# Investigate Data Leakage Case

Keywords: Windows Registry, PC information, usage Account, Application Usage

# Topics

- Key concepts (image, volume, file system)
- Gather basic PC information
- User account investigation
- Application usage investigation

# The Sleuth Kit (TSK Layers)

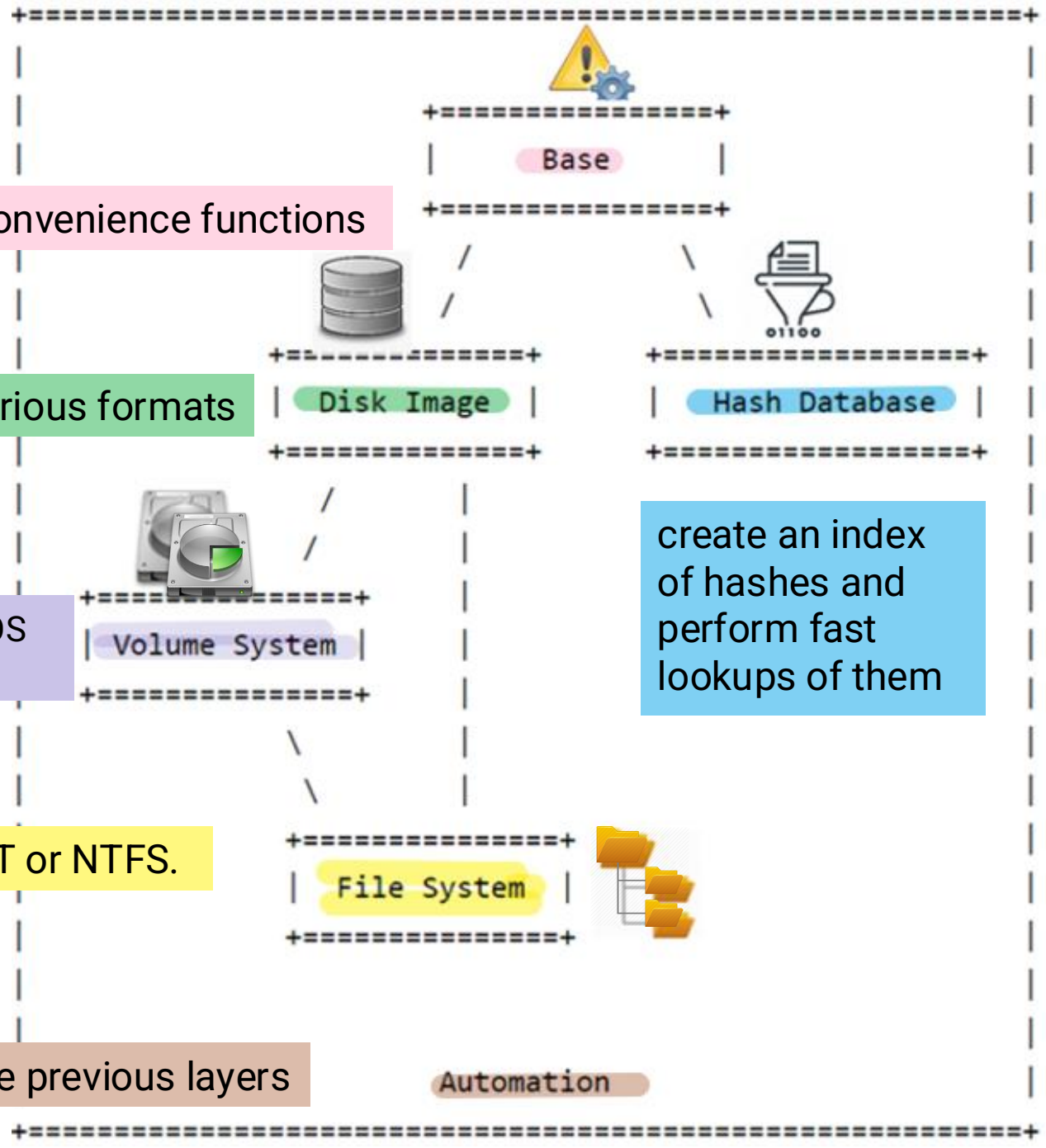
error handling, types, and convenience functions

Can open and process disk images in various formats

Processing data as a volume system, e.g., DOS partition tables

processing data as a file system, such as FAT or NTFS.

integrates all of the previous layers

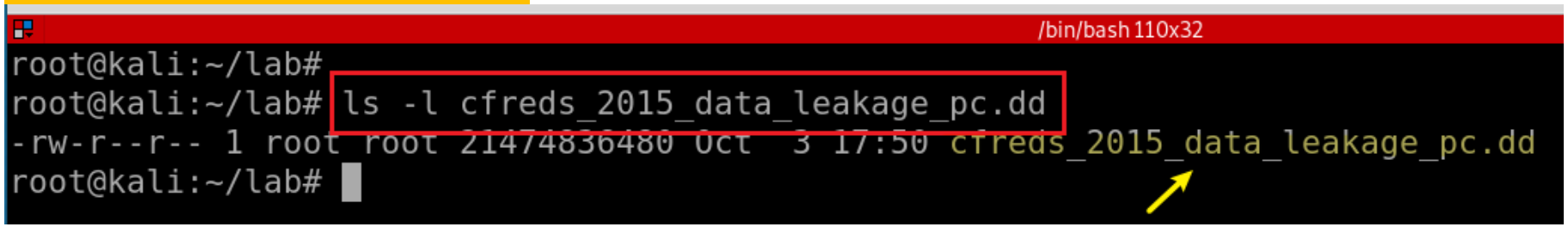



create an index of hashes and perform fast lookups of them

# Gather Basic PC Information

# 1. What are the hash values (MD5 & SHA-1) of the image? (Linux)

Verify you have the dd image



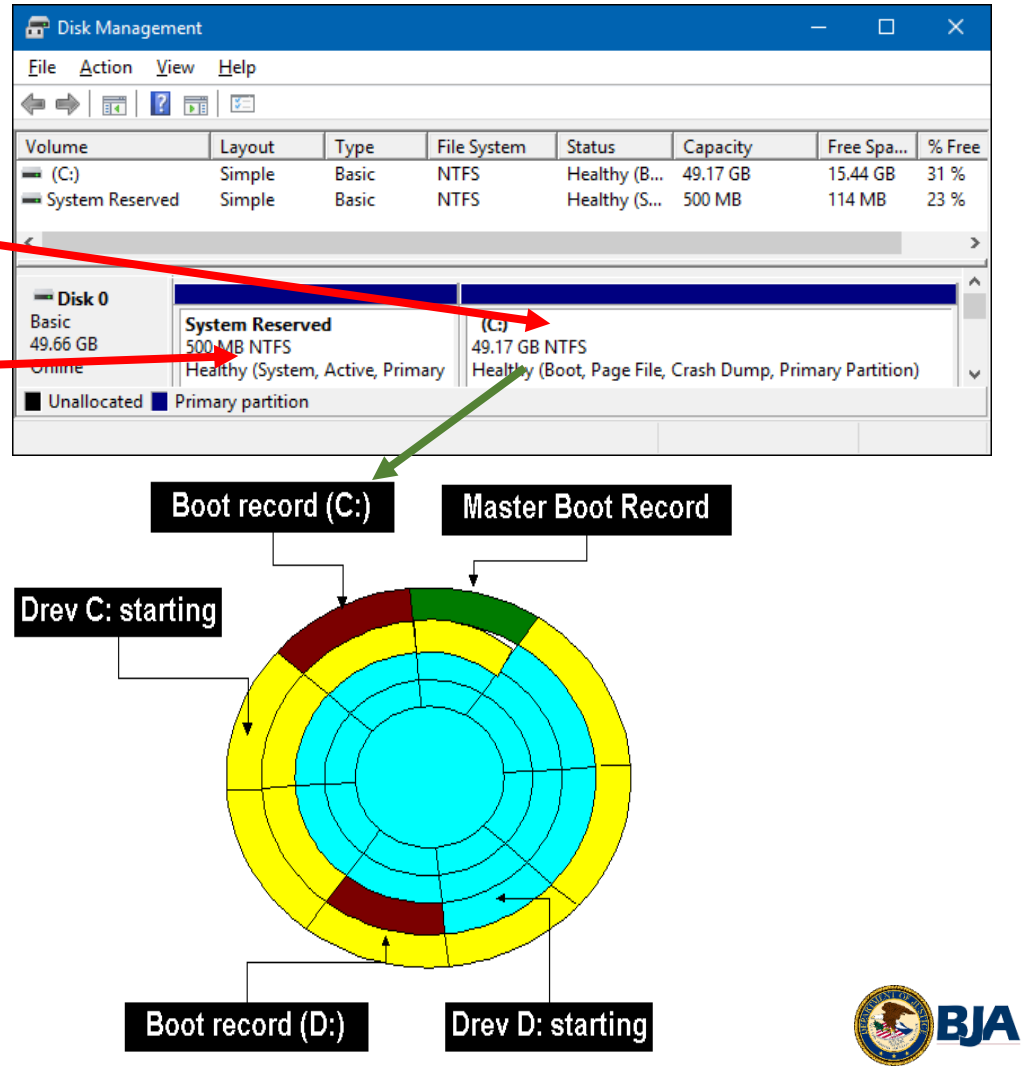
## Compute MD5 and SHA1 of the DD image

```
root@kali:~/lab# md5sum cfreds_2015_data_leakage_pc.dd
a49d1254c873808c58e6f1bcd60b5bde  cfreds_2015_data_leakage_pc.dd
root@kali:~/lab# sha1sum cfreds_2015_data_leakage_pc.dd
afe5c9ab487bd47a8a9856b1371c2384d44fd785  cfreds_2015_data_leakage_pc.dd
root@kali:~/lab#
```

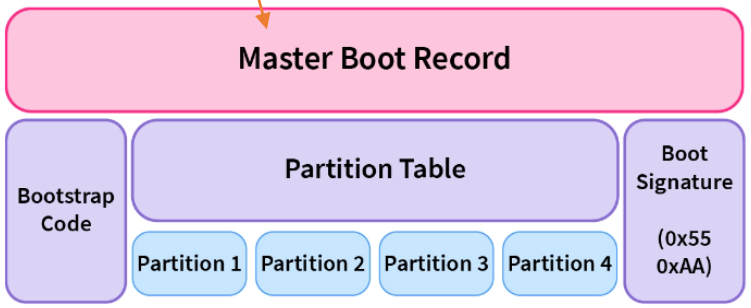
| Item | Detailed Information |
|---|---|
| Filename | cfreds_2015_data_leakage_pc |
| MD5 | A49D1254C873808C58E6F1BCD60B5BDE |
| SHA-1 | AFE5C9AB487BD47A8A9856B1371C2384D44FD785 |
| Imaging S/W | FTK Imager 3.4.0.1 |
| Image Format | DD converted from VMDK ( Some sectors were scrubbed[7] ) |
| Compression | Best (Smallest) |
| Bytes per Sector | 512 |
| Total Sectors | 41,943,040 |
| Total Size | 20.00 GB (21,474,836,480 bytes) |
| Compressed Size | 5.05 GB (5,427,795,228 bytes) ← compressed by 7zip |

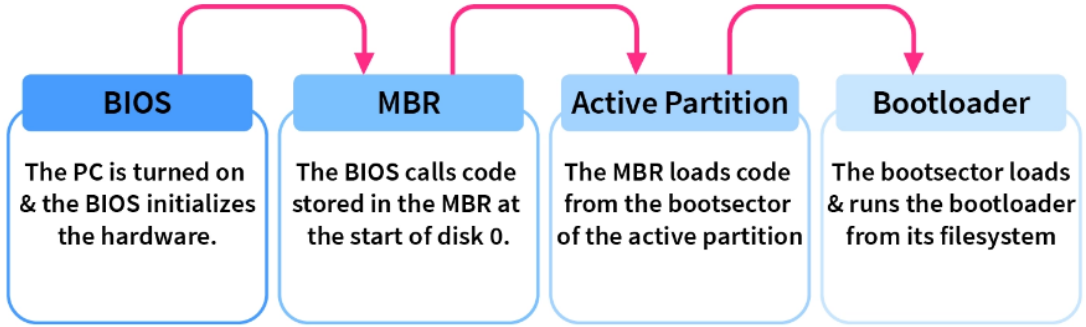
# 2.1 How to identify the partition information of PC image? (Method 1 -**fdisk**)

- What is partition/volume?
  - Boot partition
    - boot loader and kernel files for OS to start up
    - OS folder: *%systemroot%*
  - System partition
    - contains system files and device drivers that are required for the operating system to function properly
    - hidden from the user
    - contains the file system (NTFS) driver, the hardware abstraction layer (HAL), and other important system files.



| 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) |




https://www.scaler.com/topics/operating-system/master-boot-record/

Show partitions of the image

Since it is a file containing a copy of the entire disk, you can simply treat it like any other block device and run **fdisk**

```
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

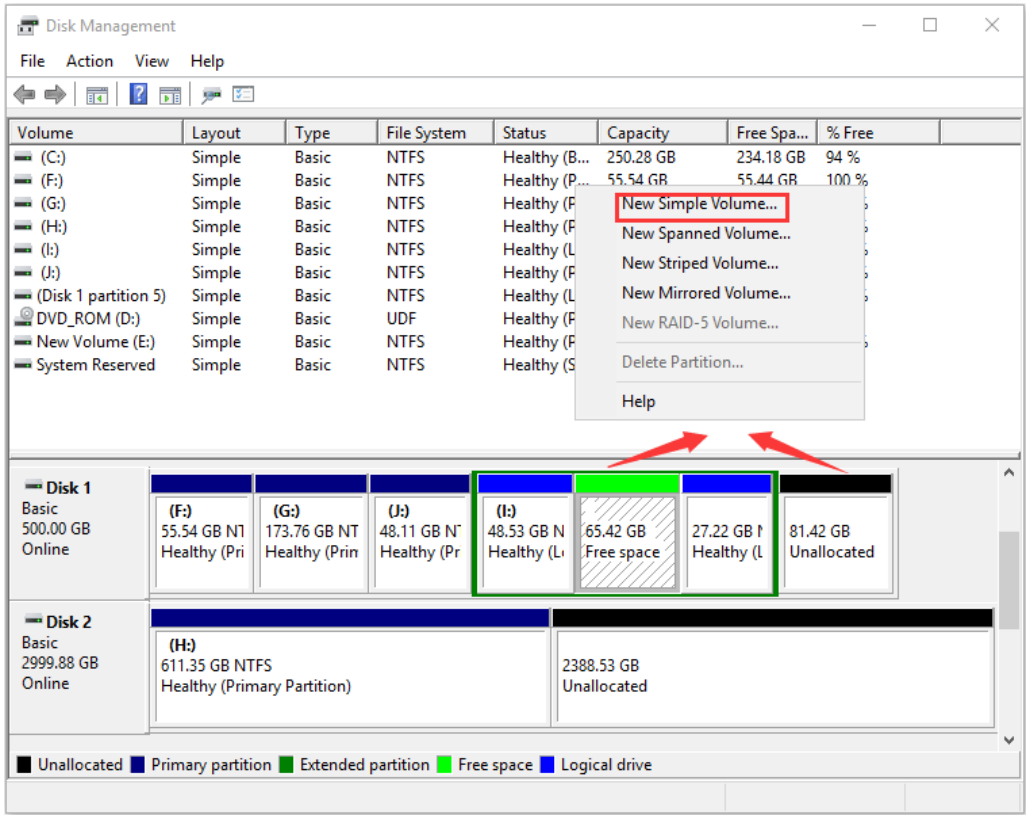
"Boot" volume: core os, load the remainder of an operating system

- What is Block devices? hard drives, CD-ROM drives, RAM
- What is fdisk ? Format disk

the device is bootable,
not a boot partition

# How to Identify the partition information of a PC image? (Method 2 -**mmls**)

- What is Unallocated Space?
  - Any physical space on a hard drive that doesn't belong to a partition.
  - No programs can write to the space.
  - The space doesn't exist to the operating system.
- To make use of unallocated space
  - you need to either create a new partition using the space or expand an existing partition.
- Media management ls (mmls):
  - Can show unallocated sectors so it can be used to search for hidden data



Show partitions and unallocated space using *mmls*

```
root@kali:~/lab# mmls cfreds_2015_data_leakage_pc.dd
DOS Partition Table
Offset Sector: 0
Units are in 512-byte sectors

      Slot    Start        End          Length       Description
000:  Meta    0000000000   0000000000   0000000001   Primary Table (#0)
001:  -------  0000000000   0000002047   0000002048   Unallocated
002:  000:000  0000002048   0000206847   0000204800   NTFS / exFAT (0x07)
003:  000:001  0000206848   0041940991   0041734144   NTFS / exFAT (0x07)
004:  -------  0041940992   0041943039   0000002048   Unallocated
root@kali:~/lab#
```

MBR

media management ls (mmls): Can show unallocated sectors so it can be used to search for hidden data

# How to Identify the partition information of PC image? (Method 3 -**parted**)

```
root@kali:~/lab# parted cfreds_2015_data_leakage_pc.dd
GNU Parted 3.3
Using /root/lab/cfreds_2015_data_leakage_pc.dd
Welcome to GNU Parted! Type 'help' to view a list of commands.
(parted) print
Model:  (file)
Disk /root/lab/cfreds_2015_data_leakage_pc.dd: 21.5GB
Sector size (logical/physical): 512B/512B
Partition Table: msdos
Disk Flags:

Number  Start   End     Size    Type     File system  Flags
 1      1049kB  106MB   105MB   primary  ntfs         boot
 2      106MB   21.5GB  21.4GB  primary  ntfs

(parted) help
```

do not list partitions whose size is greater than 2 TB

```
Device                                  Boot  Start
cfreds_2015_data_leakage_pc.dd1 *          2048
cfreds_2015_data_leakage_pc.dd2          206848
```

Display file system statistics and metadata information from a disk image (first partition)

```
root@kali:~/lab# fsstat -b 512 -o 2048 cfreds_2015_data_leakage_pc.dd
FILE SYSTEM INFORMATION
--------------------------------------------
File System Type: NTFS
Volume Serial Number: 4A180A15180A0125
OEM Name: NTFS
Volume Name: System Reserved
Version: Windows XP

METADATA INFORMATION
--------------------------------------------
First Cluster of MFT: 8533
First Cluster of MFT Mirror: 2
Size of MFT Entries: 1024 bytes
Size of Index Records: 4096 bytes
Range: 0 - 256
Root Directory: 5

CONTENT INFORMATION
--------------------------------------------
Sector Size: 512
Cluster Size: 4096
Total Cluster Range: 0 - 25598
Total Sector Range: 0 - 204798
```

-b: block size (default is 512)
-o: image offset

```
Device                                  Boot  Start
cfreds_2015_data_leakage_pc.dd1         *      2048
cfreds_2015_data_leakage_pc.dd2              206848
```

List the second partition details

```
root@kali:~/lab# fsstat -b 512 -o 206848 cfreds_2015_data_leakage_pc.dd
FILE SYSTEM INFORMATION
--------------------------------------------
File System Type: NTFS
Volume Serial Number: C8CA0C8DCA0C7A48
OEM Name: NTFS
Version: Windows XP

METADATA INFORMATION
--------------------------------------------
First Cluster of MFT: 786432
First Cluster of MFT Mirror: 2
Size of MFT Entries: 1024 bytes
Size of Index Records: 4096 bytes
Range: 0 - 78080
Root Directory: 5

CONTENT INFORMATION
--------------------------------------------
Sector Size: 512
Cluster Size: 4096
Total Cluster Range: 0 - 5216766
Total Sector Range: 0 - 41734142
```

-b: block size
-o: image offset

Serial number. Remember the #.
We will use it later.

```
Device                                  Boot  Start
cfreds_2015_data_leakage_pc.dd1         *      2048
cfreds_2015_data_leakage_pc.dd2              206848
```

# 2.2 How to show files (directories) in 2nd partition?

```
root@kali:~/lab# fls -o 206848 cfreds_2015_data_leakage_pc.dd
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
r/r 1-128-1:    $MFTMirr
d/d 57-144-5:   $Recycle.Bin
r/r 9-128-8:    $Secure:$SDS
r/r 9-144-16:   $Secure:$SDH
r/r 9-144-17:   $Secure:$SII
r/r 10-128-1:   $UpCase
d/d 13797-144-1: Documents and Settings
r/r 504-128-1:  hiberfil.sys
d/d 22329-144-1: MSOCache
r/r 58995-128-1: pagefile.sys
```

**fls**: List file and directory names in a disk image

*files in the partition 2*

**File Type | Metadata Address | File Name**

Use head/tail command to limit the number of files to display

```
| head -n 10
```

# 2.3 How to list all deleted *.docx* files in the whole partition?

- . means "any character" in a regex.
- for literal string: "\.docx"

```
root@kali:~/lab# fls -rdF -o 206848  cfreds_2015_data_leakage_pc.dd | grep .docx
r/- * 0:    Users/informant/AppData/Roaming/Microsoft/Templates/LiveContent/15/Managed/Word
 Document Building Blocks/1033/TM02835270[[fn=Photo Sidebar (Annual Report Red and Black de
sign)]].docx
r/- * 0:    Users/informant/Desktop/~$signation_Letter_(Iaman_Informant).docx
root@kali:~/lab#
```

Other useful parameters

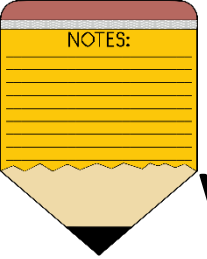
**-d** display deleted entries only
**-D** directories only
**-r** recursively display directories
**-l** long format
**-F** Display file (all non-directory) entries only
**-u** Display undeleted entries only

## Verify system information

```
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
```

## Verify Users' information

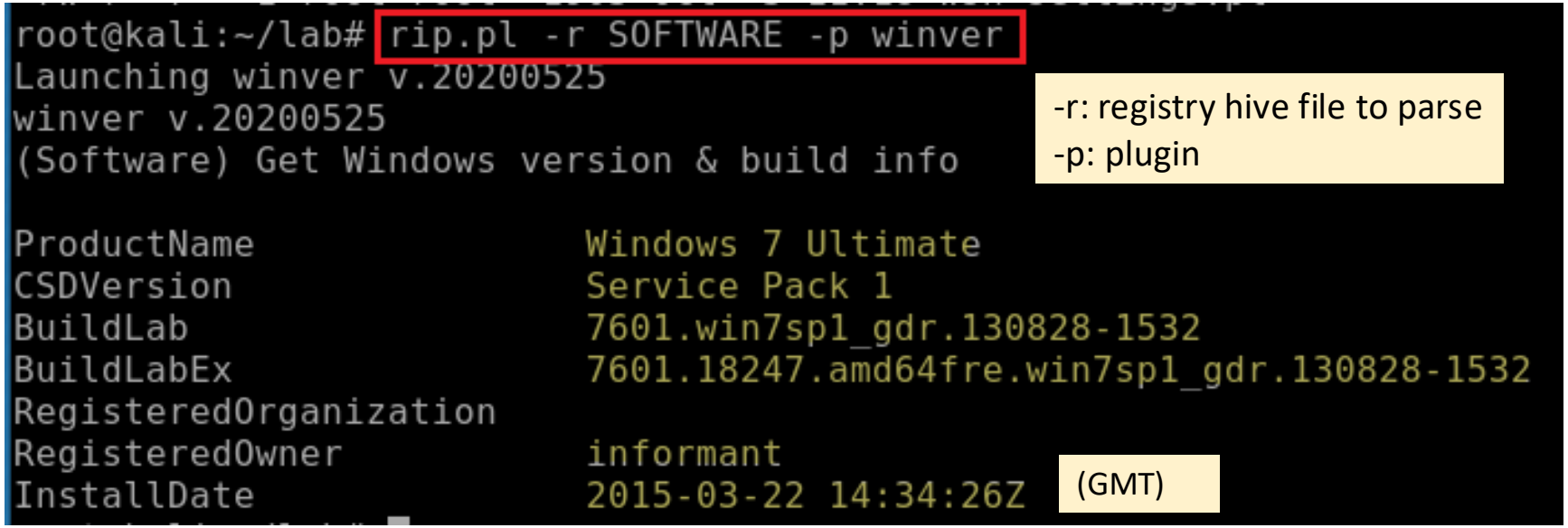
```
root@kali:~/lab# ls -l *.DAT
-rwxr-xr-x 1 root root  524288 Oct  3 20:58 NTUSER_Admin11.DAT
-rwxr-xr-x 1 root root  262144 Oct  3 20:59 NTUSER_Default.DAT
-rwxr-xr-x 1 root root 1048576 Oct  3 20:59 NTUSER_informant.DAT
-rwxr-xr-x 1 root root  524288 Oct  3 21:00 NTUSER_temporary.DAT
```




# Windows Registry Analysis Requirements

- All investigations involving Windows Registry requires
  - Installed *RegRipper* 3.0
  - Extracted files contain PC's registry information
- Verify files on the next slide before any tasks

# 3. What is the installed OS information in detail?




HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion

Show **rip.pl** command help

```
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)

Ex: C:\>rip -r c:\case\system -f system
    C:\>rip -r c:\case\ntuser.dat -p userassist
    C:\>rip -r c:\case\ntuser.dat -a
    C:\>rip -l -c

All output goes to STDOUT; use redirection (ie, > or >>) to output to a file.
```

Try it

List all 243 plugins of SOFTWARE

```
root@kali:~/lab# rip.pl -l | head -n 12
1. powershellcore v.20200525 [Software]
   - Extracts PowerShellCore settings

2. oisc v.20091125 [NTUSER.DAT]
   - Gets contents of user's Office Internet Server Cache

3. securityproviders v.20200526 [System]
   - Gets SecurityProvider value from System hive

4. mmo v.20200517 [NTUSER.DAT]
   - Checks NTUSER for Multimedia\Other values [malware]
```

-l: list

Show the location of all plugins

```
root@kali:~/lab# ls -l /usr/share/regripper/plugins/ | head -n 5
total 1320
-rw-r--r-- 1 root root  3795 Oct  3 11:18 adobe.pl
-rw-r--r-- 1 root root    57 Oct  3 11:18 all
-rw-r--r-- 1 root root  2451 Oct  3 11:18 allowedenum.pl
-rw-r--r-- 1 root root     8 Oct  3 11:18 amcache
```

# 4. What is the time zone setting?

Search for timezone plugin and the file that contains timezone

```
root@kali:~/lab# rip.pl -l | grep -i timezone
111. timezone v.20200518 [System]
   - Get TimeZoneInformation key contents
```

Run timezone plugin

```
root@kali:~/lab# rip.pl -r SYSTEM -p timezone
Launching timezone v.20200518
timezone v.20200518
(System) Get TimeZoneInformation key contents

TimeZoneInformation key
ControlSet001\Control\TimeZoneInformation
LastWrite Time 2015-03-25 10:34:25Z
  DaylightName    -> @tzres.dll,-111
  StandardName    -> @tzres.dll,-112
  Bias            -> 300 (5 hours)
  ActiveTimeBias  -> 240 (4 hours)
  TimeZoneKeyName-> Eastern Standard Timeard Time
```

The Bias property represents the difference in minutes between Greenwich Mean Time (GMT—also known as Coordinated Universal Time, or UTC) and local time. For example, Eastern time (US and Canada) has a Bias property value of -300.

'Z' stands for Zulu time, which is also GMT and UTC.

HKLM\SYSTEM\ControlSet###\Control\TimeZoneInformation

# 5. What is the computer name?

Search for computer name plugin and the file that contains timezone

Run compname plugin

```
root@kali:~/lab# rip.pl -l | grep -i name
87. compname v.20090727 [System]
   - Gets ComputerName and Hostname values from System hive
   - Gets contents of PendingFileRenameOperations value
   - Parse hive, check key/value names for RLO character
   - Check key/value names in a hive for leading null char
root@kali:~/lab#
root@kali:~/lab# rip.pl -r SYSTEM -p compname
Launching compname v.20090727
compname v.20090727
(System) Gets ComputerName and Hostname values from System hive

ComputerName    = INFORMANT-PC
TCP/IP Hostname = informant-PC
```

HKLM\SYSTEM\ControlSet###\Control\ComputerName\ComputerName (value: ComputerName)
HKLM\SYSTEM\ControlSet###\Services\Tcpip\Parameters (value: Hostname).....

# User Account Investigation

# 6. How many accounts does the system have?

*(except Administrator, Guest, systemprofile, LocalService, NetworkService)*

Search for profiles

```
root@kali:~/lab# rip.pl -l | grep -i profile
76. profilelist v.20200518 [Software]
   - Get content of ProfileList key
122. profiler v.20200525 [NTUSER.DAT, System]
   - Environment profiler information
```

resolve SIDs to user

```
root@kali:~/lab# rip.pl -r SOFTWARE -p profilelist
Launching profilelist v.20200518
profilelist v.20200518
(Software) Get content of ProfileList key

Microsoft\Windows NT\CurrentVersion\ProfileList

Path      : %systemroot%\system32\config\systemprofile
SID       : S-1-5-18
LastWrite : 2009-07-14 04:53:25Z

Path      : C:\Windows\ServiceProfiles\LocalService
SID       : S-1-5-19
LastWrite : 2015-03-25 11:14:18Z

Path      : C:\Windows\ServiceProfiles\NetworkService
SID       : S-1-5-20
LastWrite : 2015-03-25 11:14:18Z

Path      : C:\Users\informant
SID       : S-1-5-21-2425377081-3129163575-2985601102-1000
LastWrite : 2015-03-25 15:30:57Z

Path      : C:\Users\admin11
SID       : S-1-5-21-2425377081-3129163575-2985601102-1001
LastWrite : 2015-03-22 15:57:41Z

Path      : C:\Users\temporary
SID       : S-1-5-21-2425377081-3129163575-2985601102-1003
LastWrite : 2015-03-22 15:56:58Z
```

Find and search for Security Accounts Manager (SAM) information

```
root@kali:~/lab#
root@kali:~/lab# rip.pl -r SAM -P samparse | grep -E "Username|Created|Date"
Launching samparse v.20200825
Username        : Administrator [500]
Account Created : 2015-03-25 10:33:22Z
Last Login Date : 2010-11-21 03:47:20Z
Pwd Reset Date  : 2010-11-21 03:57:24Z
Pwd Fail Date   : Never
Username        : Guest [501]
Account Created : 2015-03-25 10:33:22Z
Last Login Date : Never
Pwd Reset Date  : Never
Pwd Fail Date   : Never
Username        : informant [1000]
Account Created : 2015-03-22 14:33:54Z
Last Login Date : 2015-03-25 14:45:59Z
Pwd Reset Date  : 2015-03-22 14:33:54Z
Pwd Fail Date   : 2015-03-25 14:45:43Z
Username        : admin11 [1001]
Account Created : 2015-03-22 15:51:54Z
Last Login Date : 2015-03-22 15:57:02Z
Pwd Reset Date  : 2015-03-22 15:52:10Z
Pwd Fail Date   : 2015-03-22 15:53:02Z
Username        : ITechTeam [1002]
Account Created : 2015-03-22 15:52:30Z
Last Login Date : Never
Pwd Reset Date  : 2015-03-22 15:52:45Z
Pwd Fail Date   : 2015-03-22 15:53:02Z
Username        : temporary [1003]
Account Created : 2015-03-22 15:53:01Z
Last Login Date : 2015-03-22 15:55:57Z
Pwd Reset Date  : 2015-03-22 15:53:11Z
Pwd Fail Date   : 2015-03-22 15:56:37Z
root@kali:~/lab#
```

SAM stores accounts information, e.g., passwords in a hashed format (NTLM).

*grep –E* <regular expression>

- login created
- last login: 3 days later but failed

# 6.1 What are the NTLM of these accounts?

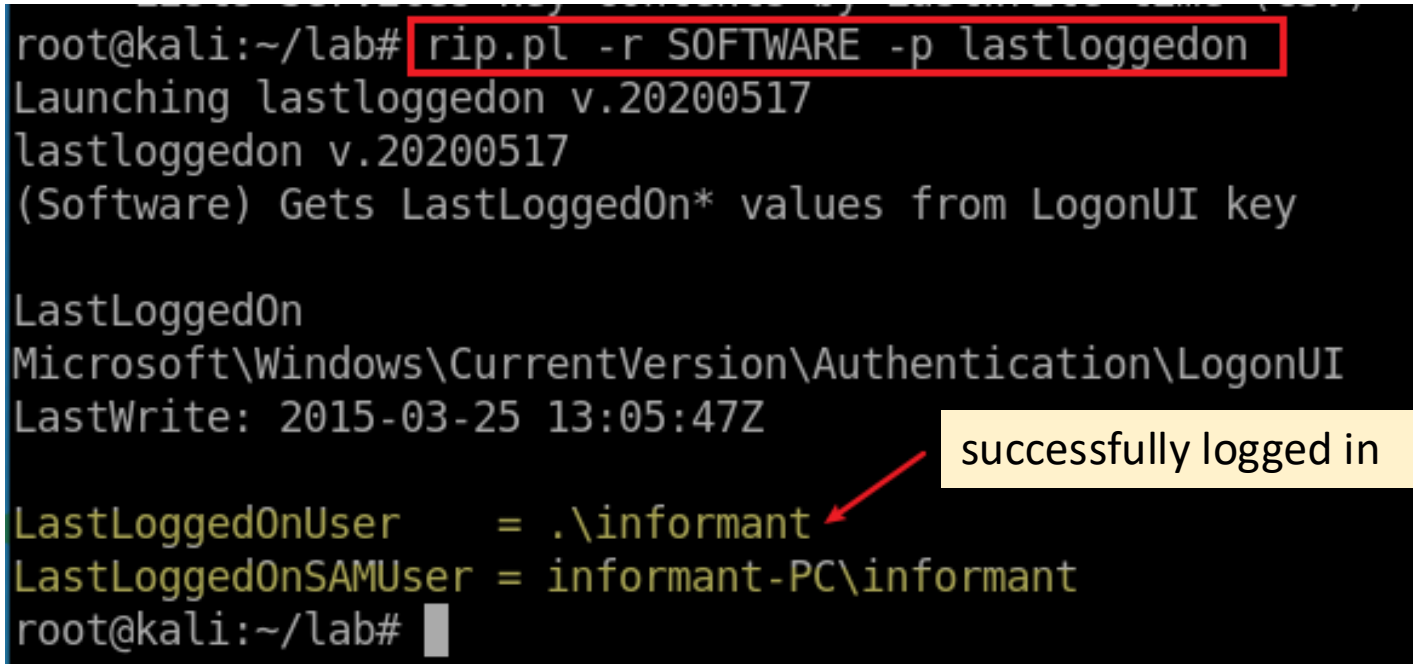
```
NTUSER.DAT                    SAM
root@kali:~/lab# impacket-secretsdump -sam SAM -security SECURITY -system SYSTEM LOCAL
Impacket v0.9.20 - Copyright 2019 SecureAuth Corporation

[*] Target system bootKey: 0xface85b8f08c42ca889ee83551ee1e6f
[*] Dumping local SAM hashes (uid:rid:lmhash:nthash)
Administrator:500:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
informant:1000:aad3b435b51404eeaad3b435b51404ee:9e3d31b073e60bfd7b07978d6f914d0a:::
admin11:1001:aad3b435b51404eeaad3b435b51404ee:21759544b2d7efccc978449463cf7e63:::
ITechTeam:1002:aad3b435b51404eeaad3b435b51404ee:75ed0cb7676889ab43764a3b7d3e6943:::
temporary:1003:aad3b435b51404eeaad3b435b51404ee:1b3801b608a6be89d21fd3c5729d30bf:::
[*] Dumping cached domain logon information (domain/username:hash)
[*] Dumping LSA Secrets
[*] DPAPI_SYSTEM
dpapi_machinekey:0x5af777761a6901635bccb536fe632146073282a3
dpapi_userkey:0xadf466891df9745e3d931833b52f0e825ab9b6d1
[*] Cleaning up...
root@kali:~/lab#
```

# 6.2 How to Crack Windows 10 passwords?

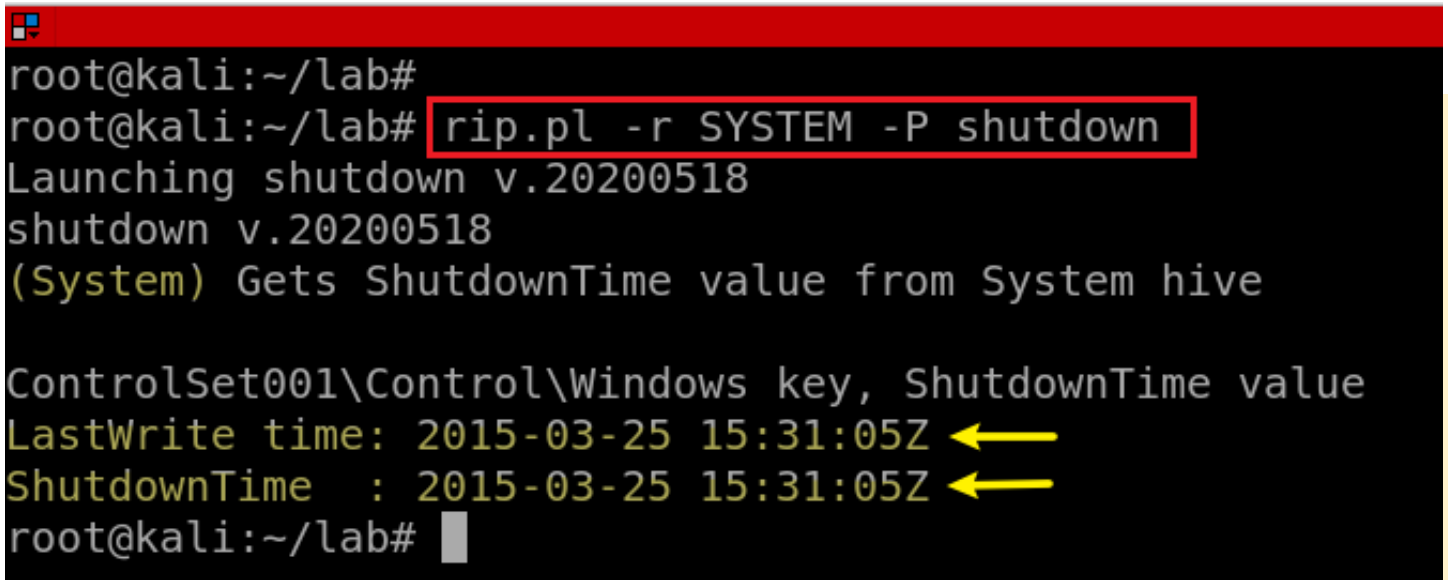
Crack Win 10 password using NTLM and Rainbow table.
Follow the PPTs

# 7. Who was the last user to logon into PC?




HKLM\Software\~
HKLM\SAM\~

# 8. When was the last recorded shutdown date/time?




A control set contains system configuration information such as device drivers and services.

- ***ControlSet001*** may be the last control set you booted with.
- ***ControlSet002*** could be what is known as the last known good control set, or the control set that last successfully booted Windows NT.

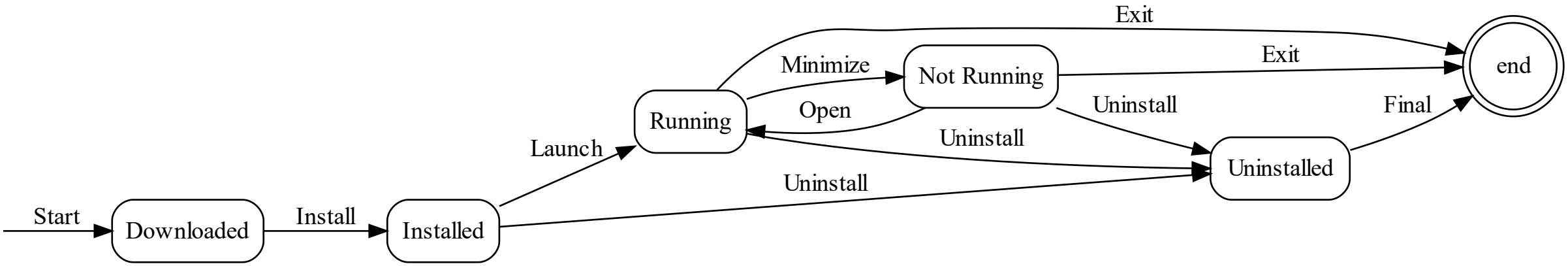
HKLM\SYSTEM\ControlSet###\Control\Windows (value: ShutdownTime)

Conclusion: informant was the last one logged on at 13:05 (previous slide) and shut down the PC at 15:31pm

# 9. Explain the information of network interface(s) with an IP address assigned by DHCP.

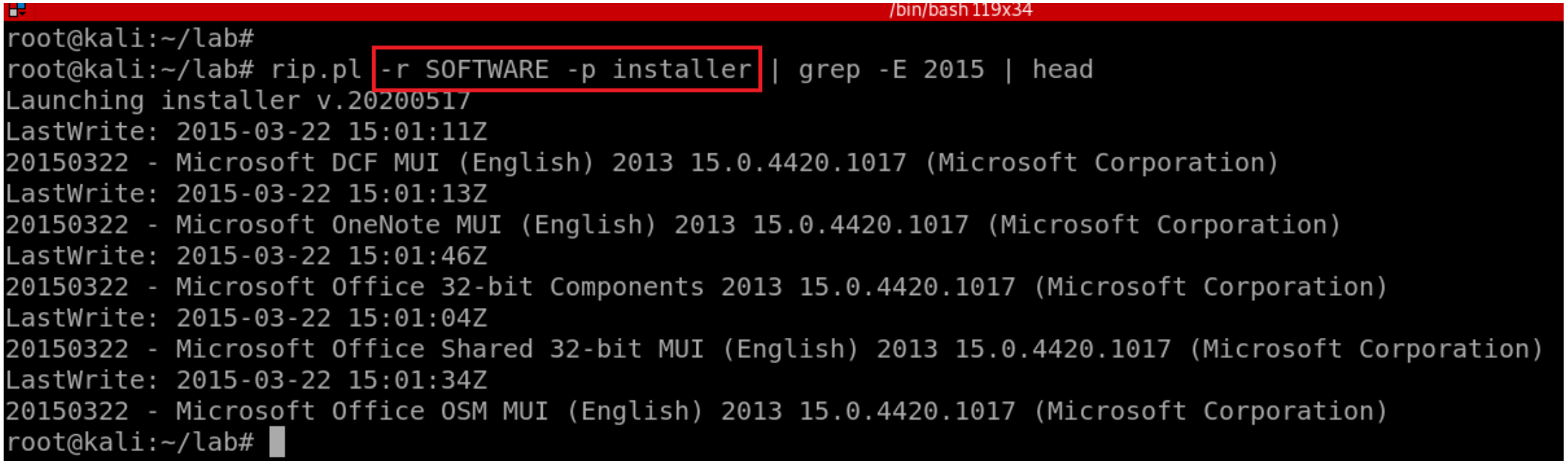

HKLM\SYSTEM\ControlSet###\Services\Tcpip\Parameters\Interfaces\{GUID}

# Application Usages Investigation



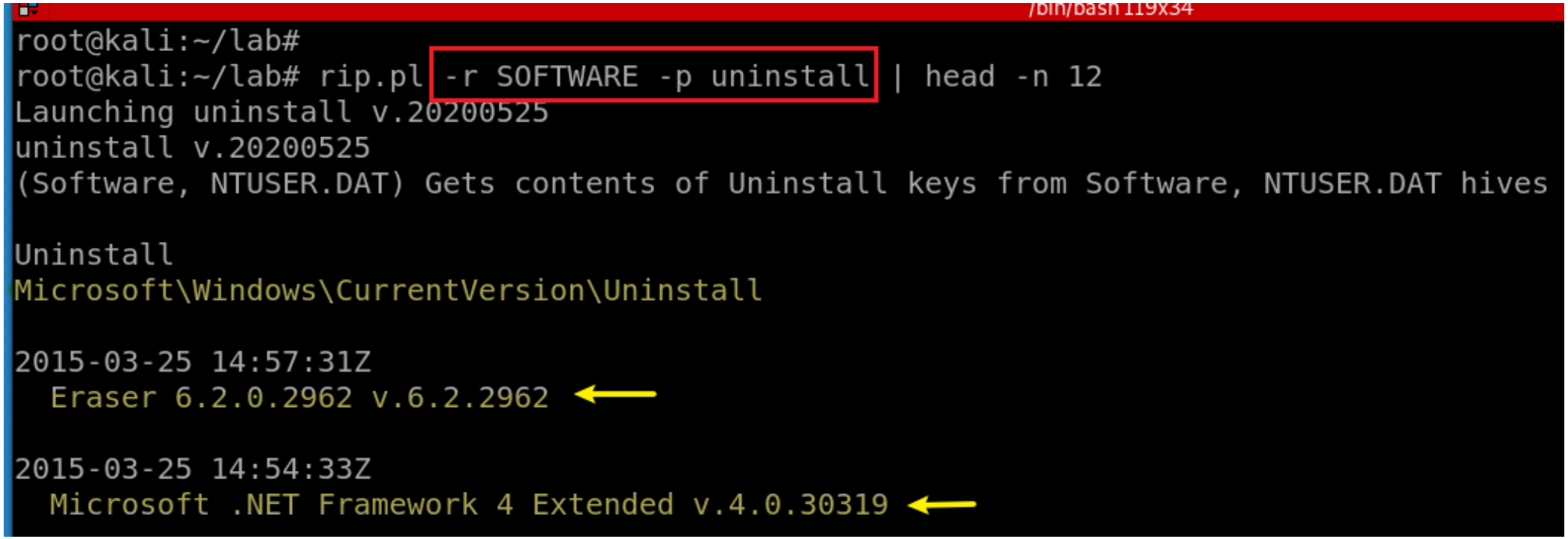

A state diagram for application usage investigations

# 10. What applications were installed by the suspect after installing OS?




HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Installer\UserData

# 10.1 What applications can be uninstalled by the suspect after installing OS?




HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Uninstall\~
HKLM\SOFTWARE\Wow6432Node\Microsoft\Windows\CurrentVersion\Uninstall\~

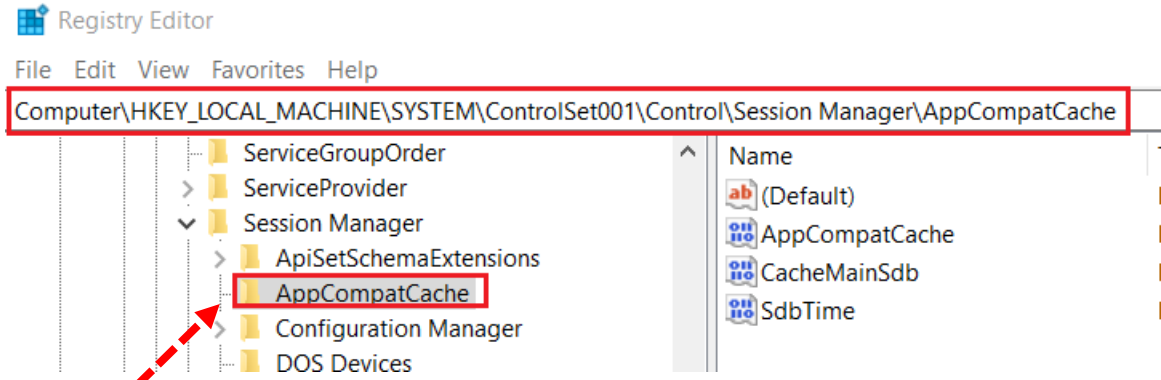
# 11. List application execution logs (Executable path, execution time, execution count…)

- *Shimcache:* speed loading
- *Amcache/RecentFileCache*
- *UserAssist*
- *Prefetch*
- *MuiCache*: Multilingual User Interface

# 11.1 Primary purpose of *Shimcache*

- The primary goal is to optimize program loading
  - speed up loading frequently executed programs by caching information about them (think about Amazon's local distribution center).
- Shimcach records execution history
  - program names, file paths, timestamps, and execution counts.
- NOT serving as a dedicated compatibility checker.
  - it is closely tied to Windows' compatibility features.
  - when a program is launched, the Shimcache checks if there are any known compatibility fixes or "shims" associated with that program.
    - shims are applied to ensure the program runs smoothly on the Windows platform.
    - not fix compatibility issues

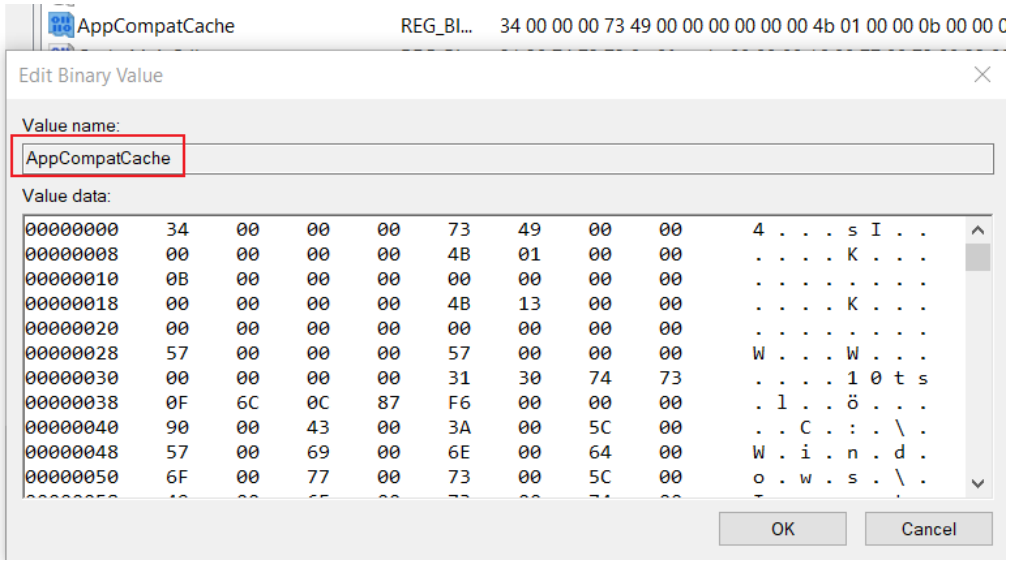
# Registry for *Shimcache*




- Known as *AppCompatCache*
  - *HKLM\SYSTEM\ControlSet###\Control\Session Manager\AppCompatCache*
- Two actions that can cause the *Shimcache* to record an entry
  - A file is executed.
    - This is recorded on all versions of Windows beginning with XP.
  - A user interactively browses a directory
    - if a directory contains the files "*foo.txt*" and "*bar.exe*", a Windows 7 system may record entries for these two files in the *Shimcache*.
    - On Windows Vista, 7, Server 2008, and Server 2012

# *Shimcache* entry

- Stores various file metadata depending on the operating system
  - File Full Path
  - File Size
  - *$Standard_Information* (SI) Last Modified time
  - *Shimcache* Last Updated time
  - Process Execution Flag : set this flag during process creation/execution
- Only contains the information prior to the system's last startup
  - current entries are stored only in memory
- The oldest data is replaced by new entries.



## Extract *shimcache* from registry

```
/bin/bash 119x34
root@kali:~/lab#
root@kali:~/lab# rip.pl -r SYSTEM -p shimcache | head -n 15
Launching shimcache v.20200428
shimcache v.20200428
(System) Parse file refs from System hive AppCompatCache data

*** ControlSet001 ***
ControlSet001\Control\Session Manager\AppCompatCache
LastWrite Time: 2015-03-25 15:31:05Z
Signature: 0xbadc0fee
Num_entries: 292
Data Length: 60816 bytes
Win2K8R2/Win7, 64-bit
C:\Windows\SysWOW64\twext.dll  2010-11-21 03:24:32
C:\Windows\System32\control.exe  2009-07-14 01:39:01  Executed
C:\Windows\system32\sppsvc.exe  2010-11-21 03:23:56  Executed
C:\Windows\System32\ieframe.dll  2015-03-22 15:16:54
C:\Windows\system32\xpsrchvw.exe  2009-07-14 01:39:59  Executed
root@kali:~/lab#
```

# 11.2 Main purpose of *Amcache*

- Assist with application compatibility (Application Compatibility Cache)
  - ensure that software updates or system changes do not break existing applications
  - used to apply compatibility fixes or "shims" to programs
  - maintain information about installed applications on the system.
    - including file paths, version numbers, and compatibility settings.
- Software Inventory
  - a repository of information about installed software.
  - `C:\Windows\AppCompat\Programs\Amcache.hve`
- Focus on software installation and updates
  - ShimCache entry is updated each time the application is executed

# *Amcache* replaces *RecentFileCache.bcf*

- *Amcache*
  - Windows 8 and Later
  - It includes information about executed programs (similar to Shimcache) and also contains details about recently accessed files and folders
- *RecentFileCache.bcf*
  - In Windows 7 and earlier versions, the RecentFileCache.bcf file was used to record recent file activity.

## Find the location of *RecenfFileCache.bcf (Win 7)* *rip.pl –r Amcache.hve –p amcache (Win 8)*

/bin/bash 115x18

```
root@kali:~/lab#
root@kali:~/lab# fls -rF -o 206848 cfreds_2015_data_leakage_pc.dd | grep -Ei 'RecentFileCache'
r/r 16029-128-4:        Windows/AppCompat/Programs/RecentFileCache.bcf
root@kali:~/lab#
```

## Show *RecenfFileCache.bcf*

/bin/bash 76x18

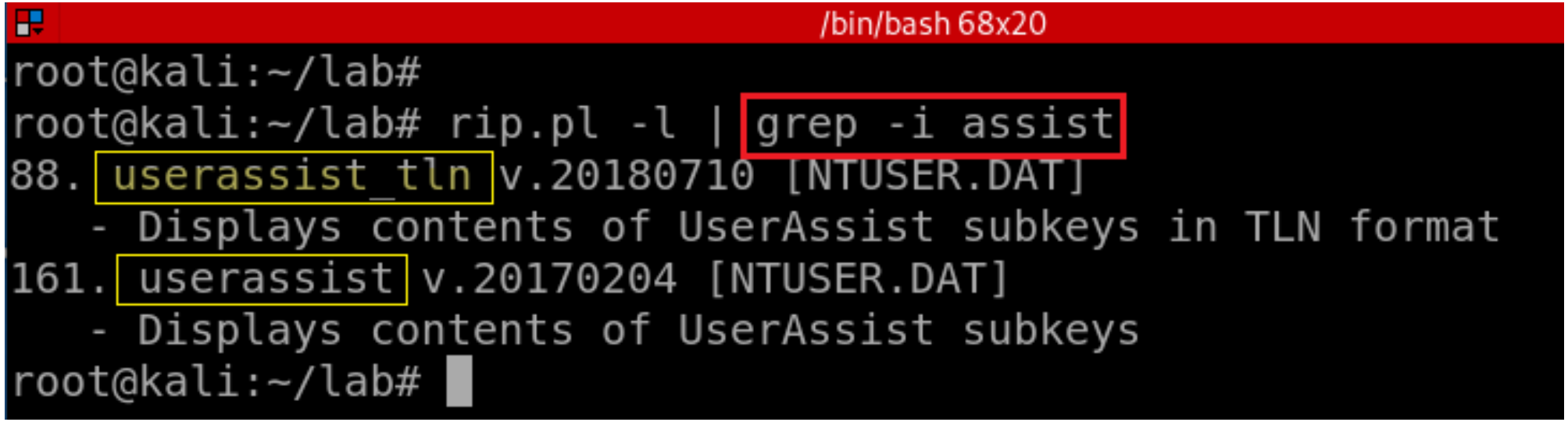
```
root@kali:~/lab#
root@kali:~/lab# icat -o 206848 cfreds_2015_data_leakage_pc.dd 16029
????"?L?8c:\program files (x86)\windows media player\setup_wm.exe0c:\a5df94f
cac8e62a530d048042c2a\setuputility.exec:\windows\syswow64\wusa.exec:\windows
\system32\wuauclt.exe c:\windows\system32\unlodctr.exe c:\windows\syswow64\u
nlodctr.exec:\windows\system32\lodctr.exe>c:\windows\microsoft.net\framework
64\v4.0.30319\regtlibv12.exe<c:\windows\microsoft.net\framework\v4.0.30319\r
egtlibv12.exe$c:\windows\syswow64\wbem\mofcomp.exe8c:\windows\microsoft.net\
framework64\v4.0.30319\ngen.exe<c:\windows\microsoft.net\framework64\v4.0.30
319\mscorsvw.exe6c:\windows\microsoft.net\framework\v4.0.30319\ngen.exe:c:\w
indows\microsoft.net\framework\v4.0.30319\mscorsvw.exeCc:\windows\microsoft.
net\framework64\v4.0.30319\servicemodelreg.exeAc:\windows\microsoft.net\fram
ework\v4.0.30319\servicemodelreg.exeAc:\windows\microsoft.net\framework64\v4
.0.30319\aspnet_regiis.exe?c:\windows\microsoft.net\framework\v4.0.30319\asp
net_regiis.exec:\windows\syswow64\ping.exe(c:\program files\ccleaner\ccleane
r64.exe&c:\program files\ccleaner\ccleaner.exeroot@kali:~/lab#
```

Table 1: Comparison of Amcache and Shimcache in Digital Forensics

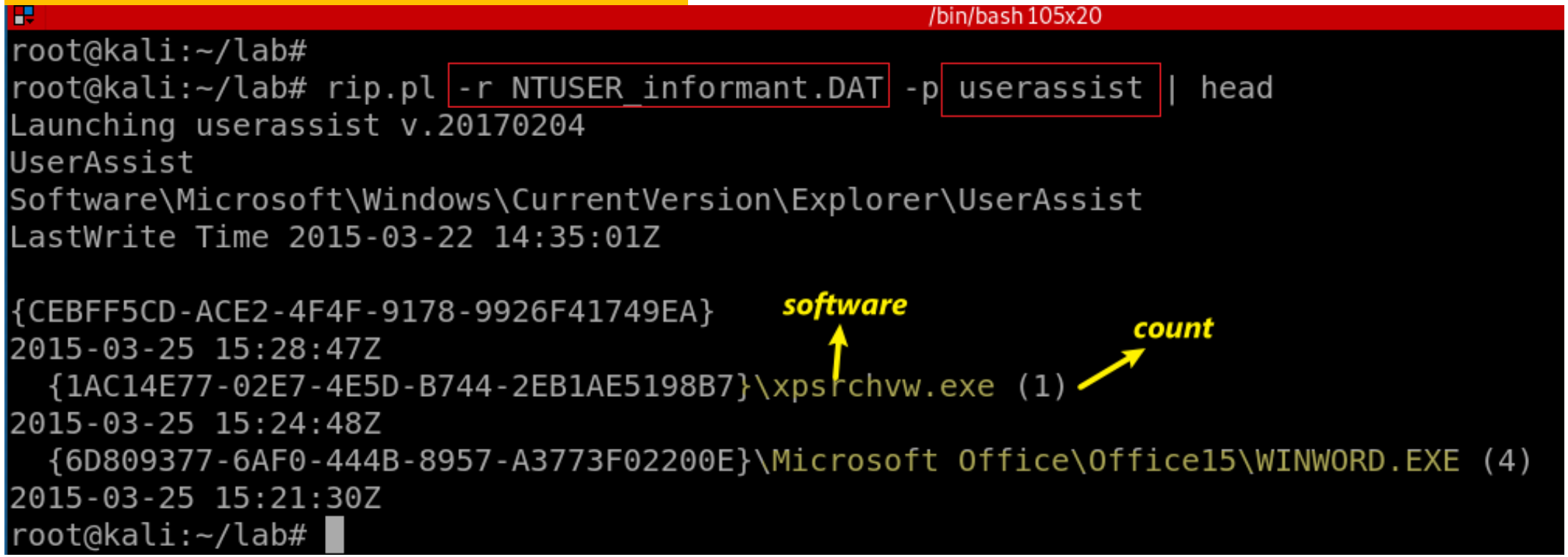
| **Aspect** | **Amcache** | **Shimcache** |
|---|---|---|
| Forensic Relevance | Records information about installed applications for compatibility testing, aiding in software inventory and compatibility analysis | Records execution history of programs, helping forensic analysts identify which programs were run and when they were last executed |
| Location | Stored in the Windows Registry as "Amcache.hve" | Stored in the Windows Registry as a key within "AppCompatCache" |
| Data | Contains information about installed applications, including file paths, version numbers, and compatibility details | Contains information about recently executed programs, including file paths, timestamps, and execution counts |
| Update Frequency | Updated periodically, especially when software changes occur on the system (e.g., application installations or updates) | Updated each time an executable is run |
| Use Cases | Used for application inventory, software compatibility testing, and ensuring that software updates or system changes do not break existing applications | Frequently used in forensic investigations to determine which programs were executed on a Windows system and when they were last run. It can provide valuable insights into user activity and potentially suspicious behavior. |
| Challenges | Requires access to the Windows Registry, which may be restricted or require administrative privileges. Interpretation may require knowledge of application compatibility. | Shimcache entries may be deleted or aged out, affecting data availability. Interpretation may require knowledge of Windows file system and program behavior. |

# 11.3 *UserAssist*

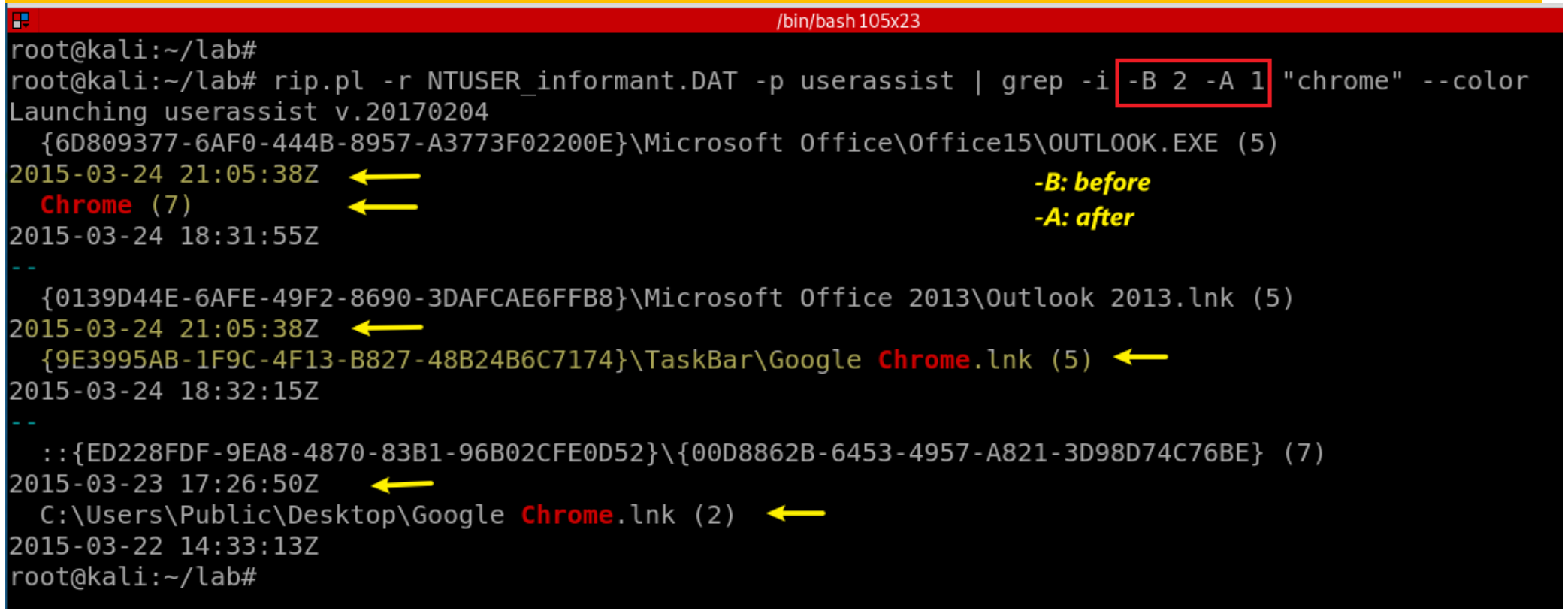
- Microsoft uses *UserAssist* to populate a user's start menu with frequently used applications.
  - Every GUI-based program launched from the desktop are tracked
- These values are located in each user's NTUSER.DAT
  - ROT-13 encoded.
  - *Timestamp of last run*
  - *Count*: *HKCU\Software\Microsoft\Windows\CurrentVersion\Explorer\UserAssist\{GUID}\Count*
- Good to analyze the behaviors of users


/bin/bash 68x20
root@kali:~/lab#
root@kali:~/lab# rip.pl -l | grep -i assist
88. userassist_tln v.20180710 [NTUSER.DAT]
- Displays contents of UserAssist subkeys in TLN format
161. userassist v.20170204 [NTUSER.DAT]
- Displays contents of UserAssist subkeys
root@kali:~/lab#

List executed programs by the user informant



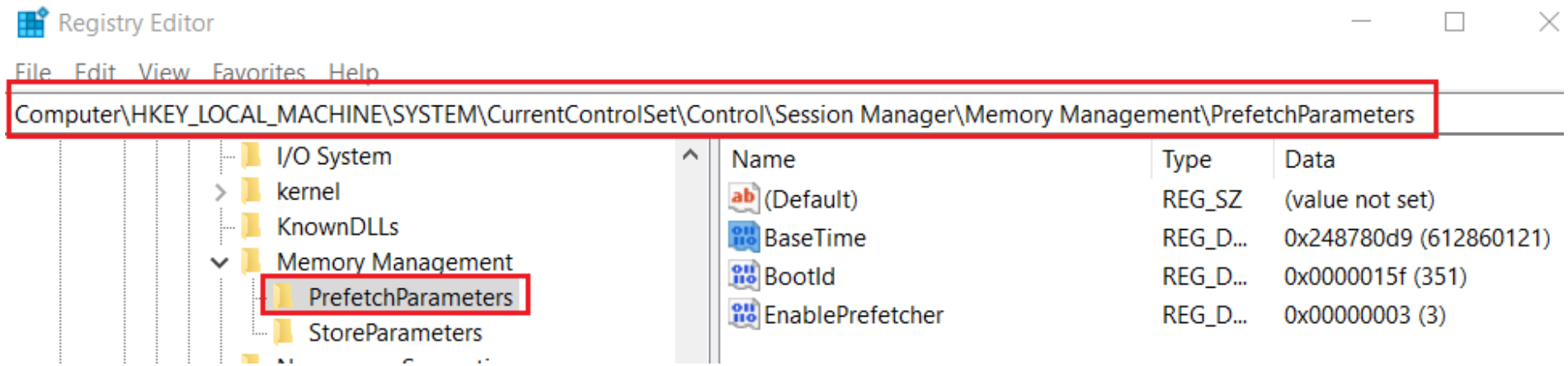
Search if "*chrome*" has been executed by the user informant. Show lines before and after the matches



# 11.4 *Prefetch*

- A memory management technology
  - Save prefetch (executables) information in .pf
    - `%SYSTEMROOT%\Prefetch\*.pf`
  - To improve customer experience,
  - Introduced by Microsoft in Windows XP and Windows 2003 Server.
- Preloads most frequently used software (with parameters) into memory
  - To speed the operating system booting and application launching.
- *SuperFetch* On Windows Vista
  - An improved version of *Prefetch*

## Prefetch registry configuration (enable/disable)



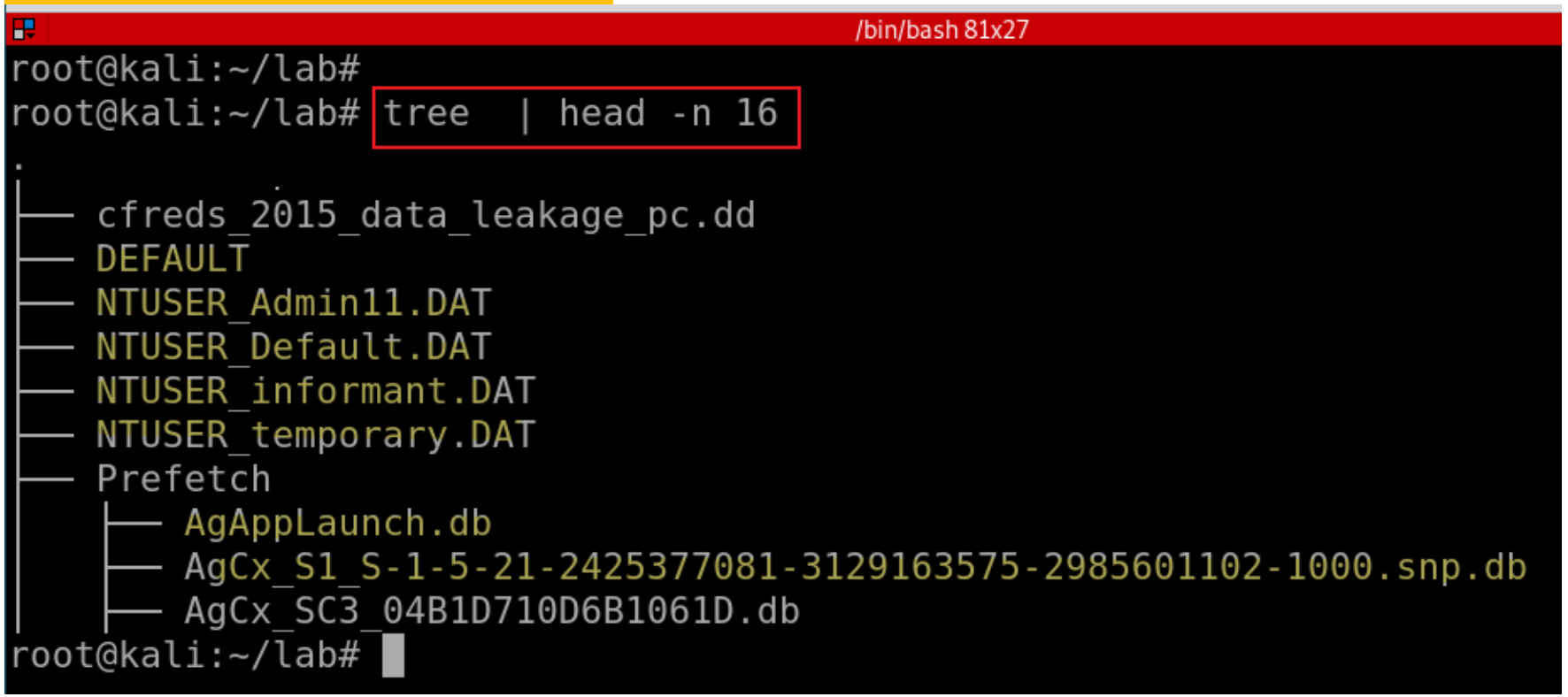

## Exam *prefetch* setting from registry

```
root@kali:~/lab#
root@kali:~/lab# rip.pl -r SYSTEM -p prefetch
Launching prefetch v.20200515
prefetch v.20200515
(System) Gets the the Prefetch Parameters

EnablePrefetcher    = 3

0 = Prefetching is disabled
1 = Application prefetching is enabled
2 = Boot prefetching is enabled
3 = Both boot and application prefetching is enabled
root@kali:~/lab#
```

Verify *Prefetch* folder has *.pf* files



Verify *prefetch* command

```
┌──(student㉿kali80)-[~/lab]
└─$ prefetch.py --help
usage: prefetch.py [-h] [-c] -f FILE

optional arguments:
  -h, --help            show this help message and exit
  -c, --csv             Present results in CSV format
  -f FILE, --file FILE  Parse a given Prefetch file
```

Parse *Prefetch* of *chrome.exe*

```
┌──(student㉿kali80)-[~/lab]
└─$ prefetch.py  -f Prefetch/CHROME.EXE-D999B1BA.pf | more

=========================
CHROME.EXE-D999B1BA.pf
=========================

Executable Name: CHROME.EXE

Run count: 71

Last Executed: 2015-03-24 21:05:38.872938

Volume Information:
   Volume Name: \DEVICE\HARDDISKVOLUME2
   Creation Date: 2015-03-25 11:08:36.956950
   Serial Number: ca0c7a48

Directory Strings:
   0: \DEVICE\HARDDISKVOLUME2\PROGRAM FILES (X86)
   1: \DEVICE\HARDDISKVOLUME2\PROGRAM FILES (X86)\GOOGLE
   2: \DEVICE\HARDDISKVOLUME2\PROGRAM FILES (X86)\GOOGLE\CHROME
   3: \DEVICE\HARDDISKVOLUME2\PROGRAM FILES (X86)\GOOGLE\CHROME\APPLICATION
   4: \DEVICE\HARDDISKVOLUME2\PROGRAM FILES (X86)\GOOGLE\CHROME\APPLICATION\41.0.2272.101
   5: \DEVICE\HARDDISKVOLUME2\PROGRAM FILES (X86)\GOOGLE\CHROME\APPLICATION\41.0.2272.101\LOCALES
--More--
```

-f FILE, --file FILE  Parse a given *Prefetch* file

resource loaded from different paths

Parse *Prefetch* of *chrome.exe* and save the results to *.CSV*

```
┌──(student㉿kali80)-[~/lab]
└─$ prefetch.py  -f Prefetch/CHROME.EXE-D999B1BA.pf -c
Timestamp,Executable Name,MFT Seq Number,MFT Entry Number,Prefetch Hash,Run Count
2015-03-24 21:05:38.872938,CHROME.EXE,d999b1ba,3,65216,71
```

-c, --csv  Present results in CSV format

Table 1: Comparison of UserAssist and Prefetch in Digital Forensics

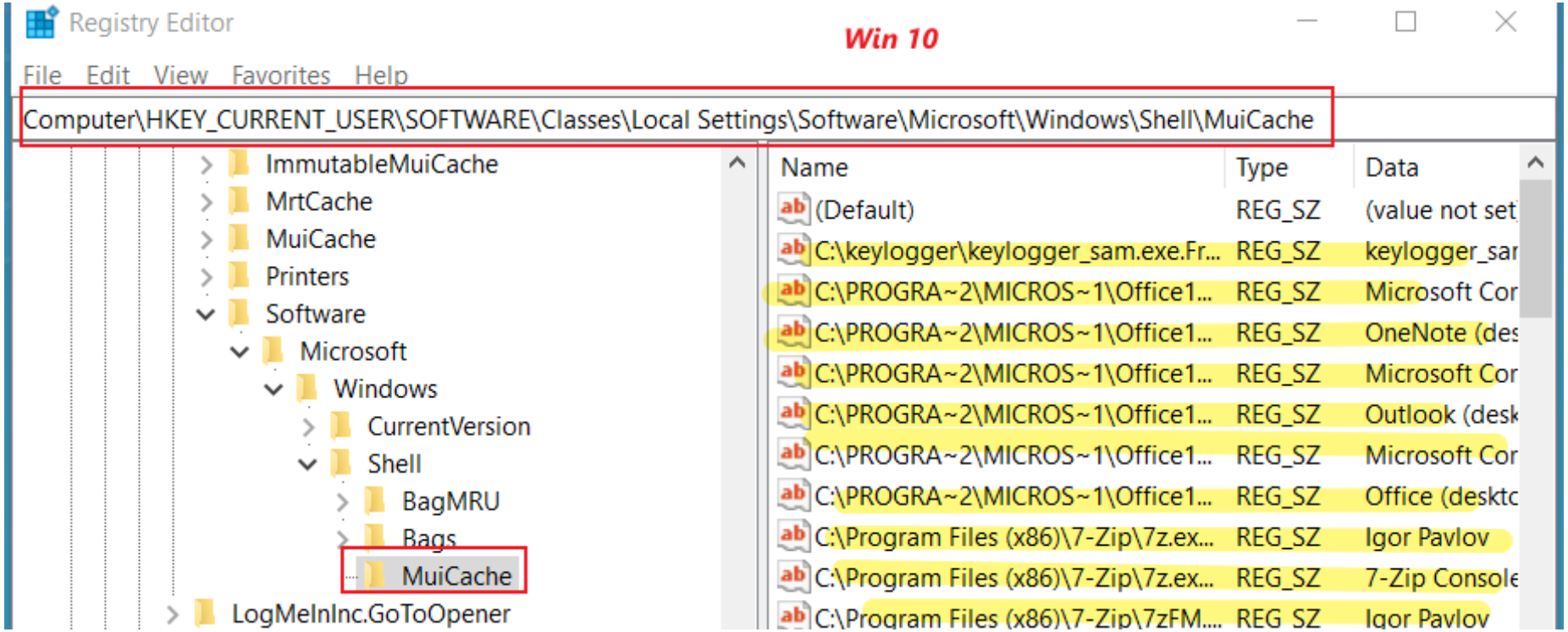
| **Aspect** | **UserAssist** | **Prefetch** |
|---|---|---|
| Forensic Relevance | Provides insights into user-specific program execution history, helping forensic analysts determine which programs were run by users | Offers information about frequently executed programs and their associated files, which can be useful for forensic investigations to understand program usage |
| Location | Registry-based, stored in the Windows Registry | File-based, stored in the 'C:' directory |
| Data | Contains data about program execution, including program names, execution counts, and timestamps | Maintains data about frequently run executables, including file names, paths, and execution order |
| Update Frequency | Updated as programs are executed, reflecting user-specific program usage | Continuously updated as programs are run to optimize program loading |
| Use Cases | Used in forensic analysis to determine which programs were executed on a Windows system and when they were last run. It can provide evidence of user activity. | Used in forensic investigations to understand which programs were frequently run on a system, aiding in timeline analysis and identifying suspicious activity |
| Challenges | Requires access to the Windows Registry, which may be restricted or require administrative privileges. Analysis may require interpreting encoded program names. | Prefetch files may be deleted or missing, affecting data availability. Interpretation of Prefetch data may require knowledge of Windows file system and program behavior. |

# 11.5 MuiCache: Multilingual User Interface

- What is MUI
  - To support multiple language for software
- Drawback
  - the MUI scheme is that it's a bit slower
- Solution: MUI caching for localized strings
  - When the right version of a string is retrieved from MUI file for a given app, it's stored in the registry.
  - Then if the string is needed again, it can be retrieved from the registry, which is faster than having to open up the MUI file again.

Windows 2000, Windows XP, Windows Server 2003:
HKEY_CURRENT_USER\Software\Microsoft\Windows\ShellNoRoam\MUICache.

Windows 10:



## Search for *muicache* plugin

```
/bin/bash 81x27
root@kali:~/lab#
root@kali:~/lab# rip.pl -l | grep -i muicache
91. muicache_tln v.20130425 [NTUSER.DAT,USRCLASS.DAT]
   - Gets EXEs from user's MUICache key (TLN)
141. muicache v.20200525 [NTUSER.DAT,USRCLASS.DAT]
   - Gets EXEs from user's MUICache key
root@kali:~/lab#
```

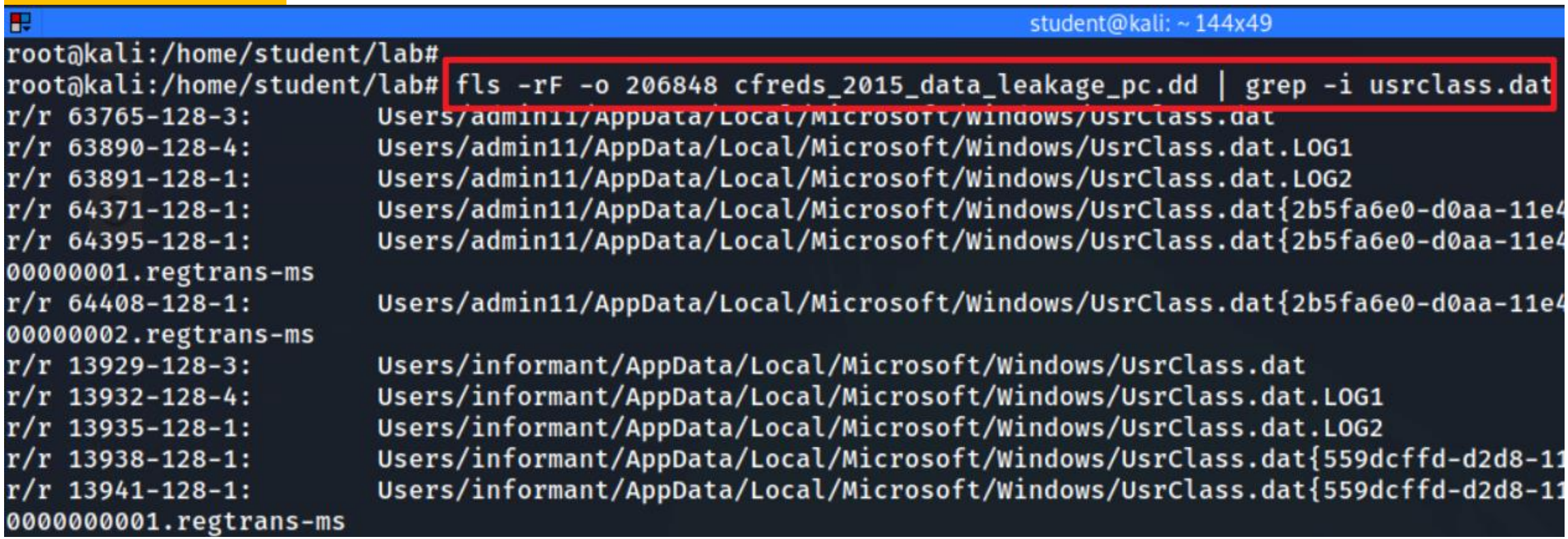
## Exam *muicache*

```
/bin/bash 81x27
root@kali:~/lab#
root@kali:~/lab# rip.pl -r NTUSER_informant.DAT -p muicache
Launching muicache v.20200525
muicache v.20200525
(NTUSER.DAT,USRCLASS.DAT) Gets EXEs from user's MUICache key

Software\Microsoft\Windows\ShellNoRoam\MUICache not found.

Local Settings\Software\Microsoft\Windows\Shell\MUICache not found.
root@kali:~/lab#
```

Search *usrclass.dat*



Extract *usrclass.dat*

```
root@kali:/home/student/lab#
root@kali:/home/student/lab# icat -o 206848 cfreds_2015_data_leakage_pc.dd 70107 >  usrclass_informant.dat
root@kali:/home/student/lab#
root@kali:/home/student/lab# ls usrclass_informant.dat
usrclass_informant.dat
root@kali:/home/student/lab#
```

Search *muicache* from *usrclass.dat*

```
root@kali:/home/student/lab#
root@kali:/home/student/lab# rip.pl -r usrclass_informant.dat -p muicache
Launching muicache v.20200525
muicache v.20200525
(NTUSER.DAT,USRCLASS.DAT) Gets EXEs from user's MUICache key

Software\Microsoft\Windows\ShellNoRoam\MUICache not found.

Local Settings\Software\Microsoft\Windows\Shell\MUICache not found.
root@kali:/home/student/lab#
```

# Summary of 11.

[File] Windows Prefetch folder \Windows\Prefetch*.pf Executable file paths and their execution timestamps (+ execution counts)

[File] IconCache \Users\informant\AppData\Local\IconCache.db Executable file paths and their icon images

[Reg] UserAssist HKU\informant\Software\Microsoft\Windows\CurrentVersion\Explorer\UserAssist*\Count\ Executable file paths and their execution timestamps (+ execution counts)

[Reg] Application Compatibility (Shimcache) HKLM\SYSTEM\ControlSet###\Control\Session Manager\AppCompatCache\ Executable file paths and their modified timestamps

[Reg] Application Compatibility Cache HKU\informant\Software\Microsoft\Windows NT\CurrentVersion\AppCompatFlags\Compatibility Assistant\ Executable file paths and their modified timestamps

[Reg] MuiCache HKU\informant\Software\Classes\Local Settings\Software\Microsoft\Windows\Shell\MuiCache\ Executable file paths