that the input blocks are synchronized with the output, preventing data loss or corruption
- `status=progress`: Displays the progress of the operation

Step 3: Once the process is complete, it's important to verify the integrity of the forensic image. This can be done by comparing the has values of the source device and the forensic image. UNIX can calculate the hash value of any file from the command line itself

```
sha256sum /dev/sda #This is the hash value of the source device
sha256sum forensic_image.dd #This is the hash value of the forensic image
created
```

Make a Forensic Image of a Partition

There are some scenarios where the entire drive need not copied, and one partition is enough. `dd` can also be used in such cases, we just have to change the input file that we are using.

UNIX treats all device drivers and hardware as files and we can use this to our advantage. Files like `dev/sda` is the entire disk itself, but the file `dev/sda1` is a partition of the disk. In order to create a forensic image of this partition, we can do this

```
sudo dd if=/dev/sda1 of=forensic_image.dd bs=64K status=progress
```

Capture Memory Dump

`dd` is not a tool that should be used to capture the memory dump, but it can be used as well. In this case we just have to target the right device file. In the case of the RAM the input file is `dev/mem`. So in order to make a Memory Dump, we can run the following command

```
sudo dd if=/dev/mem of=forensic_image.dd bs=64K status=progress
```

Best Practices

- Always make sure that you know what file corresponds to what drive as we do not want you to accidentally wipe your own data
- Always Verify the image with the help of `sha256sum` or `md5sum`
- Create multiple copies of the forensic image
- Write-lock the device that is being imaged

Disadvantages

Although `dd` is a very powerful tool, it does come with it's own set of drawbacks.

- `dd` does not preserve any kind of file system metadata like timestamps or attributes. This can be a drawback when it comes to forensic analysis.

- It's difficult to verify the integrity of the imaging as it is happening. We have to wait for the entire process, then we can verify the integrity of the image
- `dd` is a single-threaded application which means that it is very slow when compared to other tools that are made for forensic analysis
- There are chances of `dd` overwriting the existing Operating System if the input file is not specified properly

---

Alternatives

There are some alternatives that can be used for both Linux and Mac Operating Systems:

1. Guymager (Linux)
2. dmg2img (MacOS & Linux)
3. Mac Memory Dump (Only for RAM on Mac)
4. LiME => Linux Memory Extractor (Making Memory Dumps of Linux Machines)