Setting Up an Initial Lab Environment

Data Leakage Case Description

Purpose

- The purpose of this work is to learn various types of data leakage, and practice its investigation techniques.

Scenario Overview (1) - background

- Iaman Informant was working as a manager of the technology development division at a famous international company OOO
  - OOO developed state-of-the-art technologies and gadgets.
  - Mr. Informant was also very interested in IT (Information Technology), and had a slight knowledge of digital forensics.
- The information security policies of OOO include
  - All storage devices such as HDD, SSD, USB memory stick, and CD/DVD are forbidden under the 'Security Checkpoint' rules.
  - All employees are required to pass through the 'Security Checkpoint' system.
  - Confidential electronic files should be stored and kept in the authorized external storage devices and the secured network drives.
  - Confidential paper documents and electronic files can be accessed only within the allowed time range from 10:00 AM to 16:00 PM with the appropriate permissions.
  - Non-authorized electronic devices such as laptops, portable storage, and smart devices cannot be carried onto the company.

Scenario Overview (2) – data leakage

- One day, Mr. Informant received an offer from 'Spy Conspirator' to leak sensitive information related to the newest technology.
  - Mr. Conspirator was an employee of a rival company, and Mr. Informant decided to accept the offer for large amounts of money, and began establishing a detailed leakage plan.
- Mr. Informant made a deliberate effort to hide the leakage plan.
  - He discussed it with 'Mr. Conspirator' using an e-mail service like a business relationship. He also sent samples of confidential information through personal cloud storage.
- After receiving the sample data, Mr. Conspirator asked for the direct delivery of storage devices that stored the remaining (large amounts of) data.
- Eventually, Mr. Informant tried to take his storage devices away

Scenario Overview (3) – security checkpoint

- Mr. Informant and his devices were detected at the security checkpoint of the company.
  - he was suspected of leaking the company data.
- At the security checkpoint, although his devices (a USB memory stick and a CD) were briefly checked (protected with portable write blockers), there was no evidence of any leakage.
- And then, all devices were immediately transferred to the digital forensic laboratory for further analysis.
- In this scenario, find any evidence of the data leakage, and any data that might have been generated from the suspect's electronic devices.

Digital Forensic Practice Topics

Practice Point	Description	Practice Point	Description
Understanding Types of Data Leakage	- Storage devices > HDD (Hard DiskDrive), SSD (Solid State Drive) > USB flash drive, Flash memory cards > CD/DVD (with Optical Disk Drive) - Network Transmission > File sharing, Remote Desktop Connection > E-mail, SNS (Social Network Service) > Cloud services, Messenger	E-mail Forensics	- MS Outlook file examination - E-mails and attachments
Windows Forensics	- Windows event logs - Opened files and directories - Application (executable) usage history - CD/DVD burning records - External devices attached to PC - Network drive connection traces - System Caches - Windows Search databases - Volume Shadow Copy	Database Forensics	- MS Extensible Storage Engine (ESE) Database - SQLite Database
File System Forensics	- FAT, NTFS, UDF - Metadata (NTFS MFT, FAT Directory entry) - Timestamps - Transaction logs (NTFS)	Deleted Data Recovery	- Metadata based recovery - Signature & Content based recovery (aka Carving) - Recycle Bin of Windows - Unused area examination
Web Browser Forensics	- History, Cache, Cookie - Internet usage history (URLs, Search Keywords…)	User Behavior Analysis	- Constructing a forensic timeline of events - Visualizing the timeline

Learning approach for the class

Set up lab

Lab setup procedures

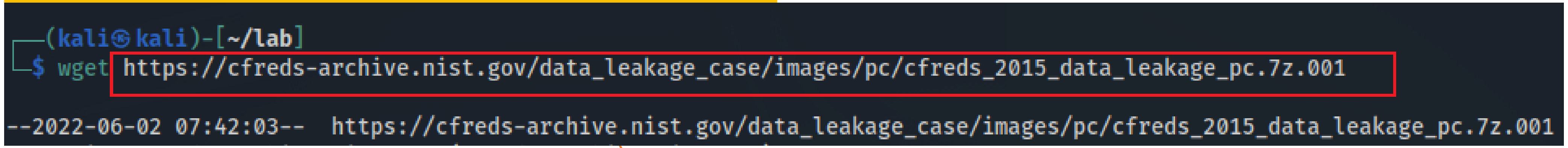
1. Install *Kali* (Optional)
2. Get the NIST data leakage case DD image
3. Exam files in the DD image
4. Extract registry files from the DD Image
5. Extract prefetched event log files from the DD Image
6. Extract security event log files from the DD Image
7. Install *tree*, Install *RegRipper 3.0*, *Windows-Prefetch-Parser*

1. Install *Kali* (Optional)

2. Get the NIST data leakage case image

How to download the DD image?

Download DD images. See notes if the link expired

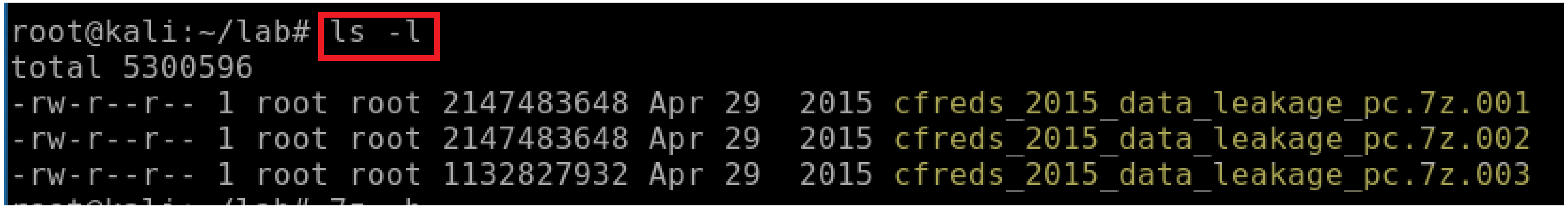


```
┌──(kali㉿kali)-[~/lab]
└─$ wget https://cfreds-archive.nist.gov/data_leakage_case/images/pc/cfreds_2015_data_leakage_pc.7z.001

--2022-06-02 07:42:03--  https://cfreds-archive.nist.gov/data_leakage_case/images/pc/cfreds_2015_data_leakage_pc.7z.001
```

Repeat **wget** command to download **.002** and **.003**

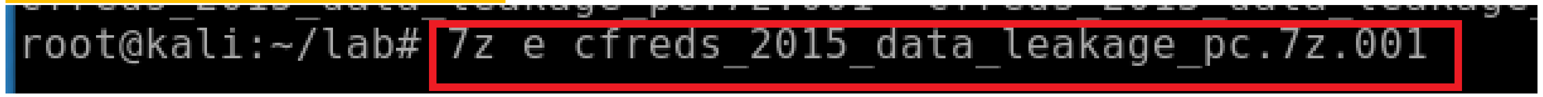
https://cfreds-archive.nist.gov/data_leakage_case/images/pc/cfreds_2015_data_leakage_pc.7z.001
https://cfreds-archive.nist.gov/data_leakage_case/images/pc/cfreds_2015_data_leakage_pc.7z.002
https://cfreds-archive.nist.gov/data_leakage_case/images/pc/cfreds_2015_data_leakage_pc.7z.003

Verify three zipped files



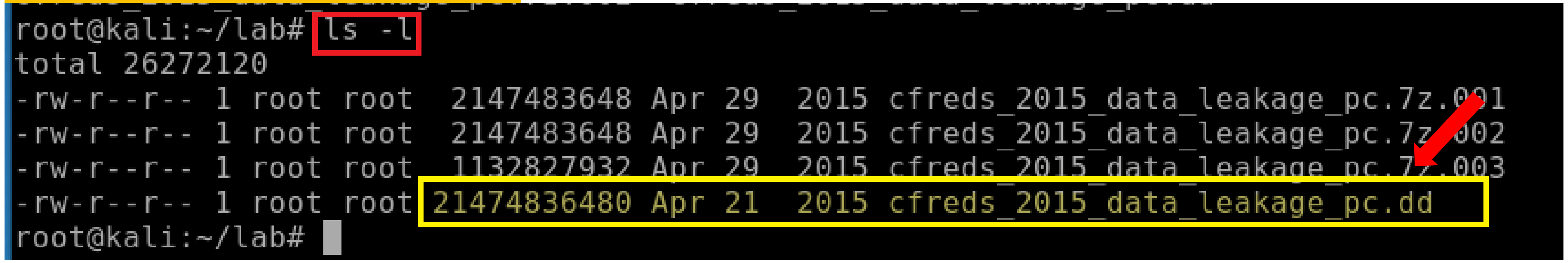
```
root@kali:~/lab# ls -l
total 5300596
-rw-r--r-- 1 root root 2147483648 Apr 29  2015 cfreds_2015_data_leakage_pc.7z.001
-rw-r--r-- 1 root root 2147483648 Apr 29  2015 cfreds_2015_data_leakage_pc.7z.002
-rw-r--r-- 1 root root 1132827932 Apr 29  2015 cfreds_2015_data_leakage_pc.7z.003
```

unzipped the DD image



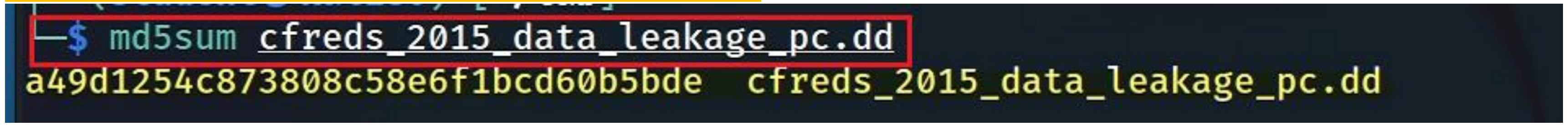
```
root@kali:~/lab# 7z e cfreds_2015_data_leakage_pc.7z.001
```

Verify the unzipped DD image



```
root@kali:~/lab# ls -l
total 26272120
-rw-r--r-- 1 root root  2147483648 Apr 29  2015 cfreds_2015_data_leakage_pc.7z.001
-rw-r--r-- 1 root root  2147483648 Apr 29  2015 cfreds_2015_data_leakage_pc.7z.002
-rw-r--r-- 1 root root  1132827932 Apr 29  2015 cfreds_2015_data_leakage_pc.7z.003
-rw-r--r-- 1 root root 21474836480 Apr 21  2015 cfreds_2015_data_leakage_pc.dd
root@kali:~/lab#
```

Verify the unzipped DD image with MD5

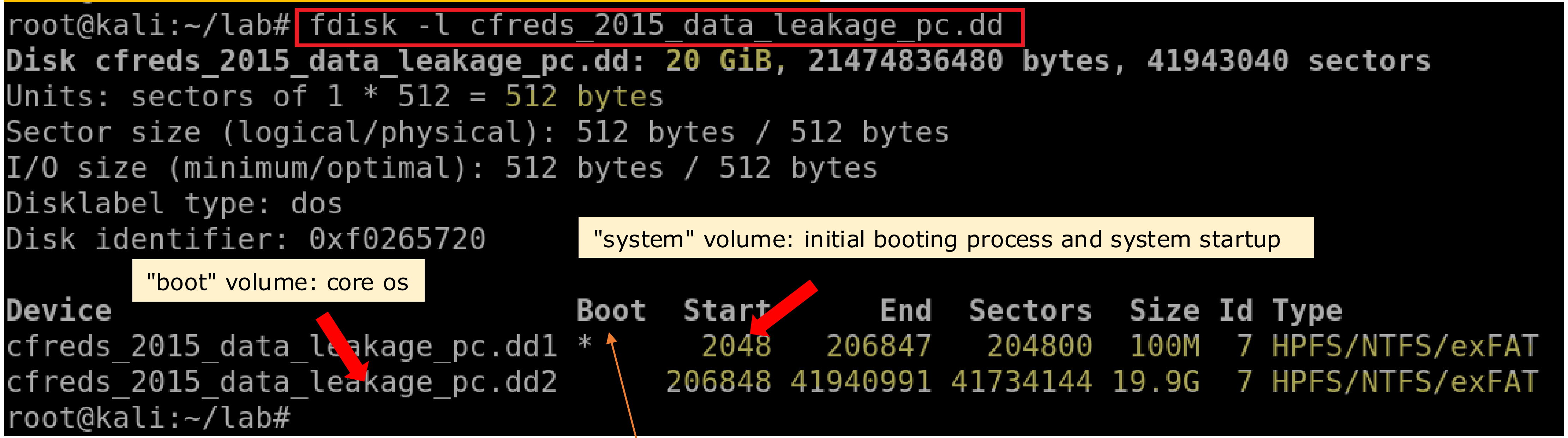


```
└─$ md5sum cfreds_2015_data_leakage_pc.dd
a49d1254c873808c58e6f1bcd60b5bde  cfreds_2015_data_leakage_pc.dd
```

3. Exam files in the DD image

Which partition is the system volume?

Exam partitions of the DD image using *fdisk* (*f*ormat *disk*)



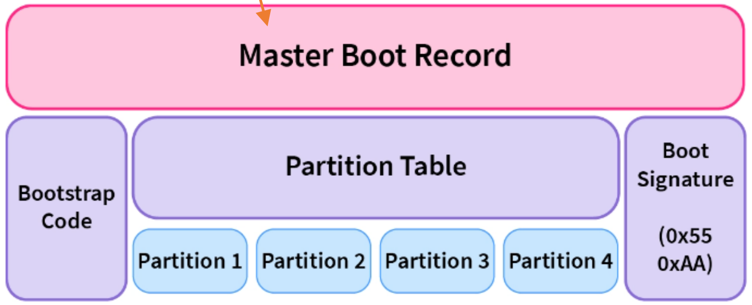
```
root@kali:~/lab# fdisk -l cfreds_2015_data_leakage_pc.dd
Disk cfreds_2015_data_leakage_pc.dd: 20 GiB, 21474836480 bytes, 41943040 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disklabel type: dos
Disk identifier: 0xf0265720

Device                          Boot   Start      End  Sectors  Size Id Type
cfreds_2015_data_leakage_pc.dd1 *       2048   206847   204800  100M  7 HPFS/NTFS/exFAT
cfreds_2015_data_leakage_pc.dd2       206848 41940991 41734144 19.9G  7 HPFS/NTFS/exFAT
root@kali:~/lab#
```

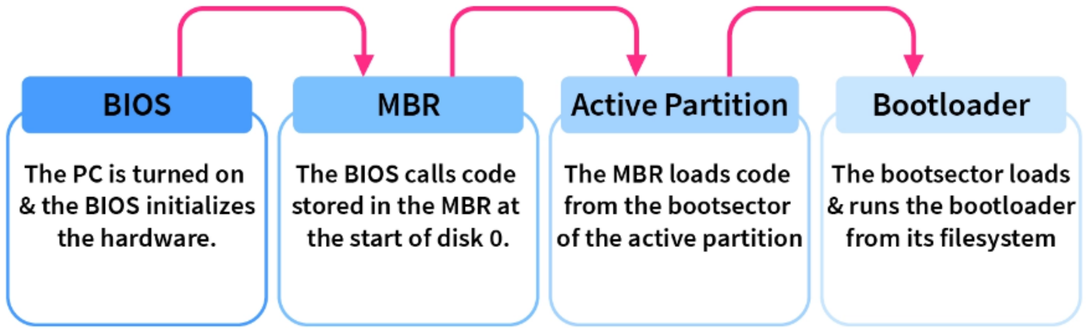
"system" volume: initial booting process and system startup

"boot" volume: core os

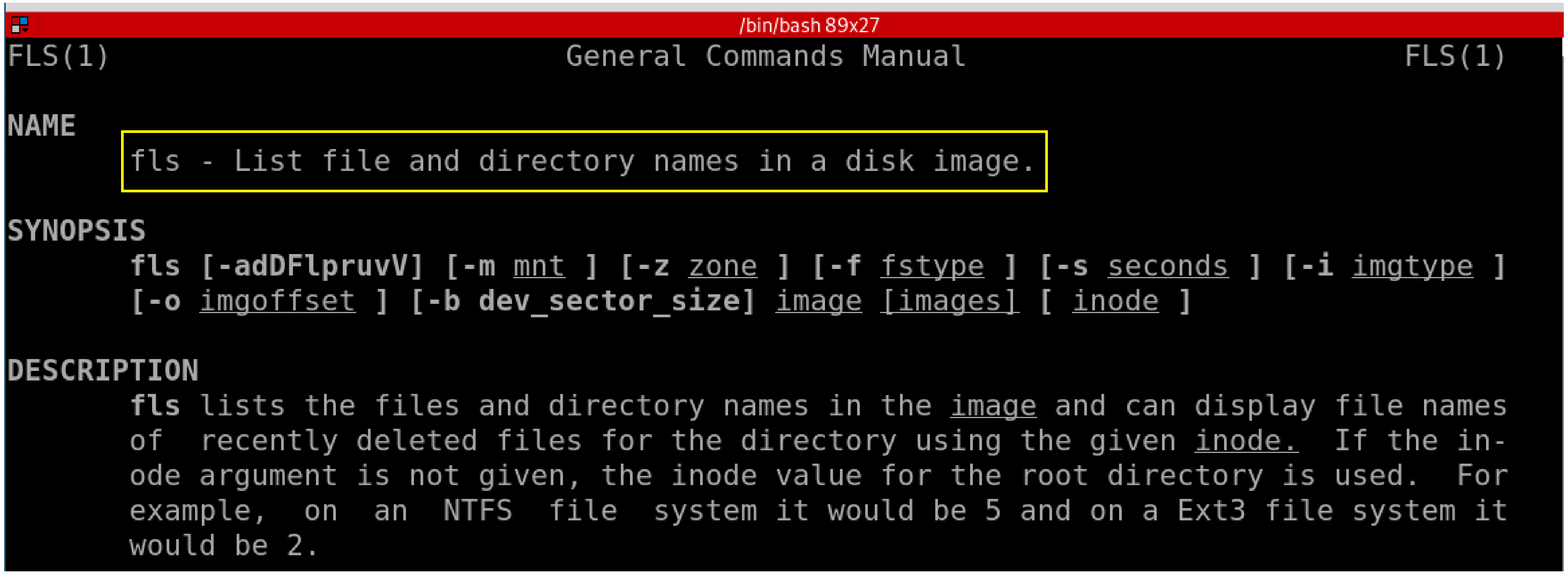
the device is bootable,
not a boot partition

System Volume	Boot Volume
contains essential system files and configurations required for the initial booting process and system startup	the core operating system files are stored
includes boot files such as the Master Boot Record (MBR) or GUID Partition Table (GPT)	contains files like the Windows system files (e.g., in Windows environments) and program files.
essential for the operating system to locate and load the necessary files for booting.	is where the operating system continues to run once the boot process is complete.
	the boot volume is assigned a drive letter (such as "C:" in Windows)



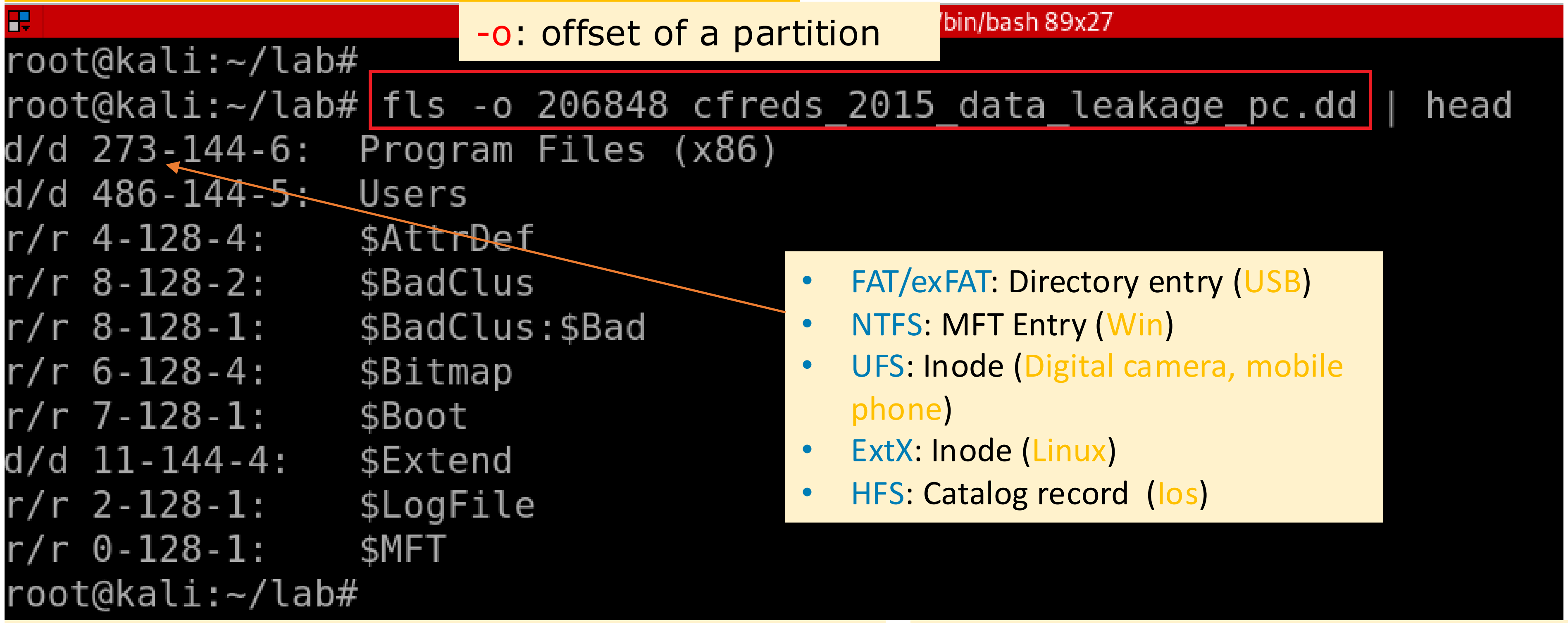

BIOS
The PC is turned on & the BIOS initializes the hardware.
MBR
The BIOS calls code stored in the MBR at the start of disk 0.
Active Partition
The MBR loads code from the bootsector of the active partition
Bootloader
The bootsector loads & runs the bootloader from its filesystem

Which command to list file/directory names in a system volume?


/bin/bash 89x27

```
FLS(1)                     General Commands Manual                     FLS(1)

NAME
       fls - List file and directory names in a disk image.

SYNOPSIS
       fls [-adDFlpruvV] [-m mnt ] [-z zone ] [-f fstype ] [-s seconds ] [-i imgtype ]
       [-o imgoffset ] [-b dev_sector_size] image [images] [ inode ]

DESCRIPTION
       fls lists the files and directory names in the image and can display file names
       of  recently deleted files for the directory using the given inode.  If the in-
       ode argument is not given, the inode value for the root directory is used.  For
       example,  on  an  NTFS  file  system it would be 5 and on a Ext3 file system it
       would be 2.
```

List file/directory names of the system volume



-o: offset of a partition

/bin/bash 89x27

```
root@kali:~/lab#
root@kali:~/lab# fls -o 206848 cfreds_2015_data_leakage_pc.dd | head
d/d 273-144-6:  Program Files (x86)
d/d 486-144-5:  Users
r/r 4-128-4:    $AttrDef
r/r 8-128-2:    $BadClus
r/r 8-128-1:    $BadClus:$Bad
r/r 6-128-4:    $Bitmap
r/r 7-128-1:    $Boot
d/d 11-144-4:   $Extend
r/r 2-128-1:    $LogFile
r/r 0-128-1:    $MFT
root@kali:~/lab#
```

- FAT/exFAT: Directory entry (USB)
- NTFS: MFT Entry (Win)
- UFS: Inode (Digital camera, mobile phone)
- ExtX: Inode (Linux)
- HFS: Catalog record (Ios)

r/ r: Regular file

- 'r': saved in the file's file name structure
- 'r' : saved in the file's metadata structure.
- For allocated files, these should always be equal.
- For deleted files, they could be different if one of the structures was reallocated to a different file type.

- d: Directory
- 273: the entry of metadata address
- metadata address: is a term that is used in the sleuth kit (TSK) as a generic term for the addresses of file system-specific data structures
- -144-6 (for NTFS), identifies the $Data attribute that this name points to.

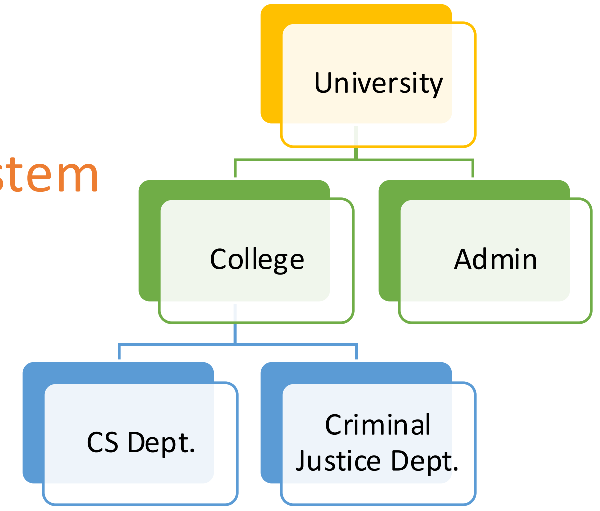
4. Extract key registry files from a DD image

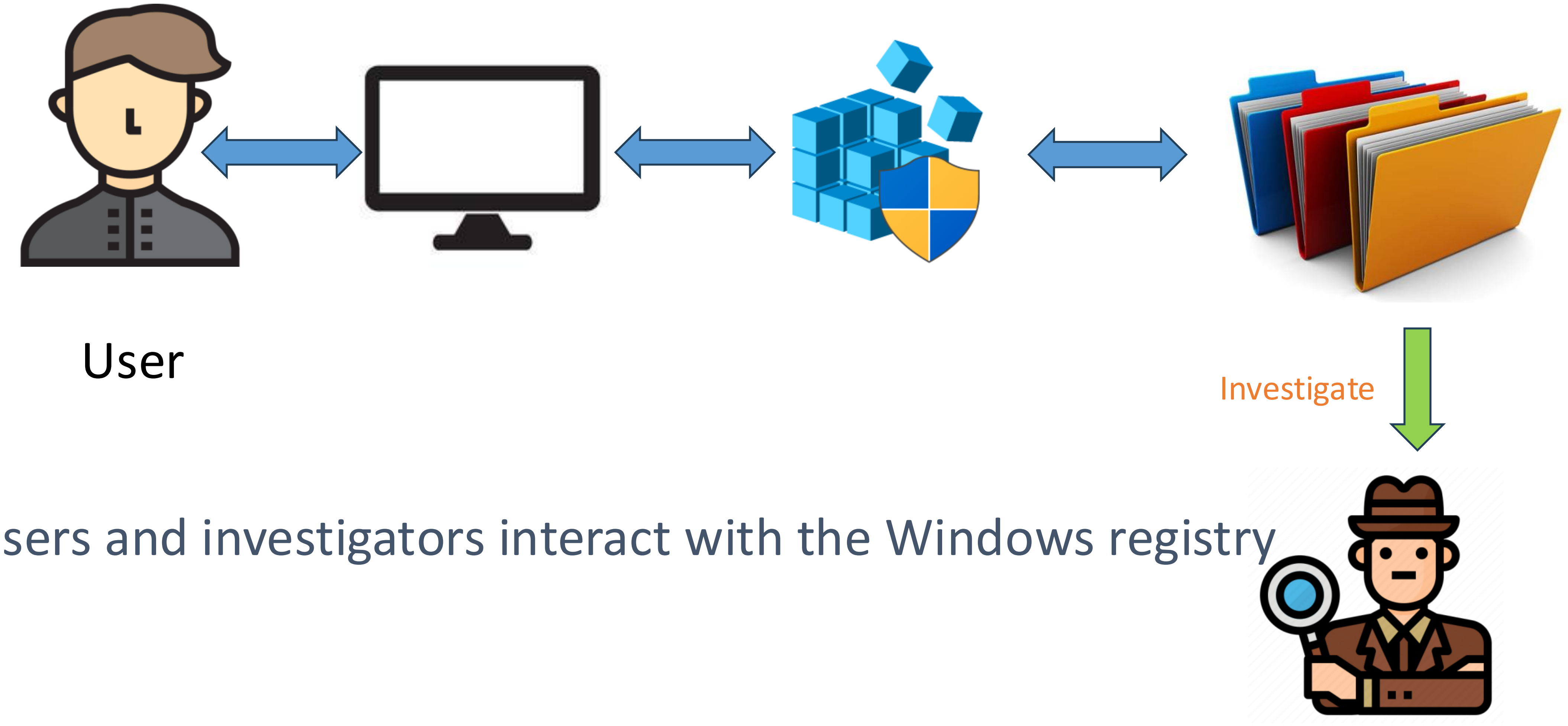
Overview

- What is Windows Registry?
- How to access registry files?
- How to copy registry files for future forensic analysis?

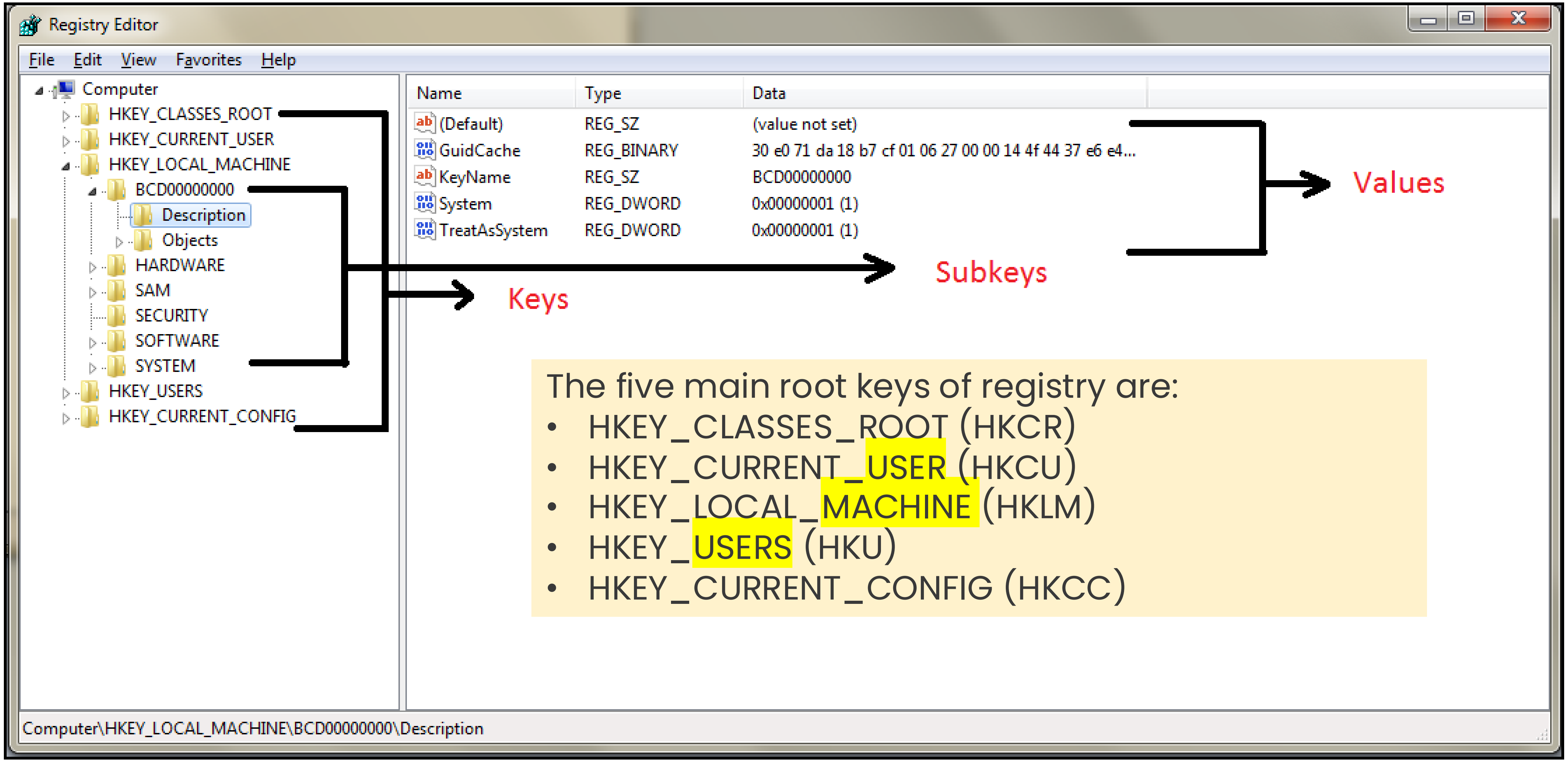
4.1 What is Windows Registry?

- The Windows Registry is a database
  - Stores low-level settings for the Microsoft Windows operating system
  - Store setting for applications that opt to use the registry.
- It is a hierarchical database
  - Store forensic information
- Components use registry
  - **Kernel**: the core component of an OS
  - **Device drivers**: software components that facilitate communication between the OS and hardware devices
  - **Services**: background processes that provide specific functions to the OS or apps
  - **Security** Accounts Manager (SAM): managing user accounts and security settings
  - **User interface**: through which users interact with and control the computer or software





Users and investigators interact with the Windows registry



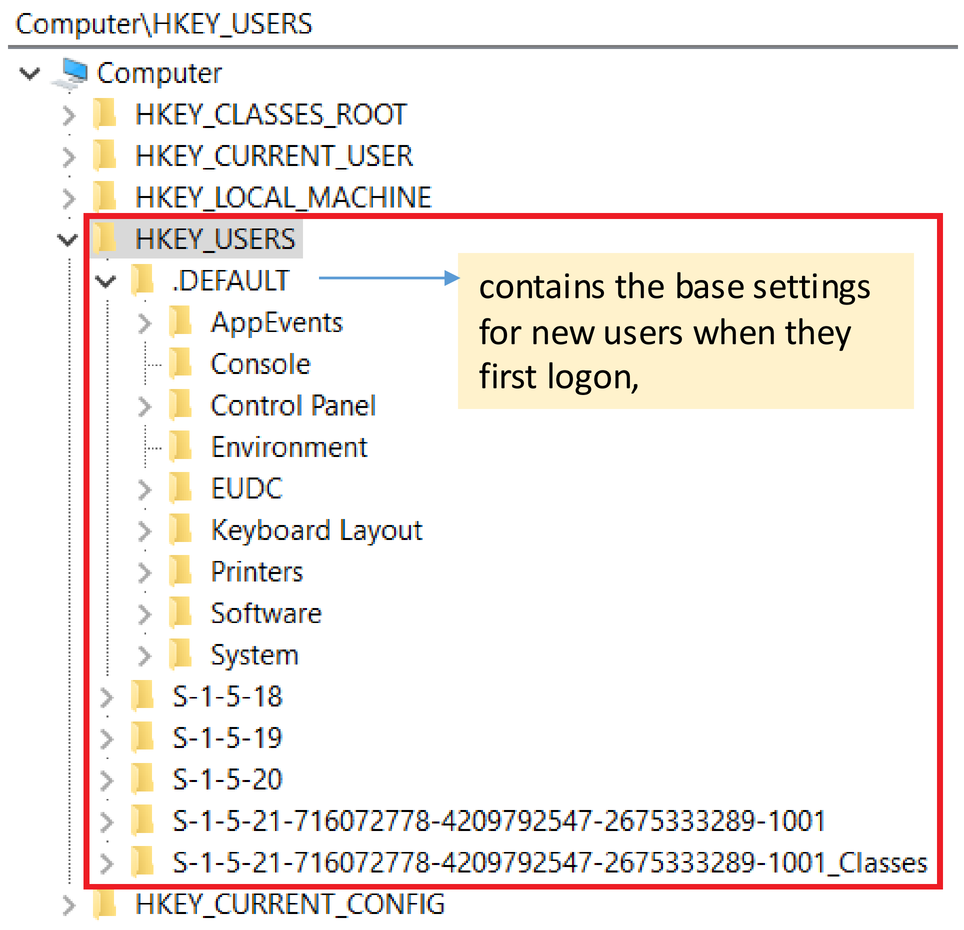
Name	Type	Data
(Default)	REG_SZ	(value not set)
GuidCache	REG_BINARY	30 e0 71 da 18 b7 cf 01 06 27 00 00 14 4f 44 37 e6 e4...
KeyName	REG_SZ	BCD00000000
System	REG_DWORD	0x00000001 (1)
TreatAsSystem	REG_DWORD	0x00000001 (1)

Computer\HKEY_LOCAL_MACHINE\BCD00000000\Description

The five main root keys of registry are:

- HKEY_CLASSES_ROOT (HKCR)
- HKEY_CURRENT_USER (HKCU)
- HKEY_LOCAL_MACHINE (HKLM)
- HKEY_USERS (HKU)
- HKEY_CURRENT_CONFIG (HKCC)

HKEY_USERS (HKU)



Computer\HKEY_USERS

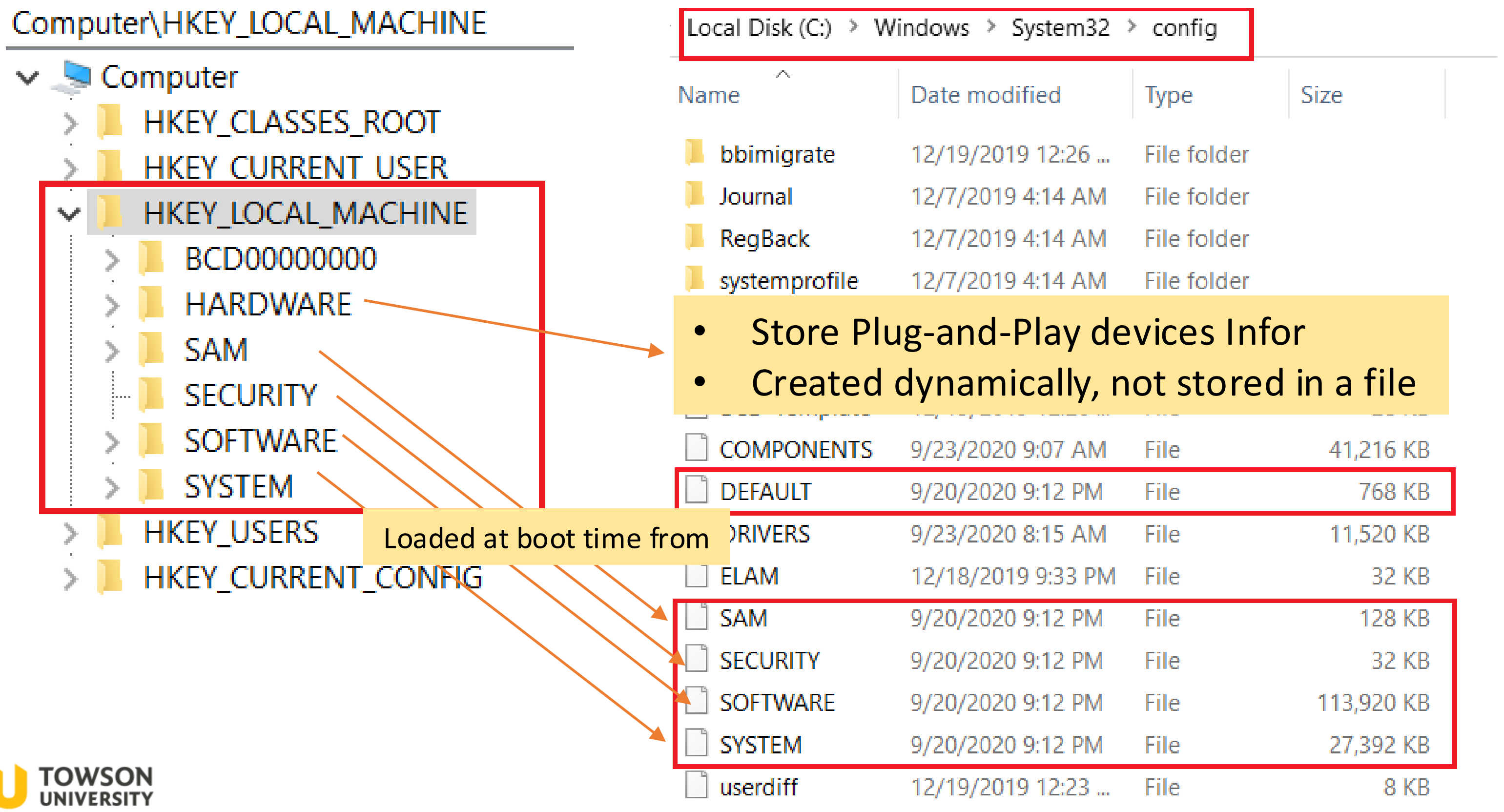
```
Computer
  HKEY_CLASSES_ROOT
  HKEY_CURRENT_USER
  HKEY_LOCAL_MACHINE
  HKEY_USERS
    .DEFAULT
      AppEvents
      Console
      Control Panel
      Environment
      EUDC
      Keyboard Layout
      Printers
      Software
      System
    S-1-5-18
    S-1-5-19
    S-1-5-20
    S-1-5-21-716072778-4209792547-2675333289-1001
    S-1-5-21-716072778-4209792547-2675333289-1001_Classes
  HKEY_CURRENT_CONFIG
```

contains the base settings for new users when they first logon,

- Contains information about all the users who logged in to the computer at some point
- When log on, the current logged in user profile is linked by HKCU
- Saved in each user's profile folder
  - C:\Users\IEUser\Ntuser.dat
  - C:\Users\ssh_Server\Ntuser.dat
  - C:\Users\Default User\Ntuser.dat

HKEY_LOCAL_MACHINE (HKLM)

- Contains computer hardware and software information
- Loaded at boot time from



```
Computer\HKEY_LOCAL_MACHINE
v Computer
  > HKEY_CLASSES_ROOT
  > HKEY_CURRENT_USER
  v HKEY_LOCAL_MACHINE
    > BCD00000000
    > HARDWARE
    > SAM
      SECURITY
    > SOFTWARE
    > SYSTEM
  > HKEY_USERS
  > HKEY_CURRENT_CONFIG
```

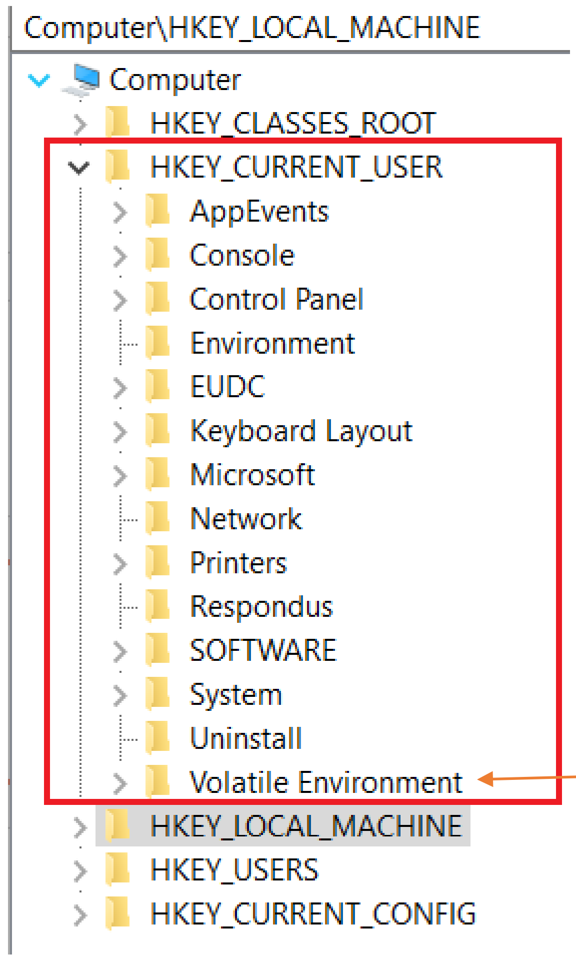
Local Disk (C:) > Windows > System32 > config

Name	Date modified	Type	Size
bbimigrate	12/19/2019 12:26 ...	File folder	
Journal	12/7/2019 4:14 AM	File folder	
RegBack	12/7/2019 4:14 AM	File folder	
systemprofile	12/7/2019 4:14 AM	File folder	
COMPONENTS	9/23/2020 9:07 AM	File	41,216 KB
DEFAULT	9/20/2020 9:12 PM	File	768 KB
DRIVERS	9/23/2020 8:15 AM	File	11,520 KB
ELAM	12/18/2019 9:33 PM	File	32 KB
SAM	9/20/2020 9:12 PM	File	128 KB
SECURITY	9/20/2020 9:12 PM	File	32 KB
SOFTWARE	9/20/2020 9:12 PM	File	113,920 KB
SYSTEM	9/20/2020 9:12 PM	File	27,392 KB
userdiff	12/19/2019 12:23 ...	File	8 KB

- Store Plug-and-Play devices Infor
- Created dynamically, not stored in a file

Loaded at boot time from

HKEY_CURRENT_USER (HKCU)

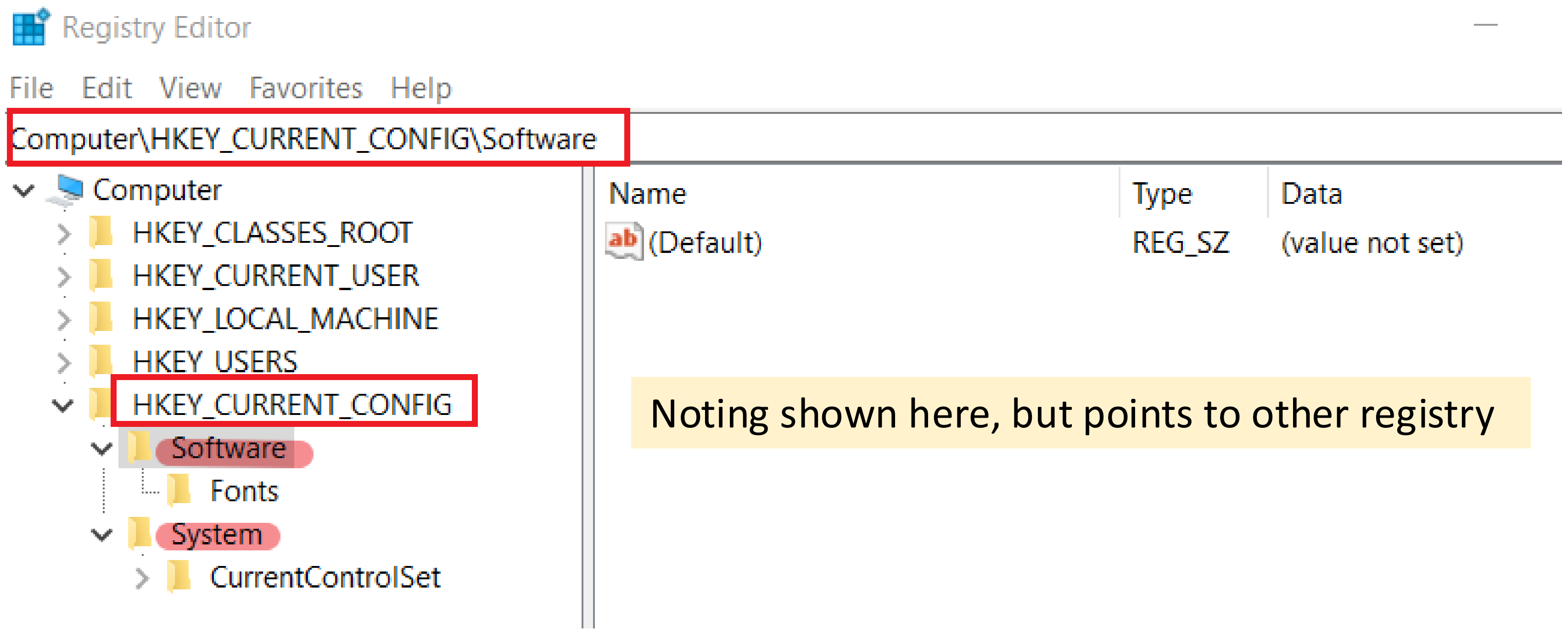


- Does not contain any data
  - A link to the subkey of HKEY_USERS
- Stores settings for currently logged-in use
  - Unloaded when the user logs out
  - If no profile is available, built from the default user
- Control everything of the current logged-in user
  - Environment variables
  - Desktop settings,
  - Network connections, printers,
  - Application preferences.
  - Keyboard layout
  - Current logged user information

Treasure for investigators

HKEY_CURRENT_CONFIG (HKCC)

- It doesn't store any information itself but instead acts as a pointer, or a shortcut, to a registry key



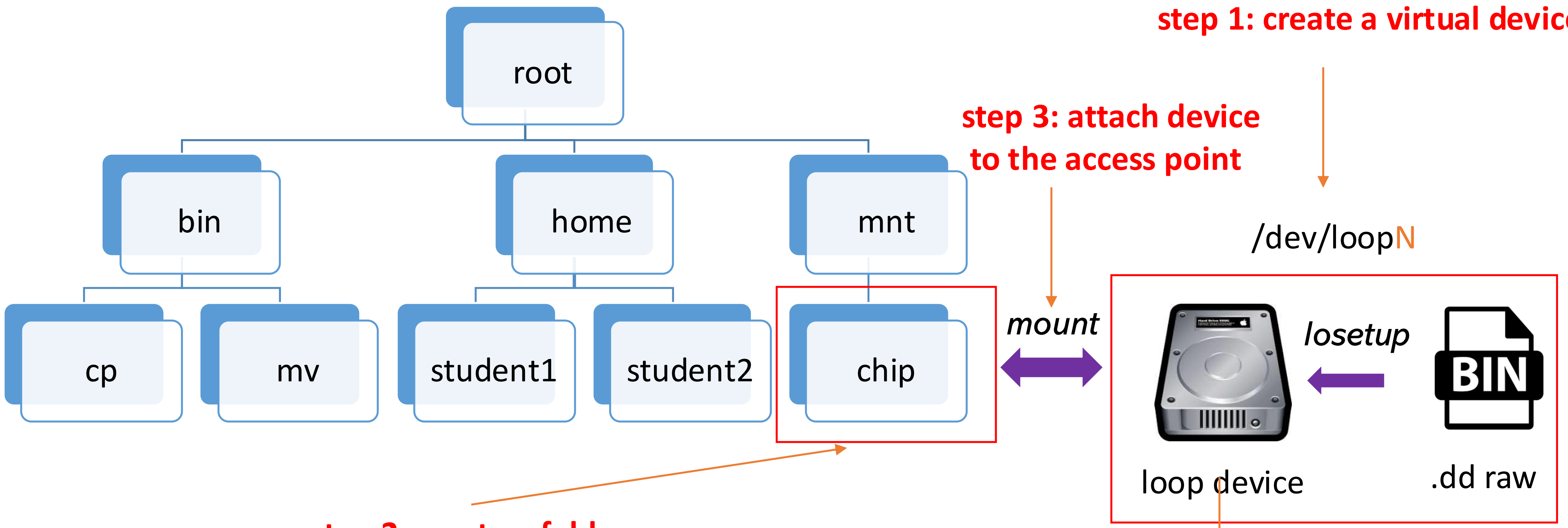
Computer\HKEY_CURRENT_CONFIG\Software

```
Computer
  HKEY_CLASSES_ROOT
  HKEY_CURRENT_USER
  HKEY_LOCAL_MACHINE
  HKEY_USERS
  HKEY_CURRENT_CONFIG
    Software
      Fonts
    System
      CurrentControlSet
```

Name	Type	Data
(Default)	REG_SZ	(value not set)

Noting shown here, but points to other registry

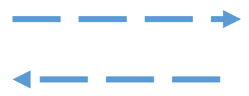
4.2 How to access registry files in dd image ? the easiest way: **Mounting**



step 2: create a folder as an access point (to a device)

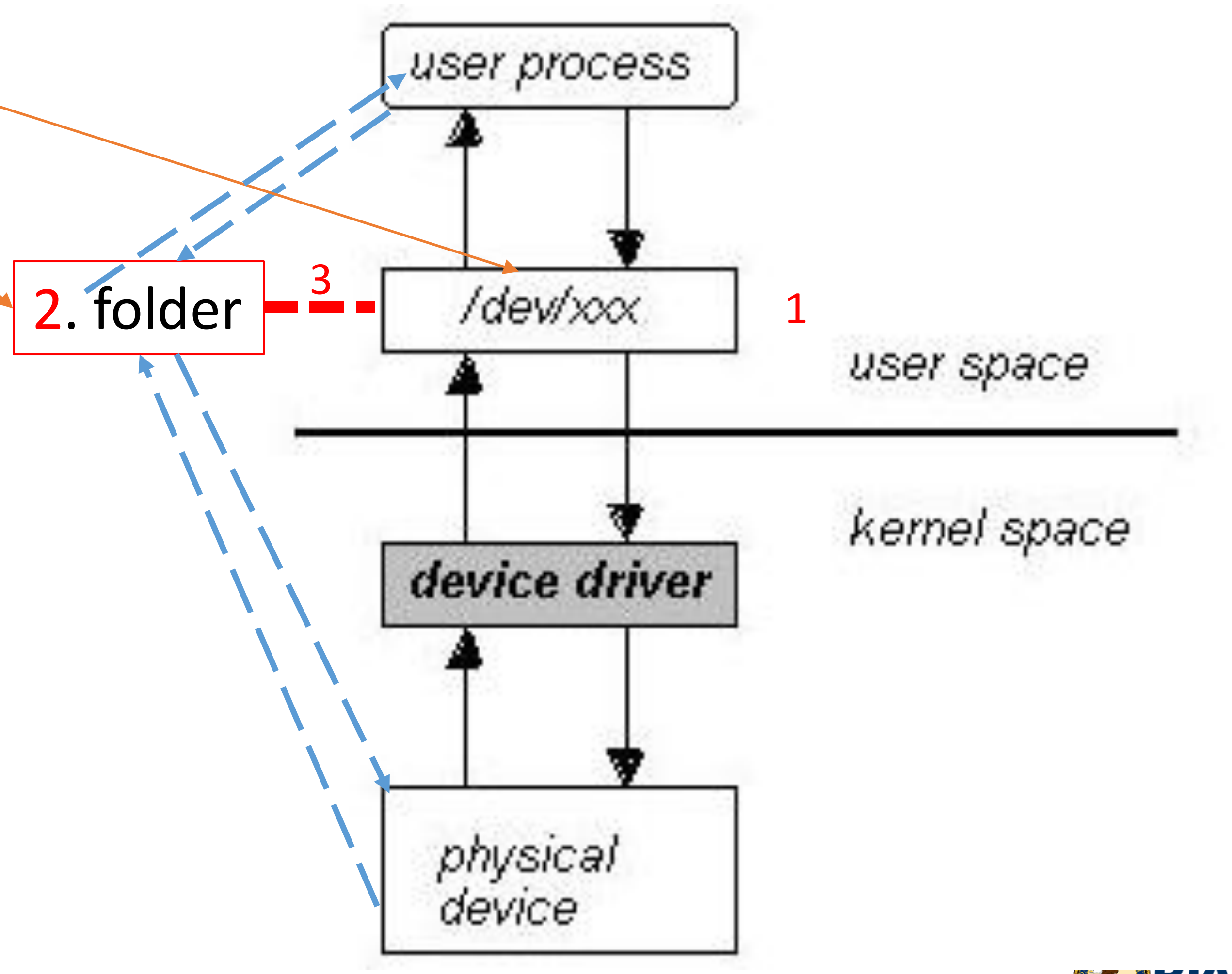
a virtual device in Unix-like systems that allows a file to be accessed as if it were a physical storage device, such as a hard drive or USB drive.

Technical view of **Mounting**



Users access a device through a folder

1. Set up a loop device
2. Create a mounting point
3. Attach the loop device to the mount point

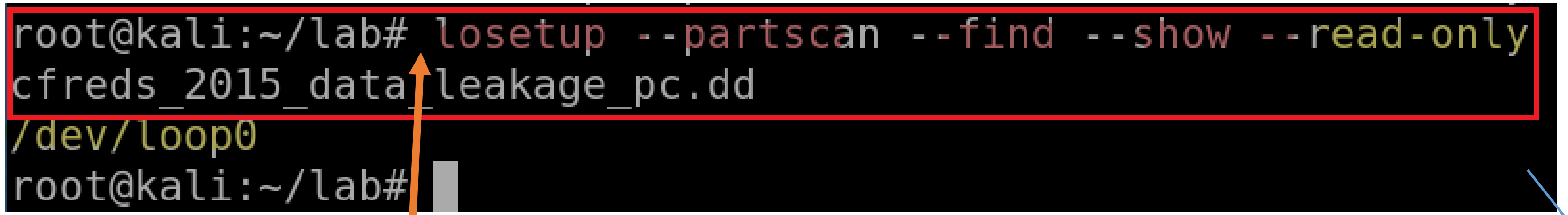


Mounting options: auto-mounting vs. manually mounting process

If your VM supports auto-mounting

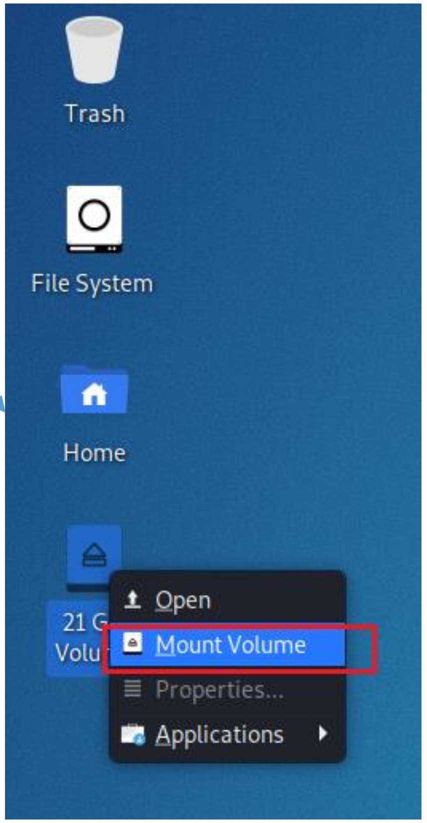
--**partscan**: scan the partition table on a newly created loop device
--**find**: find the first unused loop device.
--**show**: print device name
--**read-only**: setup read-only loop device.

Set up loop device (a pseudo-**device** that makes a file accessible as a block **device**)

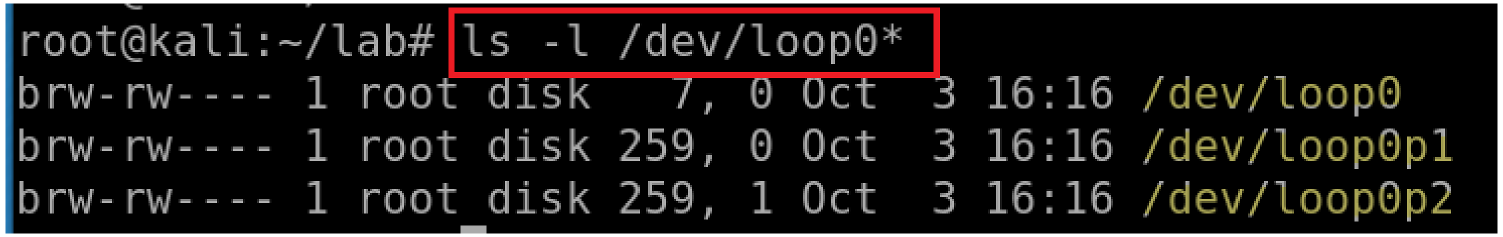


```
root@kali:~/lab# losetup --partscan --find --show --read-only
cfreds_2015_data_leakage_pc.dd
/dev/loop0
root@kali:~/lab#
```

Try to add *sudo* to get admin privilege if above cmd doesn't work.
Otherwise, you have to manually mount the image

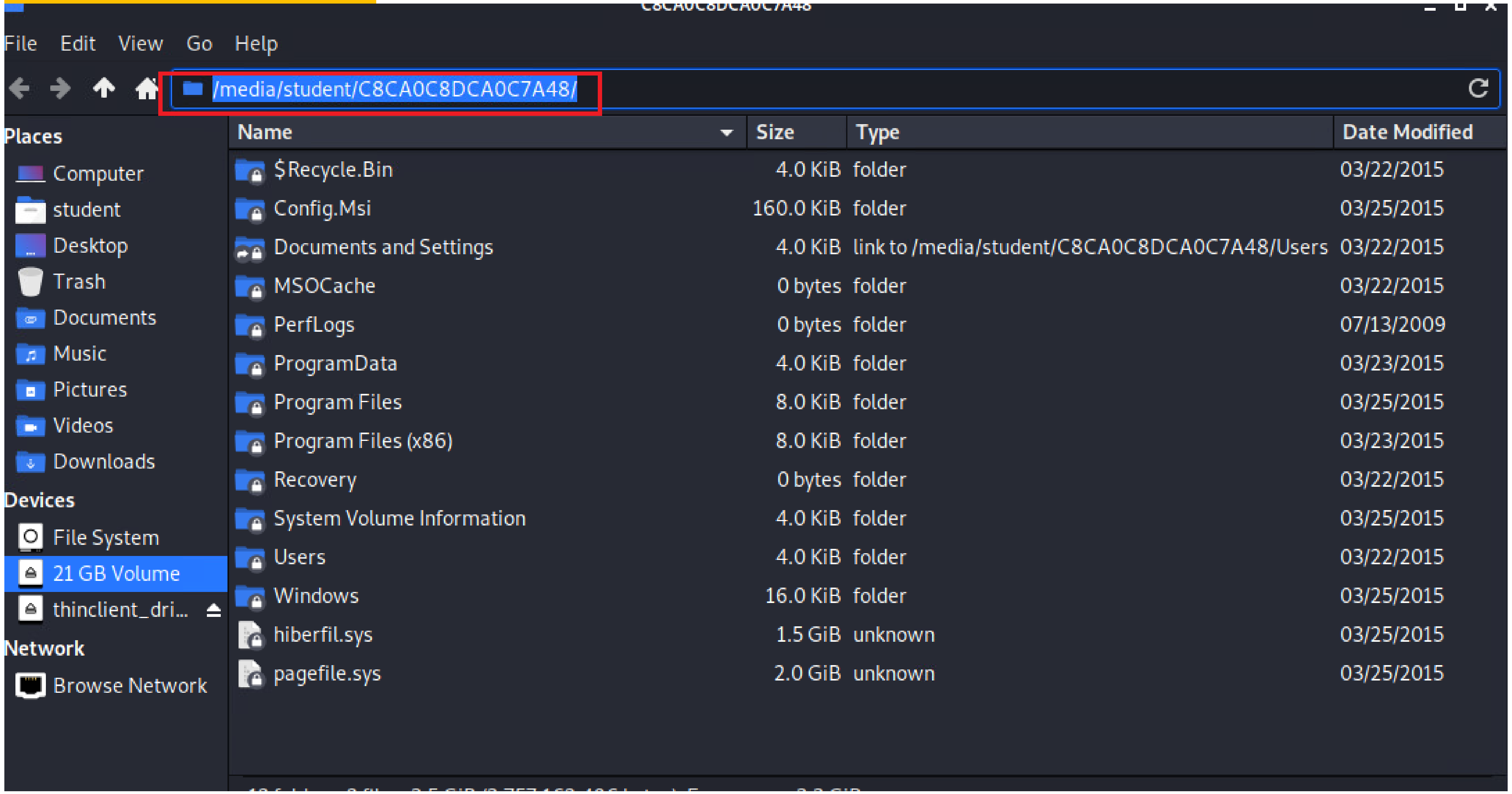


Show disk (loop0), partition 1 (loop0p1), partition 2 (loop0p2)



```
root@kali:~/lab# ls -l /dev/loop0*
brw-rw---- 1 root disk   7, 0 Oct  3 16:16 /dev/loop0
brw-rw---- 1 root disk 259, 0 Oct  3 16:16 /dev/loop0p1
brw-rw---- 1 root disk 259, 1 Oct  3 16:16 /dev/loop0p2
```

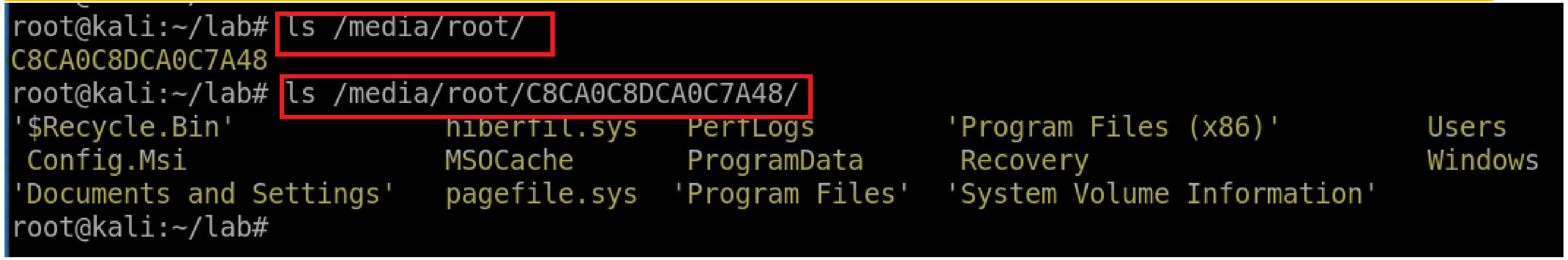
Show mounted image



/media/student/C8CA0C8DCA0C7A48/

Name	Size	Type	Date Modified
$Recycle.Bin	4.0 KiB	folder	03/22/2015
Config.Msi	160.0 KiB	folder	03/25/2015
Documents and Settings	4.0 KiB	link to /media/student/C8CA0C8DCA0C7A48/Users	03/22/2015
MSOCache	0 bytes	folder	03/22/2015
PerfLogs	0 bytes	folder	07/13/2009
ProgramData	4.0 KiB	folder	03/23/2015
Program Files	8.0 KiB	folder	03/25/2015
Program Files (x86)	8.0 KiB	folder	03/23/2015
Recovery	0 bytes	folder	03/22/2015
System Volume Information	4.0 KiB	folder	03/25/2015
Users	4.0 KiB	folder	03/22/2015
Windows	16.0 KiB	folder	03/25/2015
hiberfil.sys	1.5 GiB	unknown	03/25/2015
pagefile.sys	2.0 GiB	unknown	03/25/2015

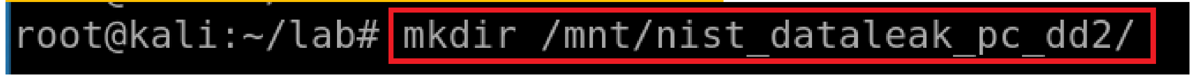
Kali 2020 automatically create the mounting point ***/media/YOUR_Account/CB…48.*** The partition 2 is mounted to the mounting point



```
root@kali:~/lab# ls /media/root/
C8CA0C8DCA0C7A48
root@kali:~/lab# ls /media/root/C8CA0C8DCA0C7A48/
'$Recycle.Bin'              hiberfil.sys   PerfLogs         'Program Files (x86)'          Users
 Config.Msi                 MSOCache       ProgramData       Recovery                      Windows
'Documents and Settings'    pagefile.sys  'Program Files'   'System Volume Information'
root@kali:~/lab#
```

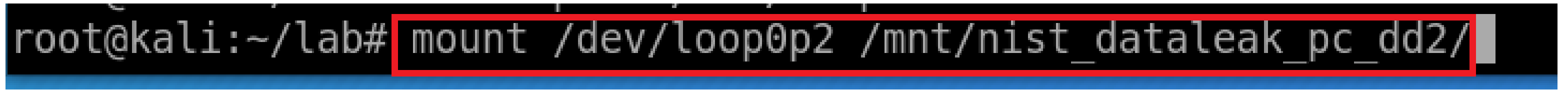
★ If your VM doesn't support auto-mounting

Create a folder as the mount point



```
root@kali:~/lab# mkdir /mnt/nist_dataleak_pc_dd2/
```

Mount partition 2 to the mounting point

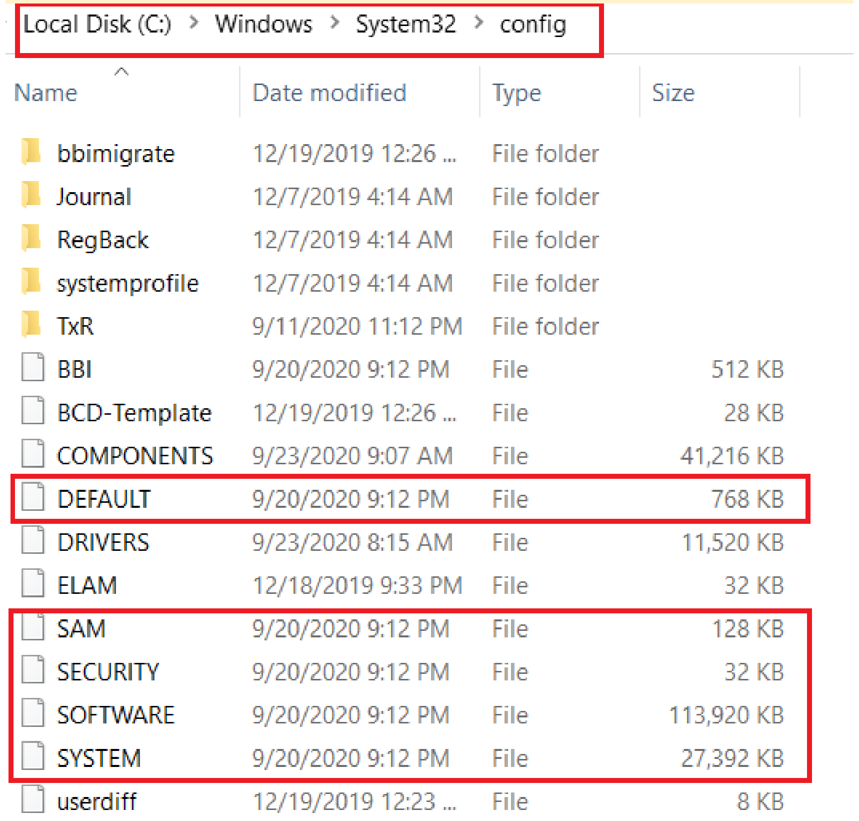


```
root@kali:~/lab# mount /dev/loop0p2 /mnt/nist_dataleak_pc_dd2/
```

Unmount command: *umount /mnt/nist_dataleak_pc_dd2/*

4.3 How to copy registry files for future forensic analysis?

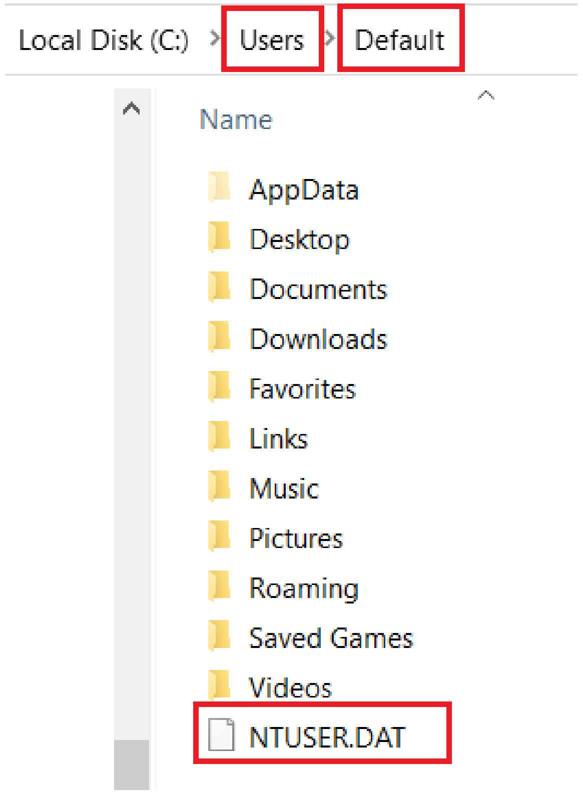
Files contain HKEY_LOCAL_MACHINE



Local Disk (C:) › Windows › System32 › config

Name	Date modified	Type	Size
bbimigrate	12/19/2019 12:26 ...	File folder	
Journal	12/7/2019 4:14 AM	File folder	
RegBack	12/7/2019 4:14 AM	File folder	
systemprofile	12/7/2019 4:14 AM	File folder	
TxR	9/11/2020 11:12 PM	File folder	
BBI	9/20/2020 9:12 PM	File	512 KB
BCD-Template	12/19/2019 12:26 ...	File	28 KB
COMPONENTS	9/23/2020 9:07 AM	File	41,216 KB
DEFAULT	9/20/2020 9:12 PM	File	768 KB
DRIVERS	9/23/2020 8:15 AM	File	11,520 KB
ELAM	12/18/2019 9:33 PM	File	32 KB
SAM	9/20/2020 9:12 PM	File	128 KB
SECURITY	9/20/2020 9:12 PM	File	32 KB
SOFTWARE	9/20/2020 9:12 PM	File	113,920 KB
SYSTEM	9/20/2020 9:12 PM	File	27,392 KB
userdiff	12/19/2019 12:23 ...	File	8 KB

Files contains HKEY_USERS

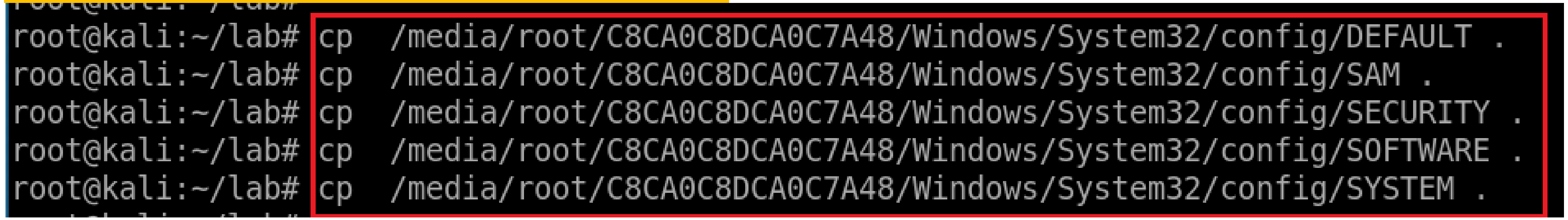


Local Disk (C:) › Users › Default

Name

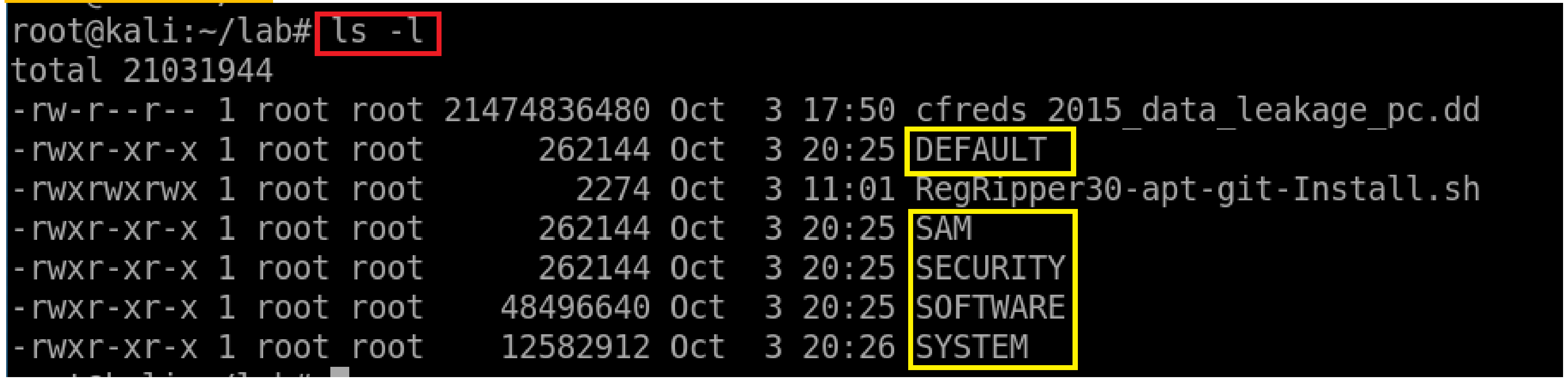
- AppData
- Desktop
- Documents
- Downloads
- Favorites
- Links
- Music
- Pictures
- Roaming
- Saved Games
- Videos
- NTUSER.DAT

Copy HKEY_LOCAL_MACHINE (Hive) files to \lab



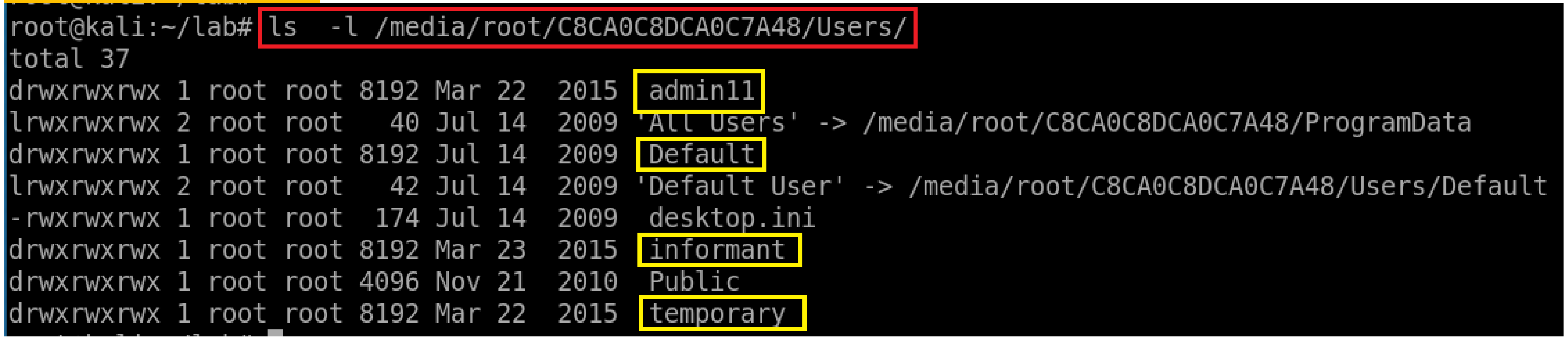
```
root@kali:~/lab# cp  /media/root/C8CA0C8DCA0C7A48/Windows/System32/config/DEFAULT .
root@kali:~/lab# cp  /media/root/C8CA0C8DCA0C7A48/Windows/System32/config/SAM .
root@kali:~/lab# cp  /media/root/C8CA0C8DCA0C7A48/Windows/System32/config/SECURITY .
root@kali:~/lab# cp  /media/root/C8CA0C8DCA0C7A48/Windows/System32/config/SOFTWARE .
root@kali:~/lab# cp  /media/root/C8CA0C8DCA0C7A48/Windows/System32/config/SYSTEM .
```

Verify five files



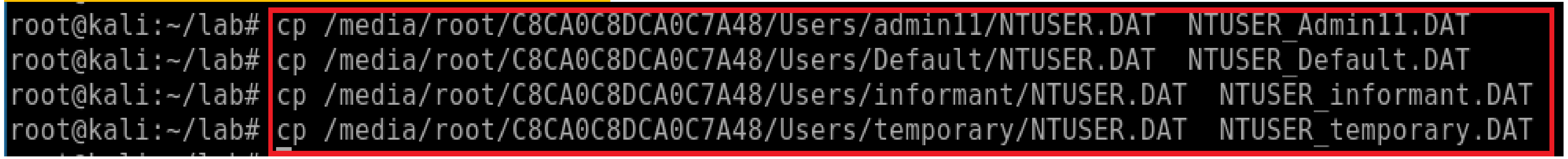
```
root@kali:~/lab# ls -l
total 21031944
-rw-r--r-- 1 root root 21474836480 Oct  3 17:50 cfreds_2015_data_leakage_pc.dd
-rwxr-xr-x 1 root root      262144 Oct  3 20:25 DEFAULT
-rwxrwxrwx 1 root root        2274 Oct  3 11:01 RegRipper30-apt-git-Install.sh
-rwxr-xr-x 1 root root      262144 Oct  3 20:25 SAM
-rwxr-xr-x 1 root root      262144 Oct  3 20:25 SECURITY
-rwxr-xr-x 1 root root    48496640 Oct  3 20:25 SOFTWARE
-rwxr-xr-x 1 root root    12582912 Oct  3 20:26 SYSTEM
```

Find users in the PC



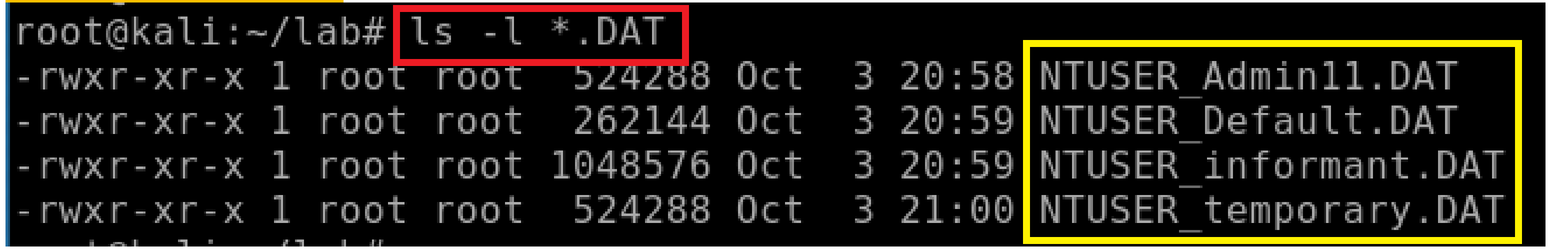
```
root@kali:~/lab# ls  -l /media/root/C8CA0C8DCA0C7A48/Users/
total 37
drwxrwxrwx 1 root root 8192 Mar 22  2015  admin11
lrwxrwxrwx 2 root root   40 Jul 14  2009 'All Users' -> /media/root/C8CA0C8DCA0C7A48/ProgramData
drwxrwxrwx 1 root root 8192 Jul 14  2009  Default
lrwxrwxrwx 2 root root   42 Jul 14  2009 'Default User' -> /media/root/C8CA0C8DCA0C7A48/Users/Default
-rwxrwxrwx 1 root root  174 Jul 14  2009  desktop.ini
drwxrwxrwx 1 root root 8192 Mar 23  2015  informant
drwxrwxrwx 1 root root 4096 Nov 21  2010  Public
drwxrwxrwx 1 root root 8192 Mar 22  2015  temporary
```

Find copy HKEY_USERS hive files to \lab



```
root@kali:~/lab# cp /media/root/C8CA0C8DCA0C7A48/Users/admin11/NTUSER.DAT  NTUSER_Admin11.DAT
root@kali:~/lab# cp /media/root/C8CA0C8DCA0C7A48/Users/Default/NTUSER.DAT  NTUSER_Default.DAT
root@kali:~/lab# cp /media/root/C8CA0C8DCA0C7A48/Users/informant/NTUSER.DAT  NTUSER_informant.DAT
root@kali:~/lab# cp /media/root/C8CA0C8DCA0C7A48/Users/temporary/NTUSER.DAT  NTUSER_temporary.DAT
```

Verify four files



```
root@kali:~/lab# ls -l *.DAT
-rwxr-xr-x 1 root root  524288 Oct  3 20:58 NTUSER_Admin11.DAT
-rwxr-xr-x 1 root root  262144 Oct  3 20:59 NTUSER_Default.DAT
-rwxr-xr-x 1 root root 1048576 Oct  3 20:59 NTUSER_informant.DAT
-rwxr-xr-x 1 root root  524288 Oct  3 21:00 NTUSER_temporary.DAT
```

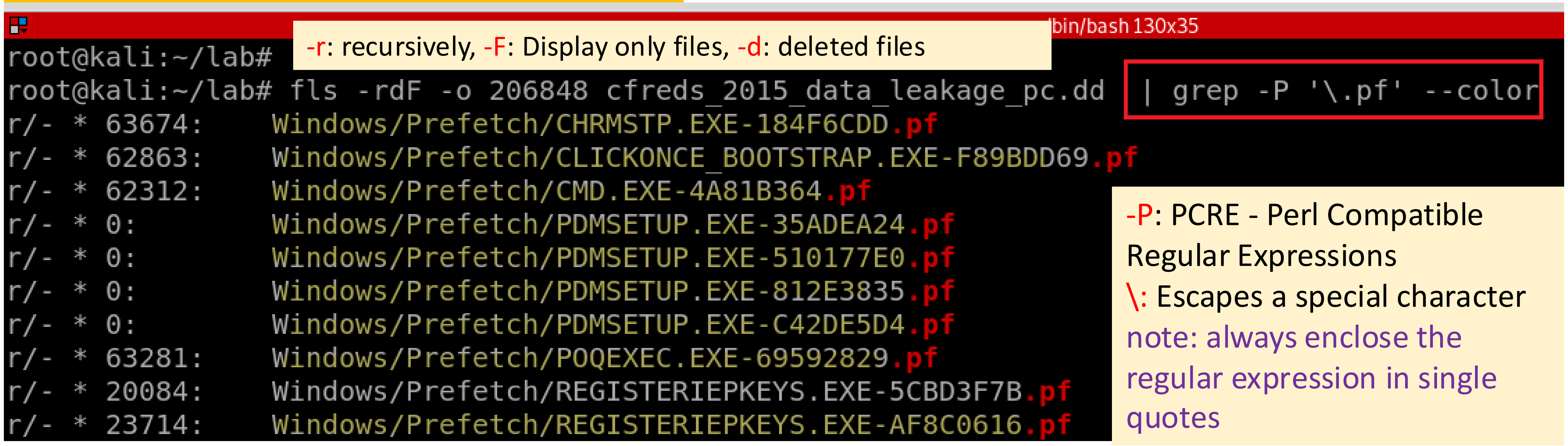
5. Extract prefetch event log files from a DD image

monitoring program execution

Prefetch introduction

- What is *Cache*?
  - Is type memory that stores data to improve performance
- What is *Prefetching*?
  - The loading of a resource (instructions, data) to cache before it is required
- Which resource are chosen for prefetching?
  - Chosen based on the user's daily behavior
  - E.g., the most used resource
- *Prefetch* log can be used for forensic analysis
  - E.g., monitoring program execution

Check *.pf* files in a DD image



```
root@kali:~/lab#
root@kali:~/lab# fls -rdF -o 206848 cfreds_2015_data_leakage_pc.dd | grep -P '\.pf' --color
r/- * 63674:    Windows/Prefetch/CHRMSTP.EXE-184F6CDD.pf
r/- * 62863:    Windows/Prefetch/CLICKONCE_BOOTSTRAP.EXE-F89BDD69.pf
r/- * 62312:    Windows/Prefetch/CMD.EXE-4A81B364.pf
r/- * 0:        Windows/Prefetch/PDMSETUP.EXE-35ADEA24.pf
r/- * 0:        Windows/Prefetch/PDMSETUP.EXE-510177E0.pf
r/- * 0:        Windows/Prefetch/PDMSETUP.EXE-812E3835.pf
r/- * 0:        Windows/Prefetch/PDMSETUP.EXE-C42DE5D4.pf
r/- * 63281:    Windows/Prefetch/POQEXEC.EXE-69592829.pf
r/- * 20084:    Windows/Prefetch/REGISTERIEPKEYS.EXE-5CBD3F7B.pf
r/- * 23714:    Windows/Prefetch/REGISTERIEPKEYS.EXE-AF8C0616.pf
```

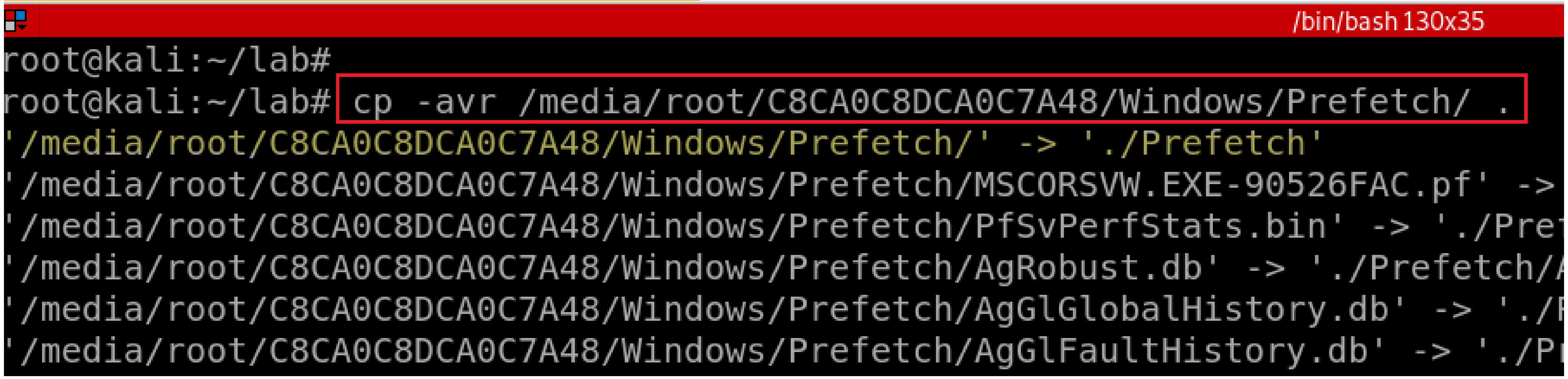
-r: recursively, -F: Display only files, -d: deleted files

-P: PCRE - Perl Compatible Regular Expressions
\: Escapes a special character
note: always enclose the regular expression in single quotes

- grep supports three regular expression syntaxes, Basic, Extended, and Perl-compatible.
- A regular expression or regex is a pattern that matches a set of strings

https://www.debuggex.com/cheatsheet/regex/pcre

Copy *Prefetch* folder to current folder



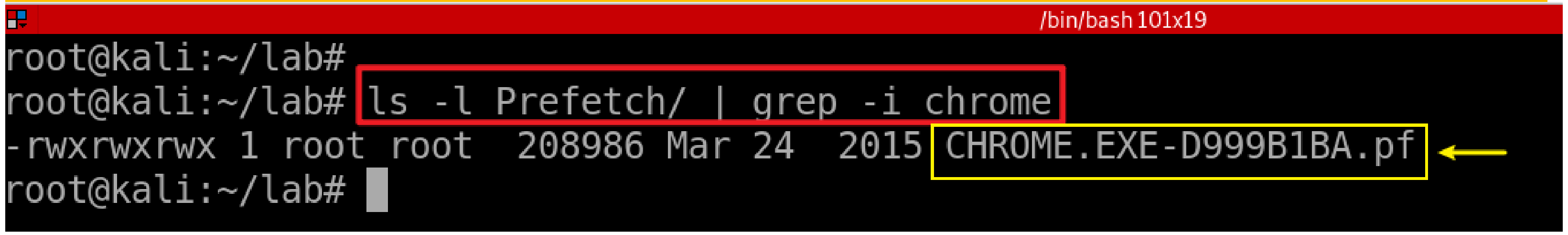
```
/bin/bash 130x35
root@kali:~/lab#
root@kali:~/lab# cp -avr /media/root/C8CA0C8DCA0C7A48/Windows/Prefetch/ .
'/media/root/C8CA0C8DCA0C7A48/Windows/Prefetch/' -> './Prefetch'
'/media/root/C8CA0C8DCA0C7A48/Windows/Prefetch/MSCORSVW.EXE-90526FAC.pf' ->
'/media/root/C8CA0C8DCA0C7A48/Windows/Prefetch/PfSvPerfStats.bin' -> './Pre
'/media/root/C8CA0C8DCA0C7A48/Windows/Prefetch/AgRobust.db' -> './Prefetch/
'/media/root/C8CA0C8DCA0C7A48/Windows/Prefetch/AgGlGlobalHistory.db' -> './
'/media/root/C8CA0C8DCA0C7A48/Windows/Prefetch/AgGlFaultHistory.db' -> './P
```

-a : Preserve the specified attributes such as directory a file mode, ownership, timestamps, if possible additional attributes: context, links, xattr, all.

-v : Verbose output.

-r : Copy directories recursively.

Verify and check .pf of chrom.exe is in *./Prefetch* folder


/bin/bash 101x19

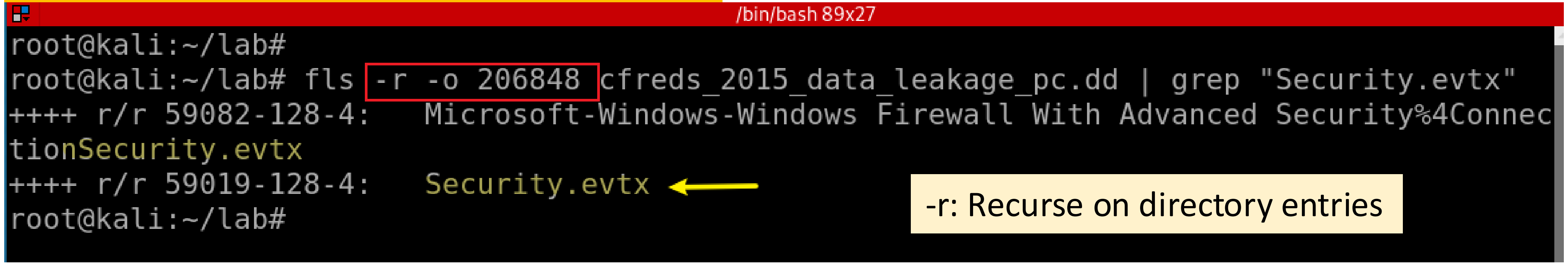
```
root@kali:~/lab#
root@kali:~/lab# ls -l Prefetch/ | grep -i chrome
-rwxrwxrwx 1 root root  208986 Mar 24  2015 CHROME.EXE-D999B1BA.pf
root@kali:~/lab#
```

6. Extract security event log files from the DD Image

Event log files overview

- Windows OS records various events for debugging
  - Created by the Windows 7 Event Viewer (*eventvwr.msc*)
- Event logs has file extension *.evtx*
  - Contains a list of events
  - Saved in a proprietary binary format
  - Only can be viewed within the Event Viewer

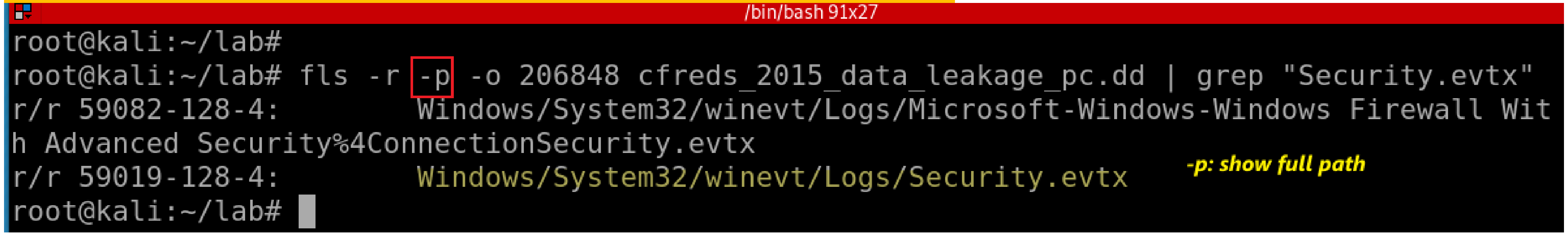
Search for "*Security.evtx*" from the DD image



```
/bin/bash 89x27
root@kali:~/lab#
root@kali:~/lab# fls -r -o 206848 cfreds_2015_data_leakage_pc.dd | grep "Security.evtx"
++++ r/r 59082-128-4:   Microsoft-Windows-Windows Firewall With Advanced Security%4Connec
tionSecurity.evtx
++++ r/r 59019-128-4:   Security.evtx
root@kali:~/lab#
```

-r: Recurse on directory entries

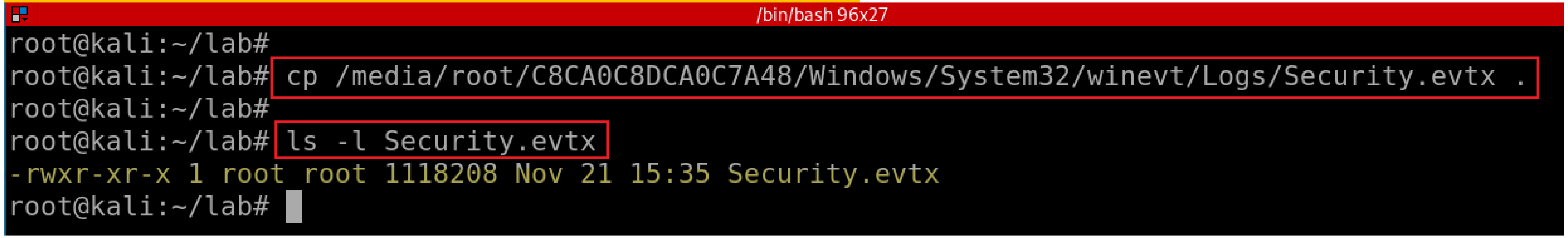
Search for "*Security.evtx*" from the DD image and show full path



```
/bin/bash 91x27
root@kali:~/lab#
root@kali:~/lab# fls -r -p -o 206848 cfreds_2015_data_leakage_pc.dd | grep "Security.evtx"
r/r 59082-128-4:        Windows/System32/winevt/Logs/Microsoft-Windows-Windows Firewall Wit
h Advanced Security%4ConnectionSecurity.evtx
r/r 59019-128-4:        Windows/System32/winevt/Logs/Security.evtx
root@kali:~/lab#
```

-p: show full path

Copy "*Security.evtx*" from the DD image to the *lab* directory

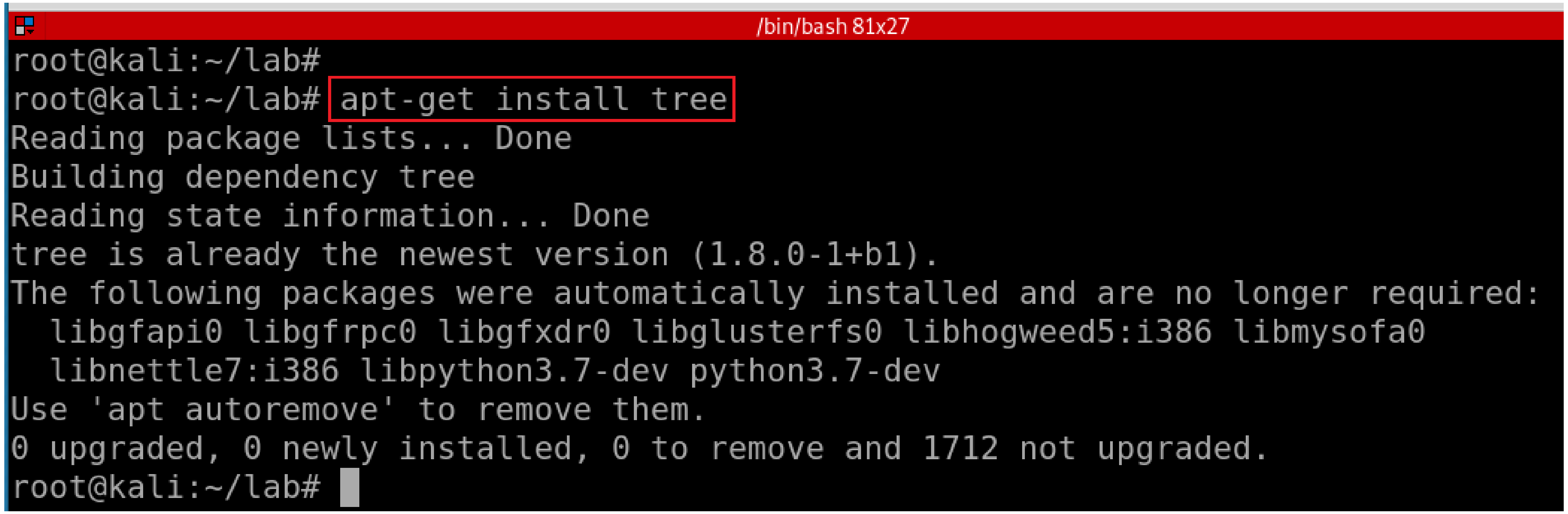


/bin/bash 96x27

```
root@kali:~/lab#
root@kali:~/lab# cp /media/root/C8CA0C8DCA0C7A48/Windows/System32/winevt/Logs/Security.evtx .
root@kali:~/lab#
root@kali:~/lab# ls -l Security.evtx
-rwxr-xr-x 1 root root 1118208 Nov 21 15:35 Security.evtx
root@kali:~/lab#
```

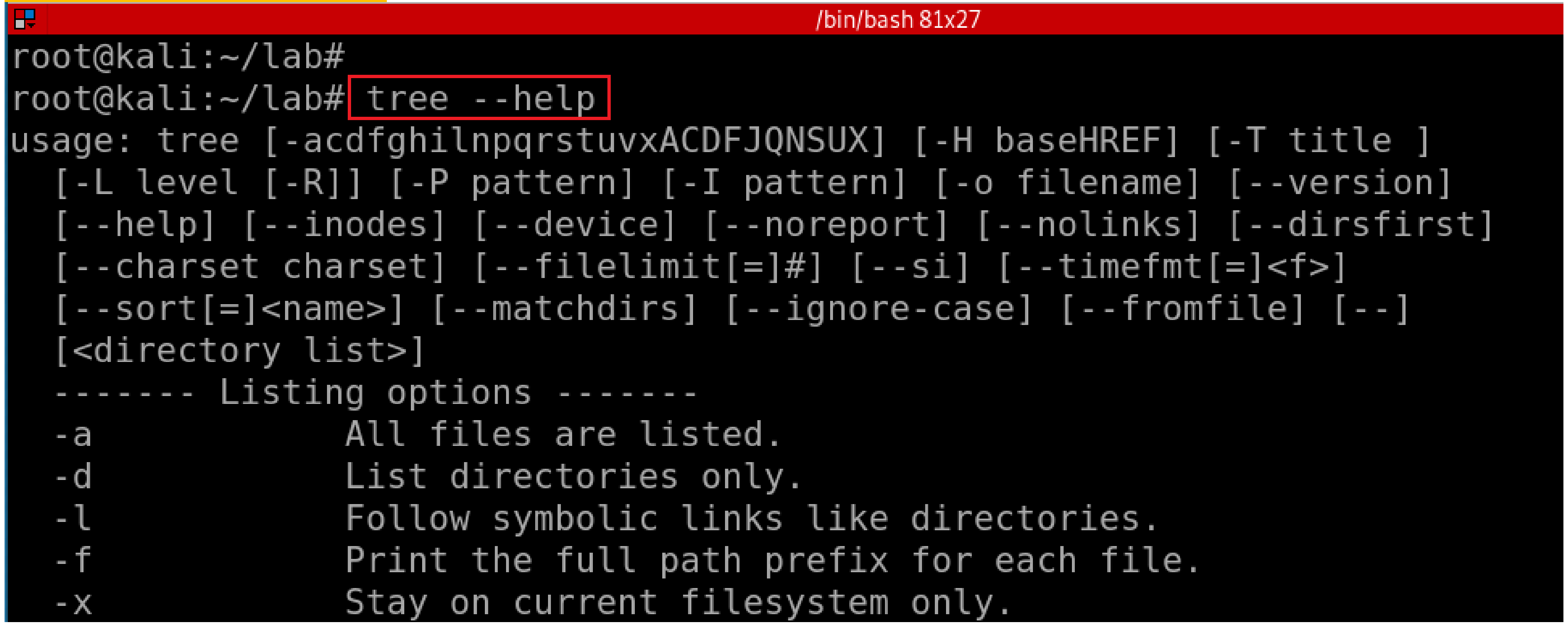
7. Install software

7.1 Install *tree*



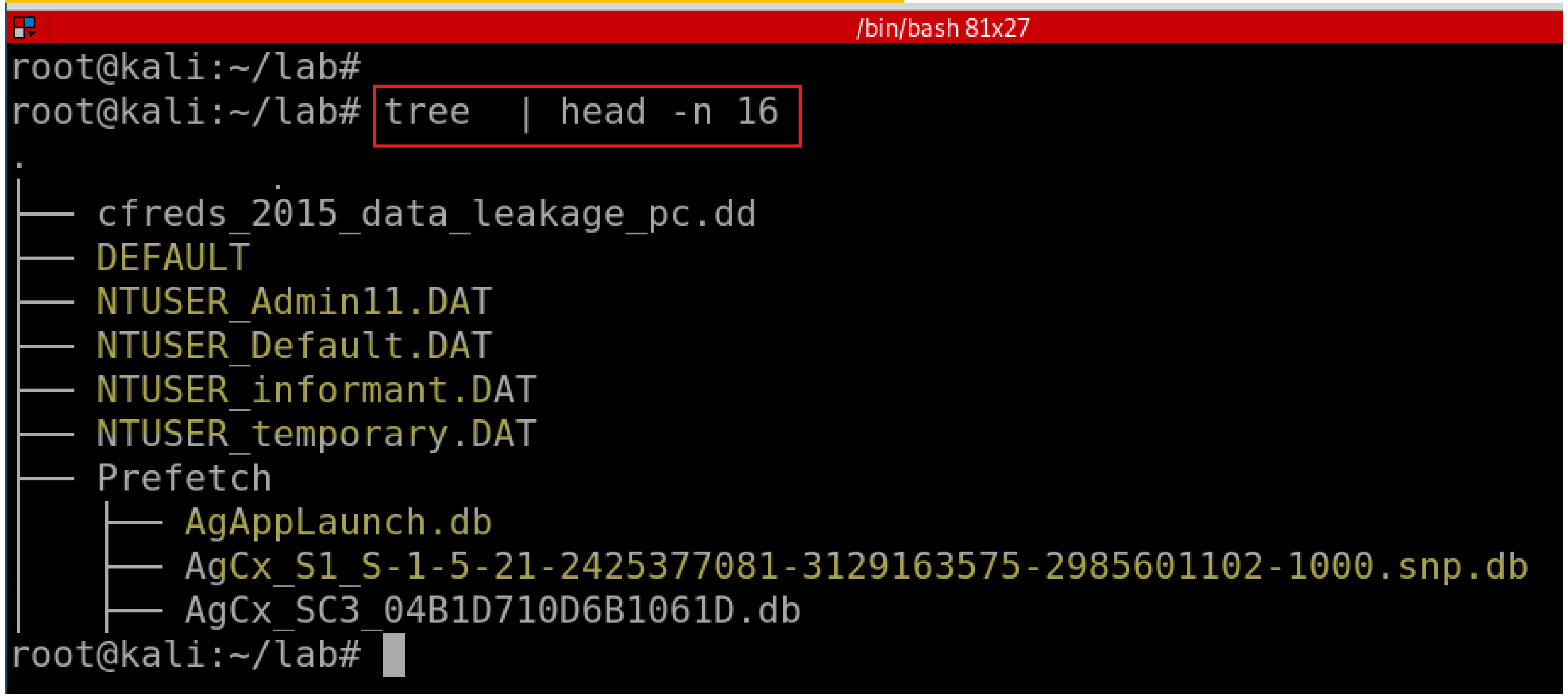
```
root@kali:~/lab#
root@kali:~/lab# apt-get install tree
Reading package lists... Done
Building dependency tree
Reading state information... Done
tree is already the newest version (1.8.0-1+b1).
The following packages were automatically installed and are no longer required:
  libgfapi0 libgfrpc0 libgfxdr0 libglusterfs0 libhogweed5:i386 libmysofa0
  libnettle7:i386 libpython3.7-dev python3.7-dev
Use 'apt autoremove' to remove them.
0 upgraded, 0 newly installed, 0 to remove and 1712 not upgraded.
root@kali:~/lab#
```

Show *tree* help menu



```
root@kali:~/lab#
root@kali:~/lab# tree --help
usage: tree [-acdfghilnpqrstuvxACDFJQNSUX] [-H baseHREF] [-T title ]
  [-L level [-R]] [-P pattern] [-I pattern] [-o filename] [--version]
  [--help] [--inodes] [--device] [--noreport] [--nolinks] [--dirsfirst]
  [--charset charset] [--filelimit[=]#] [--si] [--timefmt[=]<f>]
  [--sort[=]<name>] [--matchdirs] [--ignore-case] [--fromfile] [--]
  [<directory list>]
  ------- Listing options -------
  -a            All files are listed.
  -d            List directories only.
  -l            Follow symbolic links like directories.
  -f            Print the full path prefix for each file.
  -x            Stay on current filesystem only.
```

Show files of the current folder in a tree structure



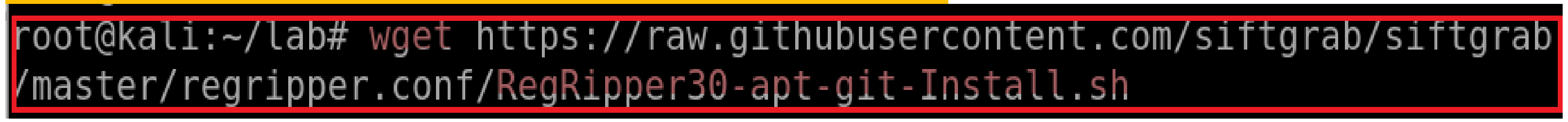
/bin/bash 81x27

```
root@kali:~/lab#
root@kali:~/lab# tree   | head -n 16
.
├── cfreds_2015_data_leakage_pc.dd
├── DEFAULT
├── NTUSER_Admin11.DAT
├── NTUSER_Default.DAT
├── NTUSER_informant.DAT
├── NTUSER_temporary.DAT
├── Prefetch
│   ├── AgAppLaunch.db
│   ├── AgCx_S1_S-1-5-21-2425377081-3129163575-2985601102-1000.snp.db
│   ├── AgCx_SC3_04B1D710D6B1061D.db
root@kali:~/lab#
```

7.2 Install *RegRipper* 3.0

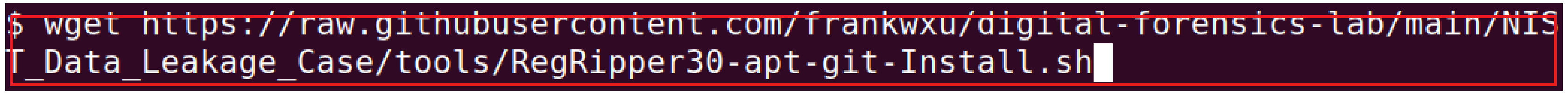
- *RegRipper* is a software tool to extract/parse information (keys, values, data) from the Registry
  - Open-source software (Windows and Linux)
  - Written in Perl
  - Consists of a framework that executes plugins
- Two basic tools
  - A command line (CLI) tool called rip
  - Graphic user interface (GUI)

Download RegRipper 3.0 installation script



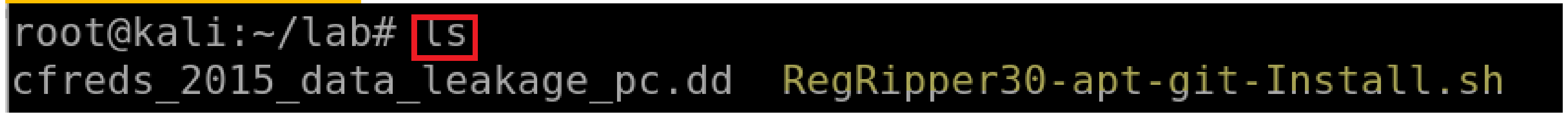
```
root@kali:~/lab# wget https://raw.githubusercontent.com/siftgrab/siftgrab
/master/regripper.conf/RegRipper30-apt-git-Install.sh
```

use the backup link if the original link does not work



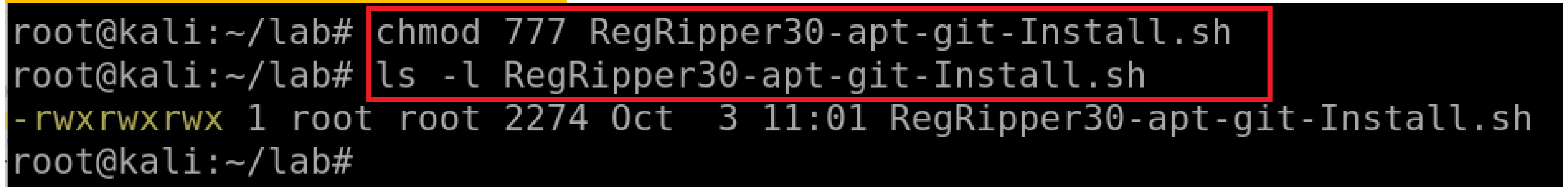
```
$ wget https://raw.githubusercontent.com/frankwxu/digital-forensics-lab/main/NIS
T_Data_Leakage_Case/tools/RegRipper30-apt-git-Install.sh
```

Verify the script



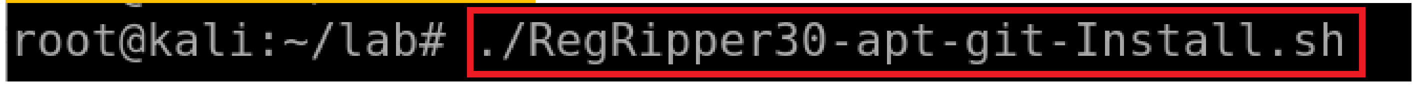
```
root@kali:~/lab# ls
cfreds_2015_data_leakage_pc.dd   RegRipper30-apt-git-Install.sh
```

Make the script executable



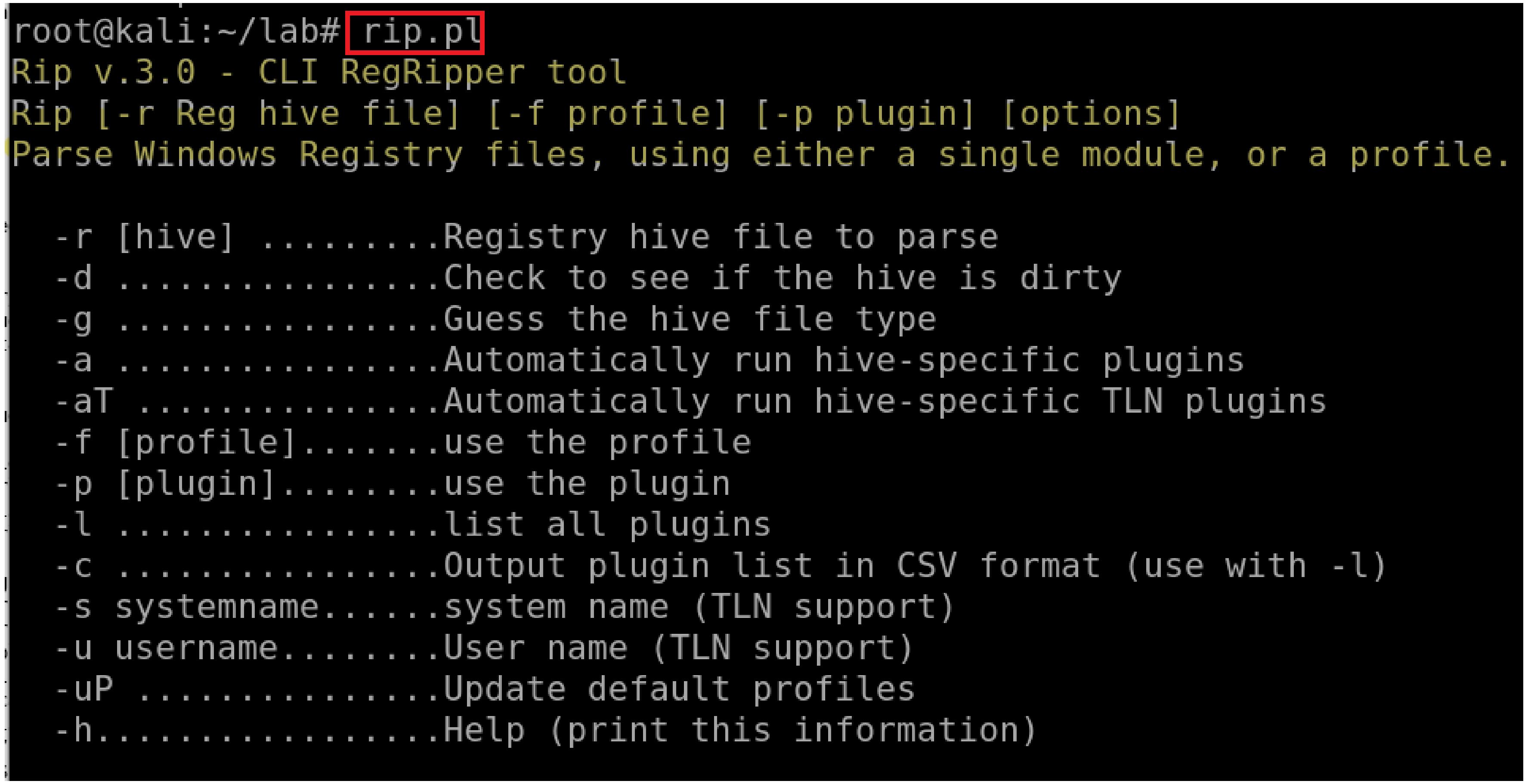
```
root@kali:~/lab# chmod 777 RegRipper30-apt-git-Install.sh
root@kali:~/lab# ls -l RegRipper30-apt-git-Install.sh
-rwxrwxrwx 1 root root 2274 Oct  3 11:01 RegRipper30-apt-git-Install.sh
root@kali:~/lab#
```

Run the script to install



```
root@kali:~/lab# ./RegRipper30-apt-git-Install.sh
```

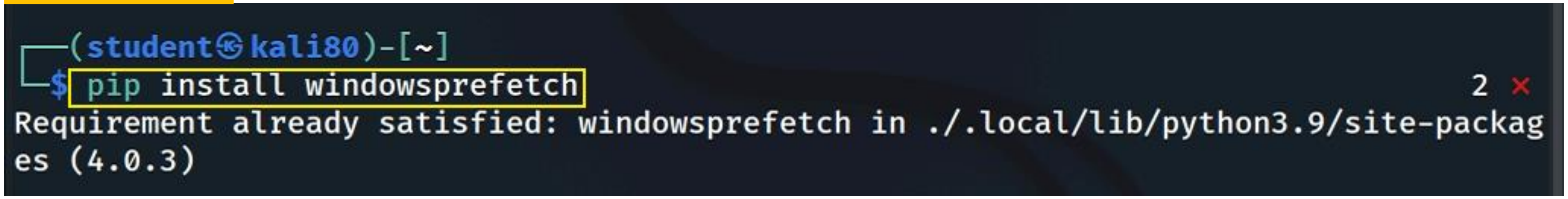
Test regripper 3.0



```
root@kali:~/lab# rip.pl
Rip v.3.0 - CLI RegRipper tool
Rip [-r Reg hive file] [-f profile] [-p plugin] [options]
Parse Windows Registry files, using either a single module, or a profile.

  -r [hive] .........Registry hive file to parse
  -d ................Check to see if the hive is dirty
  -g ................Guess the hive file type
  -a ................Automatically run hive-specific plugins
  -aT ...............Automatically run hive-specific TLN plugins
  -f [profile].......use the profile
  -p [plugin]........use the plugin
  -l ................list all plugins
  -c ................Output plugin list in CSV format (use with -l)
  -s systemname......system name (TLN support)
  -u username........User name (TLN support)
  -uP ...............Update default profiles
  -h.................Help (print this information)
```

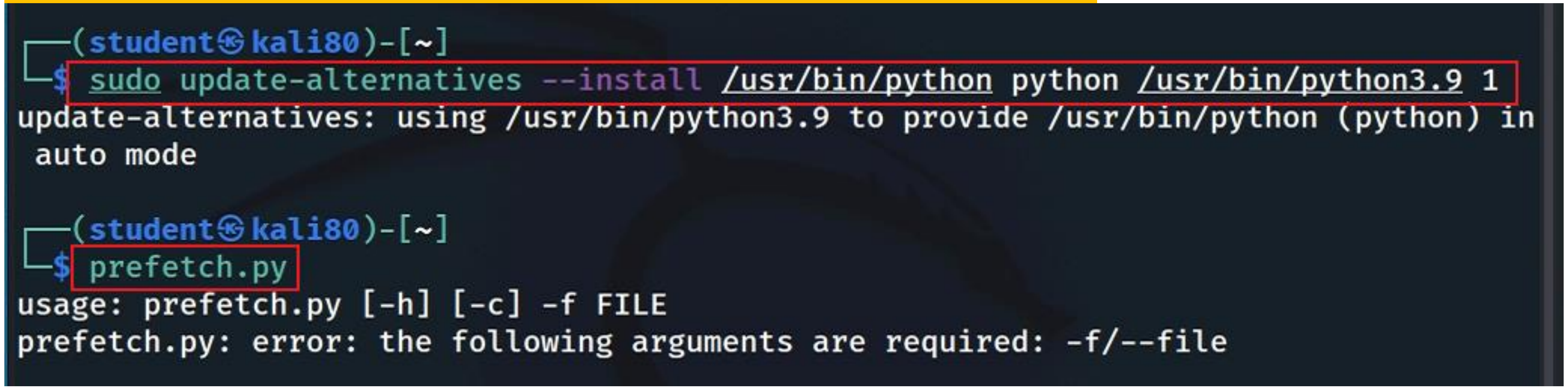
7.3 Install *Windows-Prefetch-Parser*

Install parser



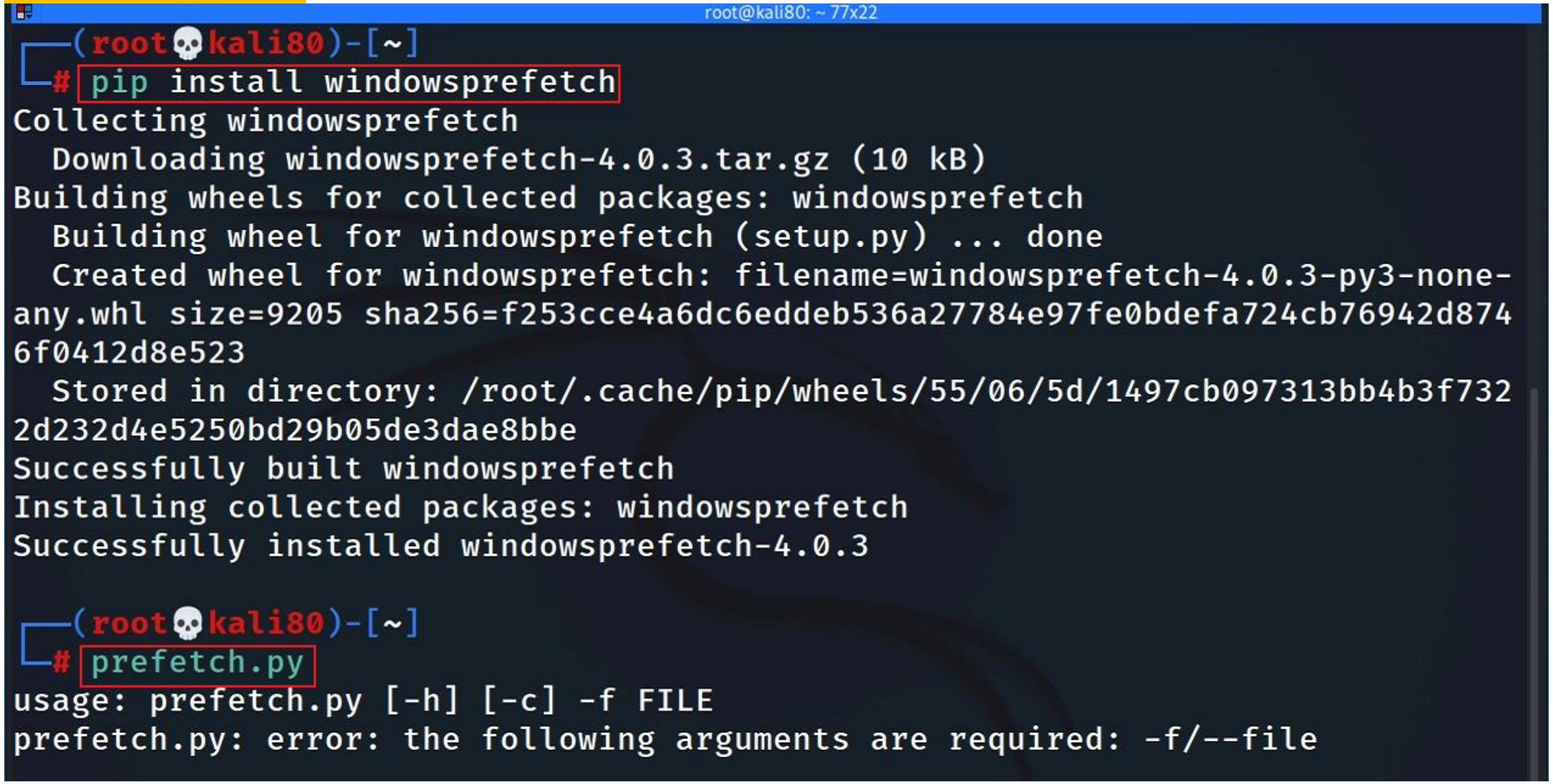
```
┌──(student㉿kali80)-[~]
└─$ pip install windowsprefetch                                              2 ✕
Requirement already satisfied: windowsprefetch in ./.local/lib/python3.9/site-packag
es (4.0.3)
```

Test if *prefetch* tool works (you can skip if your python version is 3.9)



```
┌──(student㉿kali80)-[~]
└─$ sudo update-alternatives --install /usr/bin/python python /usr/bin/python3.9 1
update-alternatives: using /usr/bin/python3.9 to provide /usr/bin/python (python) in
 auto mode

┌──(student㉿kali80)-[~]
└─$ prefetch.py
usage: prefetch.py [-h] [-c] -f FILE
prefetch.py: error: the following arguments are required: -f/--file
```

If you are the *root*



```
root@kali80: ~ 77x22
┌──(root💀kali80)-[~]
└─# pip install windowsprefetch
Collecting windowsprefetch
  Downloading windowsprefetch-4.0.3.tar.gz (10 kB)
Building wheels for collected packages: windowsprefetch
  Building wheel for windowsprefetch (setup.py) ... done
  Created wheel for windowsprefetch: filename=windowsprefetch-4.0.3-py3-none-
any.whl size=9205 sha256=f253cce4a6dc6eddeb536a27784e97fe0bdefa724cb76942d874
6f0412d8e523
  Stored in directory: /root/.cache/pip/wheels/55/06/5d/1497cb097313bb4b3f732
2d232d4e5250bd29b05de3dae8bbe
Successfully built windowsprefetch
Installing collected packages: windowsprefetch
Successfully installed windowsprefetch-4.0.3

┌──(root💀kali80)-[~]
└─# prefetch.py
usage: prefetch.py [-h] [-c] -f FILE
prefetch.py: error: the following arguments are required: -f/--file
```

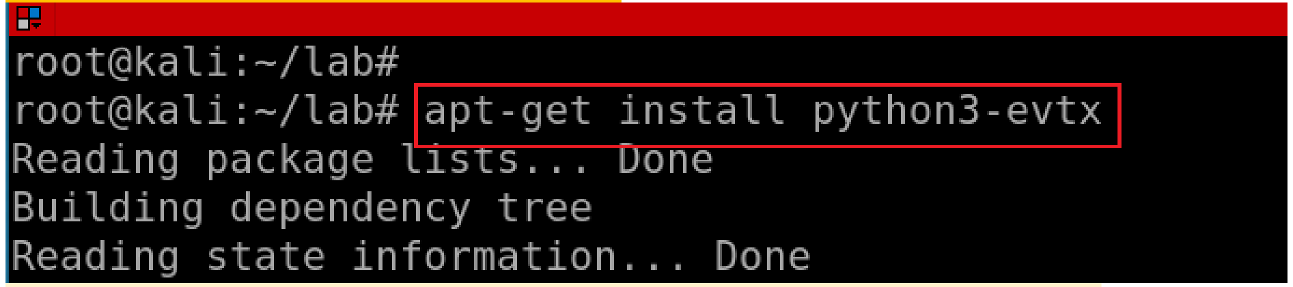
Last option: Directly build from GitHub

```
git clone https://github.com/PoorBillionaire/Windows-Prefetch-Parser.git
cd Windows-Prefetch-Parser
python3 setup.py build
python3 setup.py install
prefetch.py
```

7.4 Install Window event log parser: *Python-evtx*

- A pure Python parser for recent Windows Event Log files
  - Parse files with "*.evtx*“
- Provides programmatic access to the File and Chunk headers, record templates, and event entries

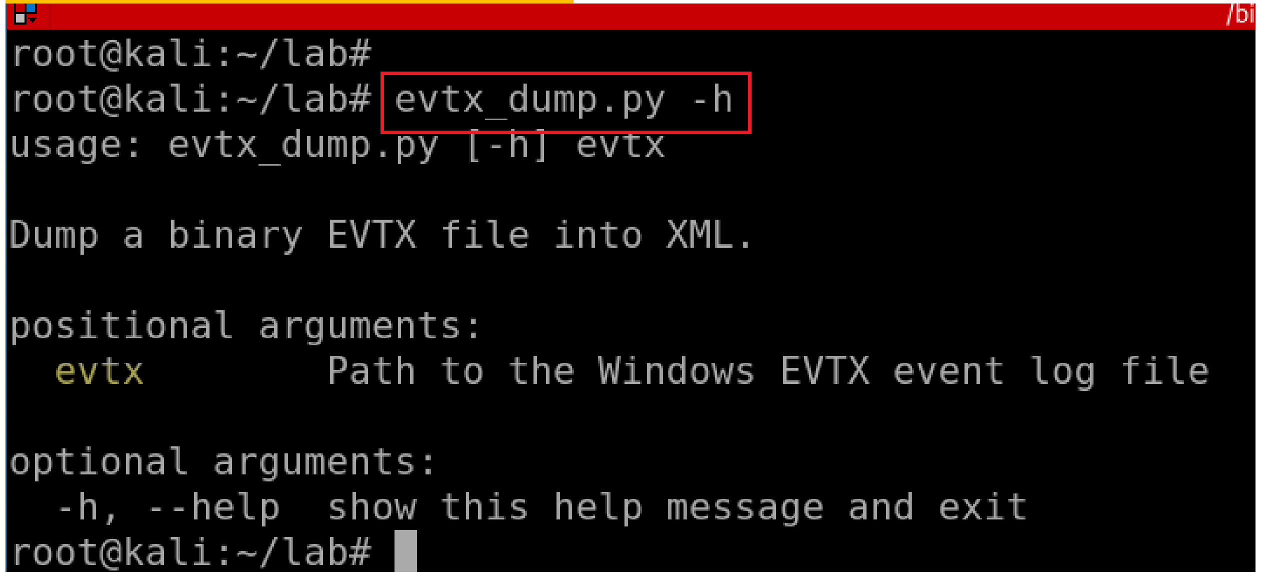
Install via apt



```
root@kali:~/lab#
root@kali:~/lab# apt-get install python3-evtx
Reading package lists... Done
Building dependency tree
Reading state information... Done
```

Run *apt-get update* if there is any error message

Verify installation



```
root@kali:~/lab#
root@kali:~/lab# evtx_dump.py -h
usage: evtx_dump.py [-h] evtx

Dump a binary EVTX file into XML.

positional arguments:
  evtx        Path to the Windows EVTX event log file

optional arguments:
  -h, --help  show this help message and exit
root@kali:~/lab#
```