
Digital Forensics (IFS4102) Lab 6:

Windows & Linux Forensics

Lab Objectives

In this lab, you will perform **additional Windows forensics tasks** as well as a **Linux forensics task**. More specifically, you want to:

1. Discover a file hidden inside NTFS's **Alternate Data Stream**.
2. Inspect and analyse a **Windows event log file**.
3. Extract and analyse the **prefetch files** of your target machine.
4. Extract and analyse the **shortcuts** of your target machine.
5. *(Optional)* Extract and analyse the **jump lists** of your target machine.
6. *(Optional)* Extract and analyse the **thumbnail caches** of your target machine.
7. *(Optional)* Perform a **Linux log analysis** using **grep**.

Task 1 (Win-FWS & Lin-FWS): NTFS' Alternate Data Stream

Notes:

- As explained in our previous Windows forensics discussion, a file can be made hidden in NTFS using its *Alternate Data Stream (ADS)* feature.
- Now, you want to **discover and remove/extract** the existence of a binary file hidden in NTFS using both **live analysis** and **disk image analysis**.

Task 1-1 (Win-FWS): Discovering a Hidden File in a NTFS

Disk using Live Analysis

Notes:

- In this task, you can discover a hidden binary file in a NTFS disk using **some commands** as part of your *live analysis*.

Steps:

1. **Create** a text file named `file.txt` as your sample *host/carrier file*.
2. **Hide** an executable file, e.g. `malware.exe`, as an *ADS stream* of the `carrier file.txt` file:

```
type malware.exe > file.txt:malware.exe
```
3. Note that the hidden binary can be **executed** simply by invoking:

```
file.txt:malware.exe
```
4. In your *live analysis*, you can **discover** the hidden executable file using the following DOS command in Windows 7 and later:

```
dir /R
```

5. Alternatively, you can use **Sysinternals' streams tool** from, which you can download from: <https://docs.microsoft.com/en-us/sysinternals/downloads/streams>.

You can run `streams` on a suspected **target file** as follows:

```
streams <file>
```

Additionally, you can use its option `-s` to **recurse** subdirectories.

You can also **remove** the available streams using `-d`:

```
streams -d <file>
```

6. Lastly, you can also use **PowerShell's commands** to detect and remove an ADS.

To **list** all data streams of a file, run:

```
Get-Item -path <file> -stream *
```

To **remove** a particular stream, run:

```
Remove-Item -path <file> -stream <stream-name>
```

Task 1-2 (Win-FWS/Lin-FWS): Discovering a Hidden Binary File in an Acquired NTFS Disk using Disk Forensics

Notes:

- Now, you want to discover and extract a hidden binary file in an acquired NTFS disk using **TSK** as part of your *static analysis*.

Steps:

1. **Hide** an executable file, e.g. `malware.exe`, again as an ADS stream of a host/carrier file, e.g. `carrier.txt`, in a NTFS disk.
2. Do a static acquisition of the target disk.
3. Now, you want to perform **disk forensics** on the target disk using **TSK**.
4. First, run **mm1s** on the target acquired disk to find out the **layout** of the disk including the **starting sector** of the target NTFS partition:

```
# mm1s <image-file>
```

5. To display all files in the NTFS file system, you can run **fls** by specifying the correct offset of the target partition:

```
# fls -o <offset-no> -r -p <image-file>
```

6. To display only all ADS-hidden files, you can **filter** the output for any names containing the colon (":") character.
7. You should be able to notice the hidden binary file, such as in the following:
`r/r 10820-128-2: path-to-folder/file.txt:malware.exe`
8. Note that the **file ID and stream information** of the hidden binary file are reported using **three numbers**, i.e. 10820-128-2:

- a. The first no, i.e. 10820, is the host file's file ID (similar to inode).

- b. The next no, i.e. 128, is the MFT attribute that corresponds to a \$DATA attribute.
 - c. The last no, i.e. 2, is the sequence ID that refers to the alternate stream.
9. You can get some **extra information** about the file whose file ID is 10820 using the TSK's `istat` command:
- ```
istat -o <offset-no> <image-file> 10820
```
10. You can then **extract** the ADS-hidden file using TSK's `icat` command:
- ```
# icat -o <offset-no> <image-file> 10820-128-2 >
<extracted-file>
```
11. Lastly, to **find out more** about the extracted binary file, you can perform the following on the extracted file:
- a. Hash look-up the binary file;
 - b. Scan it using your anti-virus software or VirusTotal;
 - c. Execute it in a sandbox environment and debug it using a Windows debugger; or
 - d. Analyze it using various static and dynamic analysis tools.

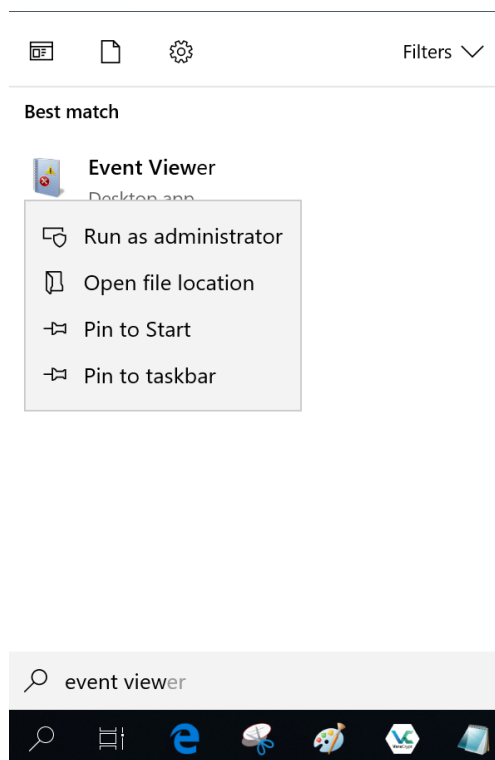
Task 2 (Win-FWS): Inspecting a Windows Event Log File

Notes:

- Now, you want to use Windows' built-in **event-viewer program** called *Event Viewer* to open and inspect a Windows event log file.
- For a sample event log file, you can download `Security.evtx` from:
https://drive.google.com/file/d/1Q4wR_cae08I0PdVSu97w5aZQI1rgYIAI/view?usp=sharing. Its MD5 value is 83fe57fd75239fda659aff2998e6f4c0.

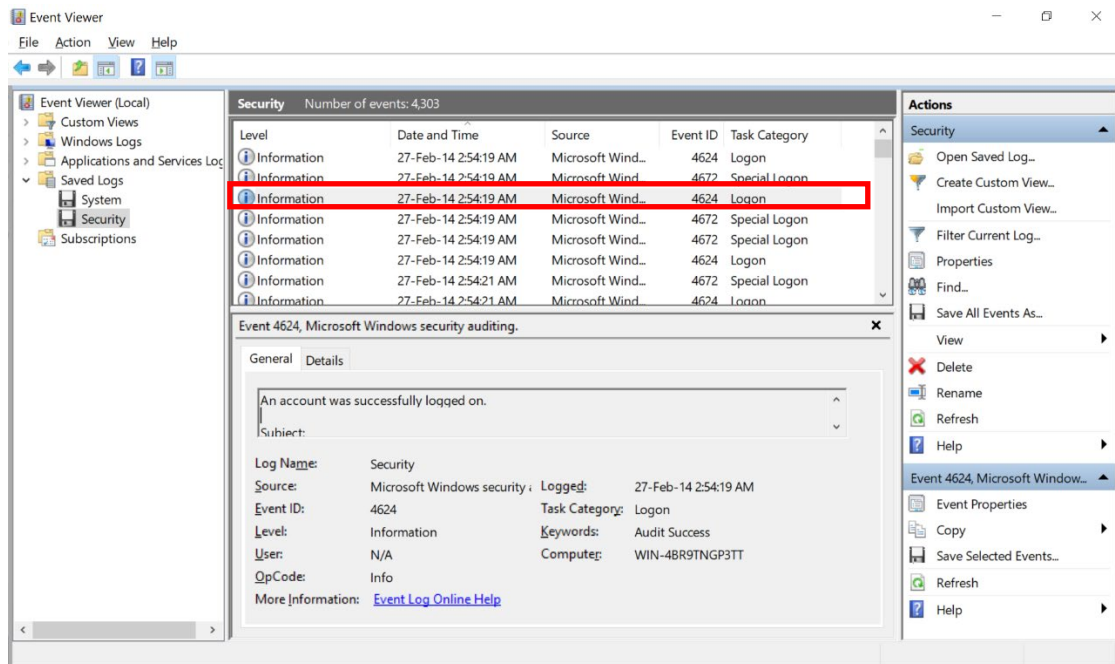
Steps:

1. Simply double-click the downloaded `Security.evtx` file to invoke the **Event Viewer** executable. Alternatively, you can go to your Windows' search button to find the "Event Viewer" program. Right-click the shown entry, and then select "Run as administrator" as shown below.

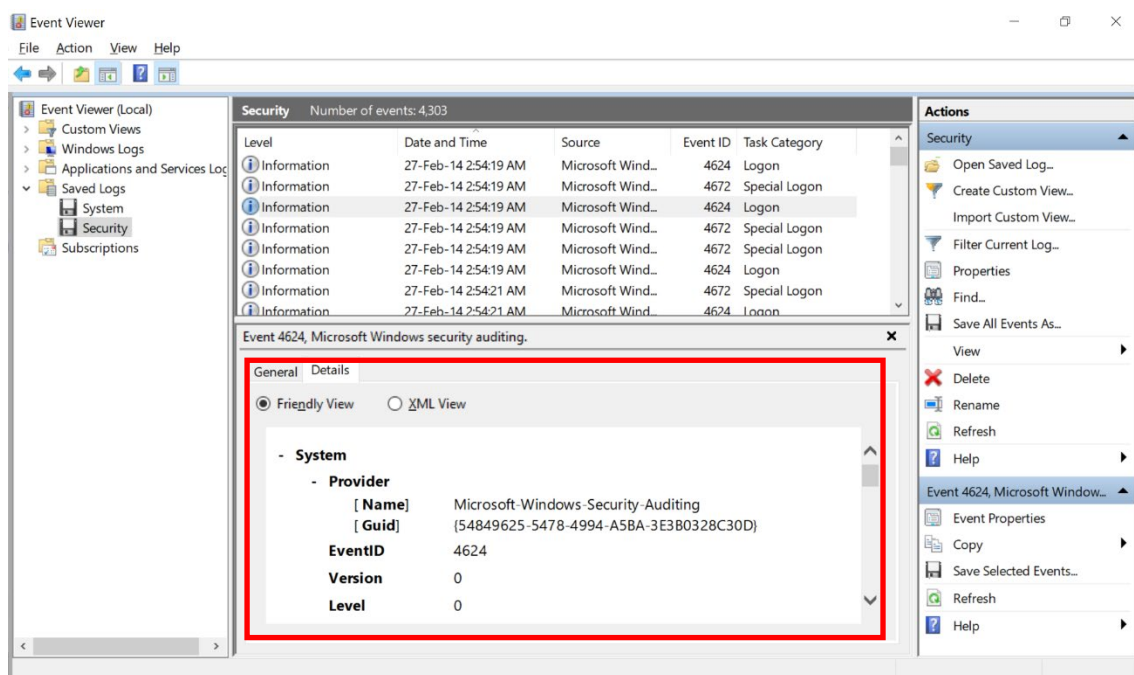


2. The event log file came from a target computer running Windows 7.

Do check **some of the events** listed by Event Viewer. You can then select the entry with “**Logon**” task category timestamped 27-Feb-14 2:54:19AM as shown below. Notice that the Event ID shown is **4624**.

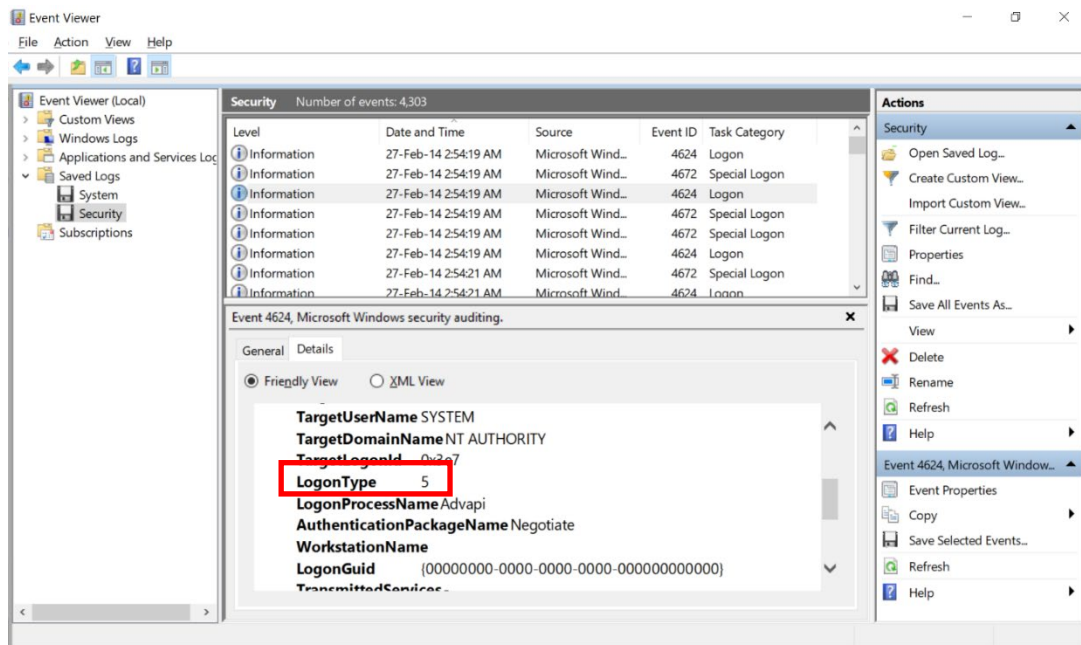


3. Click on the “Details” tab to see the details of the selected event. Expand the + symbol next to the System to expand various recorded event details.

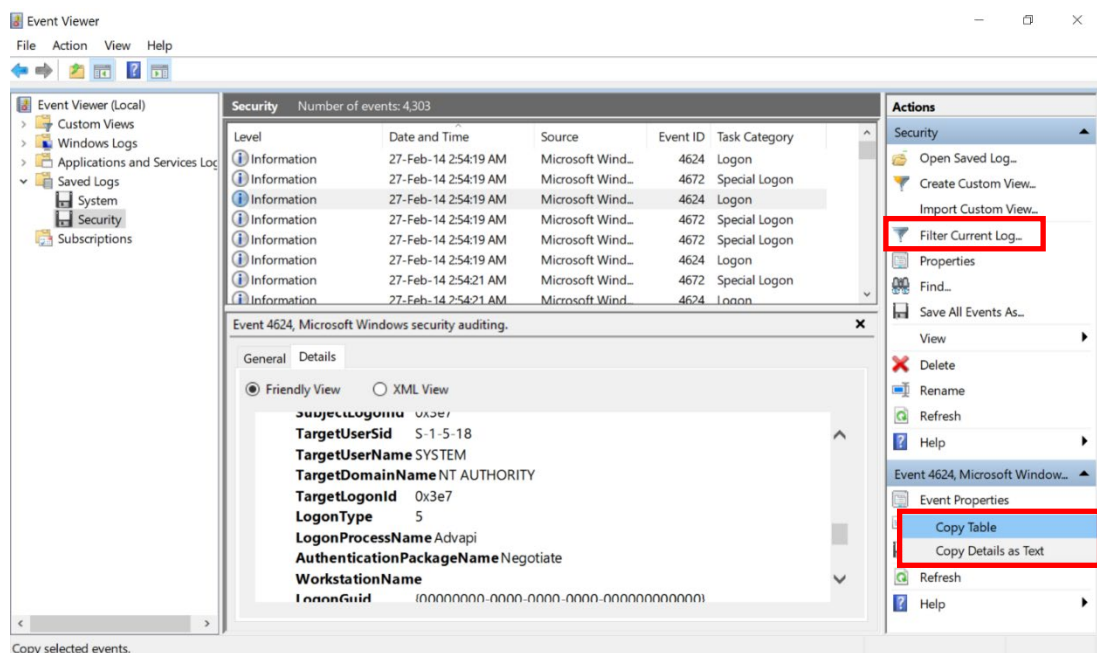


4. Do check the event's **LogonType** value. How did the user log into the system based on the value?

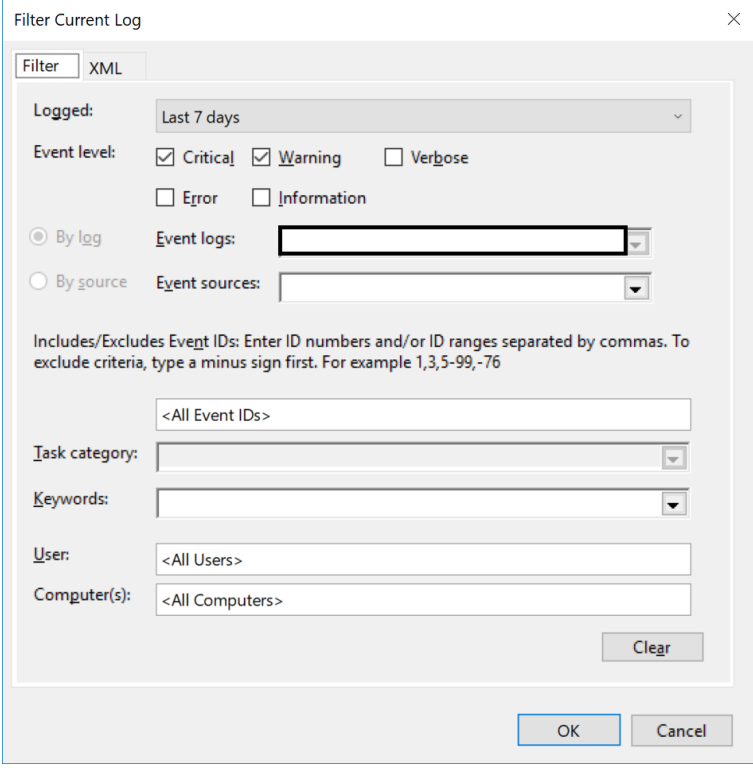
Also find out the time when the user did log on, which was stored in UTC.



5. If required for your reporting, you can export the entry information by clicking the “Copy” menu item in the “Actions” pane at the right side of the window.



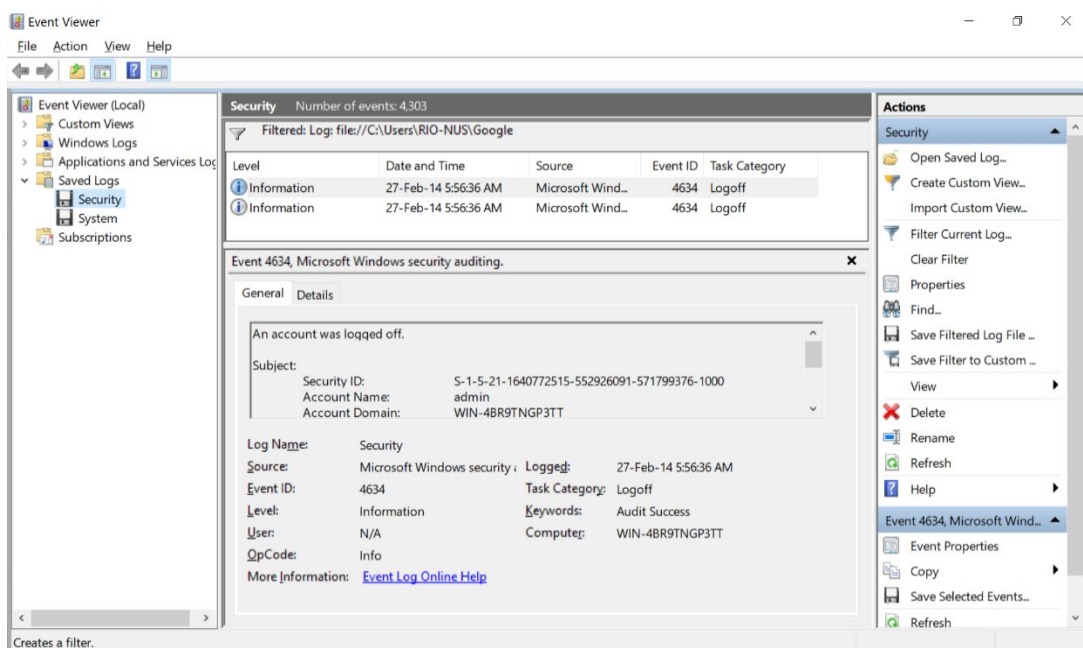
6. You can also **filter** the events by selecting the “Filter Current Log” menu item in the Actions pane. Specify your filter criteria in the shown dialog box.



The "Filter Current Log" dialog box is shown with the "Filter" tab selected. It contains the following fields and options:

- Logged:** Last 7 days (dropdown)
- Event level:**
 - ☒ Critical
 - ☒ Warning
 - ☐ Verbose
 - ☐ Error
 - ☐ Information
- By log:** ☒ (selected)
 - Event logs:** (dropdown menu)
- By source:** ☐
 - Event sources:** (dropdown menu)
- Includes/Excludes Event IDs:** Enter ID numbers and/or ID ranges separated by commas. To exclude criteria, type a minus sign first. For example 1,3,5-99,-76.
 - Field: <All Event IDs>
- Task category:** (dropdown menu)
- Keywords:** (dropdown menu)
- User:** <All Users> (dropdown menu)
- Computer(s):** <All Computers> (dropdown menu)
- Buttons:** Clear, OK, Cancel

7. Can you show **all logoff events** only? You can do so by specifying the **4634** Event ID in the “Filter Current Log” dialog box. Also check the related **4647** Event ID.



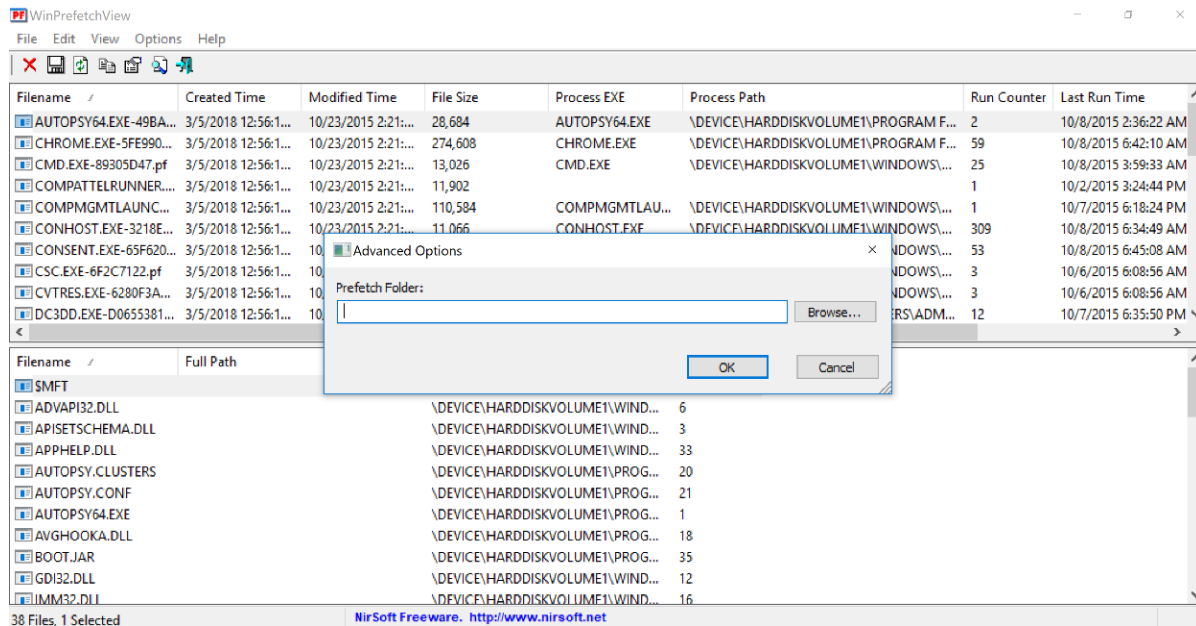
Task 3 (Win-FWS): Analyzing Windows Prefetch Files

Notes:

- Each time that you run an application in your system, a ***prefetch file***, which contains information about the files loaded by the application within the first 10 seconds, is created by Windows. The information is used for **optimizing the loading time** of the application the next time that you run it.
- In this exercise, you will use **WinPrefetchView**, a utility that reads the prefetch files stored in your system. By looking in these files, you can learn which files every application is using.

Steps:

1. **Download** WinPrefetchView from http://www.nirsoft.net/utls/win_prefetch_view.html, and extract its zipped file's contents to a folder, e.g. Desktop.
2. **Launch** WinPrefetchView.
3. On a Windows machine, it will, by default, locate the local machine's prefetch folder C:\Windows\Prefetch. If you want to **analyze** copied prefetch files from a target machine instead, select "Options", and then "Advanced Option", and subsequently click the "Browse" button to locate the folder where the copied prefetch files reside. For testing purpose in this exercise, just use the **local prefetch files** in your forensic machine.
4. WinPrefetchView will display the extracted **prefetch file entries** as shown below. The upper pane of the WinPrefetchView displays the list of all prefetch files in your system. When you select a file in the upper pane, the lower pane displays the **list of files that were loaded** by the application (within the first 10 seconds) in the previous times you used it.



5. Click an entry on the upper pane, and observe its reported **attributes**.

You should be able to find out the following:

- How many times was the application run:
by inspecting the value of the “**Run Counter**” column.
- When was the last time the application run:
by inspecting the value of the “**Last Run Time**” column.
- The executable file’s location on the hard drive:
by inspecting the value of the “**Process Path**” column.

6. Now, observe the entries show in the lower pane, which represent the files (e.g. DLL, configuration files, input files) **that were read** by the application within the first 10 seconds after it was launched.

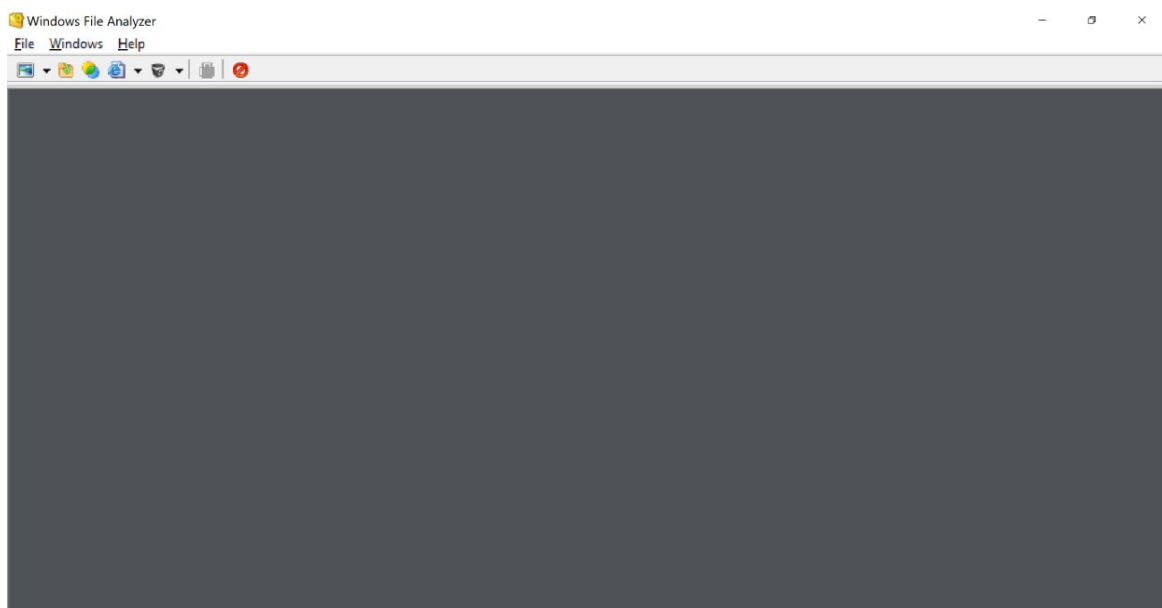
Task 4 (Win-FWS): Analyzing Windows Shortcuts Files

Notes:

- We will now analyze *shortcuts* on a target Windows machine by utilizing a tool called **Windows File Analyzer (WFA)**. Available shortcuts are useful to tell which files, application, and volumes that were previously accessed on your target machine.
- In addition to extracting and analyzing Windows shortcuts, the WFA tool can also extract and analyze prefetch files and thumbnail caches, among others. In this exercise, we will just use its shortcut analyzer feature.
- *Note:* To access the directories that store shortcut on your local machine, you may need to set your Explorer to also **show hidden/system files**.

Steps:

1. Download MiTeC **Windows File Analyzer (WFA)** from <http://www.mitec.cz/wfa.html>, and extract its zipped file's contents.
2. Launch MiTeC WFA by double-clicking its `WFA.exe` file.



3. At the main menu, select “File” then “Analyze Shortcuts...”.

For testing purpose, you can just **locate your local machine’s**

%AppData%\Microsoft\Windows\Recent\ folder (the default);

or point to a folder that stores your target machine’s shortcut files.

You can also try locating **other folders** that contain shortcuts, such as:

- a. %AppData%\Microsoft\Office\Recent\;
 - b. Desktop.
4. Select the Windows version that generated the shortcuts to analyze, and click “Ok”. The shortcuts in the target directory will be extracted and shown in the WFA’s window.
 5. You can inspect the available shortcuts and their following reported **attributes**:
 - a. Linked pathname;
 - b. The time when it was created;
 - c. The time when it was last accessed;
 - d. File size;
 - e. The serial no of the volume that stores the shortcut;
 - f. The network MAC address of the used computer.
 6. If needed, print a report on the displayed shortcuts by clicking the “Report...” button.

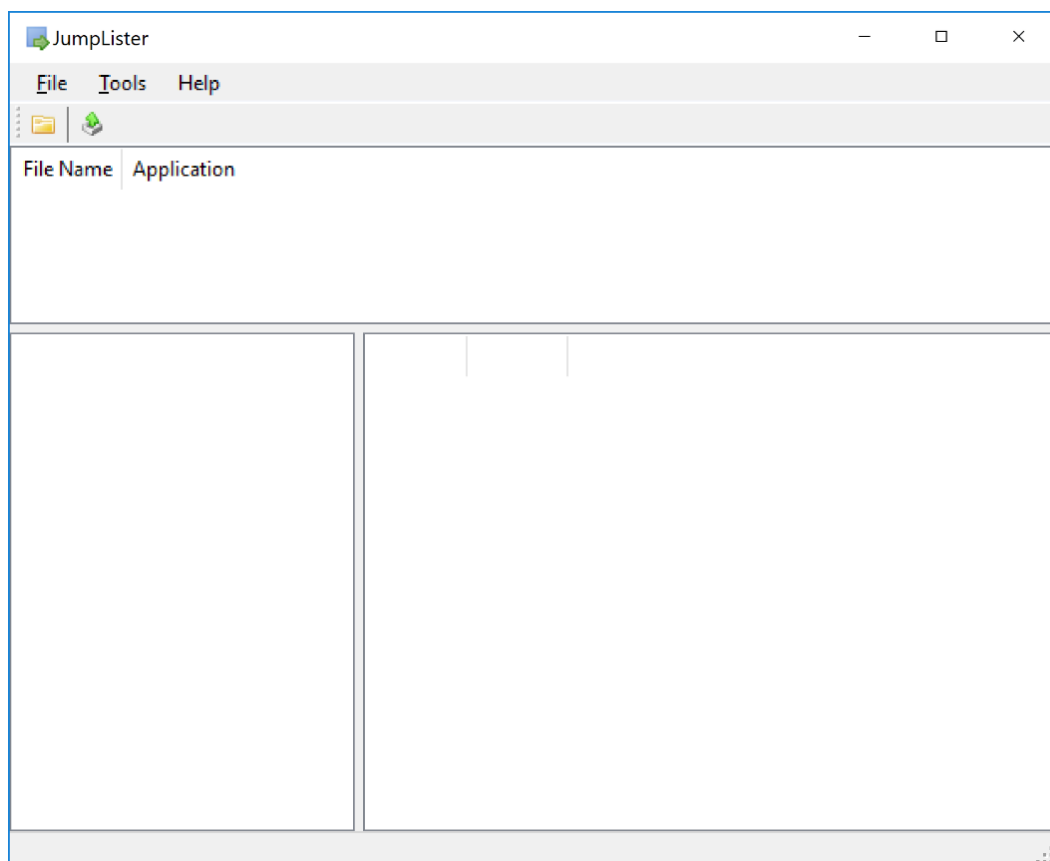
[Optional] Task 5 (Win-FWS): Analyzing Jump List Files

Notes:

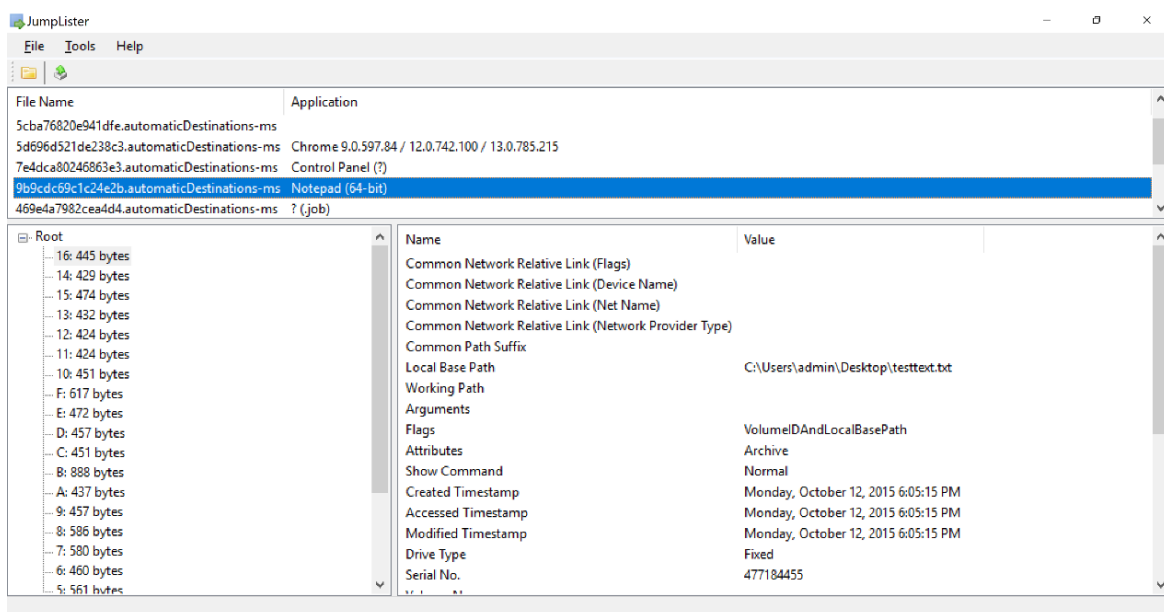
- Now, we will analyze *jump lists* on a target Windows machine, which function very much like shortcuts.
- For this, we will use a popular tool called **JumpLister** from woanware.

Steps:

1. Download JumpLister's zip file from <https://github.com/woanware/JumpLister/tree/master/Release>, and extract it.
2. Launch JumpLister by double-clicking the file `JumpLister.exe`.



3. At the main menu, select “File”, then “Load”, and **locate** the folder that stores jump lists. The folders are usually:
 - a. %APPDATA%\Microsoft\Windows\Recent\AutomaticDestinations
 - b. %APPDATA%\Microsoft\Windows\Recent\CustomDestinations.
4. If you see multiple files under a folder, shift-click and highlight **all the files** in the folder, and then click “Open”. Note that you must also correctly select the file type (i.e. “Jump List” or “Custom Jump List”) according to the folder used. The available automatic or custom jump lists will be displayed as shown below.



5. Click a jump list in the **upper pane**. The **lower left pane** will tree-list the items. Clicking an entry in this tree list will show **its content** in the lower right pane.
6. You can check various useful pieces of **information** related to a selected jump list, such as its: local base path, MAC timestamps, as well as Birth object ID timestamps.

[Optional] Task 6 (Win-FWS): Analyzing Thumbnail Caches

Notes:

- Now, we will analyze *thumbnail cache database* file(s) on your target Windows machine. You can use **MiTeC Windows File Analyzer (WFA)** from <http://www.mitec.cz/wfa.html>, which you have previously downloaded, for this exercise.

Steps:

1. Launch WFA.
2. Access the WFA's main menu, select "File", then "Analyze Thumbnail Database".
3. Open an available **thumbnail cache database** (thumbcache_*.db) file on your local machines, which is usually stored under:
C:\Users\<User>\AppData\Local\Microsoft\Windows\Explorer
folder.
4. Inspect the shown thumbnail images, together with their attributes, such as:
 - a. Filenames;
 - b. Timestamps.

***[Optional]* Task 7 (Lin-FWS): Analyzing Linux Log using Grep**

Notes:

- In this task, you want to analyse **text-based log files** on your Linux workstation by using the old handy **grep** tool.
- For a sample Linux log file, download `sample-sshd.log` from:
https://drive.google.com/file/d/1bwec2r5S3Qt_hpZAPTAU2c_YQcHtVPK9/view?usp=sharing.

Steps:

1. Use **grep** to find out the following pieces of information from the log.
2. Can you find the entries for **logged failed password** for admin?
You can just specify your grep's target string, e.g. by running:

```
grep 'Failed password for admin' sample-sshd.log
```
3. Now, you need to refer to the given “**Regular Expression Language - Quick Reference**” link (<https://docs.microsoft.com/en-us/dotnet/standard/base-types/regular-expression-language-quick-reference>).
4. How do you specify your **regex** for grep if you want to find the string 41630 or 41635? Hint: please use the **alternate construct** “|”.
5. Write a grep expression to search for **IP addresses** prefixed with “218.49.”.
6. Suppose the log file contains domain names.
Write a grep expression to search for a **domain name**.