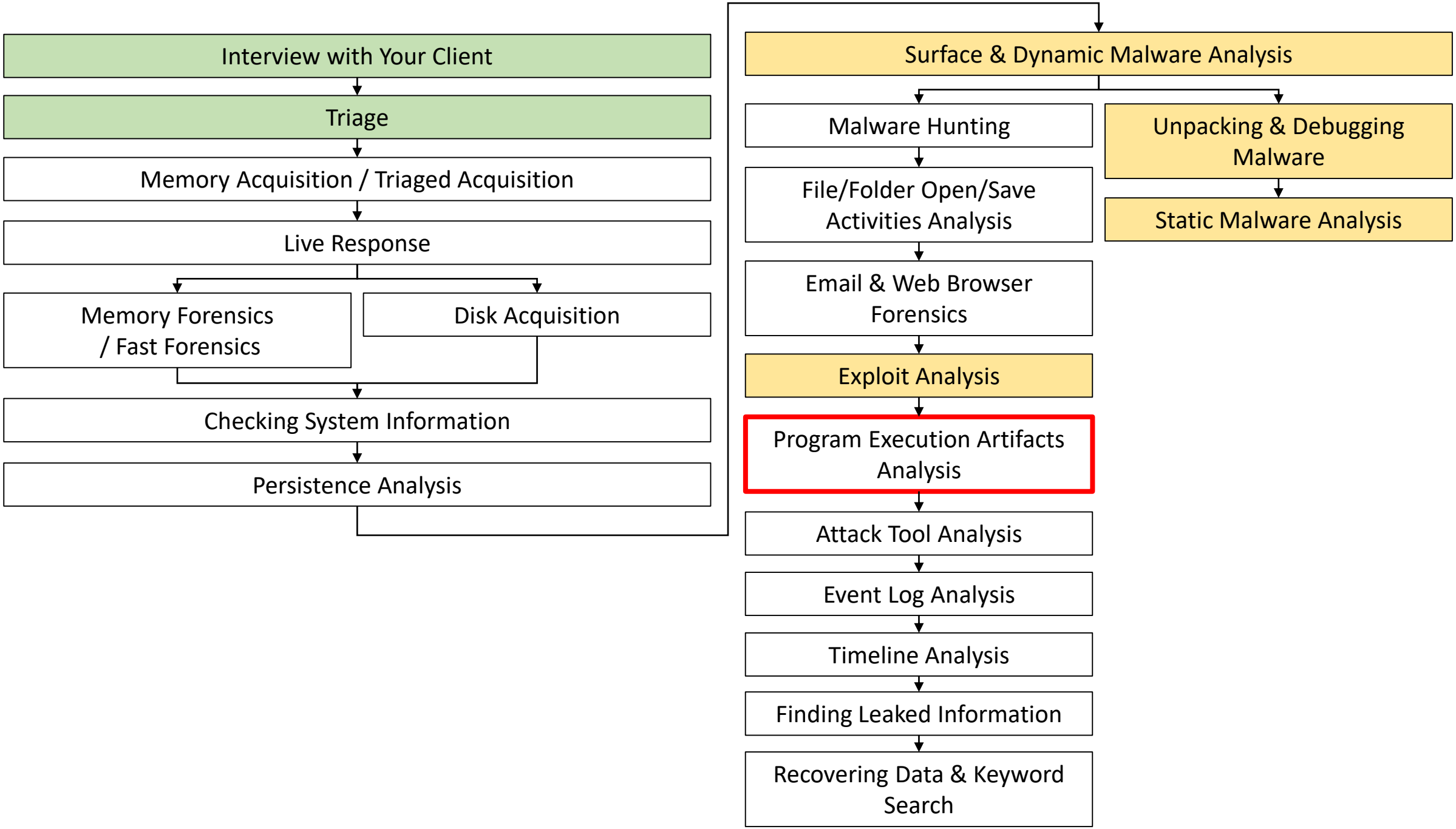
# Program Execution Artifacts Analysis


Interview with Your Client
Triage
Memory Acquisition / Triaged Acquisition
Live Response
Memory Forensics / Fast Forensics
Disk Acquisition
Checking System Information
Persistence Analysis
Surface & Dynamic Malware Analysis
Malware Hunting
File/Folder Open/Save Activities Analysis
Email & Web Browser Forensics
Exploit Analysis
Program Execution Artifacts Analysis
Attack Tool Analysis
Event Log Analysis
Timeline Analysis
Finding Leaked Information
Recovering Data & Keyword Search
Unpacking & Debugging Malware
Static Malware Analysis

# Program Execution Artifacts

- When programs are executed on Windows, several “records” of execution remains on the computer.

- By examining the “records”, it may be possible to track the programs that were executed during the intrusion.

# A Variety of Program Execution Artifacts

- There is a variety of program execution artifacts.
- The following URL can help you know what execution artifacts are in your OS.

| Artifact | Prefetch | ShimCache | MUICache | Amcache | RecentFileCache.bcf | Syscache.hve | Microsoft-Windows-TaskScheduler (200/201) | LEGACY_* Registry Keys | Microsoft-Windows-Application-Experience Program-Inventory | Microsoft-Windows-Application-Experience Program-Telemetry | Application Experience Program Telemetry |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Windows 10 | Yes | Yes | Yes | Yes | No | No | Yes | No | Yes | Yes | TBC |
| Windows 8.1 | Yes | Yes | Yes | Yes | No | No | Yes | No | Yes | Yes | TBC |
| Windows 8 | Yes | Yes | Yes | Yes | No | TBC | Yes | No | Yes | Yes | TBC |
| Windows 7 | Yes | Yes | Yes | KB2952664+ | Yes | Yes | Yes | TBC | Yes | Yes | TBC |
| Windows Vista | Yes | Yes | Yes | No | No | No | No | TBC | No | No | TBC |
| Windows XP | Yes | Yes* | Yes | No | No | No | No | Yes | No | No | TBC |
| Windows Server 2019 | If Enabled | Yes | TBC | Yes | TBC | No | TBC | TBC | TBC | TBC | TBC |
| Windows Server 2016 | If Enabled | Yes | Yes | Yes | No | No | Yes | No | TBC | TBC | TBC |
| Windows Server 2012 R2 | If Enabled | Yes | Yes | Yes | No | No | Yes | No | TBC | TBC | TBC |
| Windows Server 2012 | If Enabled | Yes | Yes | Yes | No | No | Yes | No | TBC | TBC | TBC |
| Windows Server 2008 R2 | If Enabled | Yes | Yes | TBC | Yes | Yes | Yes | TBC | TBC | TBC | Yes |
| Windows Server 2008 | If Enabled | Yes | Yes | No | No | TBC | TBC | TBC | TBC | TBC | TBC |
| Windows Server 2003 R2 | If Enabled | Yes | Yes | No | No | No | TBC | Yes | TBC | TBC | TBC |
| Windows Server 2003 | If Enabled | Yes | Yes | No | No | No | TBC | Yes | TBC | TBC | TBC |

https://blog.1234n6.com/2018/10/available-artifacts-evidence-of.html
https://1234n6-my.sharepoint.com/:x:/p/adam/EU3Fk3ec6NdPsSQx1eA1sfwB_R_fRa4tJ4c1FR6WJlWIEA?rtime=w3w-xg_o1kg

# What We Will Learn in This Section

- In this course, we will mainly learn several useful program execution artifacts which you are able to know about attackers' post-exploitation activities such as Prefetch, Shimcache, SRUM and so on. You will learn a few artifacts for identifying legitimate user's activities.

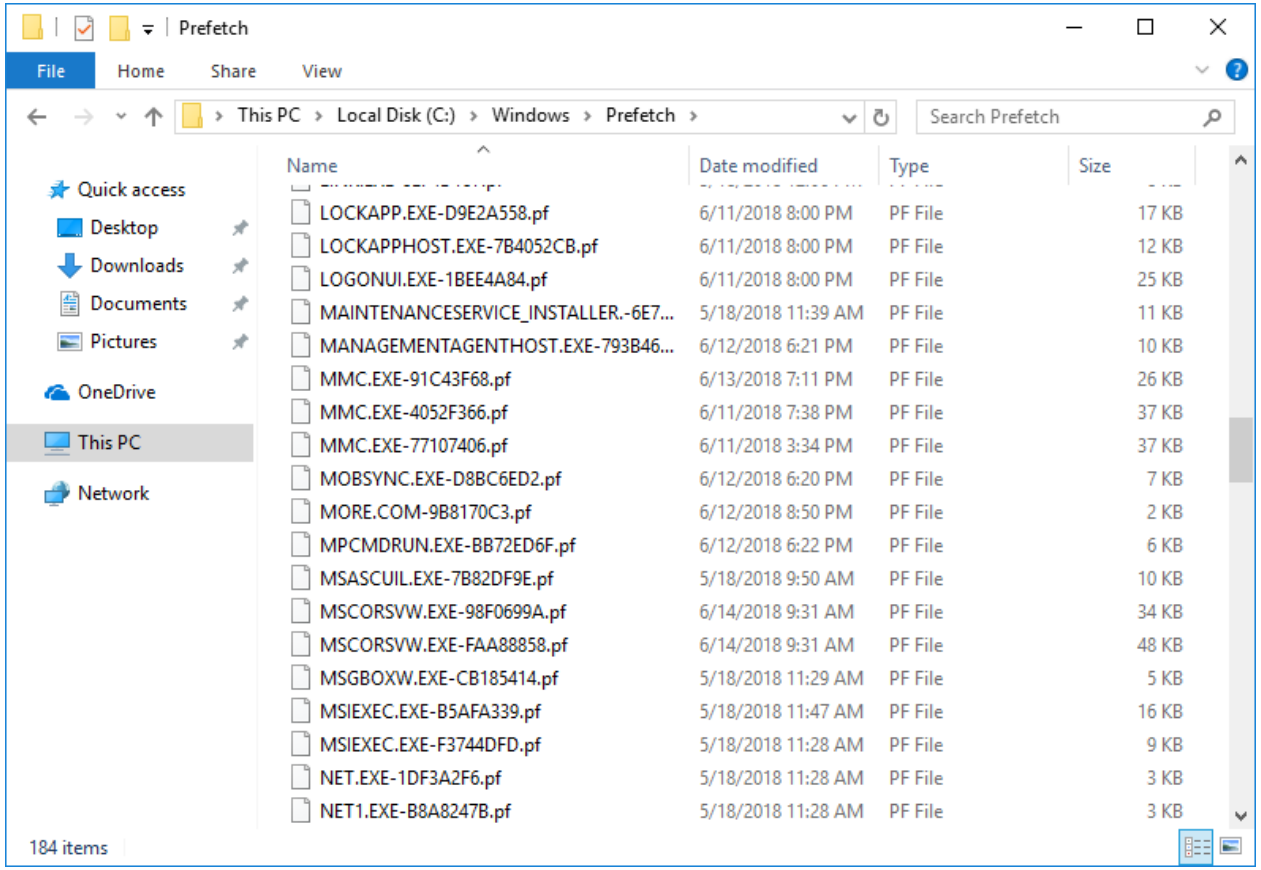
# Prefetch

# Prefetch

- A Windows function for diagnosing user behaviors and creating a list of necessary library files to improve the application execution performances.
- Prefetch was first introduced on Windows XP, and is still available on Windows 10.
- Prefetch is not always enabled: to use the feature, it may be necessary for users to manually enable it.
  - When the host is running a Windows Server OS.
  - In some cases, when the host is running on a virtual machine.

# Prefetch Files

- Prefetch files are created in “Prefetch” folder under “Windows”.
  - File extension is “.pf”.
  - If none of those files exists, prefetch is not enabled on the system.
- Prefetch files are created when the program was executed for the first time.
  - The .pf files will be updated after each program execution.
  - They may be deleted automatically in some cases.
    - E.g. when the number of prefetch files reached the maximum.

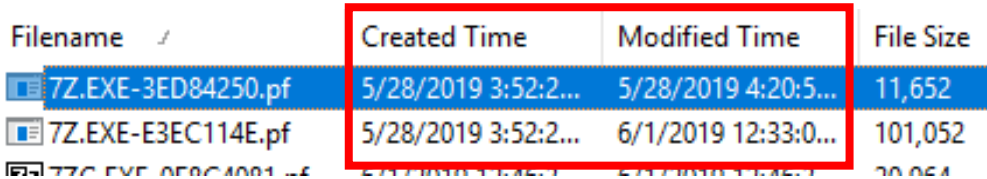


# Characteristics of Prefetch (1/2)

- Number of prefetch entries are as follows.
  - Up to 128 files for Windows 7 and older.
  - Up to 256 files for Windows 8 and newer.
    - Even when the number of prefetch entries reached these limits, files are not deleted instantaneously. They will be removed at a certain timing.
  - The number of prefetch files may be configured with the following registry value.
    - Key: "HKEY_LOCAL_MACHINE\Software\Microsoft\Windows NT\CurrentVersion\Prefetcher“
    - Value: MaxPrefetchFiles
- Each prefetch file stores 8 execution histories for Windows 8 and newer.
  - 1 execution history is stored on older Windows.

# Characteristics of Prefetch (2/2)

- Since prefetch files are updated every time the program is being executed, file system timestamps of the .pf files can be considered as a program execution history record.
  - Creation timestamp as the first execution time.
  - Modification timestamp as the last execution time.
  - However, note that a prefetch file is created or modified at the earlier one of: 10 seconds after the program was executed, or program was terminated. Therefore, there might be a maximum of 10 seconds of difference between the file timestamps and the actual execution time.



| Filename | Created Time | Modified Time | File Size |
|---|---|---|---|
| 7Z.EXE-3ED84250.pf | 5/28/2019 3:52:2... | 5/28/2019 4:20:5... | 11,652 |
| 7Z.EXE-E3EC114E.pf | 5/28/2019 3:52:2... | 6/1/2019 12:33:0... | 101,052 |

- Unlike other histories, prefetch is recorded when the programs were not executed from the Explorer process.
  - However, non-existence of prefetch files does not tell that the program has not been executed. For example, there are cases where the .pf files are not generated when the program was executed with SYSTEM privileges.

# Contents of the Prefetch Files

- What you can find from a Prefetch file includes:
  1. Name and path of the executable file and the process
  2. Number of times the program was executed
  3. Date/time when the program was last executed
     - For Windows 8 and newer, up to eight histories are recorded.
  4. Files and their paths the program is referring to, such as DLL, EXE and other read files
  5. Index of the files the program is referring to in the order of files being called from the program

1 1 1 2 3

| Filename | Created Time | Modified Time | File Size | Process EXE | Process Path | Run Counter | Last Run Time |
|---|---|---|---|---|---|---|---|
| 7Z.EXE-3ED84250.pf | 5/28/2019 3:52:2... | 5/28/2019 4:20:5... | 11,652 | 7Z.EXE | \VOLUME{0000000000000000-c6d20b19}\B... | 2 | 5/28/2019 4:20:56 PM, 5/28/ |
| 7Z.EXE-E3EC114E.pf | 5/28/2019 3:52:2... | 6/1/2019 12:33:0... | 101,052 | 7Z.EXE | C:\PROGRAM FILES\7-Zip\7z.exe | 16 | 6/1/2019 12:33:01 AM, 5/28/ |

4 4 4 5

| Filename | Full Path | Device Path | Index |
|---|---|---|---|
| 7-ZIP.CHM | C:\PROGRAM FILES\7-Zip\7-zip.chm | \VOLUME{01d5159874530c4d-7e748e... | 38 |
| 7-ZIP.DLL | C:\PROGRAM FILES\7-Zip\7-zip.dll | \VOLUME{01d5159874530c4d-7e748e... | 134 |
| 7-ZIP.DLL.TMP | C:\PROGRAM FILES\7-ZIP\7-ZIP.DLL.... | \VOLUME{01d5159874530c4d-7e748e... | 135 |
| 7-ZIP32.DLL | C:\PROGRAM FILES\7-Zip\7-zip32.dll | \VOLUME{01d5159874530c4d-7e748e... | 136 |

# Multiple Prefetch Files

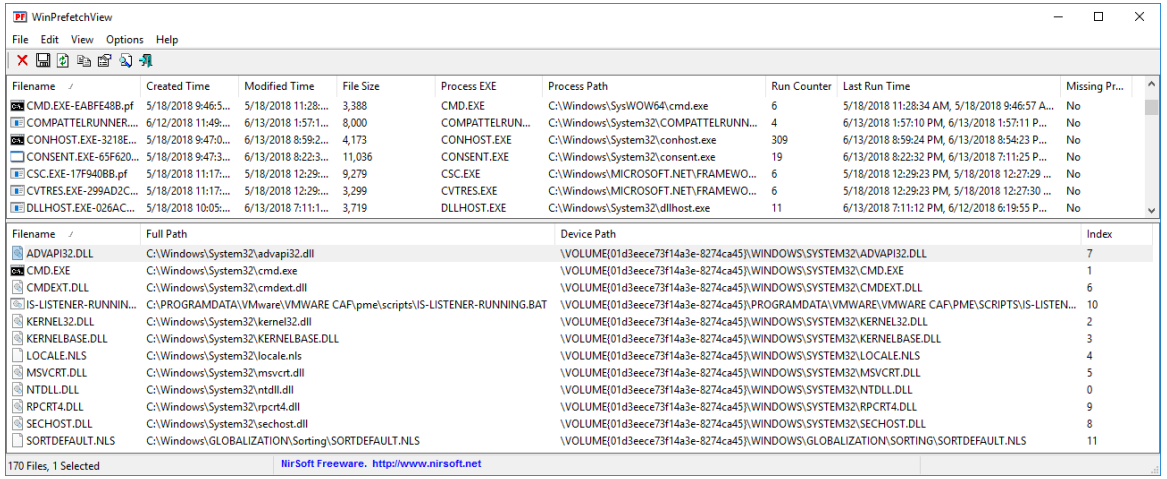
- There may be multiple Prefetch files for the same program name.

| Filename | Created Time | Modified Time | File Size | Process EXE | Process Path | Run Cou |
|---|---|---|---|---|---|---|
| IEXPLORE.EXE-1B894AFB.pf | 6/11/2018 9:41:10 PM | 6/11/2018 9:41:10 PM | 15,846 | IEXPLORE.EXE | C:\PROGRAM FILES\INTERNET EXPLORER\iexplore.exe | 1 |
| IEXPLORE.EXE-F6A52C88.pf | 6/11/2018 9:41:10 PM | 6/11/2018 9:41:13 PM | 20,367 | IEXPLORE.EXE | C:\PROGRAM FILES (X86)\INTERNET EXPLORER\iexplore.exe | 2 |

- If program(s) with a same name was executed in different paths, multiple Prefetch files would be created.

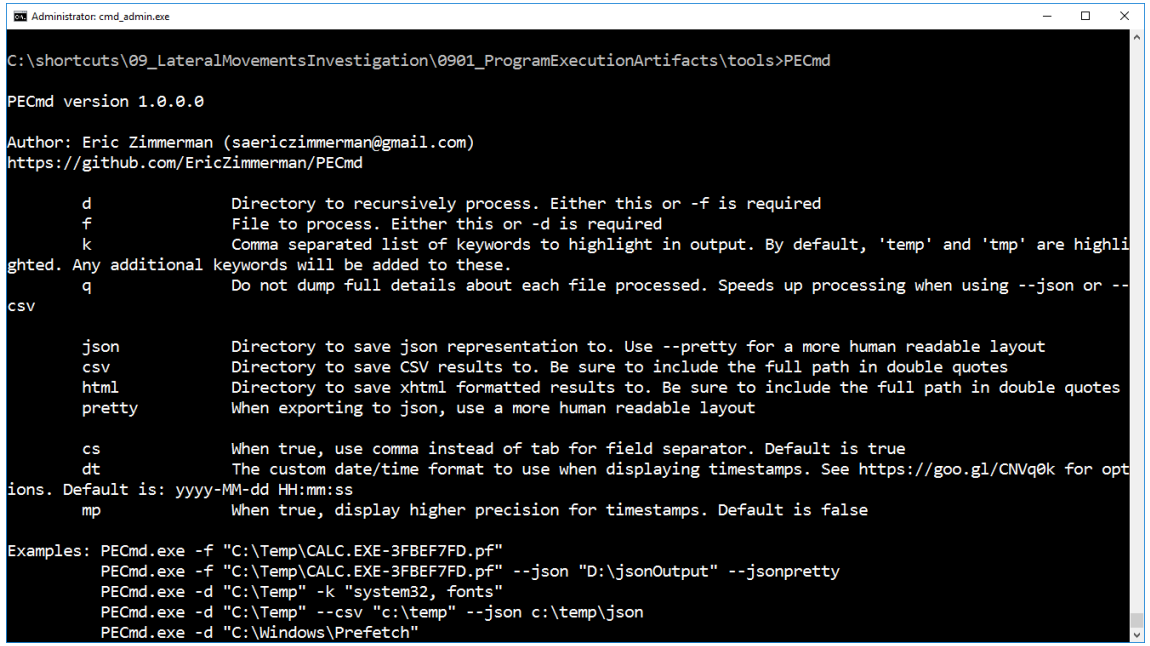
# Viewing Prefetch Files (1/2)

- Prefetch files are binary files.
  - Tools are necessary to see their contents.
- WinPrefetchView: software available on NirSoft website.
  - https://www.nirsoft.net/utils/win_prefetch_view.html
  - Note that the timestamps shown on the tool is the local time of the analysis machine by default.



# Viewing Prefetch Files (2/2)

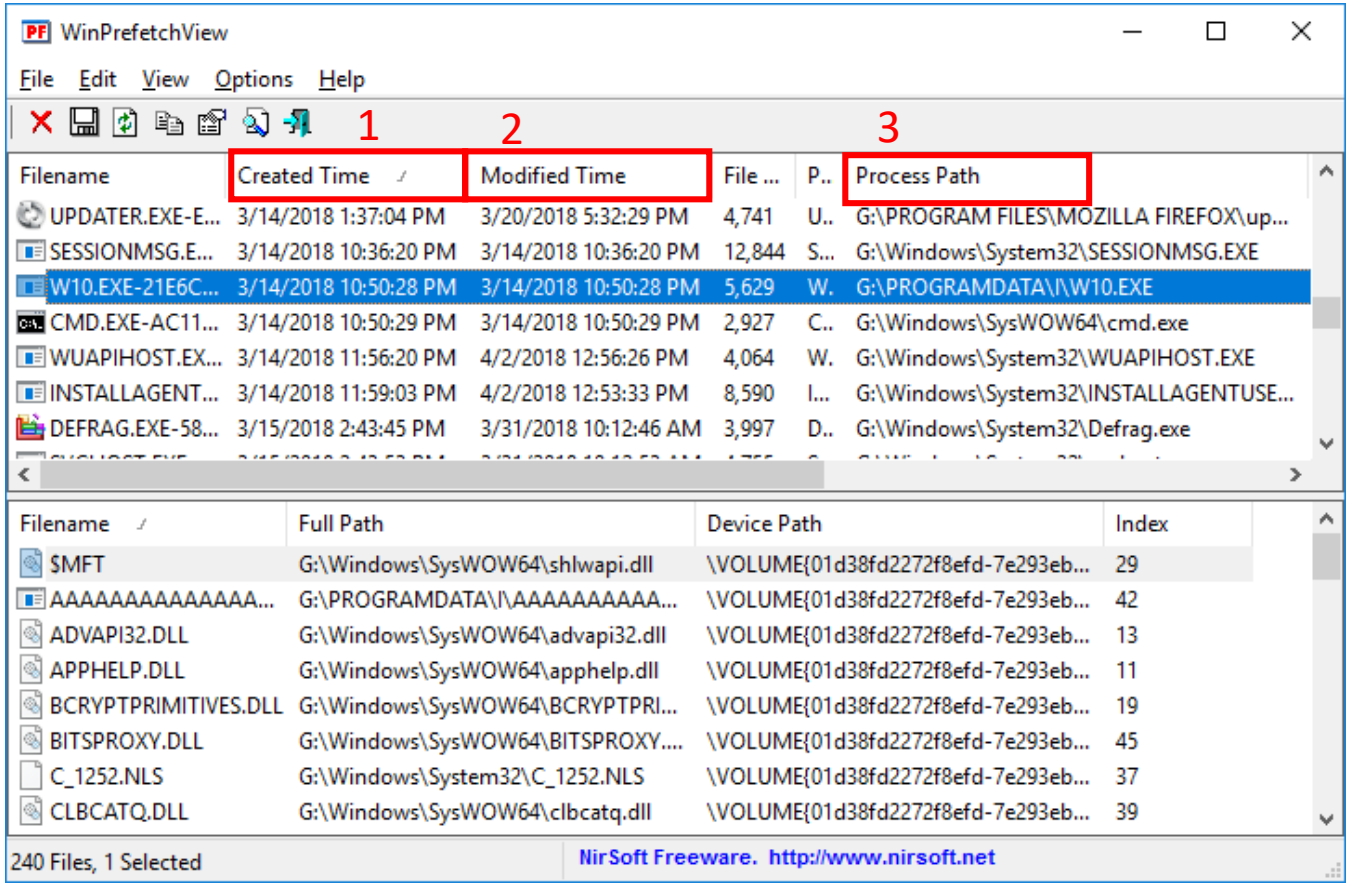
- PECmd
  - Command line tool that can parse prefetch files.
    - https://ericzimmerman.github.io/
  - WinPrefetchView shows all the “Last Run Time” in a single column, but PECmd can split them.
  - PECmd sometimes fails to resolve full paths because it uses a file name based on a “.pf” file to resolve a path, not an actual file name in a record, while WinPrefetchView can succeed. In that case, you should also check the result of WinPrefetchView.
  - Note that the default time zone will be UTC.



# Tips for Analyzing Prefetch (1/3)

Step 1: Start the analysis using WinPrefetchView:

1. Sort by Created Time
   - Timeline of the initial execution can be found.
2. Sort by Modified Time
   - Timeline of the last program execution can be found.
3. Sort by Process Path
   - File paths that do not exist on the original state of the environment may be found.
   - Even if the files are legitimate Windows executables, if the path is under “SysWOW64” folder, the program was 32-bit, not 64-bit.

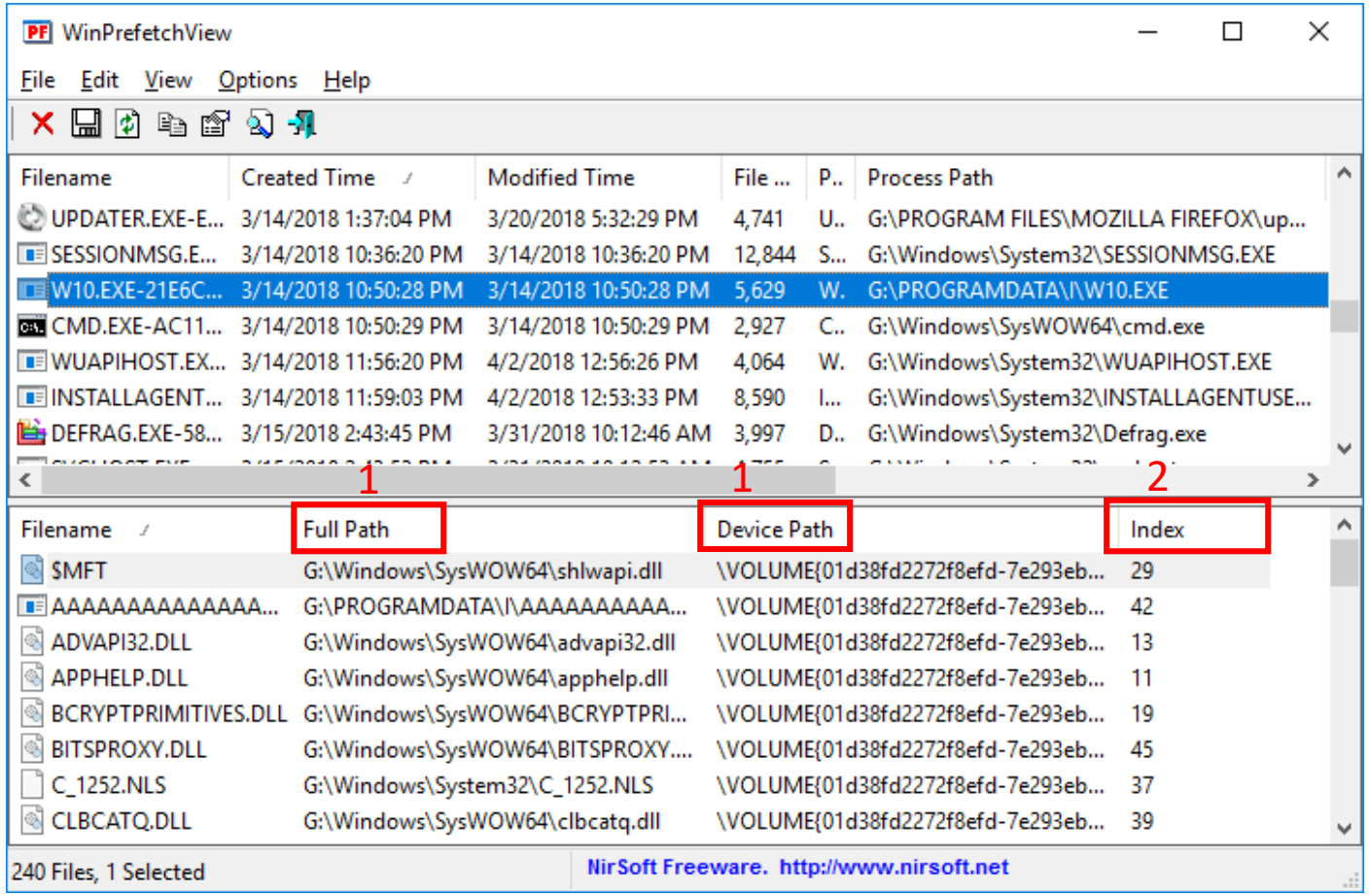



Note that SysWoW64 is NOT a directory for 64 bit, but for 32 bit.
It stands for the System directory for Windows 32 bit On Windows 64 bit.

# Tips for Analyzing Prefetch (2/3)

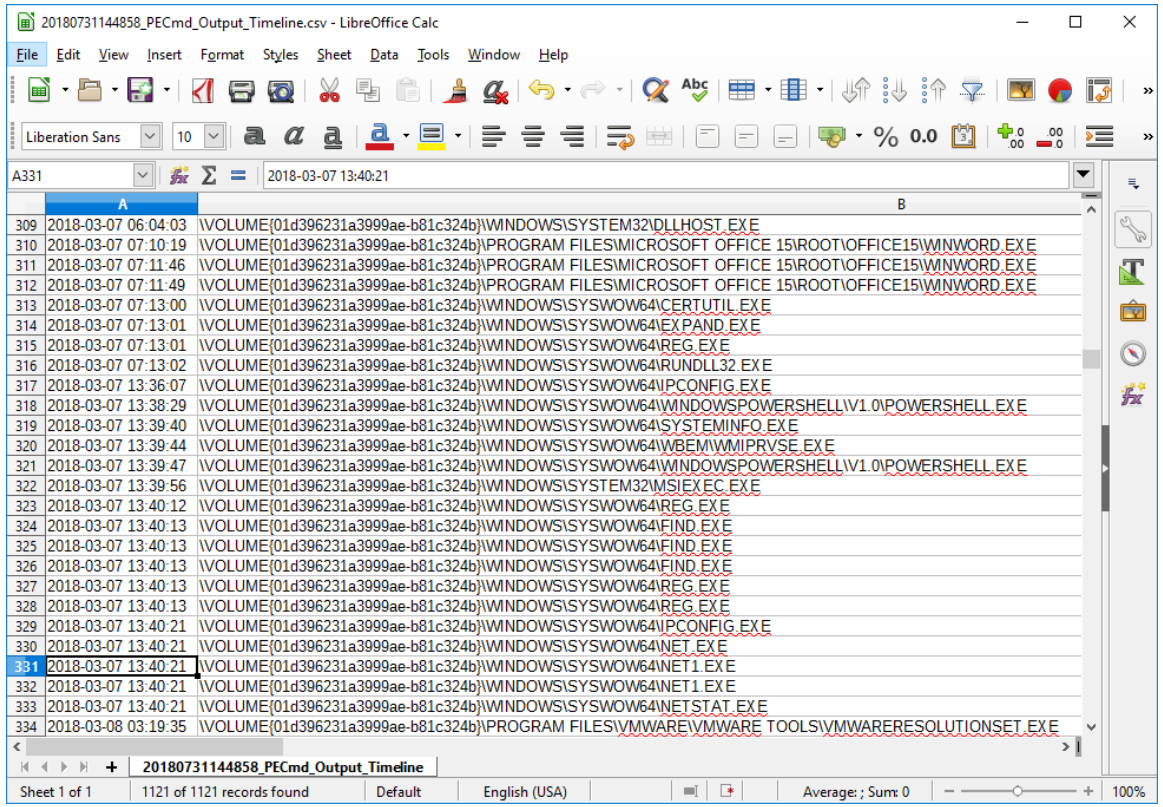
Step 2: When a suspicious entry is found, use the bottom pane to confirm the files used by the program.

1. Sort by Full Path (or Device Path)
   - File paths that do not exist on the original state of the environment may be found.
   - Even if the files are legitimate Windows files, if the path is under “SysWOW64” folder, the program was 32-bit, not 64-bit.
2. Sort by Index
   - Presents the order of files that were read from the program.

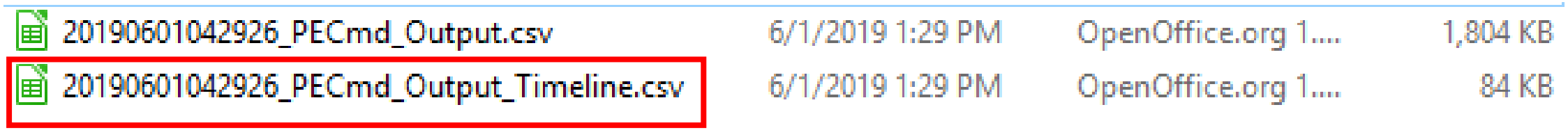


# Tips for Analyzing Prefetch (3/3)

- When looking into the “Last Run Time”, WinPrefetchView cannot treat each history independently.
- PECmd can split those Last Run Time entries.
- You can just do it like the following command.
- You will get two files and the file ends with “_Timeline.csv” is the file which is split by Last Run Time.



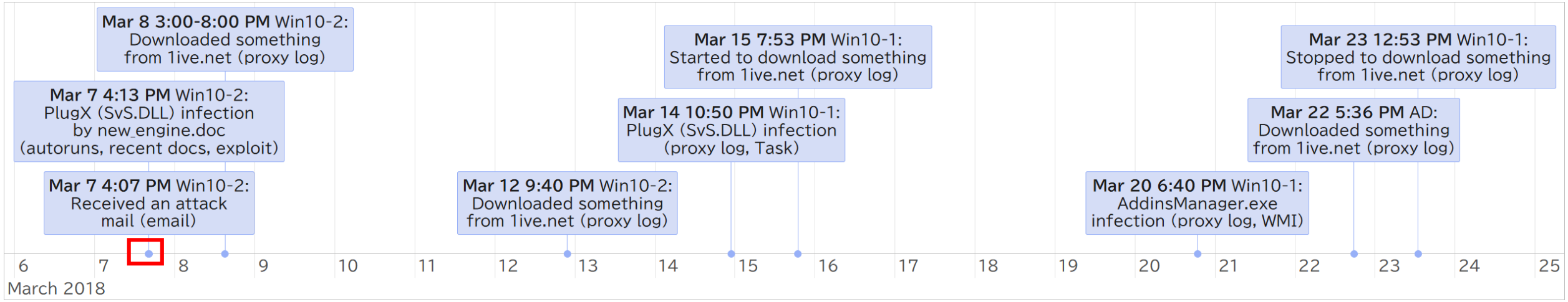

```
PECmd.exe -d PATH_TO_PREFETCH_FOLDER --csv PATH_TO_OUTPUT_FOLDER
```



# Scenario 1 Labs:
# Lab 1: Looking into Prefetch Files for Client-Win10-2

# Looking into Prefetch Files

- The goal of this exercise is to look into Prefetch files, find execution histories, and information regarding related files.
  - This helps us to find the programs used in attacks.
- Since we know the infection date/time, we will look around that time.

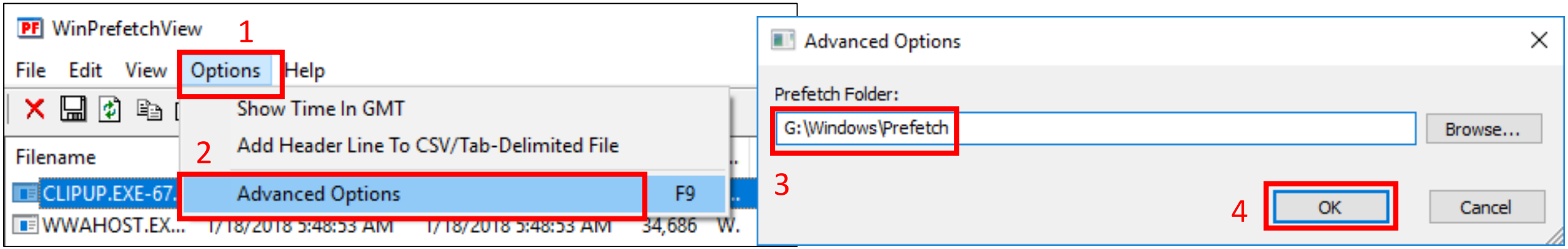


# Mounting E01 File

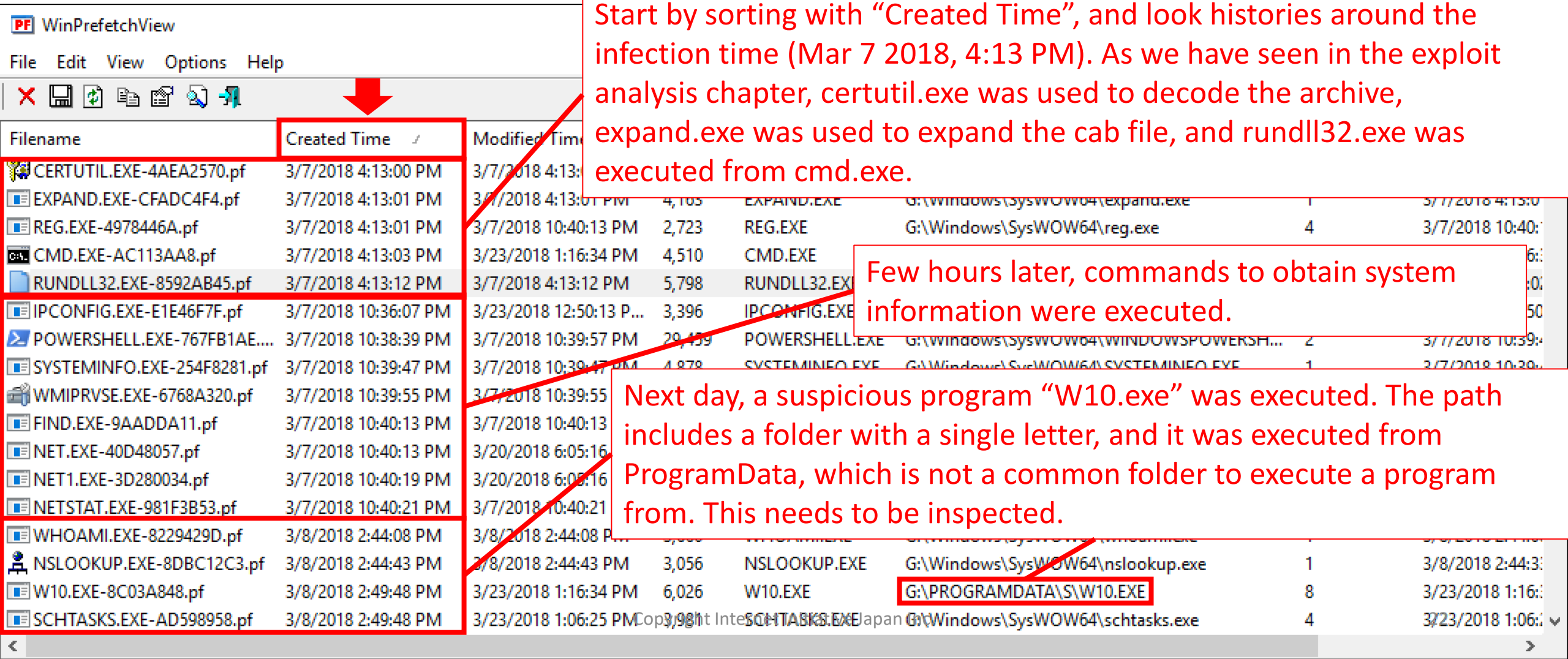
- In this exercise, we will be looking inside the E01 images.
- As we have done, open Arsenal Image Mounter from Shortcuts folder.
  - Shortcuts > 06_LateralMovementsInvestigation > 0601_ProgramExecutionArtifacts > ArsenalImageMounter.exe
  - Accept UAC when prompted.
- Say “OK” to the Arsenal Image Mounter splash screen.
- Mount **Client-Win10-2_honda.E01** file with “**Write Temporary**” option.
  - The “*.pf” files are in **%WinDir%\Prefetch** folder, where permission is necessary for opening.

# Using WinPrefetchView

- From the tools folder, open **winprefetchview.exe**.
- Once opened, the Prefetch contents for the running machine (Analysis Machine) is displayed. Select “Advanced Options” from “Options” menu.
  - Change the Prefetch Folder to **G**:\Windows\Prefetch.
  - See if you can find something interesting.



# Finding Suspicious Activities with WinPrefetchView



# Windows Commands Abused by Attackers – JPCERT/CC

- One of the studies shows that legitimate Windows commands were often used by attackers.
  - Many of the commands you can find in the WinPrefetchView are also listed.

Table 1: Initial Investigation (Top 10 commands)

| Ranking | Command | Times executed |
|---|---|---|
| 1 | tasklist | 155 |
| 2 | ver | 95 |
| 3 | ipconfig | 76 |
| 4 | systeminfo | 40 |
| 5 | net time | 31 |
| 6 | netstat | 27 |
| 7 | whoami | 22 |
| 8 | net start | 16 |
| 9 | qprocess | 15 |
| 10 | query | 14 |

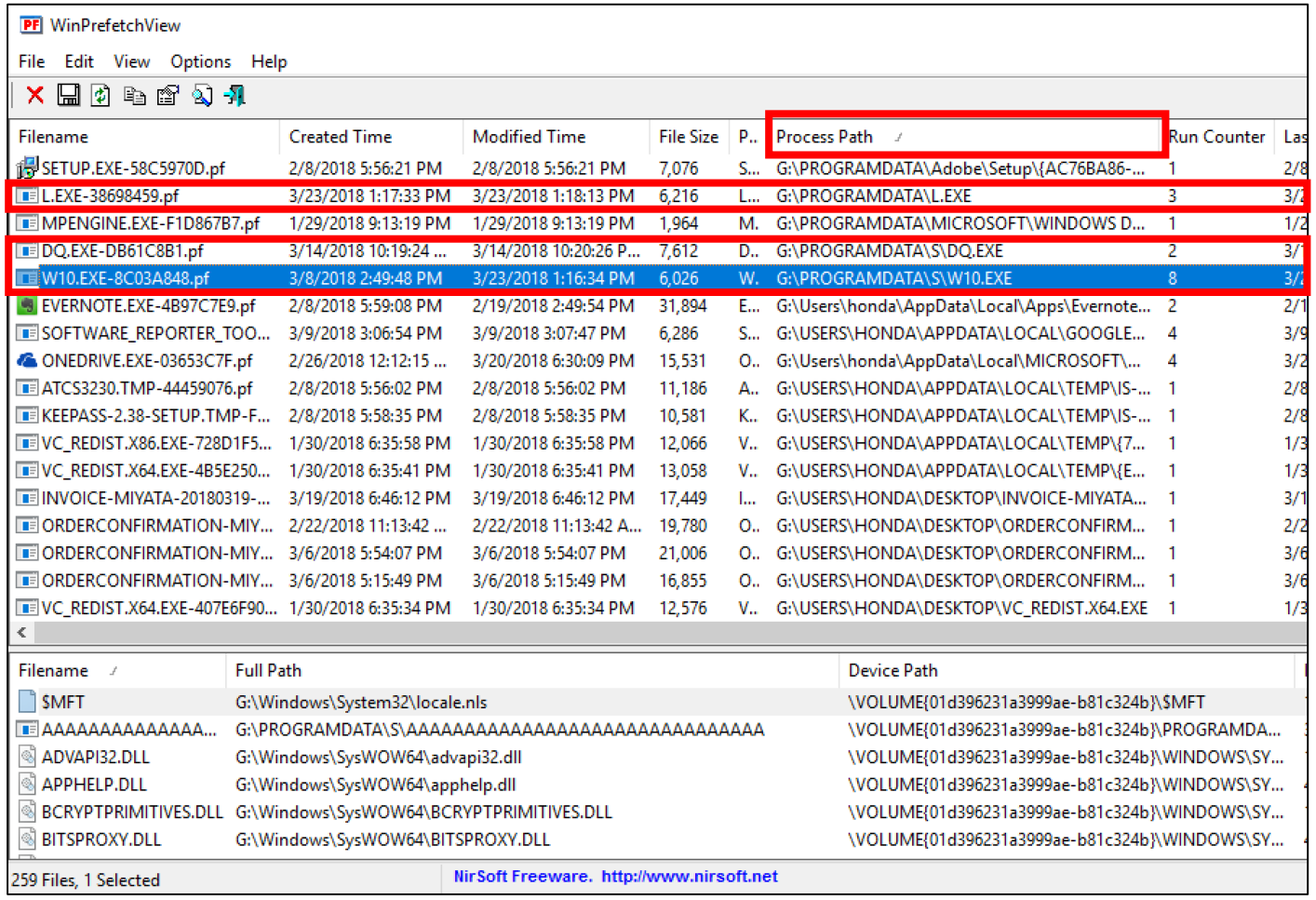
Table 2: Reconnaissance (Top 10 commands)

| Ranking | Command | Times executed |
|---|---|---|
| 1 | dir | 976 |
| 2 | net view | 236 |
| 3 | ping | 200 |
| 4 | net use | 194 |
| 5 | type | 120 |
| 6 | net user | 95 |
| 7 | net localgroup | 39 |
| 8 | net group | 20 |
| 9 | net config | 16 |
| 10 | net share | 11 |

https://blog.jpcert.or.jp/2016/01/windows-commands-abused-by-attackers.html

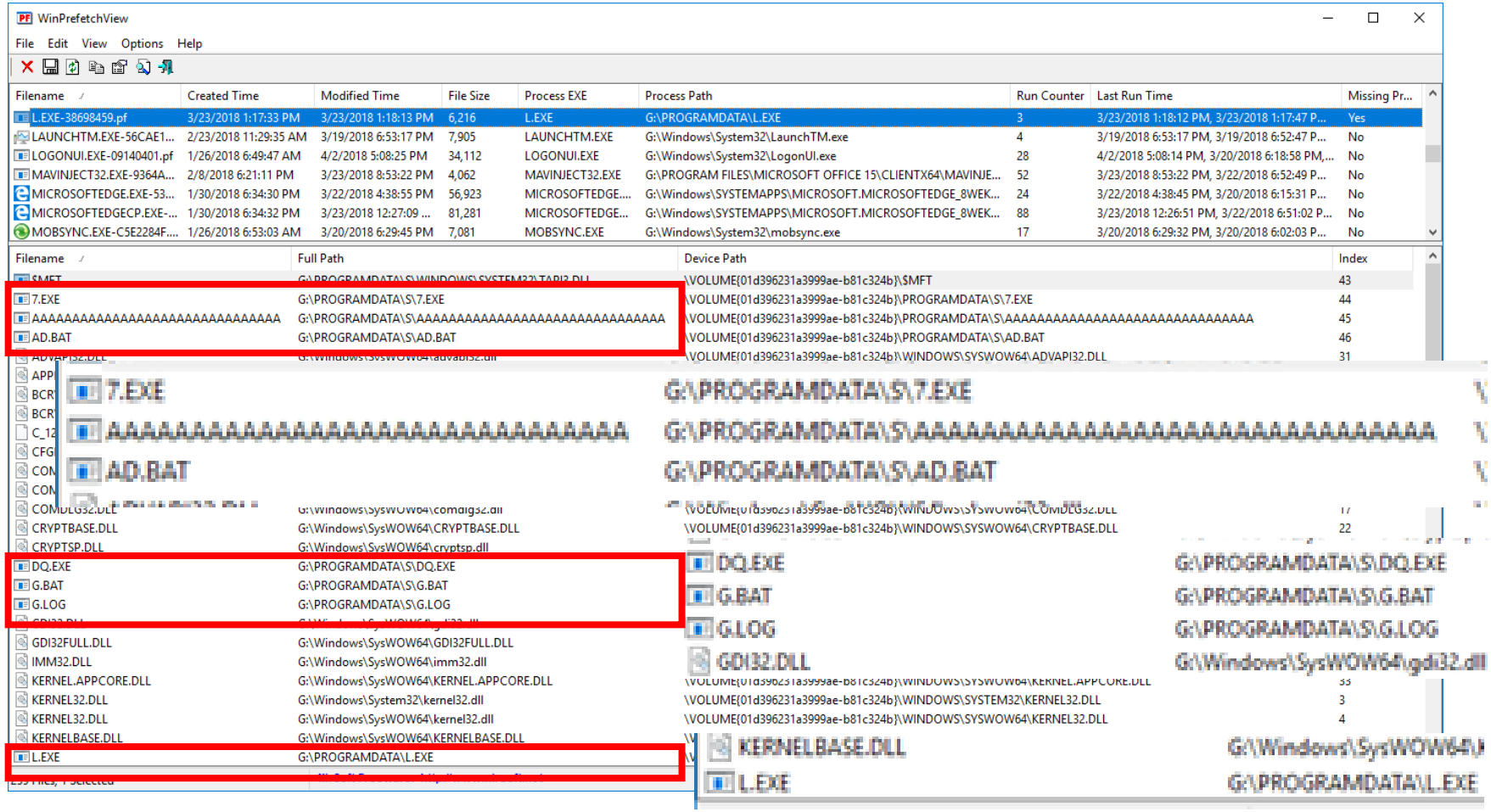
# Finding Programs in Unusual Paths

- Sort by “Process Path” to find unusual paths.
- ProgramData is not a typical path to install programs
- However, in addition to W10.exe, “DQ.EXE” and “L.EXE” were executed from the folder.



# Findings from Prefetch File Contents

- Prefetch of L.EXE recorded several interesting files.
  - Refers to other EXE and BAT files in ProgramData folder.
  - “AAAAAAAA…” file.



# Findings From the Prefetch

- Infection occurred around 4:13 PM on March 7.
- Legitimate Windows commands for retrieving system information were executed starting at 10:36 PM on March 7, and also on March 8.
- Some activities were also found on March 14 and 23.

- Suspicious files existed in paths such as “ProgramData” and “ProgramData\s”
  - L.exe, W10.exe, 7.exe, G.bat, G.log, DQ.exe, AD.BAT, AAAAAAAAAAAAAAA…

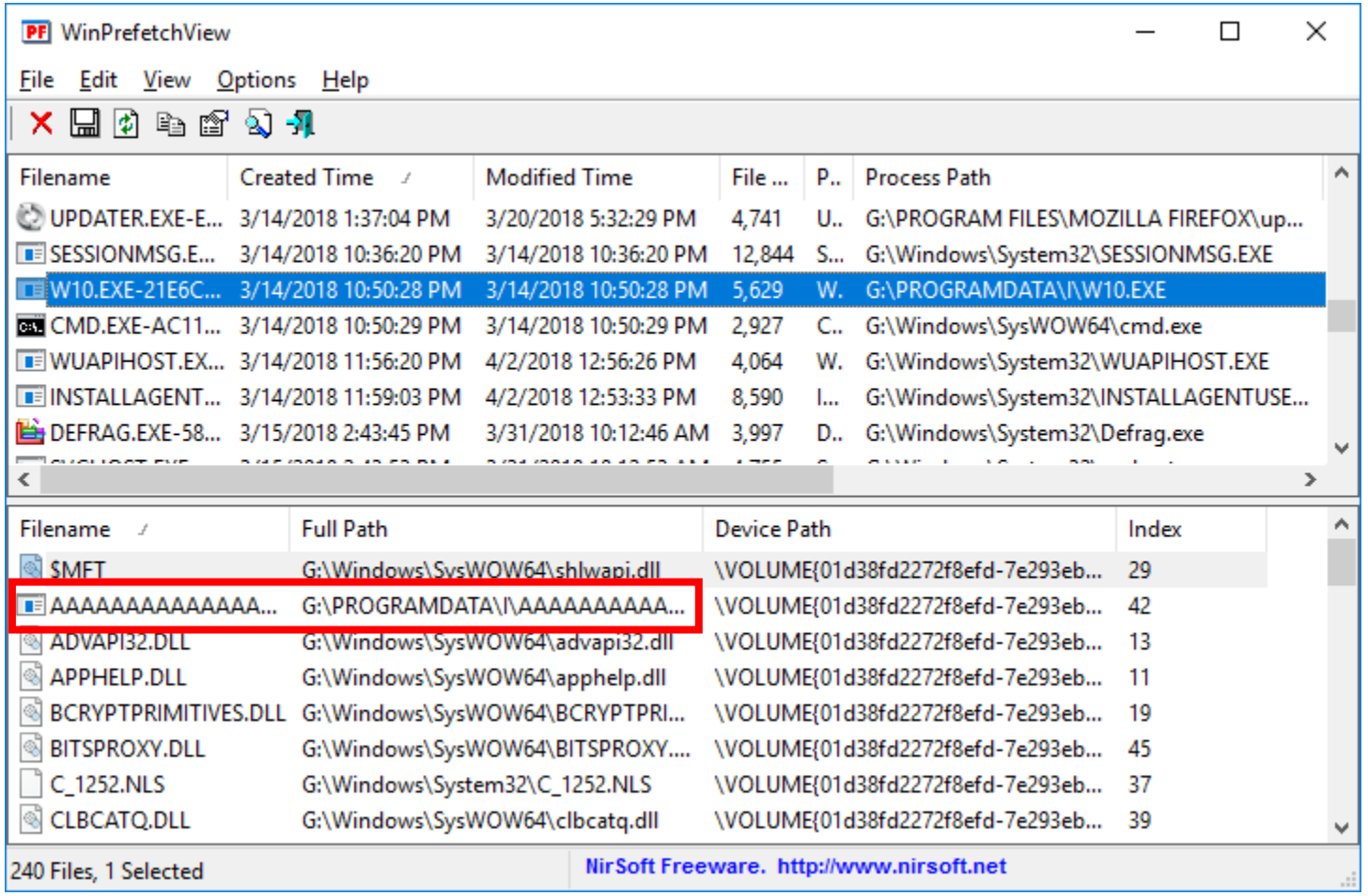
# Quick Demo: Looking into Prefetch Files for Client-Win10-1

# Quick Demo: Looking into Prefetch Files for Client-Win10-1

- Next, we will check Prefetch contents of **Client-Win10-1_toyoda.E01** file.

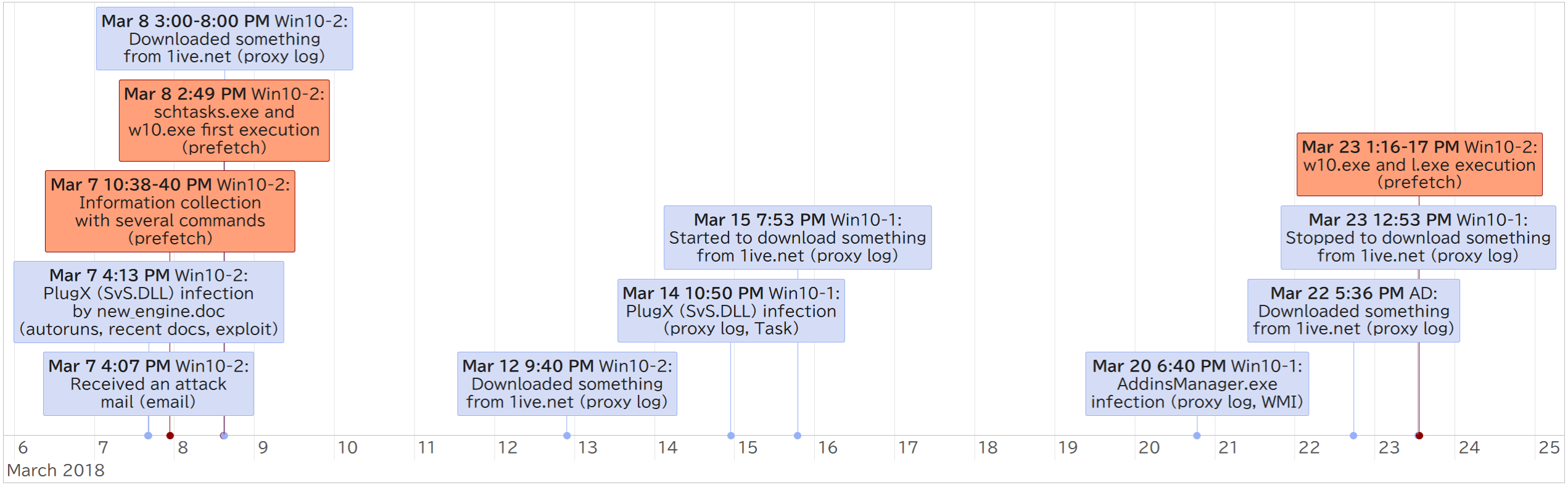
# Findings from Prefetch Files

- Similar to client-win10-2, the following points may be found.
  - A folder with single letter.
  - An executable file residing under ProgramData, which can be modified by non-privileged users.
    - “ProgramData\I” in here.
  - Suspicious reference to “AAAAAAAAAA…” file
  - Since legitimate Windows commands were executed from SysWOW64, a 32-bit program must have executed them.



# Information Obtained from Prefetch

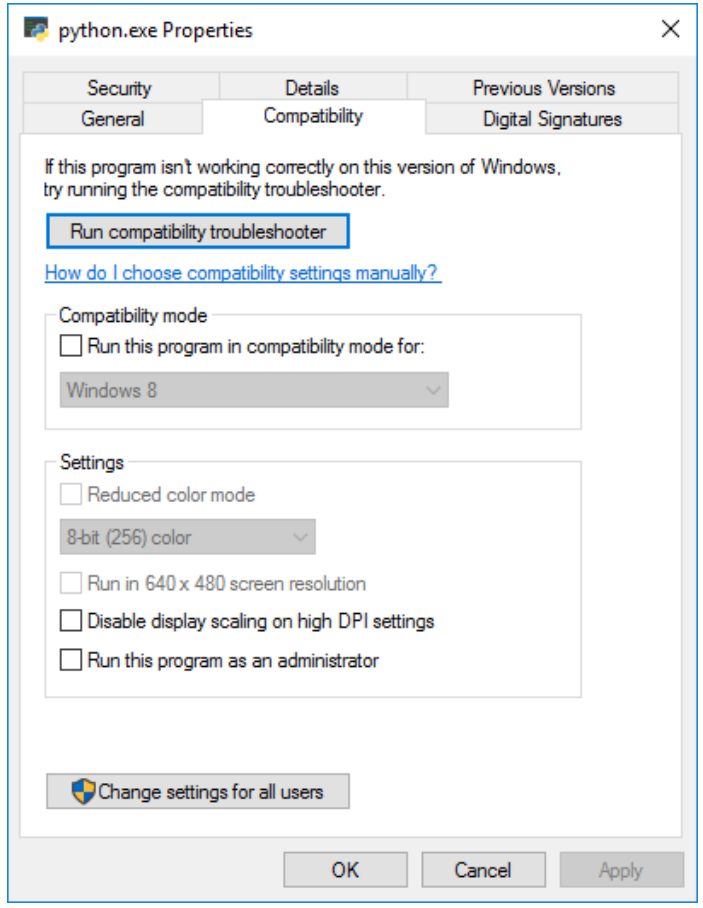
- Execution of various commands and executable files were found on Client-Win10-2.
- Execution of some commands on Client-Win10-1 can be also confirmed from the Prefetch.



# Shimcache

# Shimcache

- Since Windows XP, Windows has “application compatibility” feature.
- Shimcache is stored in registry as a binary value “AppCompatCache” under “**HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Session Manager\AppCompatCache**” key.
  - To get the Shimcache, you will need to extract the “SYSTEM” registry hive.



# Contents of Shimcache

- What you can find from a record of Shimcache includes:
  1. Path of the program
  2. Last modified timestamp of the program
     - Note that this is **not** the execution time.
  3. Flag whether the program has been executed or not (Windows Vista+ have this flag, but the flag is always false on Windows 10)
- Shimcache remains even if the original EXE was deleted.

| | | 1 | 2 | 3 |
|---|---|---|---|---|
| ControlSet | CacheEntryPosition | Path | LastModifiedTimeUTC | Executed |
| 1 | 18 | C:\Bitnami\redmine-3.4.4-2\apps\redmine\scripts\winserv.exe | 2017-02-27 09:04:34 | NA |
| 1 | 19 | C:\Windows\system32\chcp.com | 2016-07-16 11:42:45 | NA |
| 1 | 20 | C:\Windows\SysWOW64\chcp.com | 2016-07-16 11:42:45 | NA |
| 1 | 21 | C:\Bitnami\REDMIN~1.4-2\mysql\bin\mysql.exe | 2017-12-09 07:50:20 | NA |
| 1 | 22 | C:\Bitnami\REDMIN~1.4-2\ruby\bin\ruby.exe | 2016-12-02 19:03:48 | NA |
| 1 | 23 | C:\Bitnami\redmine-3.4.4-2\ruby\bin\ruby.exe | 2016-12-02 19:03:48 | NA |

# Characteristics of ShimCache

- We cannot tell program execution dates, but they are registered by reverse order of program execution.
- Programs executed with SYSTEM privilege are also recorded.
- Execution of BAT files may also be recorded in some conditions.
- If the machine was shut down inappropriately, ShimCache would not be flushed to the registry HIVE like other registry data, but it's only written to the journal LOG files.
  - In that case, tools that can parse the LOG files become necessary.
    - Yarp
    - Registry Explorer

# Characteristics of ShimCache (Cont.)

- A Shimcache entry for a certain binary that is already stored on Shimcache will be inserted again after the binary is modified and then executed. Otherwise, it won't happen again.
- Note that Shimcache may record not executed binaries at the following case.

> *Simply viewing the directory in the GUI that the extracted but not executed executable was in was enough to get it added to the shimcache*

https://www.hecfblog.com/2018/11/daily-blog-544-forensic-lunch-test.html

# Tips for Analyzing ShimCache

- Order of the records is important.
  - Do not sort them with timestamps or other columns.
  - Timestamps are “Modified Time”, and **NOT** the “Execution Time”.
  - Newer entries are recorded first, and the oldest are the last.
    - You can reverse the sort order to see it in chronological order.
  - You can also sort them with file paths to find suspicious paths.
- We should use other artifacts that have timestamps, such as Prefetch, UserAssist, AmCache or file system metadata of the command, to point out the exact time or the time range of execution timeframe.
- You may be able to trust the timestamps in ShimCache in a few cases.
  - If a program is executed from a temporary folder, it may be an installer program. In this case, the file execution time may become equal to the modification time.
    - Entries between such programs were executed at some time within that time frame.
  - Or when a BAT file created and executed repeatedly and periodically.
    - You could estimate the program execution times by observing ShimCache of those programs.

# Comparing with Prefetch

- Pros:
  - Shimcache is able to record entries even when programs were executed as SYSTEM privilege while Prefetch may not be able to record.
  - Both Prefetch and Shimcache can record executables that are not executed from the Explorer process.
  - Shimcache can record entries even where corresponding executables are located in external SMB servers while Prefetch doesn't.
  - Execution of BAT files may also be recorded while Prefetch doesn't.
- Cons:
  - Shimcache doesn't have the exact execution timestamps while Prefetch has multiple timestamps.
  - Shimcache doesn't tell the number of times which the programs were executed while Prefetch stores the number of times.

# Viewing Shimcache

- Since Shimcache is in binary format, it is difficult for the ordinal humans to read it without a parser.
- Many parsers are included in the analysis environment:
  - AppCompatCache Parser by Eric Zimmerman
  - ShimCacheParser.py
    - The tool does not run with Python 3.x
  - RegRipper
  - Registry Explorer
  - …
- In this exercise, we will use AppCompatCache Parser.

# Scenario 1 Labs:
# Lab 2: Looking into ShimCache for Client-Win10-2

# Looking into ShimCache

- Looking into ShimCache may give us information about the programs that were executed, and the programs that were already deleted from the computer.
- The goal of the exercise is to find those artifacts.
  - Try to find something that was not found in Prefetch.

- We will assume that you are still mounting client-win10-2 onto drive G of the analysis machine.
  - If you have unmounted it since the previous exercise, please remount it.

# Running AppCompatCache Parser

- Execute Command Prompt **as administrator**.
  - SYSTEM hive on the mounted image is not readable due to permissions.
    - Workarounds: open the folder once using Windows Explorer and obtain permissions, or copy the SYSTEM hive to somewhere the analyzing user can read (e.g. Desktop).
  - To make it easier, we will run the tool as administrator.
- Run AppCompatCache Parser. Enter the following **in a single line**:

```
AppCompatCacheParser.exe -f G:\Windows\System32\config\SYSTEM
--csv %USERPROFILE%\Desktop
```

  - This parses SYSTEM hive from G:\Windows\System32\config and creates output file to the user's Desktop.

# Preparation for Shimcache Analysis

- Open the AppCompatCachParser's result with CSVFileView or your favorite CSV viewer. Then reverse sort with CacheEntryPosition by clicking the header of the column **twice** so that you can look the result as chronological order.

| ControlSet | CacheEntryPosition | Path | LastModifiedTi... | Executed |
|---|---|---|---|---|
| 1 | 459 | C:\Windows\s... | 2016-07-16 11:... | NA |
| 1 | 458 | C:\Windows\s... | 2016-07-16 11:... | NA |
| 1 | 457 | C:\Windows\s... | 2016-07-16 11:... | NA |
| 1 | 456 | C:\Windows\s... | 2016-07-16 11:... | NA |
| 1 | 455 | C:\Windows\s... | 2016-07-16 11:... | NA |

- Then, press "Ctrl + F" and search "systeminfo".

# Running PECmd to Compare

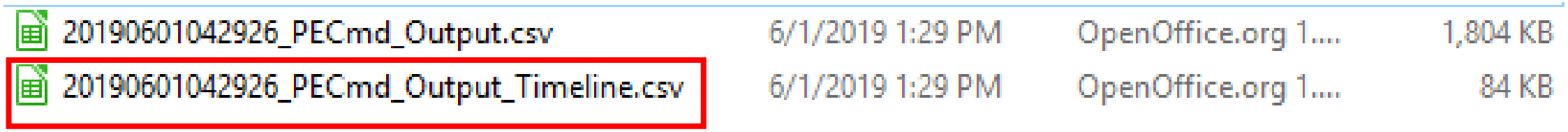
- Execute Command Prompt **as administrator**.
  - To compare the AppComaptCacheParser's result with Prefetch, execute PECmd.

```
PECmd.exe -d G:\Windows\Prefetch --csv %USERPROFILE%\Desktop
```

  - This parses Prefetch folder on G drive and creates output file to the user's Desktop.
    - -d: An option to specify Prefetch folder
    - --csv: the output folder for the result as a csv format

# The Output of PECmd

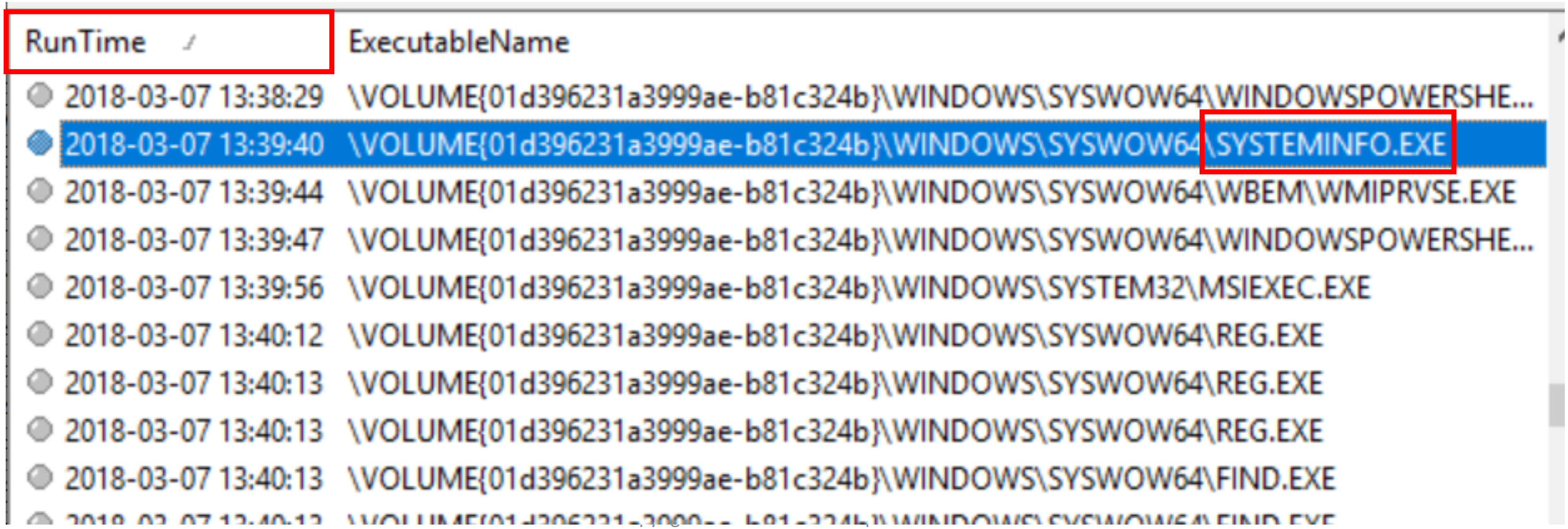
- After execution, you can find the result files in the current folder, which was specified with --csv option.



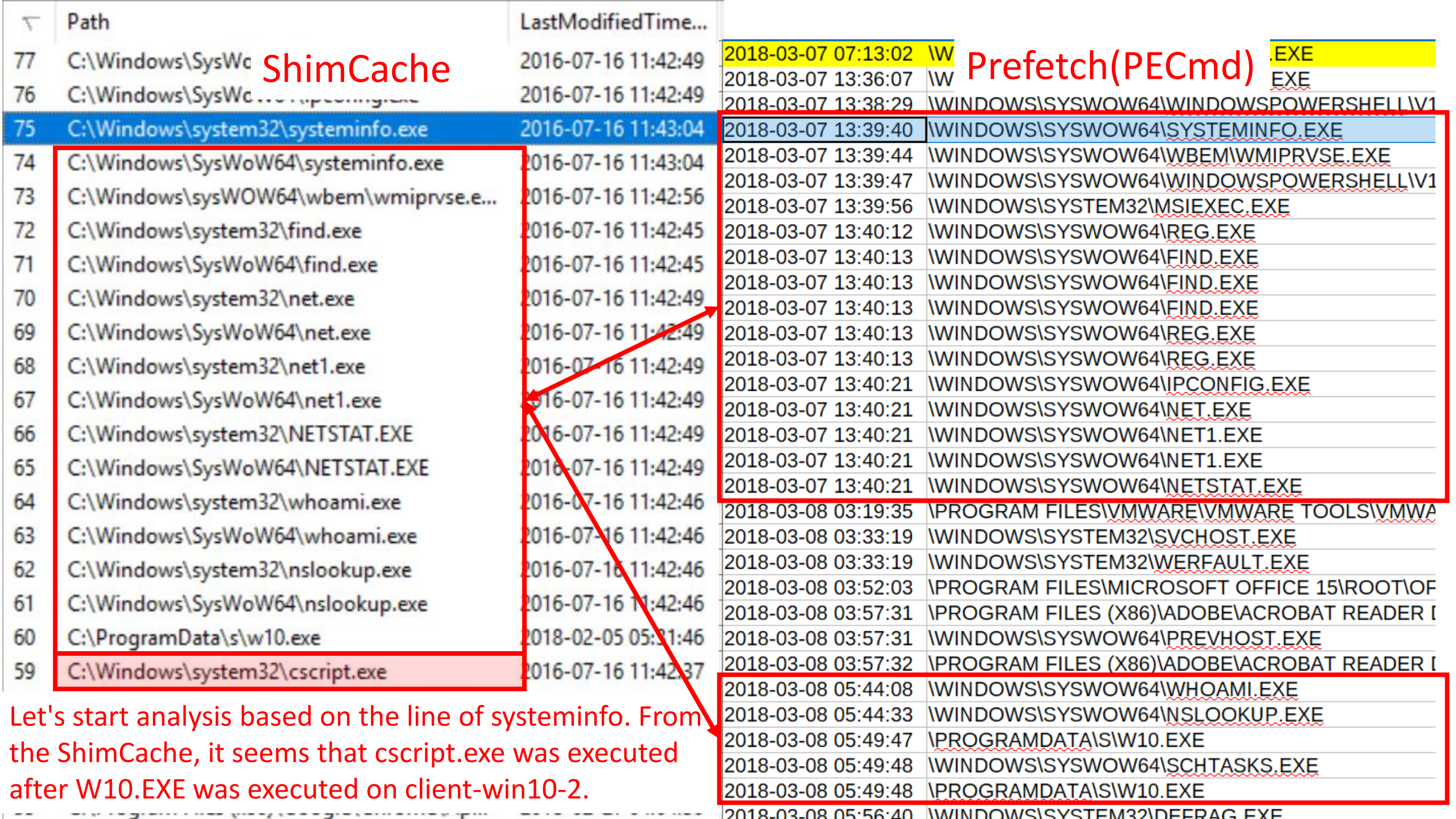

- Let’s open the file ending with ”_Timeline.csv” with CSVFileView or your favorite csv viewer. In this file, Last Run Time entries are split into each line.
- Another file is an output for parsed records of prefetch files. It is similar to the one of WinPrefetchView.

# Preparation for This Exercise

- Click “RunTime” once to sort by executed date. And press “Ctrl + F”, then search “systeminfo”.



| RunTime | ExecutableName |
|---|---|
| 2018-03-07 13:38:29 | \VOLUME{01d396231a3999ae-b81c324b}\WINDOWS\SYSWOW64\WINDOWSPOWERSHE... |
| 2018-03-07 13:39:40 | \VOLUME{01d396231a3999ae-b81c324b}\WINDOWS\SYSWOW64\SYSTEMINFO.EXE |
| 2018-03-07 13:39:44 | \VOLUME{01d396231a3999ae-b81c324b}\WINDOWS\SYSWOW64\WBEM\WMIPRVSE.EXE |
| 2018-03-07 13:39:47 | \VOLUME{01d396231a3999ae-b81c324b}\WINDOWS\SYSWOW64\WINDOWSPOWERSHE... |
| 2018-03-07 13:39:56 | \VOLUME{01d396231a3999ae-b81c324b}\WINDOWS\SYSTEM32\MSIEXEC.EXE |
| 2018-03-07 13:40:12 | \VOLUME{01d396231a3999ae-b81c324b}\WINDOWS\SYSWOW64\REG.EXE |
| 2018-03-07 13:40:13 | \VOLUME{01d396231a3999ae-b81c324b}\WINDOWS\SYSWOW64\REG.EXE |
| 2018-03-07 13:40:13 | \VOLUME{01d396231a3999ae-b81c324b}\WINDOWS\SYSWOW64\REG.EXE |
| 2018-03-07 13:40:13 | \VOLUME{01d396231a3999ae-b81c324b}\WINDOWS\SYSWOW64\FIND.EXE |




| | Path | LastModifiedTime... |
|---|---|---|
| 77 | C:\Windows\SysWo | 2016-07-16 11:42:49 |
| 76 | C:\Windows\SysWo | 2016-07-16 11:42:49 |
| 75 | C:\Windows\system32\systeminfo.exe | 2016-07-16 11:43:04 |
| 74 | C:\Windows\SysWoW64\systeminfo.exe | 2016-07-16 11:43:04 |
| 73 | C:\Windows\sysWOW64\wbem\wmiprvse.e... | 2016-07-16 11:42:56 |
| 72 | C:\Windows\system32\find.exe | 2016-07-16 11:42:45 |
| 71 | C:\Windows\SysWoW64\find.exe | 2016-07-16 11:42:45 |
| 70 | C:\Windows\system32\net.exe | 2016-07-16 11:42:49 |
| 69 | C:\Windows\SysWoW64\net.exe | 2016-07-16 11:42:49 |
| 68 | C:\Windows\system32\net1.exe | 2016-07-16 11:42:49 |
| 67 | C:\Windows\SysWoW64\net1.exe | 2016-07-16 11:42:49 |
| 66 | C:\Windows\system32\NETSTAT.EXE | 2016-07-16 11:42:49 |
| 65 | C:\Windows\SysWoW64\NETSTAT.EXE | 2016-07-16 11:42:49 |
| 64 | C:\Windows\system32\whoami.exe | 2016-07-16 11:42:46 |
| 63 | C:\Windows\SysWoW64\whoami.exe | 2016-07-16 11:42:46 |
| 62 | C:\Windows\system32\nslookup.exe | 2016-07-16 11:42:46 |
| 61 | C:\Windows\SysWoW64\nslookup.exe | 2016-07-16 11:42:46 |
| 60 | C:\ProgramData\s\w10.exe | 2018-02-05 05:31:46 |
| 59 | C:\Windows\system32\cscript.exe | 2016-07-16 11:42:37 |

| | |
|---|---|
| 2018-03-07 07:13:02 | \W .EXE |
| 2018-03-07 13:36:07 | \W EXE |
| 2018-03-07 13:38:29 | \WINDOWS\SYSWOW64\WINDOWSPOWERSHELL\V1 |
| 2018-03-07 13:39:40 | \WINDOWS\SYSWOW64\SYSTEMINFO.EXE |
| 2018-03-07 13:39:44 | \WINDOWS\SYSWOW64\WBEM\WMIPRVSE.EXE |
| 2018-03-07 13:39:47 | \WINDOWS\SYSWOW64\WINDOWSPOWERSHELL\V1 |
| 2018-03-07 13:39:56 | \WINDOWS\SYSTEM32\MSIEXEC.EXE |
| 2018-03-07 13:40:12 | \WINDOWS\SYSWOW64\REG.EXE |
| 2018-03-07 13:40:13 | \WINDOWS\SYSWOW64\FIND.EXE |
| 2018-03-07 13:40:13 | \WINDOWS\SYSWOW64\FIND.EXE |
| 2018-03-07 13:40:13 | \WINDOWS\SYSWOW64\FIND.EXE |
| 2018-03-07 13:40:13 | \WINDOWS\SYSWOW64\REG.EXE |
| 2018-03-07 13:40:13 | \WINDOWS\SYSWOW64\REG.EXE |
| 2018-03-07 13:40:21 | \WINDOWS\SYSWOW64\IPCONFIG.EXE |
| 2018-03-07 13:40:21 | \WINDOWS\SYSWOW64\NET.EXE |
| 2018-03-07 13:40:21 | \WINDOWS\SYSWOW64\NET1.EXE |
| 2018-03-07 13:40:21 | \WINDOWS\SYSWOW64\NET1.EXE |
| 2018-03-07 13:40:21 | \WINDOWS\SYSWOW64\NETSTAT.EXE |
| 2018-03-08 03:19:35 | \PROGRAM FILES\VMWARE\VMWARE TOOLS\VMWA |
| 2018-03-08 03:33:19 | \WINDOWS\SYSTEM32\SVCHOST.EXE |
| 2018-03-08 03:33:19 | \WINDOWS\SYSTEM32\WERFAULT.EXE |
| 2018-03-08 03:52:03 | \PROGRAM FILES\MICROSOFT OFFICE 15\ROOT\OF |
| 2018-03-08 03:57:31 | \PROGRAM FILES (X86)\ADOBE\ACROBAT READER D |
| 2018-03-08 03:57:31 | \WINDOWS\SYSWOW64\PREVHOST.EXE |
| 2018-03-08 03:57:32 | \PROGRAM FILES (X86)\ADOBE\ACROBAT READER D |
| 2018-03-08 05:44:08 | \WINDOWS\SYSWOW64\WHOAMI.EXE |
| 2018-03-08 05:44:33 | \WINDOWS\SYSWOW64\NSLOOKUP.EXE |
| 2018-03-08 05:49:47 | \PROGRAMDATA\S\W10.EXE |
| 2018-03-08 05:49:48 | \WINDOWS\SYSWOW64\SCHTASKS.EXE |
| 2018-03-08 05:49:48 | \PROGRAMDATA\S\W10.EXE |
| 2018-03-08 05:56:40 | \WINDOWS\SYSTEM32\DEFRAG.EXE |

# ShimCache

We found that cscript.exe was executed in between **Mar 8 2:49 PM** in JST (UTC+9:00) and **Mar 9 3:06 PM** by combining with the Prefetch result.

| | | |
|---|---|---|
| 61 | C:\Windows\SysWoW64\nslookup.exe | 2016-07-16 11:42:4 |
| 60 | C:\ProgramData\s\w10.exe | 2018-02-05 05:31:4 |
| 59 | C:\Windows\system32\cscript.exe | 2016-07-16 11:42:3 |
| 58 | C:\Users\honda\AppData\Local\Google\Chrome\User Data\SwReporter\25.141.202\software_reporter_tool.exe | 2018-01-31 04:49:5 |
| 57 | C:\Program Files (x86)\Google\Update\1.3.33.7\GoogleUpdateBroker.exe | 2018-02-08 08:58:2 |
| 56 | C:\Program Files (x86)\Google\Update\Install\{C2B0CB92-7DE6-4F7C-8963-242408D085FF}\65.0.3325.146_64.0.3282.18... | 2018-03-06 10:32:4 |

# Prefetch(PECmd)

| | |
|---|---|
| | WS\SYSWOW64\NSLOOKUP.EXE |
| 2018-03-08 05:49:47 | \PROGRAMDATA\S\W10.EXE |
| 2018-03-08 05:49:48 | \WINDOWS\SYSWOW64\SCHTASKS.EXE |
| 2018-03-08 05:49:48 | \PROGRAMDATA\S\W10.EXE |
| 2018-03-08 05:56:40 | \WINDOWS\SYSTEM32\DEFRAG.EXE |
| 2018-03-08 06:00:08 | \WINDOWS\SYSTEM32\WINDOWSPOWERSHELL\V1.0\POWERSHELL.EXE |
| 2018-03-08 07:00:01 | \WINDOWS\SYSTEM32\WINDOWSPOWERSHELL\V1.0\POWERSHELL.EXE |
| 2018-03-08 08:00:01 | \WINDOWS\SYSTEM32\WINDOWSPOWERSHELL\V1.0\POWERSHELL.EXE |
| 2018-03-08 09:00:02 | \WINDOWS\SYSTEM32\WINDOWSPOWERSHELL\V1.0\POWERSHELL.EXE |
| 2018-03-08 10:00:00 | \WINDOWS\SYSTEM32\WINDOWSPOWERSHELL\V1.0\POWERSHELL.EXE |
| 2018-03-08 11:00:00 | \WINDOWS\SYSTEM32\CMD.EXE |
| | (snip) |
| 2018-03-09 06:06:21 | \PROGRA...MOZILLA FIREFOX\FIREFOX.EXE |
| 2018-03-09 06:06:40 | \USERS\HONDA\APPDATA\LOCAL\GOOGLE\CHROME\USER DATA\SWREPORTER\25.141.202\SOFTWARE_REPOR |
| 2018-03-09 06:06:43 | \USERS\HONDA\APPDATA\LOCAL\GOOGLE\CHROME\USER DATA\SWREPORTER\25.141.202\SOFTWARE_REPOR |
| 2018-03-09 06:06:50 | \PROGRAM FILES (X86)\GOOGLE\CHROME\APPLICATION\CHROME.EXE |

# ShimCache

| | | |
|---|---|---|
| 61 | C:\Windows\SysWoW64\nslookup.exe | 2016-07-16 11:42:4 |
| 60 | C:\ProgramData\s\w10.exe | 2018-02-05 05:31:4 |
| 59 | C:\Windows\system32\cscript.exe | 2016-07-16 11:42:3 |
| 58 | C:\Users\honda\AppData\Local\Google\Chrome\User Data\SwReporter\25.141.202\software_reporter_tool.exe | 2018-01-31 04:49:5 |
| 57 | C:\Program Files (x86)\Google\Update\1.3.33.7\GoogleUpdateBroker.exe | 2018-02-08 08:58:2 |
| 56 | C:\Program Files (x86)\Google\Update\Install\{C2B0CB92-7DE6-4F7C-8963-242408D085FF}\65.0.3325.146_64.0.3282.18... | 2018-03-06 10:32:4 |

# Prefetch(PECmd)

| | |
|---|---|
| | WS\SYSWOW64\NSLOOKUP.EXE |
| 2018-03-08 05:49:47 | \PROGRAMDATA\S\W10.EXE |
| 2018-03-08 05:49:48 | \WINDOWS\SYSWOW64\SCHTASKS.EXE |
| 2018-03-08 05:49:48 | \PROGRAMDATA\S\W10.EXE |
| 2018-03-08 05:56:40 | \WINDOWS\SYSTEM32\DEFRAG.EXE |
| 2018-03-08 06:00:08 | \WINDOWS\SYSTEM32\WINDOWSPOWERSHELL\V1.0\POWERSHELL.EXE |
| 2018-03-08 07:00:01 | \WINDOWS\SYSTEM32\WINDOWSPOWERSHELL\V1.0\POWERSHELL.EXE |
| 2018-03-08 08:00:01 | \WINDOWS\SYSTEM32\WINDOWSPOWERSHELL\V1.0\POWERSHELL.EXE |
| 2018-03-08 09:00:02 | \WINDOWS\SYSTEM32\WINDOWSPOWERSHELL\V1.0\POWERSHELL.EXE |
| 2018-03-08 10:00:00 | \WINDOWS\SYSTEM32\WINDOWSPOWERSHELL\V1.0\POWERSHELL.EXE |
| 2018-03-08 11:00:00 | \WINDOWS\SYSTEM32\CMD.EXE |

w10.exe executed schtasks.exe and w10.exe itself. And then, powershell.exe was executed every hour until 7:00 PM by Task Scheduler. And cscript.exe was related to the activities. It means that cscript was executed via Powershell by Task Scheduler. In order to confirm it, you should check Windows Event logs.

# The Result of ShimCache Analysis for Client-Win10-2

- cscript.exe was executed after W10.EXE
  - cscript.exe is a program used for executing VBScript and Jscript with SYSTEM privilege in between **Mar 8 2:49 PM** in JST (UTC+9:00) and **Mar 9 3:06 PM** .
  - However, the cscript.exe isn't 32-bit binary, but 64-bit. It is likely that the file was executed by task scheduler with SYSTEM privilege or by chance at that moment.
- If we could find the script along with W10.EXE, we may be able to investigate more details.

# Scenario 1 Labs:
# Lab 3: Looking into ShimCache for Client-Win10-1

# Looking into ShimCache for Client-Win10-1

- We will assume that the OS volume of client-win10-1 is mounted to drive I.
  - If you unmounted it after the previous exercise, please remount it.
- We will compare ShimCache with Prefetch to see if entries that did not remain in Prefetch are remaining.

# Running AppCompatCache Parser

- Run AppCompatCache Parser. Enter the following **in a single line**:

```
AppCompatCacheParser.exe -f I:\Windows\System32\config\SYSTEM
--csv %USERPROFILE%\Desktop
```

  - This parses SYSTEM hive from I:\Windows\System32\config and creates an output file to the user's Desktop.
    - -f: An option to specify the SYSTEM HIVE
    - --csv: the output folder for the result as a csv format

# Running PECmd to Compare

- Execute Command Prompt **as administrator**.
  - To compare the AppComaptCacheParser's result with Prefetch, execute PECmd.

```
PECmd.exe -d I:\Windows\Prefetch --csv %USERPROFILE%\Desktop
```

  - This parses Prefetch folder on I drive and creates output file to the user's Desktop.
    - -d: An option to specify Prefetch folder
    - --csv: the output folder for the result as a csv format

# Preparation for the Analysis

- The same way as the previous exercise, let’s open the output file of the PECmd ending with ”_Timeline.csv” and sort by RunTime.
- Open the output file of the AppCompatCache Parser and reverse-sort by the CacheEntryPosition.
- Then, search w10.exe.

# Findings from ShimCache

- 32-bit cmd.exe, ipconfig.exe, and whoami.exe are recorded.
- From the prefetch results, w10.exe and cmd.exe were executed at March 14 10:50 PM (in JST:UTC+9:00) and atbroker.exe was executed at March 15 6:21 PM (in JST:UTC+9:00).
- Some temporary files were executed at March 19, 2018 on 7:04 PM (in JST:UTC+9:00). Therefore, ipconfig.exe and whoami.exe should be executed in between March 15 6:21 PM and March 19, 2018 on 7:04 PM.

ShimCache

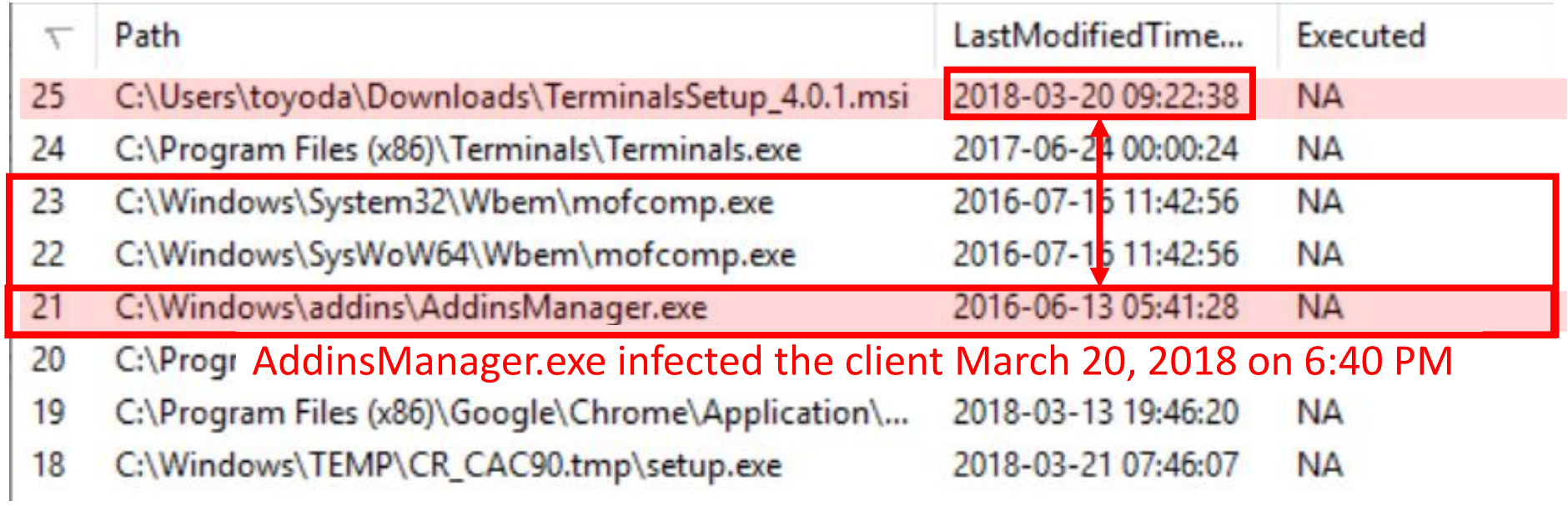
| | Path | LastModifiedTime... |
|---|---|---|
| 39 | C:\Users\ninja-rdp\AppData\Local\Microsoft\OneDri... | 2018-03-14 13:39:20 |
| 38 | C:\ProgramData\i\w10.exe | 2018-02-05 07:21:48 |
| 37 | C:\Windows\SysWoW64\cmd.exe | 2016-07-16 11:43:01 |
| 36 | C:\Windows\system32\atbroker.exe | 2016-07-16 11:42:20 |
| 35 | C:\Windows\SysWoW64\ipconfig.exe | 2016-07-16 11:42:49 |
| 34 | C:\Windows\SysWoW64\whoami.exe | 2016-07-16 11:42:46 |
| 33 | C:\Users\toyoda\Downloads\python-3.6.4-amd64.exe | 2018-03-19 10:04:10 |
| 32 | C:\Users\toyoda\AppData\Local\Temp\{9EBE94CA-7... | 2018-03-19 10:04:12 |

Prefetch(PECmd)

| Execut | |
|---|---|
| 2018-03 | EM32\BACKGROUNDTA |
| 2018-03-14 13:50:26 | \PROGRAMDATA\I\W10.EXE |
| 2018-03-14 13:50:28 | \WINDOWS\SYSWOW64\CMD.EXE |
| 2018-03-14 13:50:28 | \PROGRAMDATA\I\W10.EXE |
| 2018-03-14 13:52:42 | \WINDOWS\SYSTEM32\DLLHOST.EXE |
| 2018-03-14 13:52:43 | \WINDOWS\SYSTEM32\DLLHOST.EXE |
| 2018-03-14 13:52:51 | \WINDOWS\SYSTEM32\LOGONUI.EXE |
| 2018-03-15 05:43:43 | \WINDOWS\SYSTEM32\DEFRAG.EXE |
| 2018-03-15 05:43:43 | \WINDOWS\SYSTEM32\SVCHOST.EXE |
| 2018-03-15 09:21:23 | \WINDOWS\SYSTEM32\ATBROKER.EXE |
| 2018-03-15 09:21:25 | \WINDOWS\SYSTEM32\DLLHOST.EXE |
| 2018-03-15 09:25:55 | \PROGRAM FILES\WINDOWS DEFENDER |

# Estimating Execution Time from Nearby Entries

- AddinsManager.exe is malware that was found on client-win10-1. This is recorded in WMI, but the following output suggests that mofcomp command was used for registering the malware.
- We already know that AddinsManager.exe infected the client March 20, 2018 on 6:40 PM. Therefore, WMI must have been registered sometime between March 20, 2018 on 6:22 PM (in JST:UTC+9:00) and March 20, 2018 on 6:40 PM.



# Findings from ShimCache

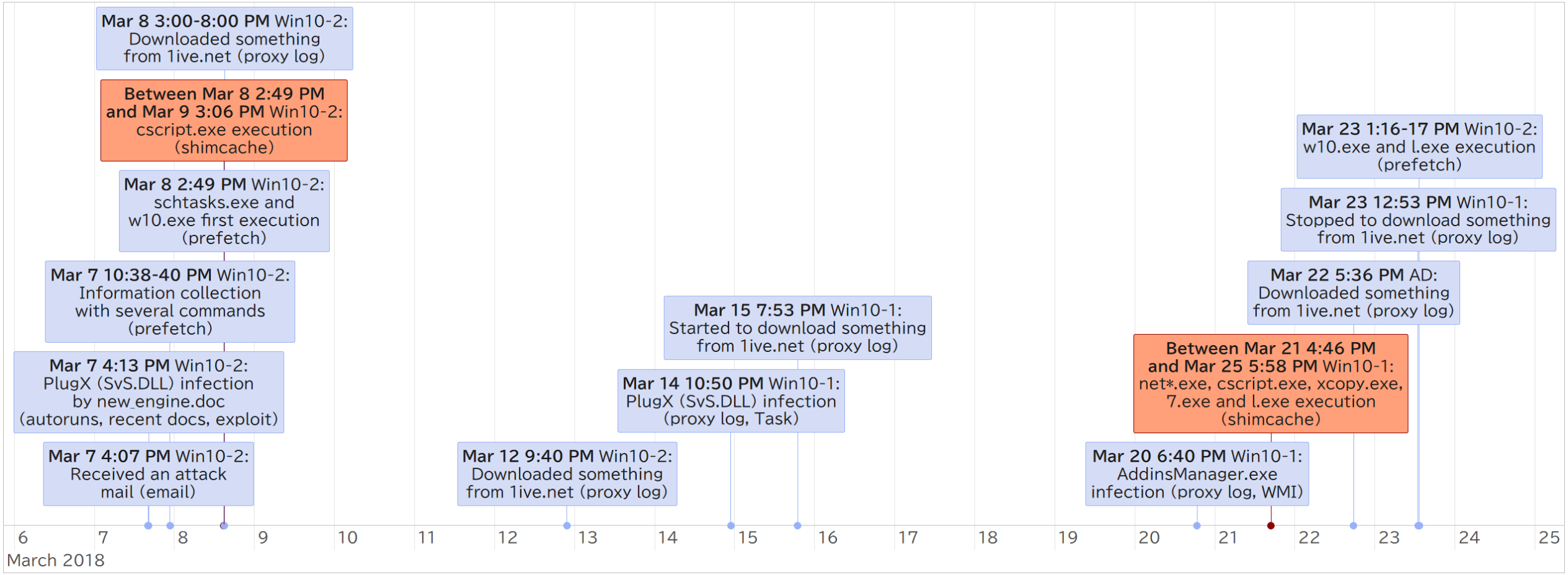
- 32-bit net*.exe, cscript.exe, and xcopy.exe are recorded.
  - Something may have been copied from the computer.
- Programs named with a single letter, such as 7.exe and l.exe, are recorded right after that.
  - Might be related to the attacks.
- Temporary files indicate that the programs were executed at some time between March 21, 2018 on 4:46 PM (in JST:UTC+9:00) and March 25, 2018 on 5:58 PM.

| | Path | LastModifiedTime... | Executed |
|---|---|---|---|
| 19 | C:\Program Files (x86)\Google\Chrome\Application\... | 2018-03-13 19:46:20 | NA |
| 18 | C:\Windows\TEMP\CR_CAC90.tmp\setup.exe | 2018-03-21 07:46:07 | NA |
| 17 | C:\Windows\SysWoW64\net.exe | 2016-07-16 11:42:49 | NA |
| 16 | C:\Windows\SysWoW64\net1.exe | 2016-07-16 11:42:49 | NA |
| 15 | C:\Windows\SysWoW64\cscript.exe | 2016-07-16 11:43:02 | NA |
| 14 | C:\Windows\SysWoW64\xcopy.exe | 2016-07-16 11:42:49 | NA |
| 13 | C:\ProgramData\s\7.exe | 2016-10-04 15:12:28 | NA |
| 12 | C:\ProgramData\l.exe | 2017-06-30 03:47:22 | NA |
| 11 | C:\Windows\Installer\MSI398C.tmp | 2018-03-25 08:58:10 | NA |
| 10 | C:\Windows\SysWoW64\werfault.exe | 2016-07-16 11:42:56 | NA |

# The Result of ShimCache Analysis

- In persistence chapter, we have found that WMI was used on client-win10-1. From the ShimCache, we found that the registration was done using mofcomp.exe.
- After W10.EXE was executed, ipconfig.exe and whoami.exe, which did not appear on Prefetch, were executed.
- net*.exe, cscript.exe, xcopy.exe, 7.exe, l.exe were executed at some time in between March 21 and 25. We should pay a close attention to the executables with a single letter.

# Scenario 1 Incident Timeline After ShimCache Analysis



# SRUM

# SRUM

- Stands for “System Resource Usage Monitor”.
  - Monitors system resource performances.
- SRUM records the CPU time and other operations, and the disk and network I/Os.
  - If an executable file appears in the SRUM, the program must have been executed.
- First introduced in Windows 8.
- It is recorded in a file “**C:\Windows\System32\sru\SRUDB.dat**”.

# Contents of SRUM

- SRUM has several sections. The contents of SRUM include:
  - Network usage: application, user SID, interface, bytes sent/received/total
  - Application resource usage: application, user SID, CPU time in foreground/background, context switches, bytes read/written, number of read/write operations, etc...
  - Energy usage
  - …
- Each SRUM entry has a timestamp of when the SRUM entry was created.
  - It records statistics once an hour or when the system shutdowns.
  - You can see information of processes that was running at that moment, like a snapshot.

# Parsing SRUM

- SRUM is not a human-readable file; it needs to be converted.
- It is written in the ESEDB format, which is used in a lot of places such as Active Directory, IE/Edge, and so on. You can parse:
  - ESEDatabaseView by Nirsoft
  - libesedb by libyal project
- However, you will need to join some tables and resolve names such as interface name. You can use a libesedb based python script called “srum-dump”.
  - https://github.com/MarkBaggett/srum-dump
- “srum-dump” can parse SRUDB.dat file, join tables and resolve names, and output to a XLSX file or several CSV files.

# Note About SRUM

- SRUM is updated once an hour or at the system shutdown.
- SRUM does not record executable files which started after a polling occurred and ended before the next polling is occurred.
- Be careful that timestamp in SRUM **does not tell exactly** when the program was first executed.

# Why This Artifact is so Important?

- We may be able to find RAT/BOT by analyzing this artifact because it has CPU usage, network I/O and disk I/O statistics every hour.
  - You can find processes that were kept executing for a long time.
  - You can find network communication spikes of suspicious processes.

# Scenario 1 Labs:
# Lab 4: Analyzing SRUM on Client-Win10-2

# Creating the Result of srum-dump

- You can use srum-dump

```
cd srum-dump
```

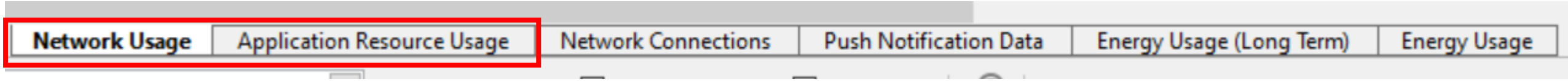
```
srum_dump -i G:\Windows\System32\sru\SRUDB.dat -r
G:\Windows\System32\config\SOFTWARE -o %USERPROFILE%\Desktop\SRUDB.dat.xlsx
```

Note that this command takes a long time! We have already prepared the command result.
The entire command needs to be entered in a single line.

- -i: Input file. You need to specify the path to a SRUDB.dat
  - The default path of SRUDB.dat is **C:\Windows\System32\sru\SRUDB.dat**
- -r: You can specify a HKLM\SOFTWARE hive to resolve WLAN's interface name.
  - The default path of SOFTWARE hive is **C:\Windows\System32\Config\SOFTWARE**
- -o: Output file. You can specify the output file path.

# Opening the output XLSX of srum-dump

- You can find six sheets in the output XLSX.

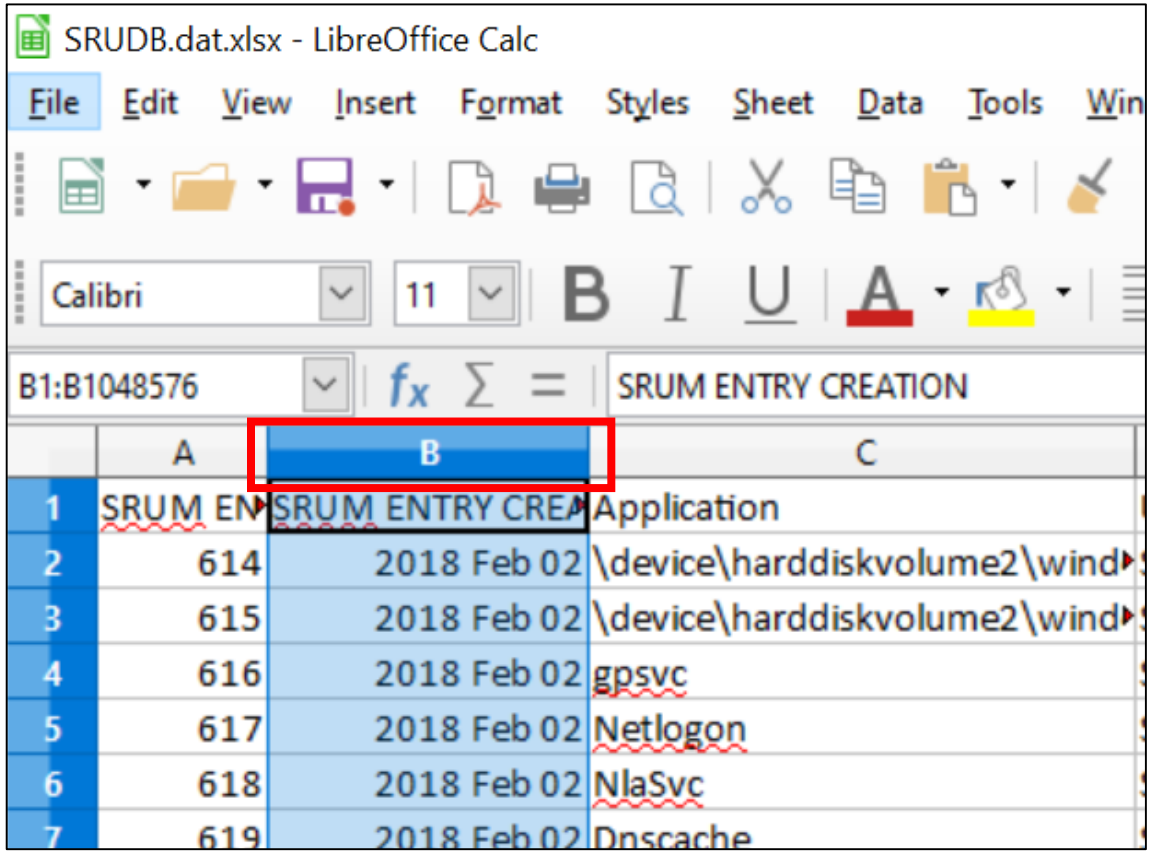



- There are two important sheets in it.
  - Network Usage
  - Application Resource Usage

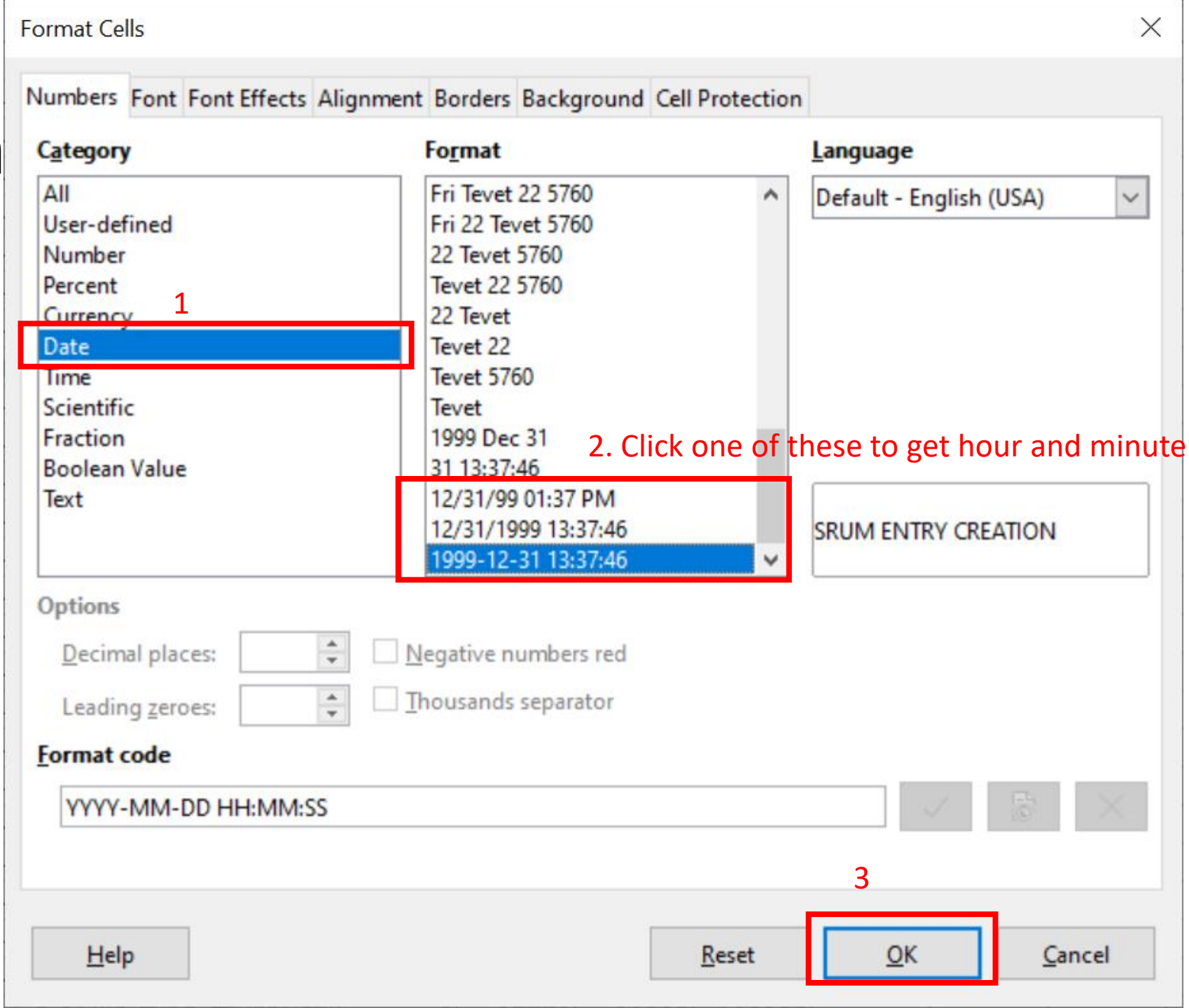
- Let’s see “Network Usage” tab first.

# Change Columns Format (1)

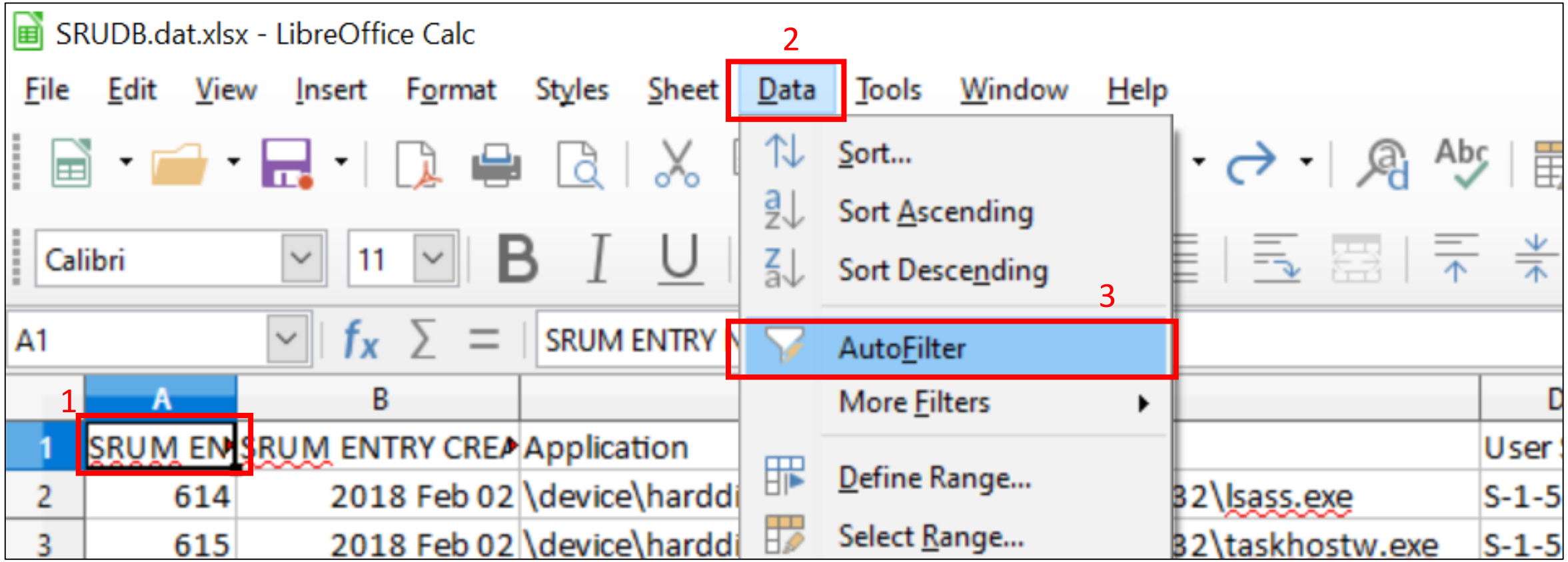
- Before starting analyzing, let's change columns format first.
- Click "Column B".
- Then press "Ctrl + 1".
- You will see "format cells" dialog.



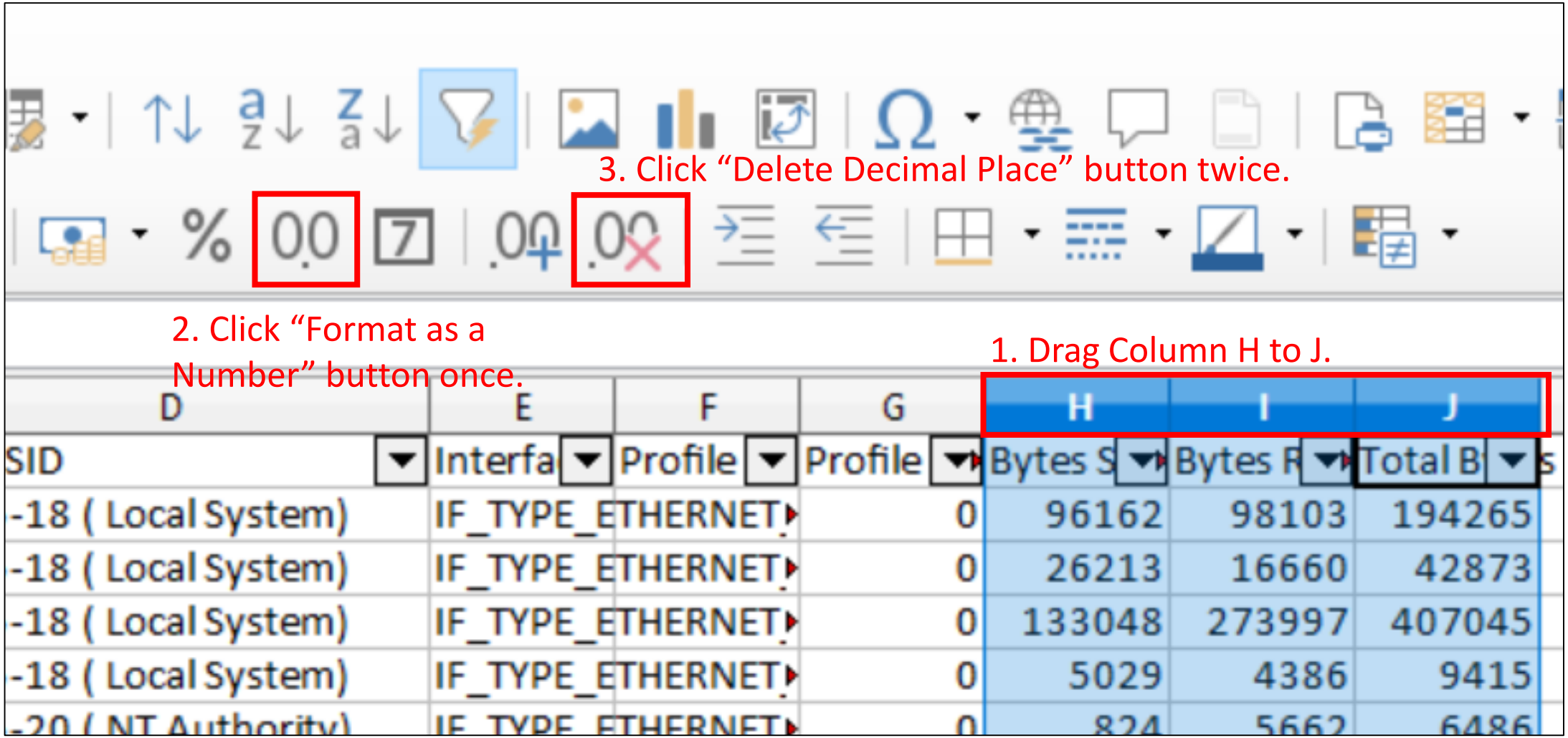
# Chan


Format Cells
Numbers
Font
Font Effects
Alignment
Borders
Background
Cell Protection
Category
All
User-defined
Number
Percent
Currency
Date
Time
Scientific
Fraction
Boolean Value
Text
1
Format
Fri Tevet 22 5760
Fri 22 Tevet 5760
22 Tevet 5760
Tevet 22 5760
22 Tevet
Tevet 22
Tevet 5760
Tevet
1999 Dec 31
31 13:37:46
12/31/99 01:37 PM
12/31/1999 13:37:46
1999-12-31 13:37:46
2. Click one of these to get hour and minute units.
Language
Default - English (USA)
SRUM ENTRY CREATION
Options
Decimal places:
Negative numbers red
Leading zeroes:
Thousands separator
Format code
YYYY-MM-DD HH:MM:SS
3
Help
Reset
OK
Cancel

# Change Columns Format (3)

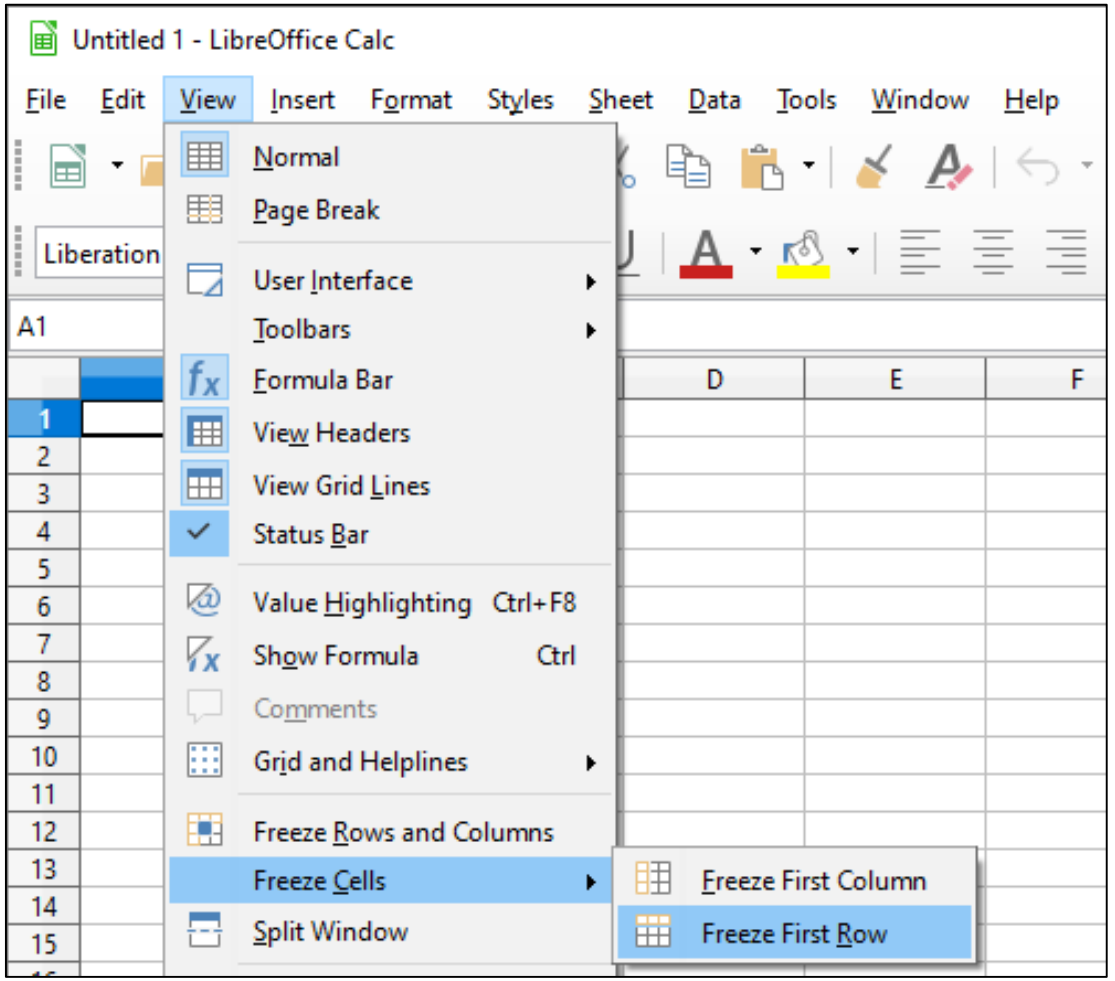


# Change Columns Format (4)



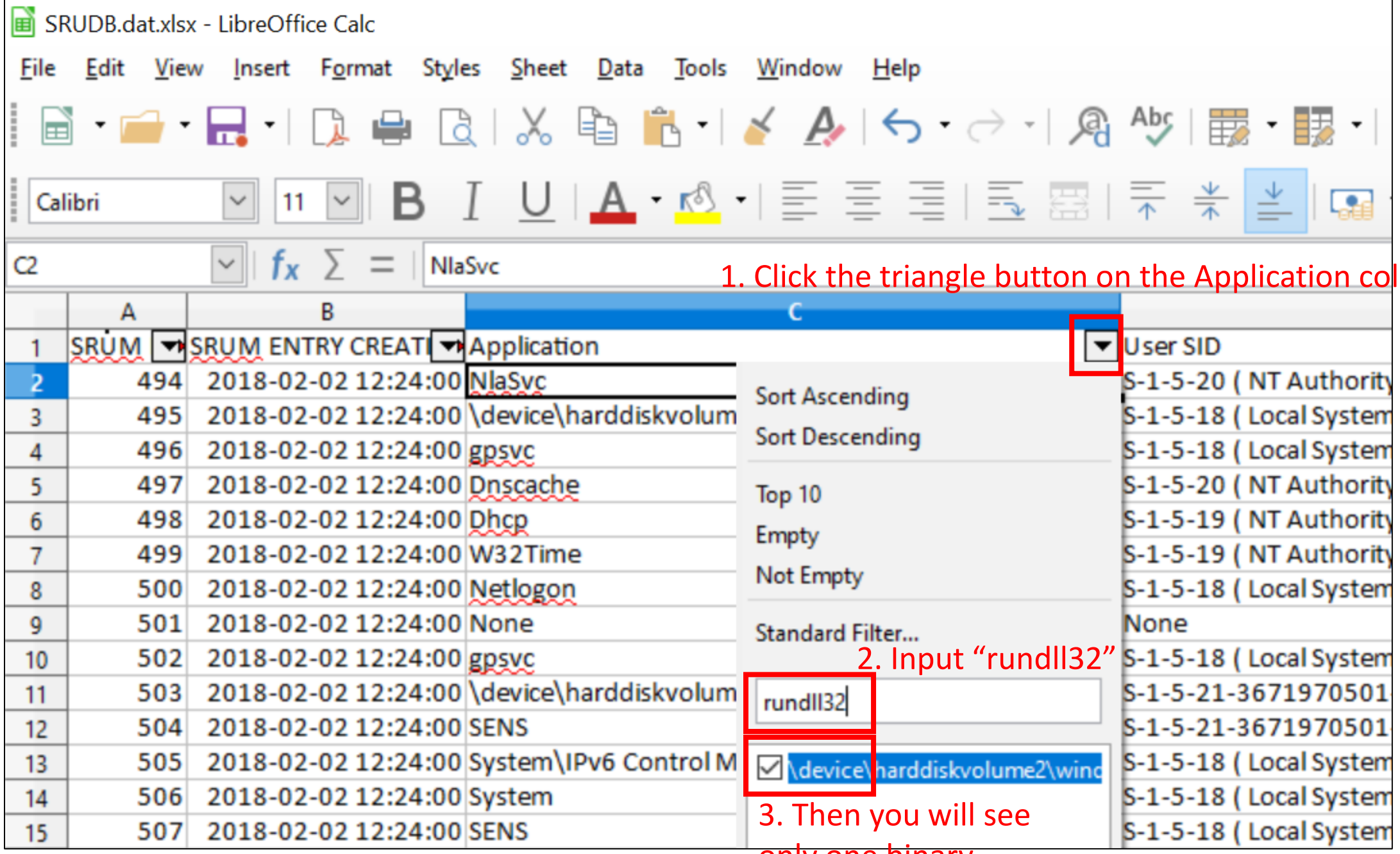
# Freeze First Row

- Use “Freeze First Row” option to keep the title row visible even when you scroll the worksheet.
- You can also apply the same option to column if there is a title column.



# The Strategy of Analyzing srum-dump Results

- We shouldn’t look each line one by one because the number of the records are huge.
- We have several pivot points that:
  - The malware process that was found from this host is “rundll32.dll (SvS.DLL)”.
  - The malware is for 32-bit environment. Therefore, program execution records of standard command-line executables which are located in the “SysWOW64” folder are suspicious because if a 32-bit program spawns another program, it should run on 32-bit environment as well.
  - The attacker used “ProgramData” and “ProgramData\s” folders as the foothold folders.
- You can also check around the timestamps of the pivot points that we found so far.
- Let’s filter with these keywords and take a look at around the timestamps.



The binary was started to be recorded at March 07 4:37 PM. Note that the timestamps are recorded in UTC. You should add nine hours for converting into JST. Since SRUM records statistics every hour, the malware was executed in around 15:37 to 16:37 for the first time. You have already known it happened at 4:13 PM though.

You can get the SID of the user that the malware executed.

| | A | B | C | D | |
|---|---|---|---|---|---|
| 1 | SRUM | SRUM_ENTRY_CREATI | Application | User SID | Inte |
| 9127 | 9619 | 2018-03-07 07:37:00 | \device\harddiskvolume2\windows\syswow64\rundll32.exe | S-1-5-21-3671970501-3975728774-4289435121-1110 | IF_ |
| 9130 | 9622 | 2018-03-07 08:37:00 | \device\harddiskvolume2\windows\syswow64\rundll32.exe | S-1-5-21-3671970501-3975728774-4289435121-1110 | IF_ |
| 9146 | 9638 | 2018-03-07 09:37:00 | \device\harddiskvolume2\windows\syswow64\rundll32.exe | S-1-5-21-3671970501-3975728774-4289435121-1110 | IF_ |
| 9162 | 9654 | 2018-03-07 10:38:00 | \device\harddiskvolume2\windows\syswow64\rundll32.exe | S-1-5-21-3671970501-3975728774-4289435121-1110 | IF_ |
| 9174 | 9666 | 2018-03-07 11:39:00 | \device\harddiskvolume2\windows\syswow64\rundll32.exe | S-1-5-21-3671970501-3975728774-4289435121-1110 | IF_ |
| 9190 | 9682 | 2018-03-07 12:40:00 | \device\harddiskvolume2\windows\syswow64\rundll32.exe | S-1-5-21-3671970501-3975728774-4289435121-1110 | IF_ |
| 9205 | 9697 | 2018-03-07 13:40:00 | \device\harddiskvolume2\windows\syswow64\rundll32.exe | S-1-5-21-3671970501-3975728774-4289435121-1110 | IF_ |
| 9219 | 9711 | 2018-03-07 13:40:00 | \Device\HarddiskVolume2\Windows\SysWOW64\rundll32.exe | S-1-5-21-3671970501-3975728774-4289435121-1110 | IF_ |
| 9223 | 9715 | 2018-03-07 14:41:00 | \device\harddiskvolume2\windows\syswow64\rundll32.exe | S-1-5-21-3671970501-3975728774-4289435121-1110 | IF_ |
| 9245 | 9737 | 2018-03-07 15:41:00 | \device\harddiskvolume2\windows\syswow64\rundll32.exe | S-1-5-21-3671970501-3975728774-4289435121-1110 | IF_ |
| 9258 | 9750 | 2018-03-07 16:42:00 | \device\harddiskvolume2\windows\syswow64\rundll32.exe | S-1-5-21-3671970501-3975728774-4289435121-1110 | IF_ |
| 9270 | 9762 | 2018-03-07 17:52:00 | \device\harddiskvolume2\windows\syswow64\rundll32.exe | S-1-5-21-3671970501-3975728774-4289435121-1110 | IF_ |

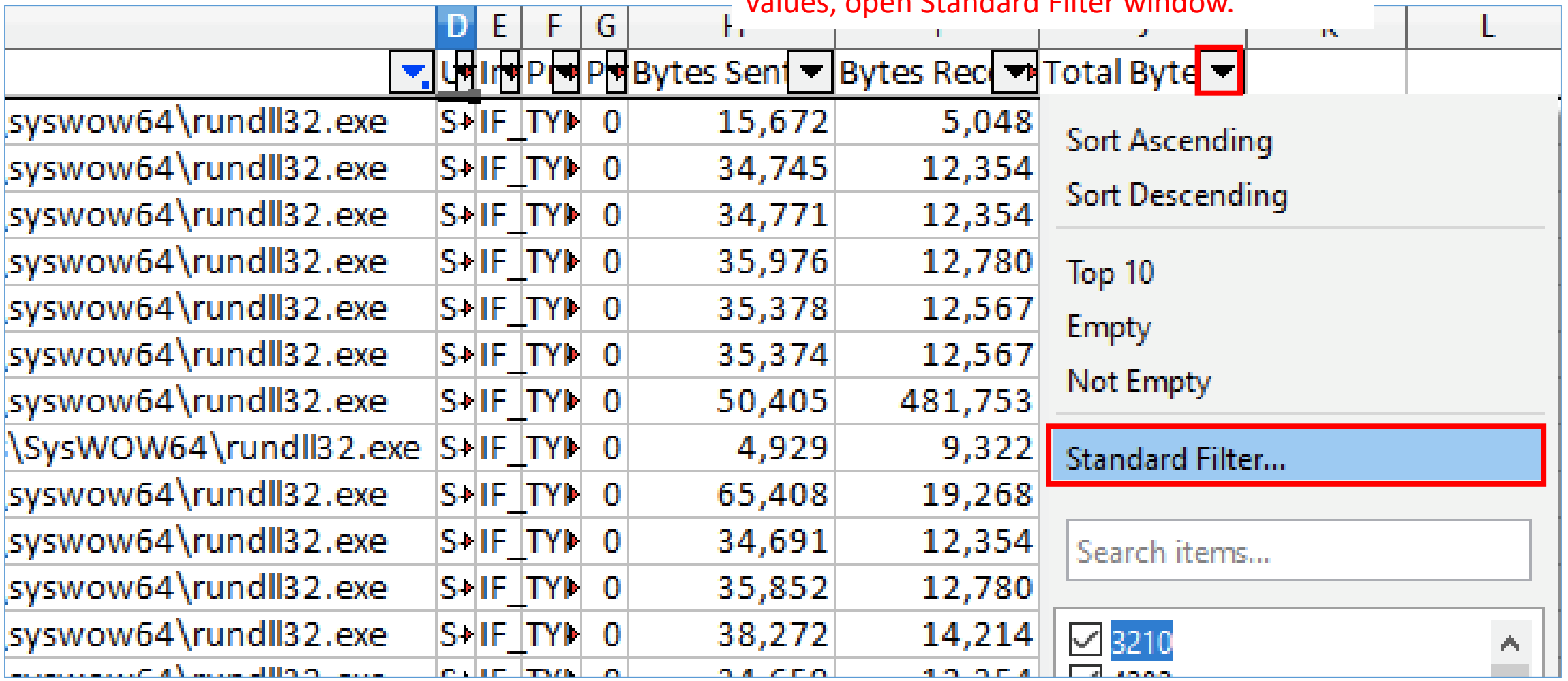
Malware seems to continue to run because it was recorded every hour.

You can see sent and received bytes that malware did in an hour.

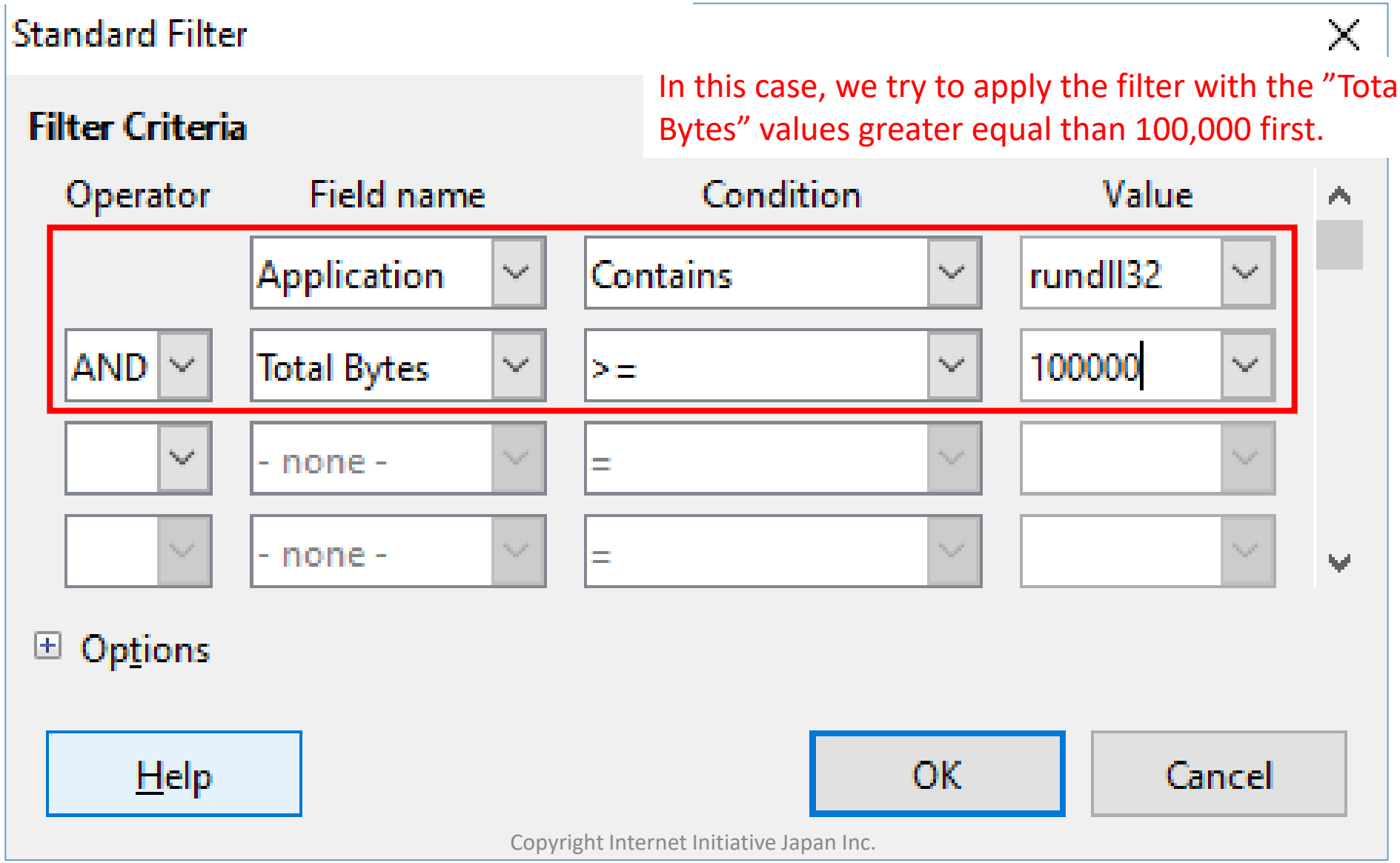
| | A | B | C | D | E | F | G | H | I | J |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | SRU | SRUM ENTRY CREATI | Application | U | In | F | P | Bytes Sent | Bytes Recei | Total B |
| 9127 | 9619 | 2018-03-07 07:37:00 | \device\harddiskvolume2\windows\syswow64\rundll32.exe | 9 | IF_T | | 0 | 15,672 | 5,048 | 20,720 |
| 9130 | 9622 | 2018-03-07 08:37:00 | \device\harddiskvolume2\windows\syswow64\rundll32.exe | 9 | IF_T | | 0 | 34,745 | 12,354 | 47,099 |
| 9146 | 9638 | 2018-03-07 09:37:00 | \device\harddiskvolume2\windows\syswow64\rundll32.exe | 9 | IF_T | | 0 | 34,771 | 12,354 | 47,125 |
| 9162 | 9654 | 2018-03-07 10:38:00 | \device\harddiskvolume2\windows\syswow64\rundll32.exe | 9 | IF_T | | 0 | 35,976 | 12,780 | 48,756 |
| 9174 | 9666 | 2018-03-07 11:39:00 | \device\harddiskvolume2\windows\syswow64\rundll32.exe | 9 | IF_T | | 0 | 35,378 | 12,567 | 47,945 |
| 9190 | 9682 | 2018-03-07 12:40:00 | \device\harddiskvolume2\windows\syswow64\rundll32.exe | 9 | IF_T | | 0 | 35,374 | 12,567 | 47,941 |
| 9205 | 9697 | 2018-03-07 13:40:00 | \device\harddiskvolume2\windows\syswow64\rundll32.exe | 9 | IF_T | | 0 | 50,405 | 481,753 | 532,158 |
| 9219 | 9711 | 2018-03-07 13:40:00 | \Device\HarddiskVolume2\Windows\SysWOW64\rundll32.exe | 9 | IF_T | | 0 | 4,929 | 9,322 | 14,251 |
| 9223 | 9715 | 2018-03-07 14:41:00 | \device\harddiskvolume2\windows\syswow64\rundll32.exe | 9 | IF_T | | 0 | 65,408 | 19,268 | 84,676 |
| 9245 | 9737 | 2018-03-07 15:41:00 | \device\harddiskvolume2\windows\syswow64\rundll32.exe | 9 | IF_T | | 0 | 34,691 | 12,354 | 47,045 |
| 9258 | 9750 | 2018-03-07 16:42:00 | \device\harddiskvolume2\windows\syswow64\rundll32.exe | 9 | IF_T | | 0 | 35,852 | 12,780 | 48,632 |

You can see malware communicates relatively a lot around this time.

In order to filter entries with “Total Bytes” values, open Standard Filter window.

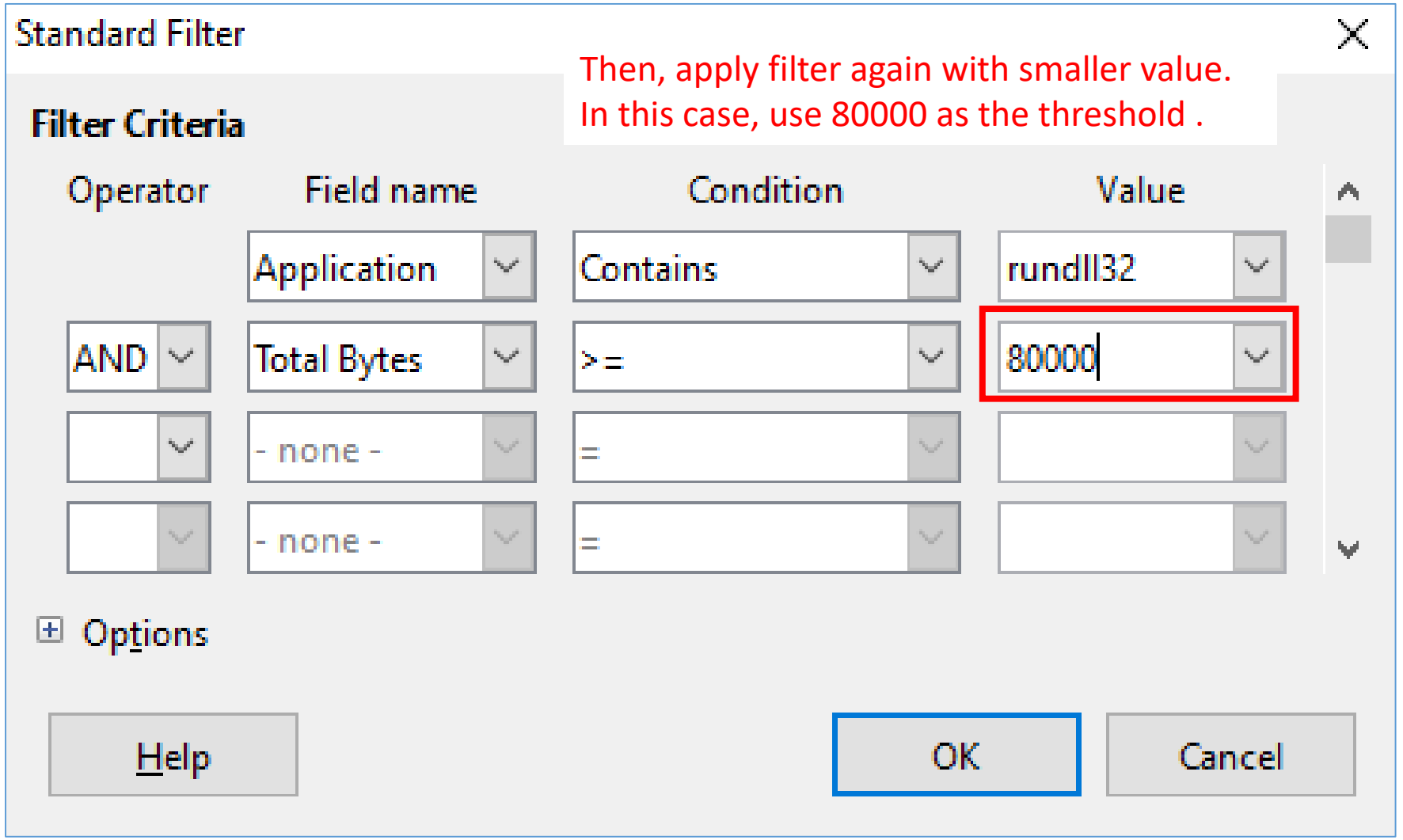


Then, set the filter as the below figure.



There are few entries having the “Total Bytes” value greater than 100000.

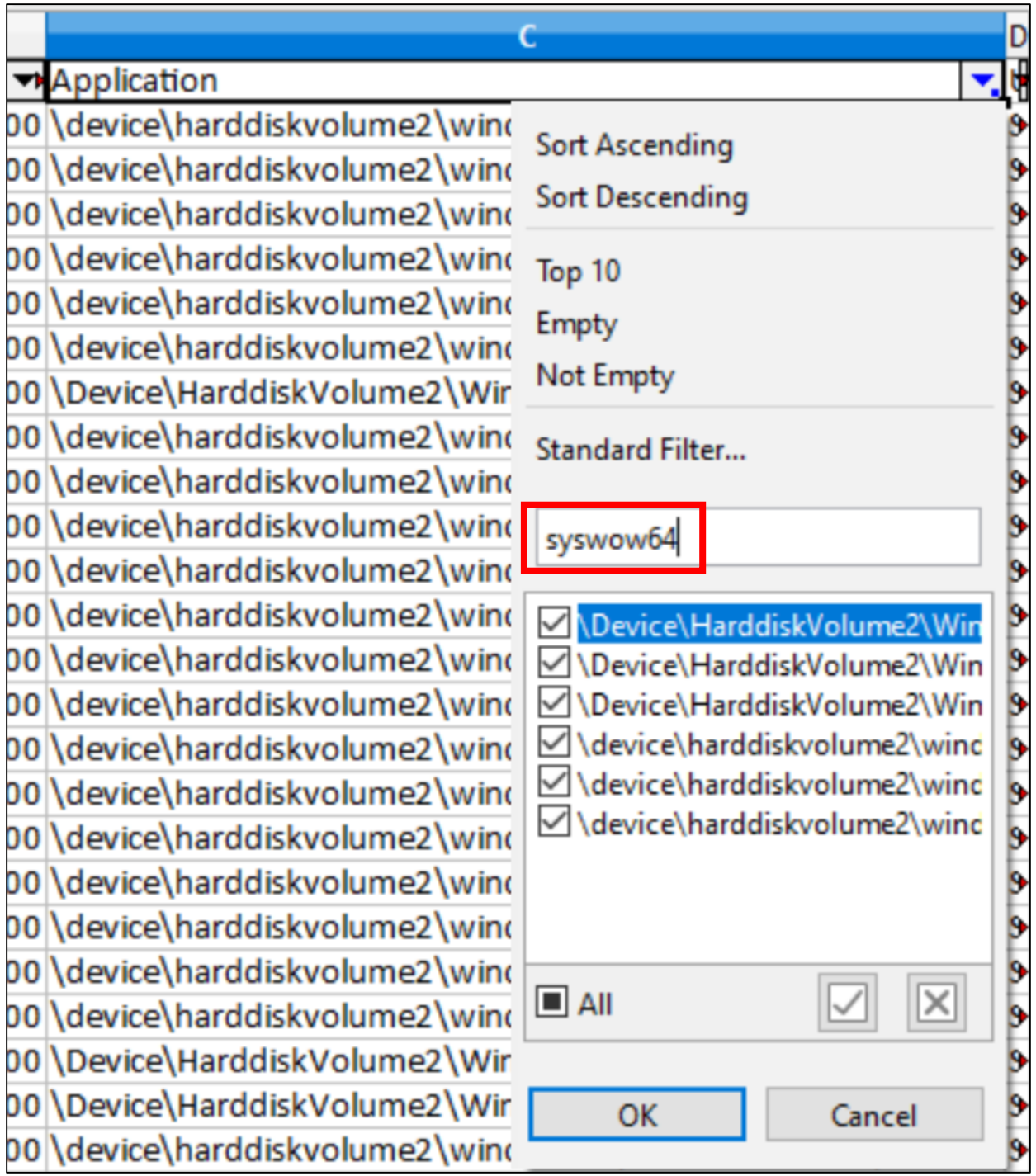
| SRUM ENTRY CREAT | Application | U | In | P | P | Bytes Sent | Bytes Rec | Total Bytes |
|---|---|---|---|---|---|---|---|---|
| 2018-03-07 13:40:00 | \device\harddiskvolume2\windows\syswow64\rundll32.exe | S | IF_TY | | 0 | 50,405 | 481,753 | 532,158 |
| 2018-03-09 06:06:00 | \device\harddiskvolume2\windows\syswow64\rundll32.exe | S | IF_TY | | 0 | 88,798 | 18,062 | 106,860 |
| 2018-03-12 12:34:00 | \device\harddiskvolume2\windows\syswow64\rundll32.exe | S | IF_TY | | 0 | 86,125 | 14,214 | 100,339 |
| 2018-03-12 13:35:00 | \device\harddiskvolume2\windows\syswow64\rundll32.exe | S | IF_TY | | 0 | 110,178 | 20,513 | 130,691 |
| 2018-03-14 13:03:00 | \device\harddiskvolume2\windows\syswow64\rundll32.exe | S | IF_TY | | 0 | 190,827 | 36,817 | 227,644 |
| 2018-03-14 14:04:00 | \device\harddiskvolume2\windows\syswow64\rundll32.exe | S | IF_TY | | 0 | 141,900,642 | 16,798,753 | 158,699,395 |
| 2018-03-23 04:16:00 | \device\harddiskvolume2\windows\syswow64\rundll32.exe | S | IF_TY | | 0 | 77,141 | 25,684 | 102,825 |
| | | | | | | | | |
| | | | | | | | | |


Standard Filter
Then, apply filter again with smaller value.
In this case, use 80000 as the threshold .
Filter Criteria
Operator
Field name
Condition
Value
Application
Contains
rundll32
AND
Total Bytes
>=
80000
- none -
=
- none -
=
Options
Help
OK
Cancel

| SRU | SRUM ENTRY CREAT | Application | L | I | P | P | Bytes Sent | Bytes Rec | Total Bytes |
|---|---|---|---|---|---|---|---|---|---|
| 9697 | 2018-03-07 13:40:00 | \device\harddiskvolume2\windows\syswow64\rundll32.exe | S | IF_TY | | 0 | 50,405 | 481,753 | 532,158 |
| 9715 | 2018-03-07 14:41:00 | \device\harddiskvolume2\windows\syswow64\rundll32.exe | S | IF_TY | | 0 | 65,408 | 19,268 | 84,676 |
| 9951 | 2018-03-08 05:57:00 | \device\harddiskvolume2\windows\syswow64\rundll32.exe | S | IF_TY | | 0 | 65,799 | 26,401 | 92,200 |
| 10305 | 2018-03-09 06:06:00 | \device\harddiskvolume2\windows\syswow64\rundll32.exe | S | IF_TY | | 0 | 88,798 | 18,062 | 106,860 |
| 11682 | 2018-03-12 12:34:00 | \device\harddiskvolume2\windows\syswow64\rundll32.exe | S | IF_TY | | 0 | 86,125 | 14,214 | 100,339 |
| 11699 | 2018-03-12 13:35:00 | \device\harddiskvolume2\windows\syswow64\rundll32.exe | S | IF_TY | | 0 | 110,178 | 20,513 | 130,691 |
| 12556 | 2018-03-14 13:03:00 | \device\harddiskvolume2\windows\syswow64\rundll32.exe | S | IF_TY | | 0 | 190,827 | 36,817 | 227,644 |
| 12575 | 2018-03-14 14:04:00 | \device\harddiskvolume2\windows\syswow64\rundll32.exe | S | IF_TY | | 0 | 141,900,642 | 16,798,753 | 158,699,395 |
| 15733 | 2018-03-23 04:16:00 | \device\harddiskvolume2\windows\syswow64\rundll32.exe | S | IF_TY | | 0 | 77,141 | 25,684 | 102,825 |
| 15752 | 2018-03-23 05:16:00 | \device\harddiskvolume2\windows\syswow64\rundll32.exe | S | IF_TY | | 0 | 65,494 | 18,368 | 83,862 |

There are several spikes of the communications for the malware process.
The attacker might have had something activities around these time periods.

- March 7 9:40 PM - 11:41 PM (in JST: UTC+9:00)
- March 8 1:56 PM - 2:57 PM
- March 9 2:06 PM - 3:06 PM
- March 12 8:33 PM - 10:35 PM
- March 14 9:02 PM - 11:04 PM
- March 23 12:16 PM - 2:16 PM



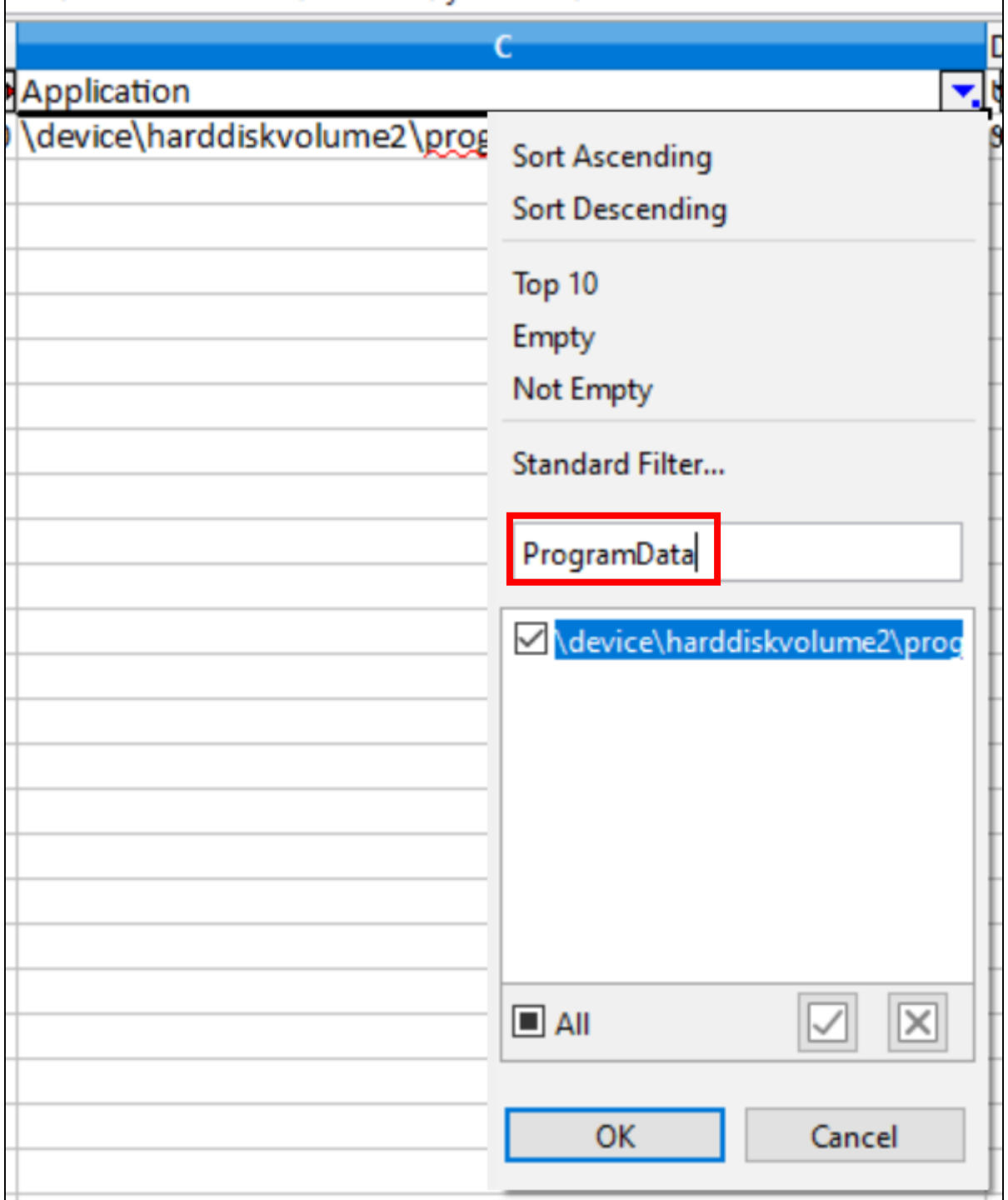

When you filter with "syswow64"...,

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 9682 | 2018-03-07 12:40:00 | \device\harddiskvolume2\windows\syswow64\rundll32.exe | IF_T | 0 | 35,374 | 12,567 | 47 |
| 9697 | 2018-03-07 13:40:00 | \device\harddiskvolume2\windows\syswow64\rundll32.exe | IF_T | 0 | 50,405 | 481,753 | 532 |
| 9711 | 2018-03-07 13:40:00 | \Device\HarddiskVolume2\Windows\SysWOW64\rundll32.exe | IF_T | 0 | 4,929 | 9,322 | 14 |
| 9715 | 2018-03-07 14:41:00 | \device\harddiskvolume2\windows\syswow64\rundll32.exe | IF_T | 0 | 65,408 | 19,268 | 84 |
| 9722 | 2018-03-07 14:41:00 | \Device\HarddiskVolume2\Windows\SysWOW64\net1.exe | IF_T | 0 | 18,886 | 26,741 | 45 |

We can find net1.exe and nslookup.exe at the same time as the malware's spikes on the communications.

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 9934 | 2018-03-08 04:56:00 | \device\harddiskvolume2\windows\syswow64\rundll32.exe | IF_T | 0 | 42,335 | 12,654 | 54 |
| 9951 | 2018-03-08 05:57:00 | \device\harddiskvolume2\windows\syswow64\rundll32.exe | IF_T | 0 | 65,799 | 26,401 | 92 |
| 9965 | 2018-03-08 05:57:00 | \device\harddiskvolume2\windows\syswow64\nslookup.exe | IF_T | 0 | 255 | 530 | |
| 9966 | 2018-03-08 05:57:00 | \Device\HarddiskVolume2\Windows\SysWOW64\rundll32.exe | IF_T | 0 | 1,946 | 5,016 | 6 |
| 9968 | 2018-03-08 06:58:00 | \device\harddiskvolume2\windows\syswow64\rundll32.exe | IF_T | 0 | 47,905 | 15,654 | 63 |
| 9991 | 2018-03-08 07:58:00 | \device\harddiskvolume2\windows\syswow64\rundll32.exe | IF_T | 0 | 42,542 | 12,567 | 55 |

Note that these binaries are false positive and they are not related to this incident because of the dates.
- searchprotocolhost.exe
- msiexec.exe
- IMEWDBLD.EXE

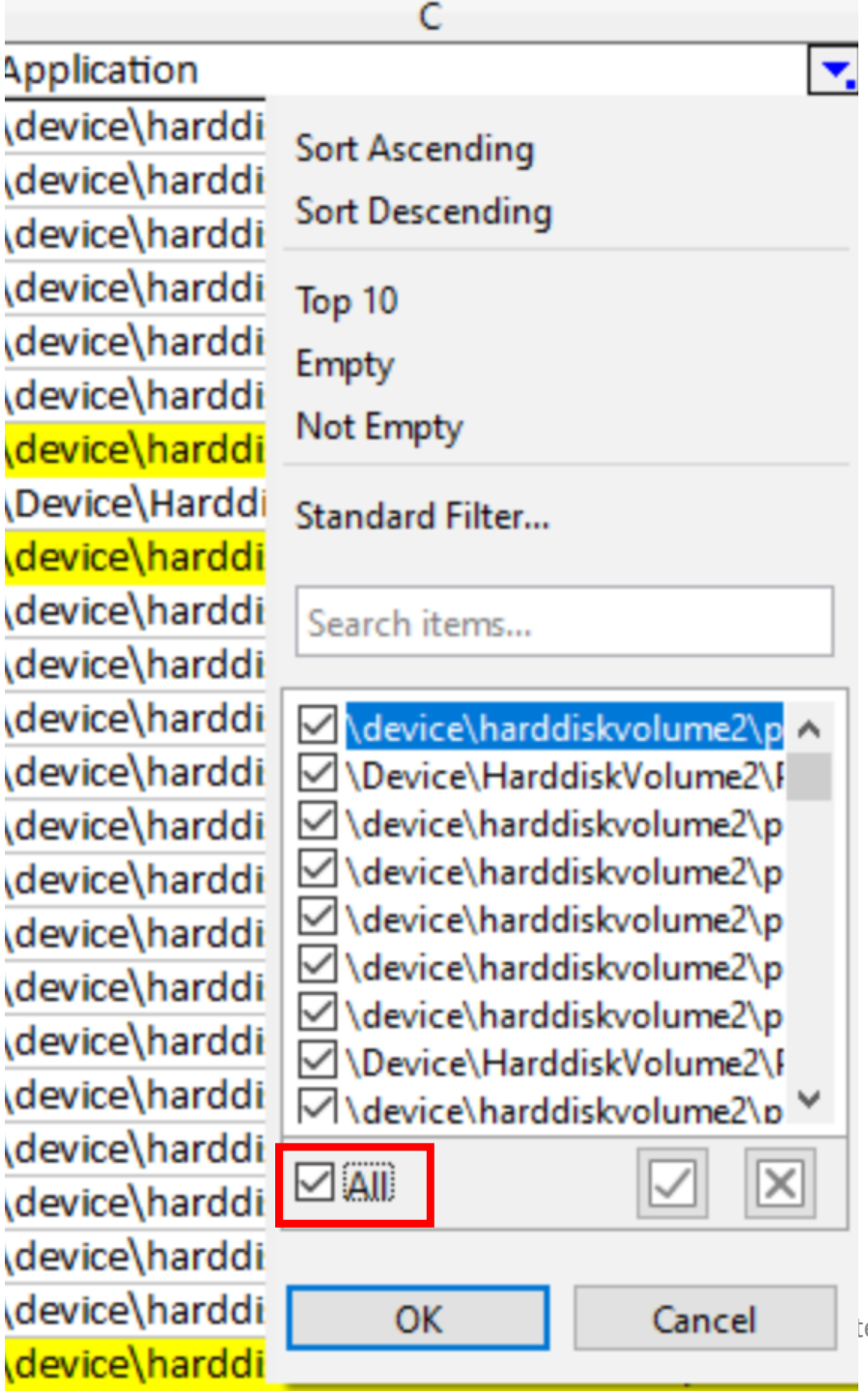



When you filter with "ProgramData"...,

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 12541 | 2018-03-14 12:02:00 | \device\harddiskvolume2\windows\syswow64\rundll32.exe | 9 | IF_T | 0 | 40,412 | 12,780 | 53 |
| 12556 | 2018-03-14 13:03:00 | \device\harddiskvolume2\windows\syswow64\rundll32.exe | 9 | IF_T | 0 | 190,827 | 36,817 | 227 |
| 12569 | 2018-03-14 13:03:00 | \Device\HarddiskVolume2\Windows\SysWOW64\rundll32.exe | 9 | IF_T | 0 | 1,346 | 2,936 | 4 |
| 12579 | 2018-03-14 14:04:00 | \device\harddiskvolume2\windows\syswow64\rundll32.exe | 9 | IF_T | 0 | 141,900,642 | 16,798,753 | |

We can also find dq.exe at the same time as the malware's spikes on a communication.

| A | B | C | D | E | F | G | H | I | J |
|---|---|---|---|---|---|---|---|---|---|
| SRU | SRUM_ENTRY_CREATI | Application | U | In | P | P | Bytes Sent | Bytes Rece | Total |
| 12584 | 2018-03-14 14:04:00 | \device\harddiskvolume2\programdata\s\dq.exe | 9 | IF_T | | 0 | 5,941 | 9,274 | 15 |




Finally, check records  having the same time as the malware's spikes on a communication.

| | | | | | | |
|---|---|---|---|---|---|---|
| 2018-03-07 12:40:00 | \device\harddiskvolume2\windows\system32\browser_broker.exe | S-1 | IF_TYP | 0 | 560 | 4,48 |
| 2018-03-07 13:40:00 | \device\harddiskvolume2\windows\syswow64\rundll32.exe | S-1 | IF_TYP | 0 | 50,405 | 481,75 |
| 2018-03-07 13:40:00 | \device\harddiskvolume2\program files\microsoft office 15\root\office | S-1 | IF_TYP | 0 | 23,644 | 18,36 |
| 2018-03-07 13:40:00 | \device\harddiskvolume2\windows\system32\wbem\wmiprvse.exe | S-1 | IF_TYP | 0 | 1,254 | 6,75 |
| 2018-03-07 14:41:00 | \device\harddiskvolume2\windows\system32\ | | | | | |
| 2018-03-07 14:41:00 | Dnscache | | | | | |
| 2018-03-07 14:41:00 | \device\harddiskvolume2\windows\syswow64\rundll32.exe | S-1 | IF_TYP | 0 | 65,408 | 19,26 |
| 2018-03-07 14:41:00 | System\SMB | S-1 | IF_TYP | 0 | 76,146 | 71,38 |
| 2018-03-07 14:41:00 | \device\harddiskvolume2\program files\microsoft office 15\root\office | S-1 | IF_TYP | 0 | 23,944 | 18,30 |
| 2018-03-07 14:41:00 | \device\harddiskvolume2\windows\system32\wbem\wmiprvse.exe | S-1 | IF_TYP | 0 | 3,360 | 21,20 |

wmiprvse.exe processes imply WMI related activities.

| | | | | | | |
|---|---|---|---|---|---|---|
| 2018-03-12 13:35:00 | \device\harddiskvolume2\windows\syswow64\rundll32.exe | S-1 | IF_TYP | 0 | 110, | |
| 2018-03-12 13:35:00 | \d | | | | | |
| 2018-03-12 13:35:00 | \device\harddiskvolume2\windows\system32\windowspowershell\v1.0\powershell.exe | S-1 | IF_TYP | 0 | 19, | |
| 2018-03-12 13:35:00 | \Device\HarddiskVolume2\Windows\System32\cscript.exe | S-1 | IF_TYP | 0 | | |

These powershell.exe and cscript.exe might have been called by the attacker via the malware.

| | | | | | | |
|---|---|---|---|---|---|---|
| 2018-03-14 14:04:00 | \device\harddiskvolume2\windows\syswow64\rundll32.exe | S-1 | IF_TYP | 0 | 141,900,642 | 16,79 |
| 2018-03-14 14:04:00 | \device\harddiskvolume2\windows\system32\lsass.exe | S-1 | IF_TYP | 0 | 3,897 | |
| 2018-03-14 14:04:00 | \device\harddiskvolume2\windows\system32\mstsc.exe | S-1 | IF_TYP | 0 | 695,983 | 3,63 |

mstsc.exe is the remote desktop client. It might be related to the attacker's lateral movement activity.

- Next, let’s take a look at “Application Resource Usage” tab.
- If you filter with “syswow64”...,

You have already known these periods are suspicious.
- March 7 9:40 PM - 11:41 PM (in JST: UTC+9:00)
- March 8 1:56 PM - 2:57 PM
- March 9 2:06 PM - 3:06 PM
- March 12 8:33 PM - 10:35 PM
- March 14 9:02 PM - 11:04 PM
- March 23 12:16 PM - 2:16 PM

| | | |
|---|---|---|
| 44406 | 2018-03-07 07:37:00 | \Device\HarddiskVolume2\Windows\SysWOW64\cmd.exe |
| 44407 | 2018-03-07 07:37:00 | \Device\HarddiskVolume2\Windows\SysWOW64\rundll32.exe |
| 44464 | 2018-03-07 08:37:00 | \Device\HarddiskVolume2\Windows\SysWOW64\rundll32.exe |
| 44521 | 2018-03-07 09:37:00 | \Device\HarddiskVolume2\Windows\SysWOW64\rundll32.exe |
| 44576 | 2018-03-07 10:38:00 | \Device\HarddiskVolume2\Windows\SysWOW64\rundll32.exe |
| 44632 | 2018-03-07 11:39:00 | \Device\HarddiskVolume2\Windows\SysWOW64\rundll32.exe |
| 44686 | 2018-03-07 12:40:00 | \Device\HarddiskVolume2\Windows\SysWOW64\rundll32.exe |
| 44726 | 2018-03-07 13:40:00 | \Device\HarddiskVolume2\Windows\SysWOW64\wbem\WmiPrvSE.exe |
| 44745 | 2018-03-07 13:40:00 | \Device\HarddiskVolume2\Windows\SysWOW64\cmd.exe |
| 44746 | 2018-03-07 13:40:00 | \Device\HarddiskVolume2\Windows\SysWOW64\rundll32.exe |
| 44747 | 2018-03-07 13:40:00 | \Device\HarddiskVolume2\Windows\SysWOW64\WindowsPowerShell\v1.0\powershell.exe |
| 44790 | 2018-03-07 14:41:00 | \Device\HarddiskVolume2\Windows\SysWOW64\wbem\WmiPrvSE.exe |
| 44810 | 2018-03-07 14:41:00 | \Device\HarddiskVolume2\Windows\SysWOW64\cmd.exe |
| 44811 | 2018-03-07 14:41:00 | \Device\HarddiskVolume2\Windows\SysWOW64\rundll32.exe |

These are close to the infection date.

These are close to the information collection date.

- Although, we couldn't find cmd.exe, powershell.exe and WmiPrvSE.exe on Network Usage tab, we did in Application Resource Usage tab!

| | | | |
|---|---|---|---|
| 42102 | 45683 | 2018-03-08 05:57:00 | \Device\HarddiskVolume2\Windows\SysWOW64\SearchProtocolHost.exe |
| 42107 | 45688 | 2018-03-08 05:57:00 | \Device\HarddiskVolume2\Windows\SysWOW64\cmd.exe |
| 42108 | 45689 | 2018-03-08 05:57:00 | \Device\HarddiskVolume2\Windows\SysWOW64\rundll32.exe |
| 42109 | 45690 | 2018-03-08 05:57:00 | \Device\HarddiskVolume2\Windows\SysWOW64\prevhost.exe |
| 42169 | 45750 | 2018-03-08 06:58:00 | \Device\HarddiskVolume2\Windows\SysWOW64\SearchProtocolHost.exe |
| 42175 | 45756 | 2018-03-08 06:58:00 | \Device\HarddiskVolume2\Windows\SysWOW64\cmd.exe |
| 42176 | 45757 | 2018-03-08 06:58:00 | \Device\HarddiskVolume2\Windows\SysWOW64\rundll32.exe |
| 42231 | 45812 | 2018-03-08 07:58:00 | \Device\HarddiskVolume2\Windows\SysWOW64\SearchProtocolHost.exe |
| 42237 | 45818 | 2018-03-08 07:58:00 | \Device\HarddiskVolume2\Windows\SysWOW64\rundll32.exe |

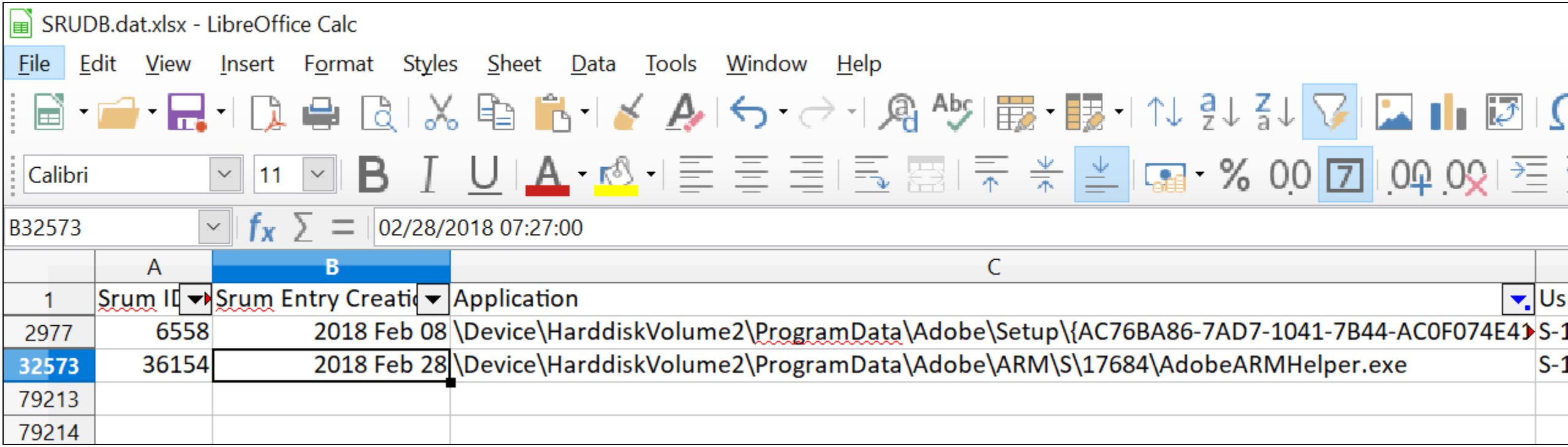
| | | | |
|---|---|---|---|
| 49160 | 52741 | 2018-03-12 12:34:00 | \Device\HarddiskVolume2\Windows\SysWOW64\rundll32.exe |
| 49231 | 52812 | 2018-03-12 13:35:00 | \Device\HarddiskVolume2\Windows\SysWOW64\cmd.exe |
| 49232 | 52813 | 2018-03-12 13:35:00 | \Device\HarddiskVolume2\Windows\SysWOW64\rundll32.exe |
| 49301 | 52882 | 2018-03-12 14:36:00 | \Device\HarddiskVolume2\Windows\SysWOW64\rundll32.exe |

You can find several 32-bit cmd.exe around the spiked date except for the records at March 9.

| | | | |
|---|---|---|---|
| 52522 | 56103 | 2018-03-14 12:02:00 | \Device\HarddiskVolume2\Windows\SysWOW64\rundll32.exe |
| 52596 | 56177 | 2018-03-14 13:03:00 | \Device\HarddiskVolume2\Windows\SysWOW64\cmd.exe |
| 52597 | 56178 | 2018-03-14 13:03:00 | \Device\HarddiskVolume2\Windows\SysWOW64\rundll32.exe |
| 52677 | 56258 | 2018-03-14 14:04:00 | \Device\HarddiskVolume2\Windows\SysWOW64\cmd.exe |
| 52678 | 56259 | 2018-03-14 14:04:00 | \Device\HarddiskVolume2\Windows\SysWOW64\rundll32.exe |
| 52752 | 56333 | 2018-03-14 15:04:00 | \Device\HarddiskVolume2\Windows\SysWOW64\rundll32.exe |

| | | | |
|---|---|---|---|
| 65582 | 69163 | 2018-03-23 04:16:00 | \Device\HarddiskVolume2\Windows\SysWOW64\rundll32.exe |
| 65584 | 69165 | 2018-03-23 04:16:00 | \Device\HarddiskVolume2\Windows\SysWOW64\SearchProtocolHost.exe |
| 65595 | 69176 | 2018-03-23 04:16:00 | \Device\HarddiskVolume2\Windows\SysWOW64\cmd.exe |
| 65652 | 69233 | 2018-03-23 05:16:00 | \Device\HarddiskVolume2\Windows\SysWOW64\rundll32.exe |
| 65654 | 69235 | 2018-03-23 05:16:00 | \Device\HarddiskVolume2\Windows\SysWOW64\SearchProtocolHost.exe |
| 65659 | 69240 | 2018-03-23 05:16:00 | \Device\HarddiskVolume2\Windows\SysWOW64\cmd.exe |

- There is no interesting result for filtering with “ProgramData”.

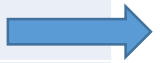



| | A | B | C | |
|---|---|---|---|---|
| 1 | Srum ID | Srum Entry Creati | Application | Us |
| 2977 | 6558 | 2018 Feb 08 | \Device\HarddiskVolume2\ProgramData\Adobe\Setup\{AC76BA86-7AD7-1041-7B44-AC0F074E41 | S-1 |
| 32573 | 36154 | 2018 Feb 28 | \Device\HarddiskVolume2\ProgramData\Adobe\ARM\S\17684\AdobeARMHelper.exe | S-1 |
| 79213 | | | | |
| 79214 | | | | |

# Summary for This Lab

- You was able to find several suspicious spiked communications from malware.

| Date Range (in JST: UTC+9:00) | Assumed Activities | Sent/Recv Bytes |
|---|---|---|
| March 7 9:40 PM - 11:41 PM | Information Collection | 50,405/481,753 |
| March 8 1:56 PM - 2:57 PM | SchedTasks and W10 execution | 65,799/26,401 |
| March 9 2:06 PM - 3:06 PM | ??? (w/o cmd.exe) | 88,798/18,062 |
| March 12 8:33 PM - 10:35 PM | ??? (w/ cmd.exe) | 110,178/20,513 |
| March 14 9:02 PM - 11:04 PM | Lateral Movement Occurred? (Client-Win10-1 was infected around this date) | 141,900,642/16,798,753 |
| March 23 12:16 PM - 2:16 PM | l.exe execution | 77,141/25,684 |



We got a possible new activity by the attacker.

# Scenario 1 Labs:
# Lab 5: Analyzing SRUM on Client-Win10-1

# Creating the Result of srum-dump

- You can use srum-dump

```
cd C:\tools\srum-dump\
```

```
srum_dump -i E:\Artifacts\scenario1_srum\Client-Win10-1_toyoda\SRUDB.dat -r
E:\Artifacts\scenario1_srum\Client-Win10-1_toyoda\SOFTWARE -o
E:\Artifacts\scenario1_srum\Client-Win10-1_toyoda\SRUDB.dat.xlsx
```

Note that this command takes a long time! We have already prepared the command result.
This is one line. Don't start a new line.

- -i: Input file. You need to specify the path to a SRUDB.dat
  - The default path of SRUDB.dat is **C:\Windows\System32\sru\SRUDB.dat**
- -r: You can specify a HKLM\SOFTWARE hive to resolve WLAN's interface name.
  - The default path of SOFTWARE hive is **C:\Windows\System32\Config\SOFTWARE**
- -o: Output file. You can specify the output file path.

# The Strategy of Analyzing srum-dump Results

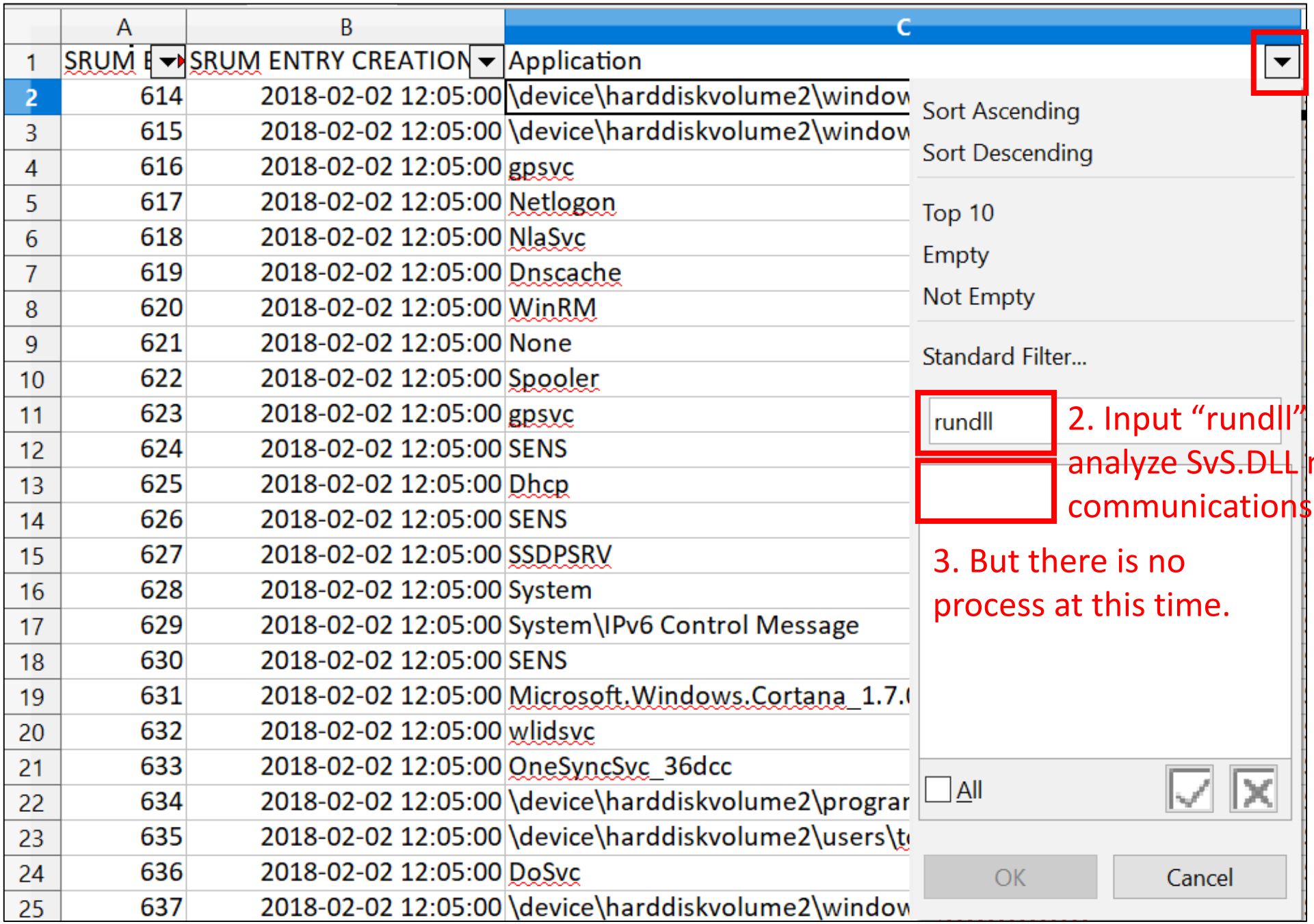
- We shouldn’t look each line one by one because the number of the records are huge.
- We have several pivot points that:
  - Malware processes that were found from this host are “rundll32.dll (SvS.DLL)” and “AddinsManager.exe”.
  - Both of the malware are for 32-bit environment. Therefore, program execution records of standard command-line executables which are located in the “SysWow64” folder are suspicious.
  - The attacker used “ProgramData” and “ProgramData\s” folders as the foothold folders.
- You can also check around the timestamps of the pivot points that we found so far.
- Let’s filter with these keywords and take a look at around the timestamps.

# Preparation for This Exercise

- First, let’s open the path below.
  - `E:\Artifacts\scenario1_srum\Client-Win10-1_toyoda\SRUDB.dat.xlsx`

- And change columns format as we mentioned in the previous exercise.

1. Click the triangle button on the Application column.



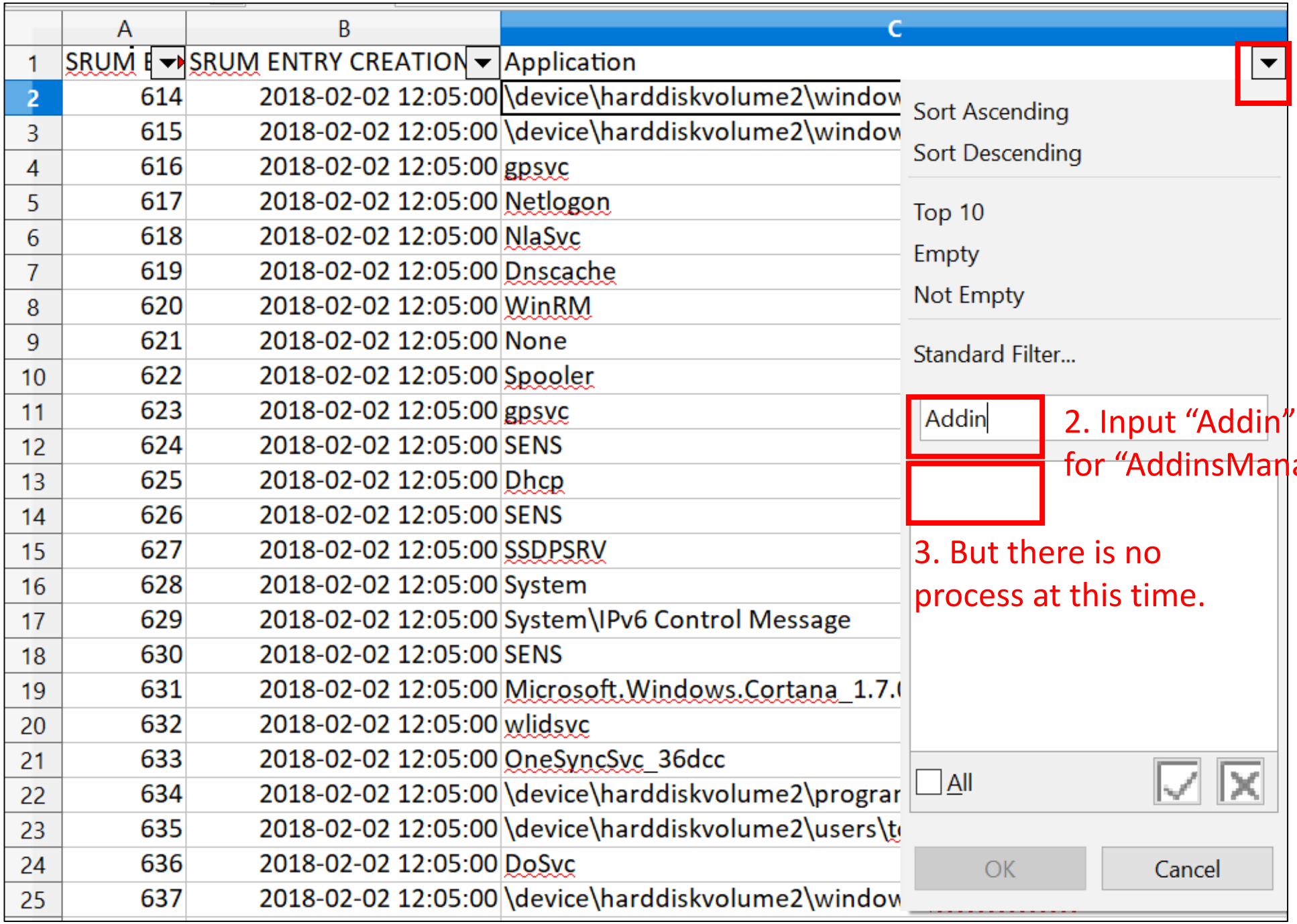
| | A | B | C |
|---|---|---|---|
| 1 | SRUM E | SRUM ENTRY CREATION | Application |
| 2 | 614 | 2018-02-02 12:05:00 | \device\harddiskvolume2\windows |
| 3 | 615 | 2018-02-02 12:05:00 | \device\harddiskvolume2\windows |
| 4 | 616 | 2018-02-02 12:05:00 | gpsvc |
| 5 | 617 | 2018-02-02 12:05:00 | Netlogon |
| 6 | 618 | 2018-02-02 12:05:00 | NlaSvc |
| 7 | 619 | 2018-02-02 12:05:00 | Dnscache |
| 8 | 620 | 2018-02-02 12:05:00 | WinRM |
| 9 | 621 | 2018-02-02 12:05:00 | None |
| 10 | 622 | 2018-02-02 12:05:00 | Spooler |
| 11 | 623 | 2018-02-02 12:05:00 | gpsvc |
| 12 | 624 | 2018-02-02 12:05:00 | SENS |
| 13 | 625 | 2018-02-02 12:05:00 | Dhcp |
| 14 | 626 | 2018-02-02 12:05:00 | SENS |
| 15 | 627 | 2018-02-02 12:05:00 | SSDPSRV |
| 16 | 628 | 2018-02-02 12:05:00 | System |
| 17 | 629 | 2018-02-02 12:05:00 | System\IPv6 Control Message |
| 18 | 630 | 2018-02-02 12:05:00 | SENS |
| 19 | 631 | 2018-02-02 12:05:00 | Microsoft.Windows.Cortana_1.7.( |
| 20 | 632 | 2018-02-02 12:05:00 | wlidsvc |
| 21 | 633 | 2018-02-02 12:05:00 | OneSyncSvc_36dcc |
| 22 | 634 | 2018-02-02 12:05:00 | \device\harddiskvolume2\prograr |
| 23 | 635 | 2018-02-02 12:05:00 | \device\harddiskvolume2\users\t( |
| 24 | 636 | 2018-02-02 12:05:00 | DoSvc |
| 25 | 637 | 2018-02-02 12:05:00 | \device\harddiskvolume2\windows |



2. Input “rundll” to analyze SvS.DLL malware communications.

3. But there is no process at this time.

1. Click the triangle button on the Application column.



| | A | B | C |
|---|---|---|---|
| 1 | SRUM E | SRUM ENTRY CREATION | Application |
| 2 | 614 | 2018-02-02 12:05:00 | \device\harddiskvolume2\windows |
| 3 | 615 | 2018-02-02 12:05:00 | \device\harddiskvolume2\windows |
| 4 | 616 | 2018-02-02 12:05:00 | gpsvc |
| 5 | 617 | 2018-02-02 12:05:00 | Netlogon |
| 6 | 618 | 2018-02-02 12:05:00 | NlaSvc |
| 7 | 619 | 2018-02-02 12:05:00 | Dnscache |
| 8 | 620 | 2018-02-02 12:05:00 | WinRM |
| 9 | 621 | 2018-02-02 12:05:00 | None |
| 10 | 622 | 2018-02-02 12:05:00 | Spooler |
| 11 | 623 | 2018-02-02 12:05:00 | gpsvc |
| 12 | 624 | 2018-02-02 12:05:00 | SENS |
| 13 | 625 | 2018-02-02 12:05:00 | Dhcp |
| 14 | 626 | 2018-02-02 12:05:00 | SENS |
| 15 | 627 | 2018-02-02 12:05:00 | SSDPSRV |
| 16 | 628 | 2018-02-02 12:05:00 | System |
| 17 | 629 | 2018-02-02 12:05:00 | System\IPv6 Control Message |
| 18 | 630 | 2018-02-02 12:05:00 | SENS |
| 19 | 631 | 2018-02-02 12:05:00 | Microsoft.Windows.Cortana_1.7.( |
| 20 | 632 | 2018-02-02 12:05:00 | wlidsvc |
| 21 | 633 | 2018-02-02 12:05:00 | OneSyncSvc_36dcc |
| 22 | 634 | 2018-02-02 12:05:00 | \device\harddiskvolume2\prograr |
| 23 | 635 | 2018-02-02 12:05:00 | \device\harddiskvolume2\users\t( |
| 24 | 636 | 2018-02-02 12:05:00 | DoSvc |
| 25 | 637 | 2018-02-02 12:05:00 | \device\harddiskvolume2\windows |



2. Input “Addin” to get entries for “AddinsManager.exe”

3. But there is no process at this time.

# Note of SRUM's Network Usage

- We couldn't get the results for both malware processes due to some reasons at this time. Unfortunately, it seems that SRUM sometimes doesn't record some of the communications…
- Let's continue the analysis. Next, let's check binaries in the syswow64 directory.

These are no related to this incidents because of the date.

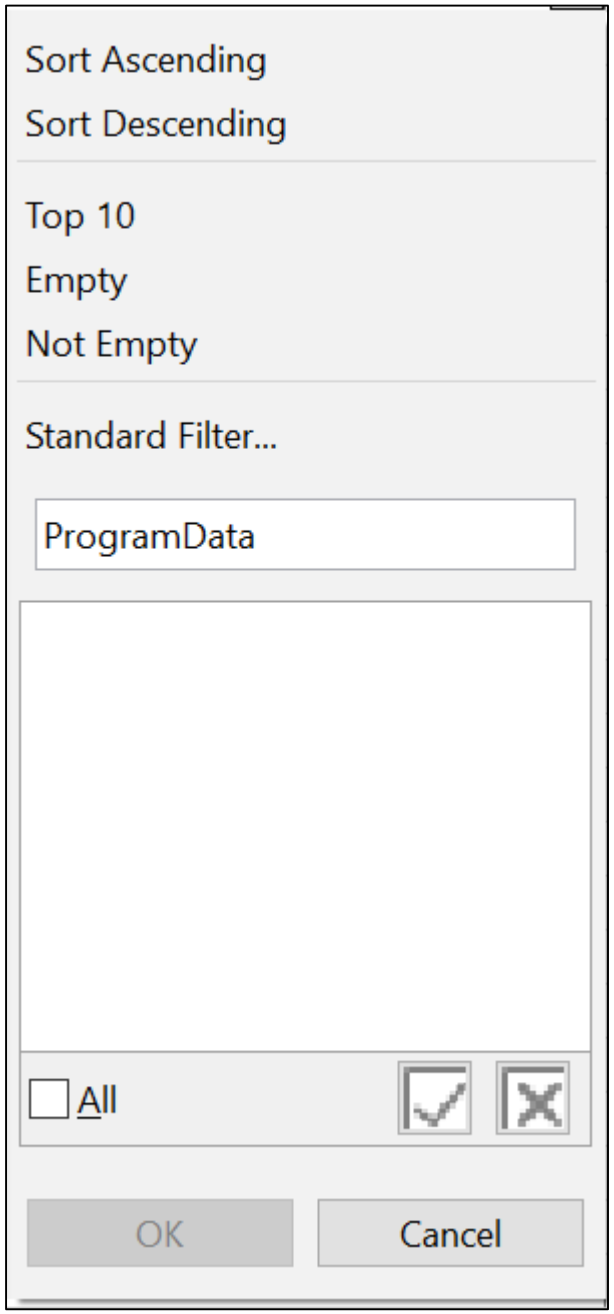
These binaries imply the attacker might have used NET USER/GROUP command to get accounts information or NET USE command to mount network share with SMB.

The attacker used certain scripts written in JScript or VBScript.

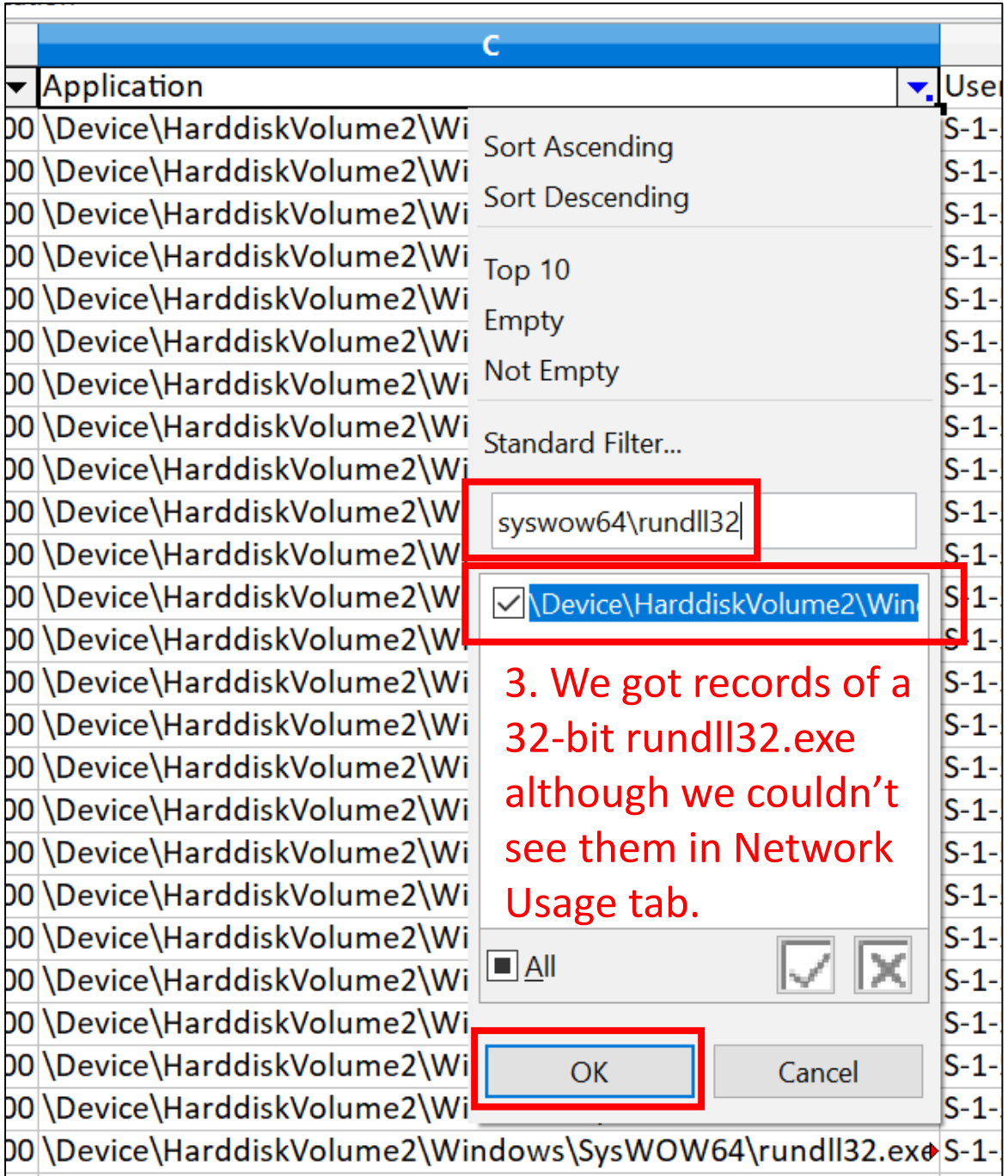
| | A | B | C | D | E | F | G | H | I |
|---|---|---|---|---|---|---|---|---|---|
| | SRU | SRUM ENTRY CRE | Application | User SID | I | | | Bytes | Bytes Re |
| | 1394 | 2018-02-08 07:54:00 | \Device\HarddiskVolume2\Windows\SysWOW64\msiexec.exe | S-1-5-21-3671970501-397 | U | | # | 0 | 0 |
| | 1430 | 2018-02-08 08:38:00 | \device\harddiskvolume2\windows\syswow64\msiexec.exe | S-1-5-21-3671970501-397 | IF | | # | 454 | 4,494 |
| | 1432 | 2018-02-08 08:38:00 | \Device\HarddiskVolume2\Windows\SysWOW64\msiexec.exe | S-1-5-21-3671970501-397 | U | | # | 0 | 0 |
| 6 | 14158 | 2018-03-22 09:07:00 | \Device\HarddiskVolume2\Windows\SysWOW64\net1.exe | S-1-5-18 ( Local System) | U | | # | 16,192 | 13,449 |
| 8 | 14160 | 2018-03-22 09:07:00 | \Device\HarddiskVolume2\Windows\SysWOW64\net.exe | S-1-5-18 ( Local System) | U | | # | 4,038 | 3,493 |
| 9 | 14161 | 2018-03-22 09:07:00 | \Device\HarddiskVolume2\Windows\SysWOW64\cscript.exe | S-1-5-18 ( Local System) | U | | # | 8,976 | 6,727 |
| 3 | 14165 | 2018-03-22 09:07:00 | \Device\HarddiskVolume2\Windows\SysWOW64\cmd.exe | S-1-5-18 ( Local System) | U | | # | 15,552 | 27,196 |
| 5 | 14167 | 2018-03-22 10:07:00 | \Device\HarddiskVolume2\Windows\SysWOW64\cmd.exe | S-1-5-18 ( Local System) | U | | # | 41,196 | 90,164 |
| 6 | 14168 | 2018-03-22 10:07:00 | \Device\HarddiskVolume2\Windows\SysWOW64\xcopy.exe | S-1-5-18 ( Local System) | U | | # | 330,941 | 64,173,558 |
| 9 | 14171 | 2018-03-22 10:07:00 | \Device\HarddiskVolume2\Windows\SysWOW64\net.exe | S-1-5-18 ( Local System) | U | | # | 0 | 0 |
| 0 | 14172 | 2018-03-22 10:07:00 | \Device\HarddiskVolume2\Windows\SysWOW64\cscript.exe | S-1-5-18 ( Local System) | U | | # | 3,814 | 4,360 |
| 5 | 14227 | 2018-03-23 04:20:00 | \Device\HarddiskVolume2\Windows\SysWOW64\cscript.exe | S-1-5-18 ( Local System) | U | | # | 2,289 | 2,554 |
| 6 | 14228 | 2018-03-23 04:20:00 | \Device\HarddiskVolume2\Windows\SysWOW64\cmd.exe | S-1-5-18 ( Local System) | U | | # | 8,601 | 16,616 |
| 7 | 14229 | 2018-03-23 04:20:00 | \Device\HarddiskVolume2\Windows\SysWOW64\xcopy.exe | S-1-5-18 ( Local System) | U | | # | 46,219 | 32,670,032 |

The attacker used xcopy.exe at least twice. And the first one is likely that the attacker mounted a remote file system with SMB and copy data from the system. For the second one, the attacker also used xcopy command to get data from a remote host because this record is in the Network Usage tab. The traffic were approximately 64 MB and 33 MB.

- There is no entry when you filter with “ProgramData”.
- But we was able to find remote file copy operation by the attacker.
- Next, let’s check Application Resource Usage tab.

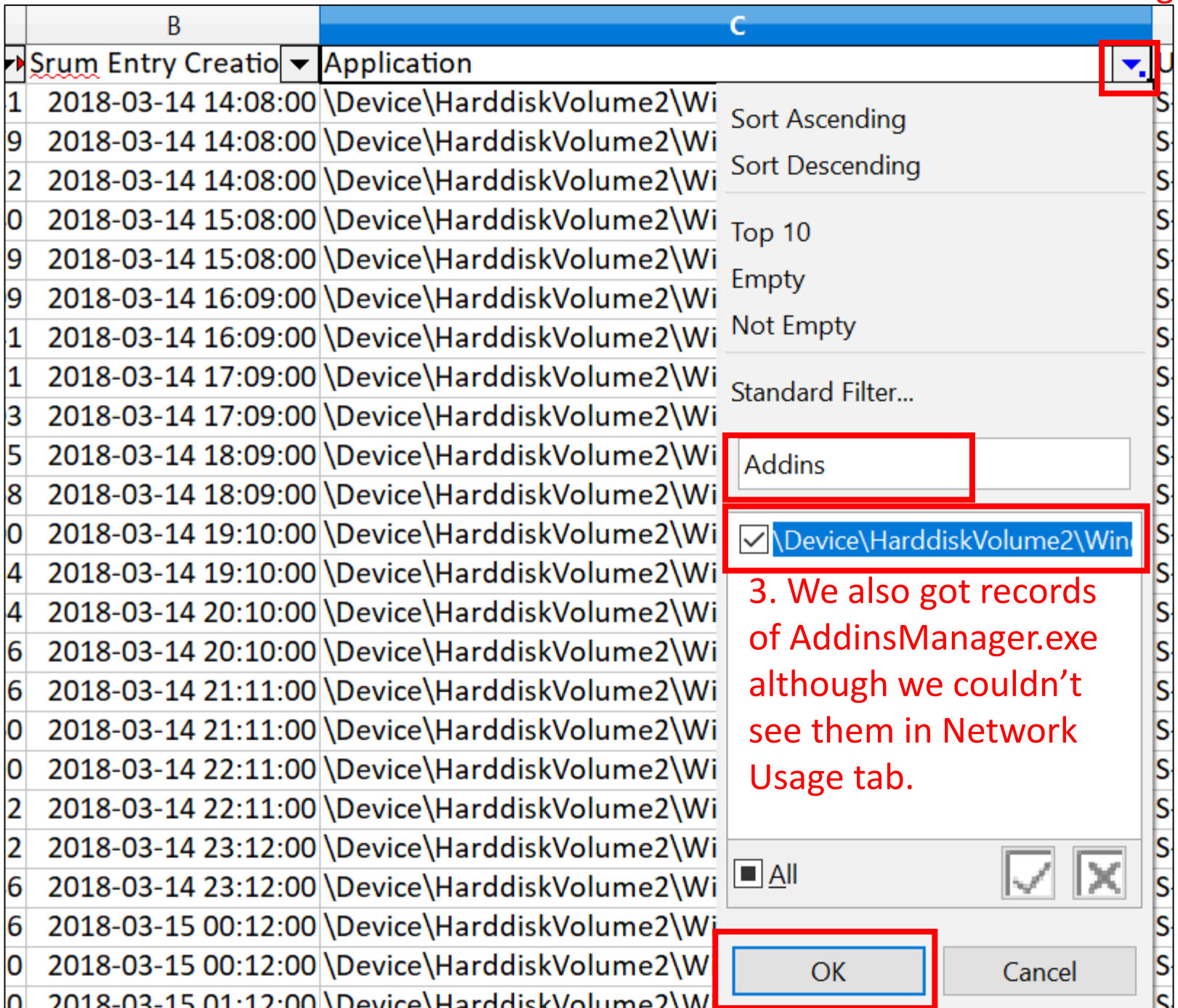


1. Click the triangle button on the Application column.




2. Input "syswow64\rundll32" to get entries for "SvS.DLL".

1. Click the triangle button on the Application column.




2. Input "Addins" to get entries for "AddinsManager.exe".

Malware run on Local System account.

| | | | | |
|---|---|---|---|---|
| 52203 | 57336 | 2018-03-15 09:19:00 | \Device\HarddiskVolume2\Windows\SysWOW64\rundll32.exe | S-1-5-21-3671970501-397572 |
| 52248 | 57381 | 2018-03-15 10:20:00 | \Device\HarddiskVolume2\Windows\SysWOW64\rundll32.exe | S-1-5-18 ( Local System) |
| 52250 | 57383 | 2018-03-15 10:20:00 | \Device\HarddiskVolume2\Windows\SysWOW64\cmd.exe | S-1-5-18 ( Local System) |
| 52287 | 57420 | 2018-03-15 10:20:00 | \Device\HarddiskVolume2\Windows\SysWOW64\rundll32.exe | S-1-5-21-3671970501-397572 |

Rundll32.exe with cmd.exe at these timestamps.
The attacker is like to have been doing something.

| | | | | |
|---|---|---|---|---|
| 58074 | 63207 | 2018-03-19 09:02:00 | \Device\HarddiskVolume2\Windows\SysWOW64\rundll32.exe | S-1-5-21-3671970501-397572 |
| 58103 | 63236 | 2018-03-19 10:03:00 | \Device\HarddiskVolume2\Windows\SysWOW64\rundll32.exe | S-1-5-18 ( Local System) |
| 58104 | 63237 | 2018-03-19 10:03:00 | \Device\HarddiskVolume2\Windows\SysWOW64\cmd.exe | S-1-5-18 ( Local System) |
| 58141 | 63274 | 2018-03-19 10:03:00 | \Device\HarddiskVolume2\Windows\SysWOW64\rundll32.exe | S-1-5-21-3671970501-397572 |
| 58174 | 63307 | 2018-03-19 11:03:00 | \Device\HarddiskVolume2\Windows\SysWOW64\rundll32.exe | S-1-5-18 ( Local System) |
| 58175 | 63308 | 2018-03-19 11:03:00 | \Device\HarddiskVolume2\Windows\SysWOW64\cmd.exe | S-1-5-18 ( Local System) |
| 58214 | 63347 | 2018-03-19 11:03:00 | \Device\HarddiskVolume2\Windows\SysWOW64\rundll32.exe | S-1-5-21-3671970501-397572 |

| | | | | |
|---|---|---|---|---|
| 59597 | 64730 | 2018-03-20 09:38:00 | \Device\HarddiskVolume2\Windows\SysWOW64\rundll32.exe | S-1-5-21-3671970501-397572 |
| 59630 | 64763 | 2018-03-20 10:41:00 | \Device\HarddiskVolume2\Windows\SysWOW64\rundll32.exe | S-1-5-18 ( Local System) |
| 59656 | 64789 | 2018-03-20 10:41:00 | \Device\HarddiskVolume2\Windows\SysWOW64\cmd.exe | S-1-5-18 ( Local System) |
| 59676 | 64809 | 2018-03-20 10:41:00 | \Device\HarddiskVolume2\Windows\SysWOW64\rundll32.exe | S-1-5-21-3671970501-397572 |

The attacker might have mounted a network drive, and execute JScripts or VBScripts.

| | | | |
|---|---|---|---|
| 2018-03-22 08:06:00 | \Device\HarddiskVolume2\Windows\SysWOW64\rundll32.exe | S-1-5-18 ( Local System) | 6.91E+09 |
| 2018-03-22 08:06:00 | \Device\HarddiskVolume2\Windows\SysWOW64\cmd.exe | S-1-5-18 ( Local System) | 27107134 |
| 2018-03-22 08:06:00 | \Device\HarddiskVolume2\Windows\SysWOW64\rundll32.exe | S-1-5-21-3671970501-3975728774-4289435121-1106 | 4.93E+08 |
| 2018-03-22 09:07:00 | \Device\HarddiskVolume2\Windows\SysWOW64\rundll32.exe | S-1-5-18 ( Local System) | 7.09E+09 |
| 2018-03-22 09:07:00 | \Device\HarddiskVolume2\Windows\SysWOW64\cmd.exe | S-1-5-18 ( Local System) | 7.35E+08 |
| 2018-03-22 09:07:00 | \Device\HarddiskVolume2\Windows\SysWOW64\cscript.exe | S-1-5-18 ( Local System) | 7.7E+08 |
| 2018-03-22 09:07:00 | \Device\HarddiskVolume2\Windows\SysWOW64\net.exe | S-1-5-18 ( Local System) | 56454596 |
| 2018-03-22 09:07:00 | \Device\HarddiskVolume2\Windows\SysWOW64\rundll32.exe | S-1-5-18 ( Local System) | 4.58E+08 |
| 2018-03-22 10:07:00 | \Device\HarddiskVolume2\Windows\SysWOW64\rundll32.exe | S-1-5-18 ( Local System) | 7.21E+09 |
| 2018-03-22 10:07:00 | \Device\HarddiskVolume2\Windows\SysWOW64\cmd.exe | S-1-5-18 ( Local System) | 7.64E+08 |
| 2018-03-22 10:07:00 | \Device\HarddiskVolume2\Windows\SysWOW64\xcopy.exe | S-1-5-18 ( Local System) | 4.47E+08 |
| 2018-03-22 10:07:00 | \Device\HarddiskVolume2\Windows\SysWOW64\rundll32.exe | S-1-5-21-3671970501-3975728774-4289435121-1106 | 5.85E+08 |
| 2018-03-22 11:08:00 | \Device\HarddiskVolume2\Windows\SysWOW64\rundll32.exe | S-1-5-18 ( Local System) | 9.51E+09 |
| 2018-03-22 11:08:00 | \Device\HarddiskVolume2\Windows\SysWOW64\cmd.exe | S-1-5-18 ( Local System) | 74524422 |
| 2018-03-22 11:08:00 | \W64\xcopy.exe | S-1-5-18 ( Local System) | 0 |
| 2018-03-22 11:08:00 | \W64\rundll32.exe | S-1-5-21-3671970501-3975728774-4289435121-1106 | 1.1E+09 |
| 2018-03-22 12:08:00 | \W64\rundll32.exe | S-1-5-18 ( Local System) | 6.92E+09 |
| 2018-03-22 12:08:00 | \W64\xcopy.exe | S-1-5-18 ( Local System) | 0 |
| 2018-03-22 12:08:00 | \W64\rundll32.exe | S-1-5-21-3671970501-3975728774-4289435121-1106 | 5.17E+08 |
| 2018-03-22 13:09:00 | \W64\rundll32.exe | S-1-5-18 ( Local System) | 6.92E+09 |
| 2018-03-22 13:09:00 | \W64\xcopy.exe | S-1-5-18 ( Local System) | 0 |

As we have already seen these activities before, xcopy was executed for a relatively long time and they copied data from external hosts.

xcopy is recorded repeatedly, but CPU wasn't used at all here. It might have been a zombie process or have remained for some reasons.

When you filter with “Syswow64”, you can find several rundll32 with cmd.exe. The attacker was doing something.

| | | | |
|---|---|---|---|
| 2018-03-23 03:19:00 | \Device\HarddiskVolume2\Windows\SysWOW64\rundll32.exe | S-1-5-18 ( Local System) | 6.38E+09 |
| 2018-03-23 03:19:00 | \Device\HarddiskVolume2\Windows\SysWOW64\xcopy.exe | S-1-5-18 ( Local System) | 0 |
| 2018-03-23 03:19:00 | \Device\HarddiskVolume2\Windows\SysWOW64\rundll32.exe | S-1-5-21-3671970501-3975728774-4289435121-1106 | 5.36E+08 |
| 2018-03-23 04:20:00 | \Device\HarddiskVolume2\Windows\SysWOW64\rundll32.exe | S-1-5-18 ( Local System) | 1.16E+10 |
| 2018-03-23 04:20:00 | \Device\HarddiskVolume2\Windows\SysWOW64\cmd.exe | S-1-5-18 ( Local System) | 7.06E+08 |
| 2018-03-23 04:20:00 | \Device\HarddiskVolume2\Windows\SysWOW64\xcopy.exe | S-1-5-18 ( Local System) | 1.81E+08 |
| 2018-03-23 04:20:00 | \Device\HarddiskVolume2\Windows\SysWOW64\rundll32.exe | S-1-5-21-3671970501-3975728774-4289435121-1106 | 2.61E+09 |
| 2018-03-23 05:20:00 | \Device\HarddiskVolume2\Windows\SysWOW64\rundll32.exe | S-1-5-18 ( Local System) | 6.97E+09 |
| 2018-03-23 05:20:00 | \Device\HarddiskVolume2\Windows\SysWOW64\cmd.exe | S-1-5-18 ( Local System) | 1.3E+08 |
| 2018-03-23 05:20:00 | \Device\HarddiskVolume2\Windows\SysWOW64\xcopy.exe | S-1-5-18 ( Local System) | 0 |
| 2018-03-23 05:20:00 | \Device\HarddiskVolume2\Windows\SysWOW64\rundll32.exe | S-1-5-21-3671970501-3975728774-4289435121-1106 | 5.74E+08 |
| 2018-03-23 06:21:00 | \Device\HarddiskVolume2\Windows\SysWOW64\rundll32.exe | S-1-5-18 ( Local System) | 7.71E+09 |
| 2018-03-23 06:21:00 | \Device\HarddiskVolume2\Windows\SysWOW64\xcopy.exe | S-1-5-18 ( Local System) | 0 |
| 2018-03-23 06:21:00 | \Device\HarddiskVolume2\Windows\SysWOW64\rundll32.exe | S-1-5-21-3671970501-3975728774-4289435121-1106 | 7.85E+08 |
| 2018-03-23 07:22:00 | \Device\HarddiskVolume2\Windows\SysWOW64\rundll32.exe | S-1-5-18 ( Local System) | 7.88E+09 |
| 2018-03-23 07:22:00 | \Device\HarddiskVolume2\Windows\SysWOW64\xcopy.exe | S-1-5-18 ( Local System) | 0 |

When you filter with “ProgramData”, you can find execution of 7.exe and l.exe the activities happened at the same time as the above activities.

| | Srum Entry Creatio | Application | User SID |
|---|---|---|---|
| 3 | 2018-02-08 07:54:00 | \Device\HarddiskVolume2\ProgramData\Adobe\Setup\{AC76BA86-7A | S-1-5-21-3671970501-3975728774-4289435121- |
| 7 | 2018-03-23 04:20:00 | \Device\HarddiskVolume2\ProgramData\s\7.exe | S-1-5-18 ( Local System) |
| 8 | 2018-03-23 05:20:00 | \Device\HarddiskVolume2\ProgramData\l.exe | S-1-5-18 ( Local System) |

# Summary for This Lab

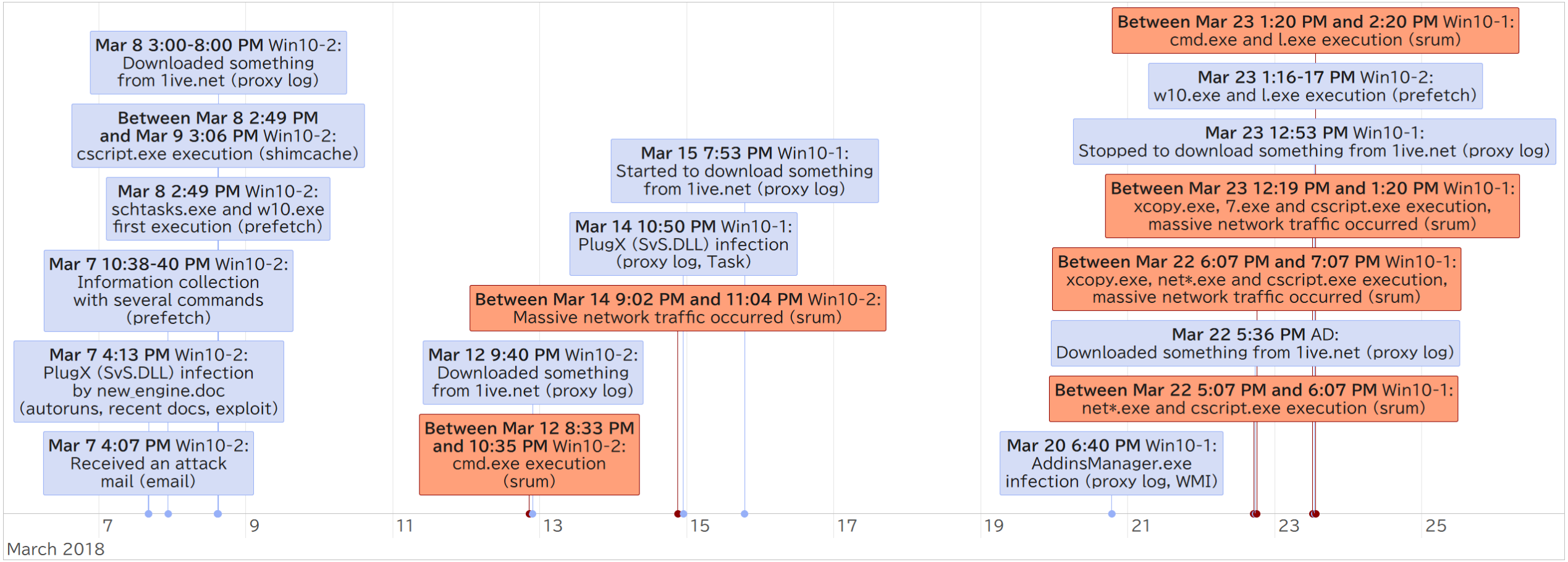
- You was able to find several attacker's activities.

| Date Range (in JST: UTC+9:00) | Assumed Activities | Sent/Recv Bytes |
|---|---|---|
| Mar 15 6:19 PM - 7:20 PM | rundll32 w/ cmd | |
| Mar 19 6:02 PM - 8:03 PM | rundll32 w/ cmd | |
| Mar 20 6:38 PM - 7:41 PM | rundll32 w/ cmd | |
| Mar 22 5:07 PM - 6:07 PM | rundll32 w/ net, cscript | |
| Mar 22 6:07 PM - 7:07 PM | rundll32 w/ xcopy, net, cscript | 330,942/64,173,558 |
| Mar 23 12:19 PM - 1:20 PM | rundll32 w/ xcopy, cscript, 7.exe | 46,219/32,670,032 |
| Mar 23 1:20 PM - 2:20 PM | rundll32 w/ cmd, l.exe | |

We got attacker's new activities.

We were able to narrow down these attacker's activities rather than shimcache analysis and we got the exact number of the traffic volume.

# Scenario 1 Incident Timeline After SRUM Analysis



# UserAssist

# UserAssist

- Keeps track of frequently used applications.
  - The data contains path to the EXE file, number of times the program was executed, and timestamp the UserAssist entry was modified.
    - When the entry is updated, the timestamp will change; timestamp can be interpreted as the last time the program was executed.
- The UserAssist is recorded in registry under “HKEY_CURRENT_USER\SOFTWARE\Microsoft\Windows\CurrentVersion\Explorer\UserAssist” key.
  - Each entry name is “ROT13” of the EXE file path.
  - Each data is in binary.

# ROT13

- ROT13 means to rotate the alphabet for 13 letters.
  - Numbers and symbols remain unchanged.

| First 13 Letters | A | B | C | D | E | F | G | H | I | J | K | L | M |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Later 13 Letters | N | O | P | Q | R | S | T | U | V | W | X | Y | Z |

- For example, “P:\Hfref\gneb” refers to...?

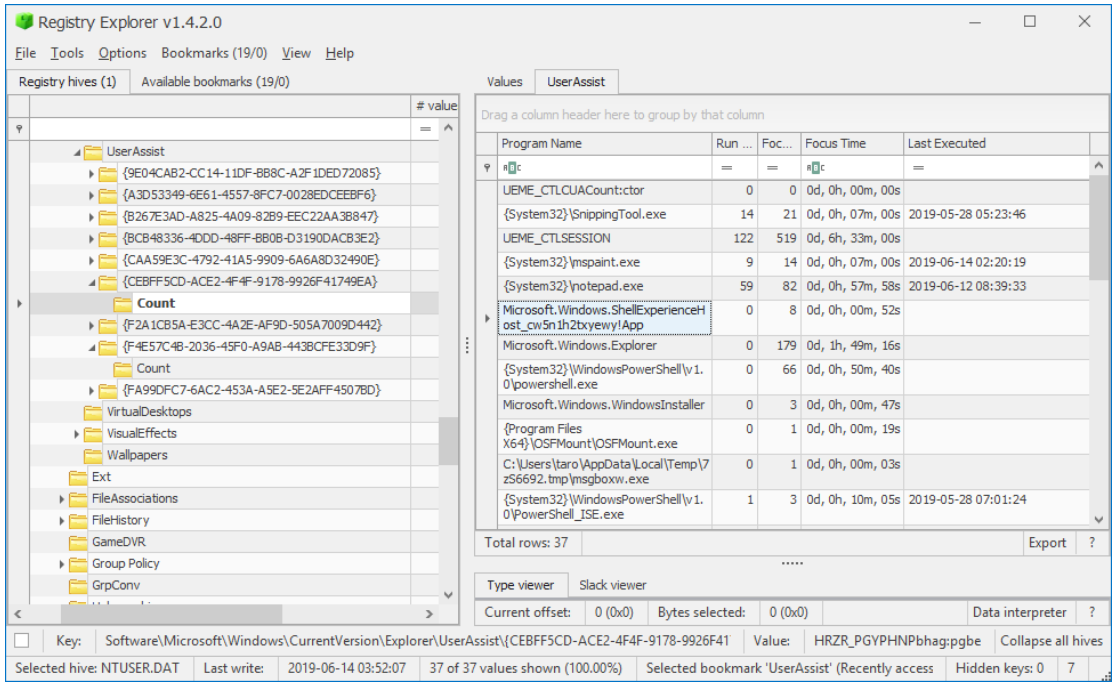
C:\Users\taro

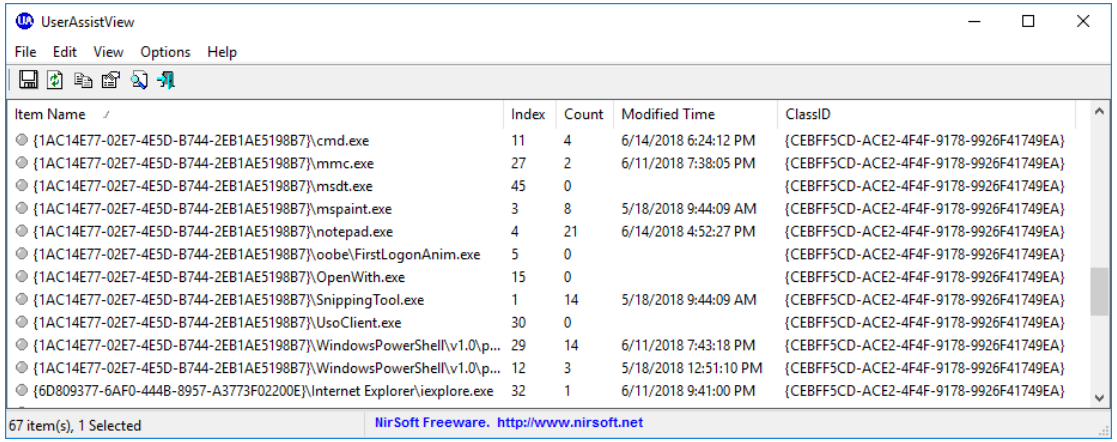
# Characteristics of UserAssist

- Only the programs executed from Windows Explorer are recorded in UserAssist.
- Therefore, it is useful for investigating legitimate user activities.
  - For example, malware infection with social engineering, or internal attacks triggered by the user.
- If the attacker used GUI, the UserAssist may be recorded.
- Or if malware used Startup as a persistence, the evidence would be recorded in UserAssist.
  - In that case, runcount would be kept to be zero and the timestamp would be 1601/01/01.
- A researcher claimed that it can also record Scheduled Task's activities in a certain condition, although we couldn't reproduce it.
  - https://medium.com/@forensic_matt/no-run-counts-in-userassist-4f2a78cb9e2c

# Parsing UserAssist

- It will be easier to use the tool to read the UserAssist data.
  - To read the name of UserAssist entries, it is necessary to calculate ROT13.
  - The contents of UserAssist data are in binary format.
- Registry Explorer by Eric Zimmarman
  - It can parse UserAssist both online and offline.
- UserAssistView by NirSoft
  - Useful for parsing UserAssist of the **running machine**.
  - Since it cannot read contents of the offline hosts, one way is to use the Registry Explorer for viewing them.



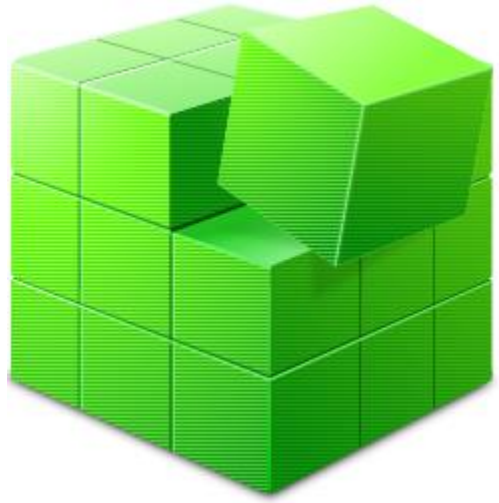



# Scenario 1 Labs:
# Lab 6: Parsing UserAssist for Client-Win10-2

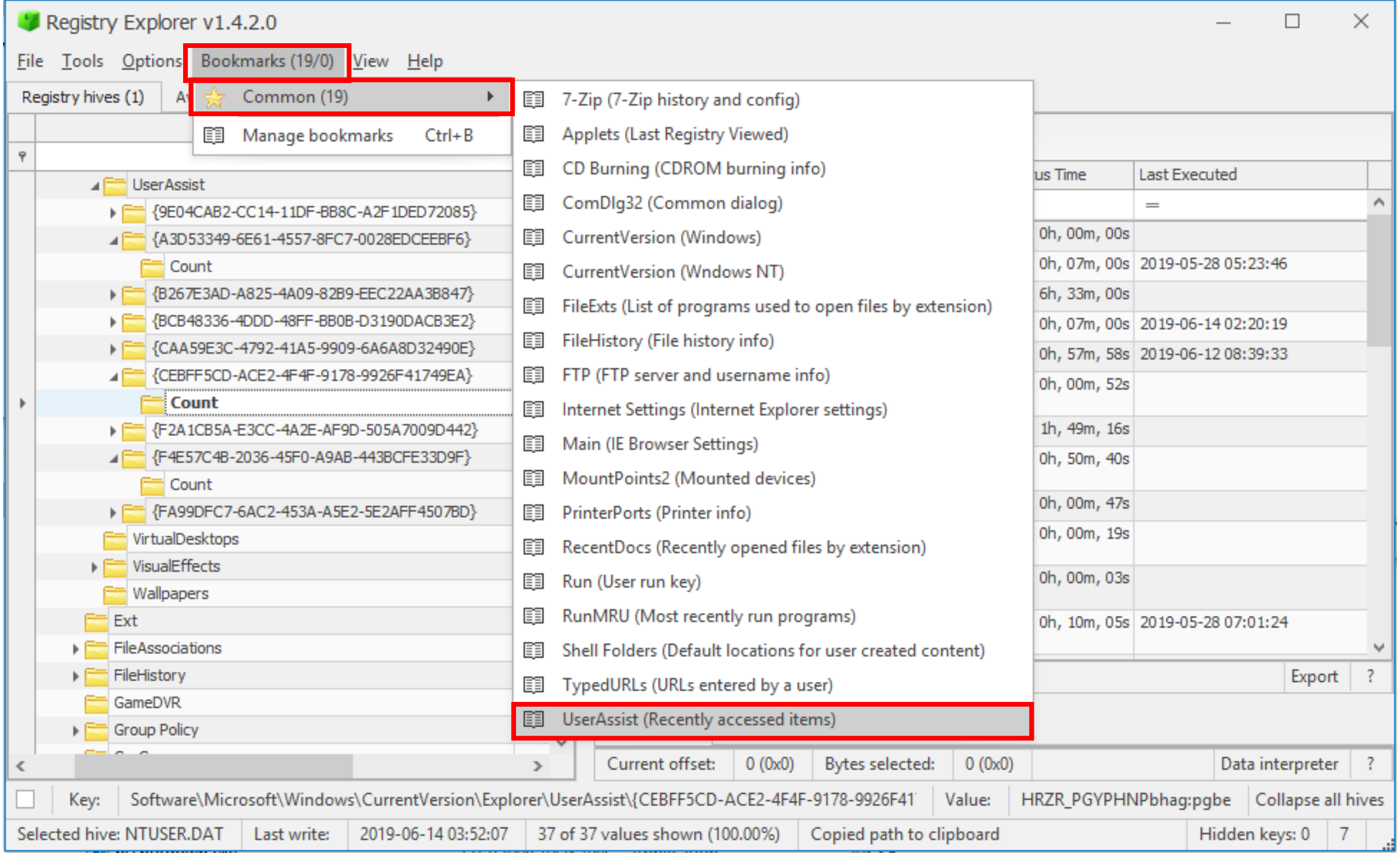
# Parsing UserAssist

- The goal is to look into UserAssist to see if any interesting data is remaining.

# Opening Per-User Hive

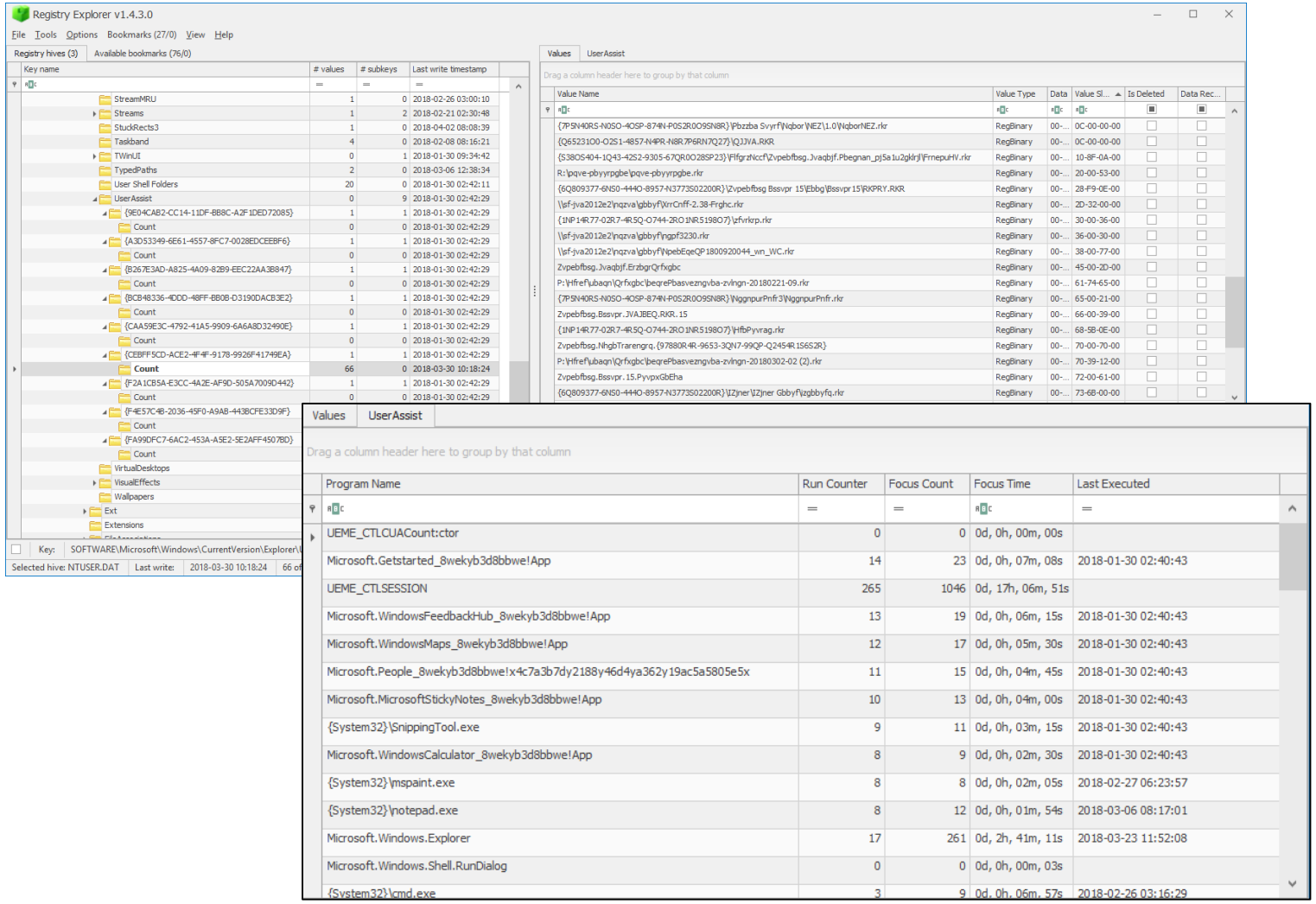


- Since UserAssist resides in per-user registry hive, we will use Registry Explorer to see the data.
- When Registry Explorer is opened, select "**Load offline hive**" under "File" menu.
- Open **G:\Users\honda\NTUSER.DAT** hive.
  - If the disk image is not mounted as "G", modify the drive letter.
  - When you open "honda" folder, you will be asked to obtain permission.
- There is a bookmark for UserAssist in Registry Explorer.


Registry Explorer v1.4.2.0
File
Tools
Options
Bookmarks (19/0)
View
Help
Registry hives (1)
Common (19)
Manage bookmarks Ctrl+B
7-Zip (7-Zip history and config)
Applets (Last Registry Viewed)
CD Burning (CDROM burning info)
ComDlg32 (Common dialog)
CurrentVersion (Windows)
CurrentVersion (Wndows NT)
FileExts (List of programs used to open files by extension)
FileHistory (File history info)
FTP (FTP server and username info)
Internet Settings (Internet Explorer settings)
Main (IE Browser Settings)
MountPoints2 (Mounted devices)
PrinterPorts (Printer info)
RecentDocs (Recently opened files by extension)
Run (User run key)
RunMRU (Most recently run programs)
Shell Folders (Default locations for user created content)
TypedURLs (URLs entered by a user)
UserAssist (Recently accessed items)
UserAssist
{9E04CAB2-CC14-11DF-BB8C-A2F1DED72085}
{A3D53349-6E61-4557-8FC7-0028EDCEEBF6}
Count
{B267E3AD-A825-4A09-82B9-EEC22AA3B847}
{BCB48336-4DDD-48FF-BB0B-D3190DACB3E2}
{CAA59E3C-4792-41A5-9909-6A6A8D32490E}
{CEBFF5CD-ACE2-4F4F-9178-9926F41749EA}
Count
{F2A1CB5A-E3CC-4A2E-AF9D-505A7009D442}
{F4E57C4B-2036-45F0-A9AB-443BCFE33D9F}
Count
{FA99DFC7-6AC2-453A-A5E2-5E2AFF4507BD}
VirtualDesktops
VisualEffects
Wallpapers
Ext
FileAssociations
FileHistory
GameDVR
Group Policy
us Time
Last Executed
0h, 00m, 00s
0h, 07m, 00s
2019-05-28 05:23:46
6h, 33m, 00s
0h, 07m, 00s
2019-06-14 02:20:19
0h, 57m, 58s
2019-06-12 08:39:33
0h, 00m, 52s
1h, 49m, 16s
0h, 50m, 40s
0h, 00m, 47s
0h, 00m, 19s
0h, 00m, 03s
0h, 10m, 05s
2019-05-28 07:01:24
Export
?
Current offset:
0 (0x0)
Bytes selected:
0 (0x0)
Data interpreter
?
Key:
Software\Microsoft\Windows\CurrentVersion\Explorer\UserAssist\{CEBFF5CD-ACE2-4F4F-9178-9926F41
Value:
HRZR_PGYPHNPbhag:pgbe
Collapse all hives
Selected hive: NTUSER.DAT
Last write:
2019-06-14 03:52:07
37 of 37 values shown (100.00%)
Copied path to clipboard
Hidden keys: 0
7

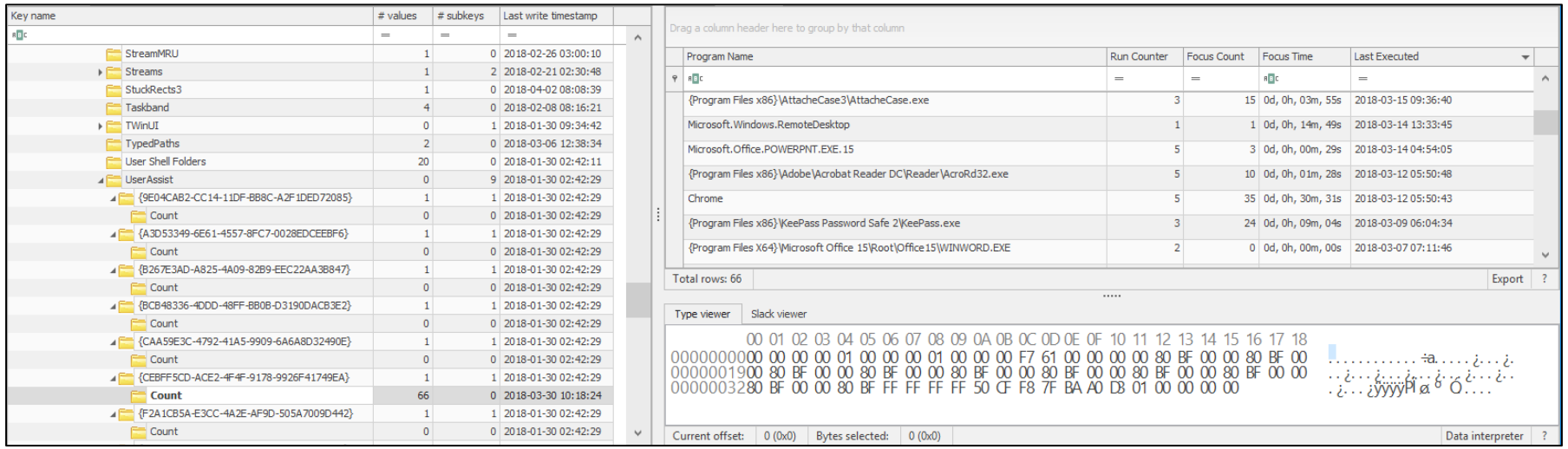
# Observing UserAssist

- When you look into the UserAssist, you can see that the “Value Name” and the “Program Name” columns are in ROT13 relations.
- You can also see the parameters such as Run Counter, Focus Count, Focus Time and the Last Executed date/time.



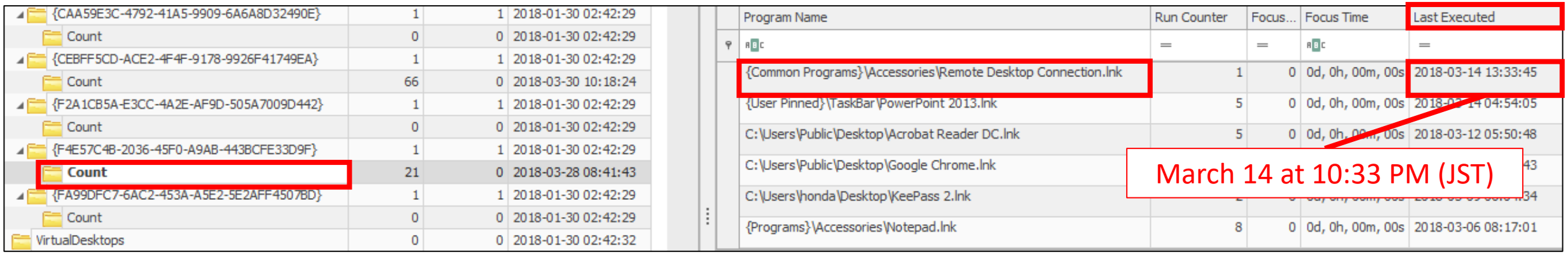
# Observing EXE UserAssist for User honda

- UserAssist mainly consists of keys that record EXE files and LNK files.
  - {CEBFF5CD-ACE2-4F4F-9178-9926F41749EA} : for EXE
  - {F4E57C4B-2036-45F0-A9AB-443BCFE33D9F} : for LNK (shortcuts)
- The figure below shows EXE records in NTUSER.DAT, for user honda on client-win10-2.
- If you sort it with “Last Executed” column, it can be viewed in chronological order.
- Since UserAsisst is recorded when the programs were executed from the Windows Explorer, CLI operations over RAT will not be recorded in it.



# Observing LNK UserAssist for User honda

- The image below is UserAssist of LNK on client-win10-2.
- On March 14 at 10:50 PM (JST:UTC+9:00), when infection of client-win10-1 occurred, execution of Remote Desktop Connection is recorded.
  - We should figure out if the user happened to run the Remote Desktop client at this time, or it was opened by the attackers.



| | | | |
|---|---|---|---|
| {CAA59E3C-4792-41A5-9909-6A6A8D32490E} | 1 | 1 | 2018-01-30 02:42:29 |
| Count | 0 | 0 | 2018-01-30 02:42:29 |
| {CEBFF5CD-ACE2-4F4F-9178-9926F41749EA} | 1 | 1 | 2018-01-30 02:42:29 |
| Count | 66 | 0 | 2018-03-30 10:18:24 |
| {F2A1CB5A-E3CC-4A2E-AF9D-505A7009D442} | 1 | 1 | 2018-01-30 02:42:29 |
| Count | 0 | 0 | 2018-01-30 02:42:29 |
| {F4E57C4B-2036-45F0-A9AB-443BCFE33D9F} | 1 | 1 | 2018-01-30 02:42:29 |
| **Count** | 21 | 0 | 2018-03-28 08:41:43 |
| {FA99DFC7-6AC2-453A-A5E2-5E2AFF4507BD} | 1 | 1 | 2018-01-30 02:42:29 |
| Count | 0 | 0 | 2018-01-30 02:42:29 |
| VirtualDesktops | 0 | 0 | 2018-01-30 02:42:32 |

| Program Name | Run Counter | Focus... | Focus Time | Last Executed |
|---|---|---|---|---|
| {Common Programs}\Accessories\Remote Desktop Connection.lnk | 1 | 0 | 0d, 0h, 00m, 00s | 2018-03-14 13:33:45 |
| {User Pinned}\TaskBar\PowerPoint 2013.lnk | 5 | 0 | 0d, 0h, 00m, 00s | 2018-03-14 04:54:05 |
| C:\Users\Public\Desktop\Acrobat Reader DC.lnk | 5 | 0 | 0d, 0h, 00m, 00s | 2018-03-12 05:50:48 |
| C:\Users\Public\Desktop\Google Chrome.lnk | | | | 43 |
| C:\Users\honda\Desktop\KeePass 2.lnk | | | | 34 |
| {Programs}\Accessories\Notepad.lnk | 8 | 0 | 0d, 0h, 00m, 00s | 2018-03-06 08:17:01 |

March 14 at 10:33 PM (JST)

# Amcache.hve

# Amcache.hve

- Amcache.hve is another application compatibility feature that was introduced in Windows 8.
  - In Windows Vista and 7, another feature “RecentFile.bcf” was used.
  - Amcache.hve was later added to Windows 7.
- It is recorded in **%WinDir%\AppCompat\Programs\Amcache.hve** file.
  - Requires Administrative privilege to open the path.
  - The file cannot be copied while Windows is running.
  - %WinDir% is typically mapped to C:\Windows.
- The file is in Registry (hive) format.

# Contents of the Amcache.hve

- What you can find from Amcache.hve include:
  - The list of program EXE files, with their path, size, sha-1 hash and other application properties.
  - The list of device driver files, with their properties.
- You can regard each key of the last written time as the last execution times.

# Value Names

- In Amcache.hve, value names are numbers; without knowing their meanings, it is difficult to determine which item is important and which one is not.

| Number | Contents |
|---|---|
| 0 | Product Name |
| 1 | Company Name |
| 2 | File Version |
| 3 | Language Code |
| 4 | SwitchBackContext |
| 5 | File Version |
| 6 | File Size |
| 7 | PE Header Field: SizeOfImage |
| 8 | Hash of PE Header |
| 9 | PE Header Field: Checksum |

| Number | Contents |
|---|---|
| c | File Description |
| F | Build Timestamp |
| 11 | Last Modified Timestamp |
| 12 | Created Timestamp |
| 15 | Full path to the File |
| 17 | Last Modified Timestamp 2 |
| 100 | Program ID |
| 101 | SHA1 hash of the File |
| *a,b,d,10,16* | *Unknown* |

# Tools for Amcache.hve

- Amcache Parser by Eric Zimmerman
  - https://github.com/EricZimmerman/AmcacheParser

```
Amcacheparser -f PATH_TO_AMCACHE --csv PATH_TO_OUTPUT_FOLDER
```

- Python-registry (amcache.py)
  - https://github.com/williballenthin/python-registry

```
py amcache.py PATH_TO_AMCACHE > PATH_TO_OUTPUT_FILE
```

# Note for Amcache.hve

- Note that different Amcache versions have different features. For example:
  - Amcache on Windows 10 records related to drivers. However, it no longer records non-GUI applications while former version record them. Therefore, we might not be able to get attackers' activities with it on Windows 10.
  - You can see the detail analysis of Amcache at the following URL.
    - http://www.ssi.gouv.fr/uploads/2019/01/anssi-coriin_2019-analysis_amcache.pdf
  - David Cowen said:

> *Amcache added the executable we left on the desktop but did not execute on Friday at 5am UTC saturday, the process event log showed it was a background task manager.*

https://www.hecfblog.com/2018/11/daily-blog-550-forensic-lunch-test.html
https://www.hecfblog.com/2018/11/daily-blog-551-forensic-lunch-test.html
https://www.hecfblog.com/2018/11/daily-blog-552-forensic-lunch-test.html

# SuperFetch

# SuperFetch

- SuperFetch is an addition to Prefetch for performance improvements.
  - First introduced in Windows Vista.
- Prefetch primarily focused on boot and application startup. The SuperFetch focuses on historical information and memory management.
- SuperFetch data is stored in the following database files under “C:\Windows\Prefetch”: AgAppLaunch.db, AgGlFaultHistory.db, AgGlFgAppHistory.db, AgGlGlobalHistory.db, AgGlUAD_P_<SID>.db, AgGlUAD_<SID>.db and AgRobust.db.
  - Need Administrative privileges to see them.

# Tools for Analyzing SuperFetch

- CrowdResponse by CrowdStrike
  - https://www.crowdstrike.com/resources/community-tools/crowdresponse/
- rewolf-superfetch-dumper
  - https://github.com/rwfpl/rewolf-superfetch-dumper
- libagdb by libyal
  - https://github.com/libyal/libagdb

# Analyzing Superfetch by CrowdResponse

- For example, you can use it with these two commands to get outputs on 64 bit environment.

```
CrowdResponse64.exe @SuperFetch -6s "F:\Windows\Prefetch" > superfetch_cr.xml
CRconvert.exe -s -t -f superfetch_cr.xml -o output_dir
```

| | A | B | C | D |
|---|---|---|---|---|
| 1 | process | launchcount | fgcount | path |
| 2 | CONHOST.EXE | 4 | 209 | ? CONHOST.EXE |
| 3 | GPUPDATE.EX | 22 | 241 | ? GPUPDATE.EXE |
| 4 | MPCMDRUN.EX | 15 | 152 | ? MPCMDRUN.EXE |
| 5 | ? | 0 | 231 | ? CDIR-COLLECTOR.EXE |
| 6 | CONSENT.EXE | 18 | 163 | ? CONSENT.EXE |
| 7 | DSMUSERTASK | 2 | 63 | ? DSMUSERTASK.EXE |
| 8 | FIREFOX.EXE | 0 | 151 | ? FIREFOX.EXE |
| 9 | OUTLOOK.EXE | 0 | 219 | ? OUTLOOK.EXE |

| | A | B | C |
|---|---|---|---|
| 1 | timestamp | period | path |
| 2 | 2018-01-26T16:56:36Z | Weekday 12 to 6AM | ? MPCMDRUN.EXE |
| 3 | 2018-01-27T14:07:24Z | Weekday 12 to 6AM | ? MPCMDRUN.EXE |
| 4 | 2018-01-27T14:08:48Z | Weekday 12 to 6AM | ? SIHOST.EXE |
| 5 | 2018-01-27T14:08:48Z | Weekday 12 to 6AM | ? EXPLORER.EXE |
| 6 | 2018-01-27T14:08:48Z | Weekday 12 to 6AM | ? |
| 7 | 2018-01-27T14:08:48Z | Weekday 12 to 6AM | ? |
| 8 | 2018-01-27T14:08:49Z | Weekday 12 to 6AM | ? OUTLOOK.EXE |
| 9 | 2018-01-27T14:08:54Z | Weekday 12 to 6AM | ? |

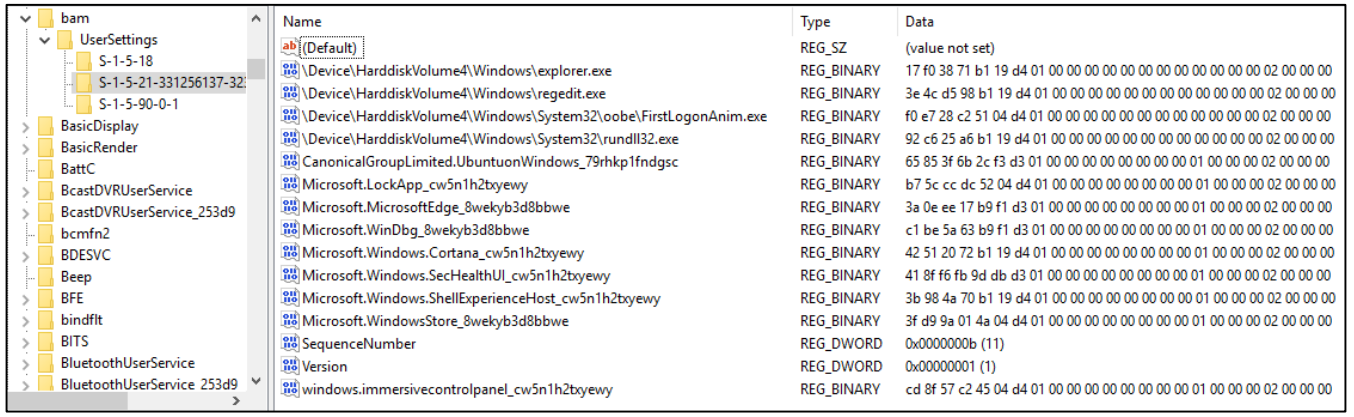
# BAM

# BAM

- Stands for "**Background Activity Moderator**"
  - We are **not** talking about "Business Activity Monitoring" in this class.
- BAM resides in registry under "**HKEY_LOCAL_MACHINE\SYSTEM\ControlSet001\Services\bam\UserSettings**" key.
- This is relatively new feature that was introduced in Windows 10 version 1709, and we have confirmed to exist at least until on 1903.
  - The service does not exist in Windows 8.1 and older, and earlier versions of Windows 10.

# BAM Contents

- BAM also has list of applications, last executed timestamps, and some other data.
- Since BAM is relatively new, some history cleaning tools may not be aware of the BAM.
  - Could be one of the locations where the histories may remain even when other artifacts were cleared.



# IconCache.db

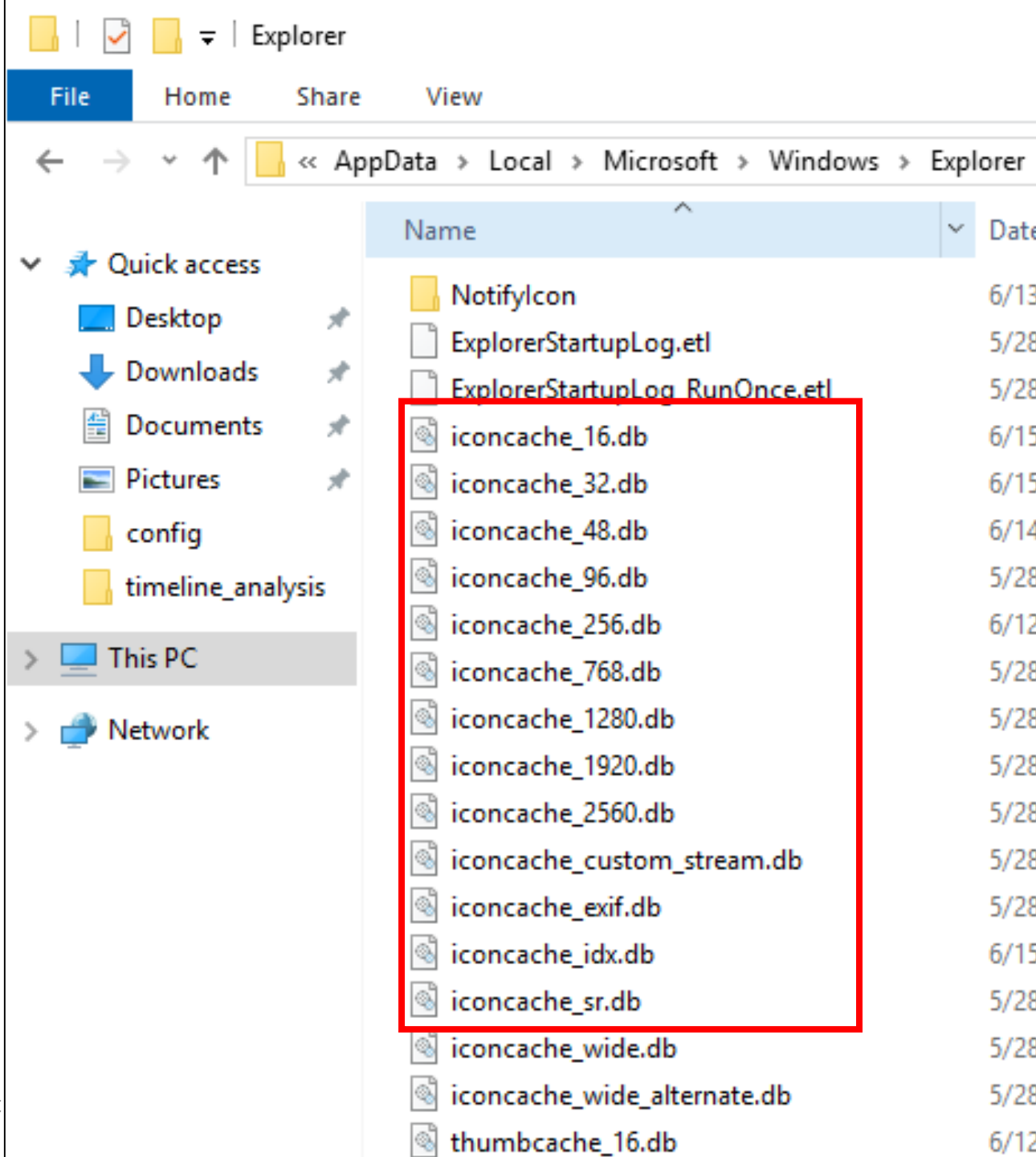
# IconCache.db (1)

- Strictly speaking, This artifact isn’t for confirming program execution. However, you might be able to find unknown executable names and their icons which a user displayed executables in the Explorer process by analyzing IconCache.db.

- You can find it in the following path.
  - Windwos 7 or later
    - C:\Users\<user_name>\AppData\Local
      - You can get file paths of executables by analyzing it.
  - Windows 8 and 10
    - C:\Users\<user_name>\AppData\Local\Microsoft\Windows\Explorer
      - You can get icons of executables by analyzing them.

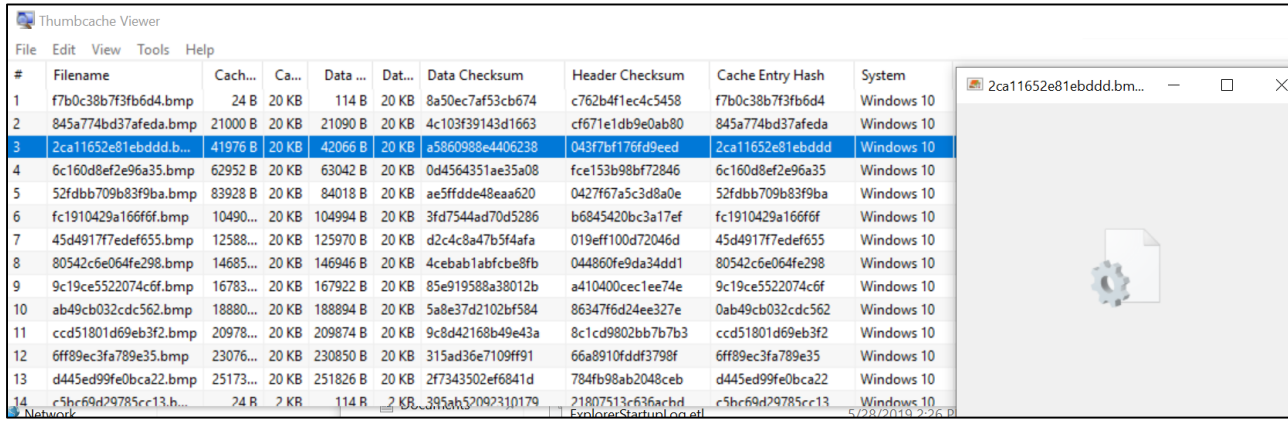
# IconCache.db (2)

- There is a variety of IconCache.db files in Windows 8 and later.



# Tools for IconCache.db

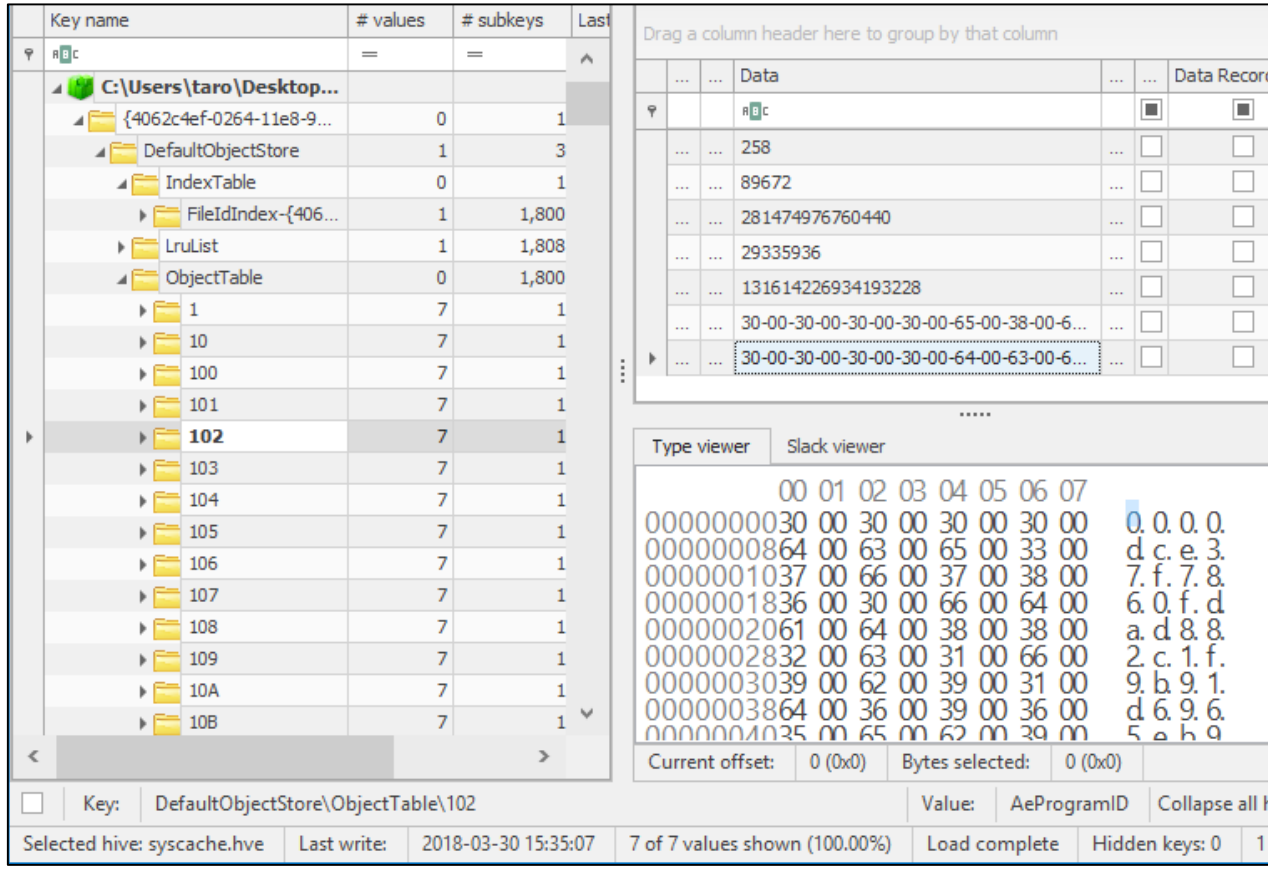
- You will need different tools for different paths because the file formats are different.
  - C:\Users\<user_name>\AppData\Local\IconCache.db
    - strings
      - We are not sure the file format. However, the paths strings are stored as UTF-16-LE. You can use strings command.
  - C:\Users\<user_name>\AppData\Local\Microsoft\Windows\Explorer\IconCache*.db
    - libwtcdb by libyal (<= Win 8.1)
      - https://github.com/libyal/libwtcdb
    - Thumbcache Viewer
      - https://thumbcacheviewer.github.io/
      - The file format is the same as thumbnailcache.



# Syscache

# SysCache

- SysCache is a program execution artifact only on Windows 7 and Windows Server 2008 R2.
  - https://dfir.ru/2018/12/02/the-cit-database-and-the-syscache-hive/
  - https://dfir.ru/2019/01/04/what-writes-to-the-syscache-hive/
  - https://www.hecfblog.com/2018/12/daily-blog-563-forensic-lunch-test.html
- Tools
  - Syscache is a HIVE. You can parse it with registry parsers such as Registry Explorer and yarp. However, you will need to parse data on your own.
- File Path:
  - C:\System Volume Information\Syscache.hve



# CITDatabase

# CITDatabase

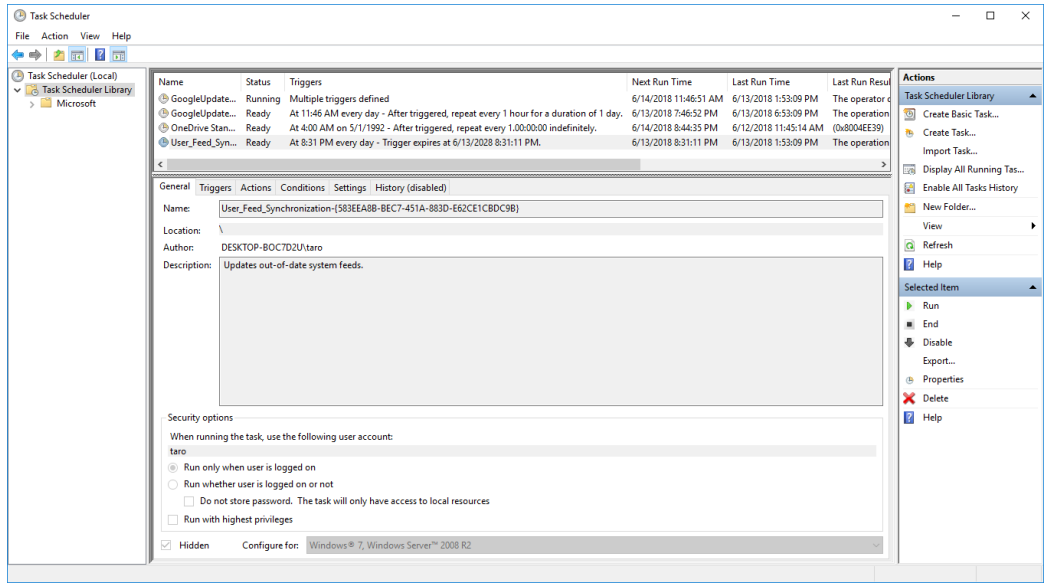
- CITDatabase is a program execution artifact on Windows 7 and Windows 8.* only.
  - https://twitter.com/errno_fail/status/1130150447157796864
  - https://twitter.com/errno_fail/status/1068654591389118470
- You can get paths of executable files.

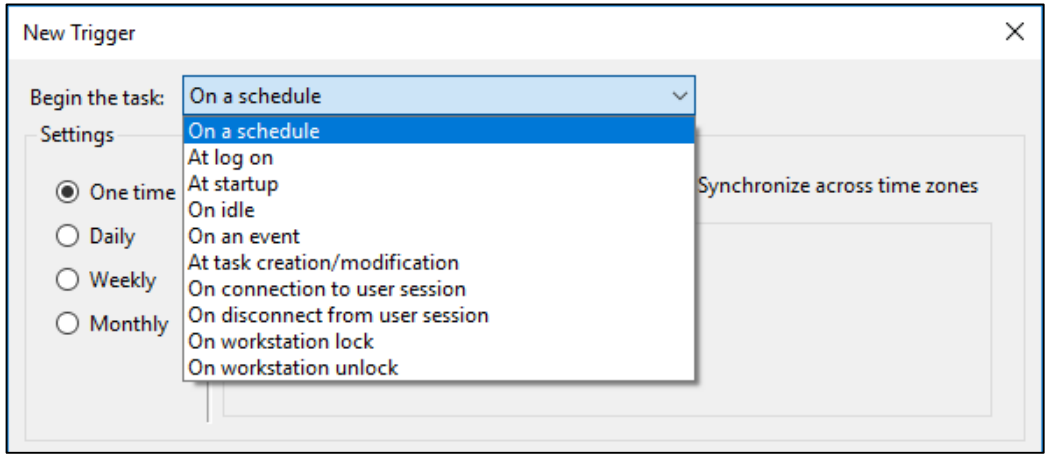
- CITDatabase path
  - HKEY_LOCAL_MACHINE\Software\Microsoft\Windows NT\CurrentVersion\AppCompatFlags\CIT\System
- Tool
  - https://gist.github.com/msuhanov/356b724f9a4403059667142 7adb6cfc6

# Task Scheduler

# Task Scheduler

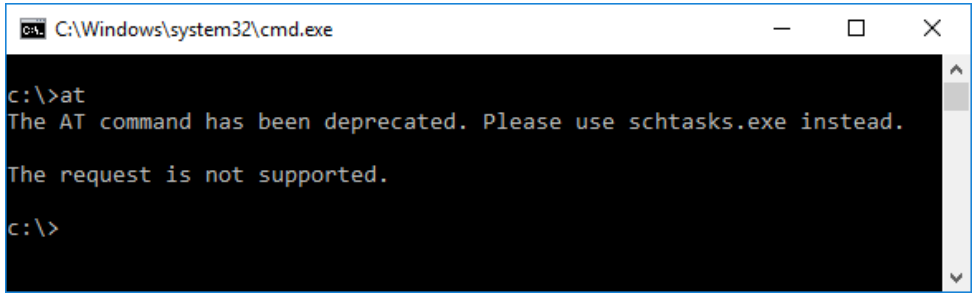
- Task Scheduler is used to execute a program/command at a specified time.
  - Specified “time” does not have to be a chronological time.
    - There are options to execute the task “at log on”, “at startup”, and so on.
    - Some malware use the “at log on” to make them executed when the Windows starts up.

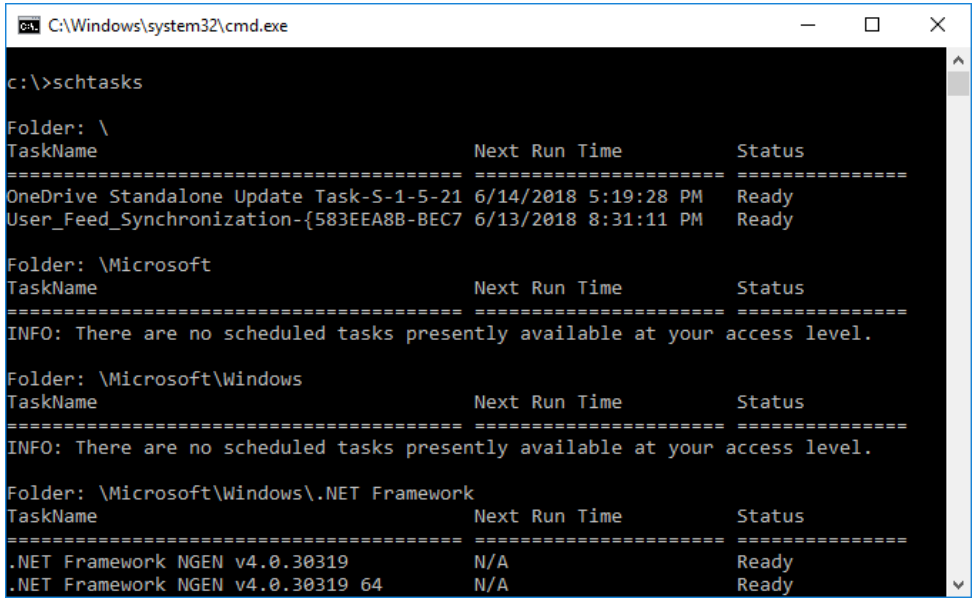


# Task Scheduler Commands

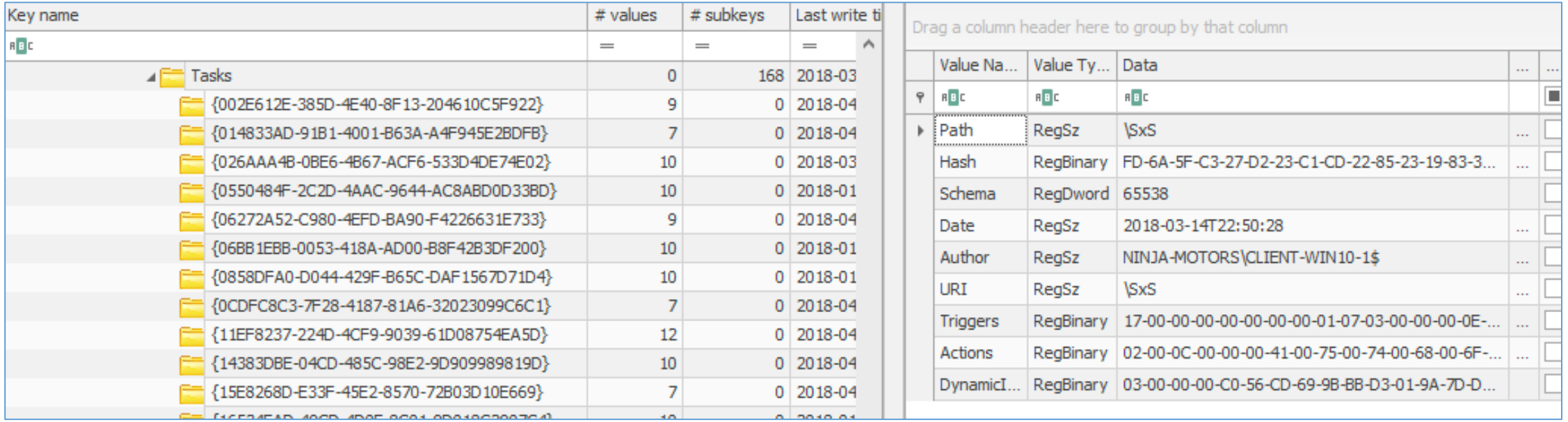
- at.exe
  - Used on Windows 7 and older.
  - Command still exists on Windows 8 and later, but it will just show a message saying that the at command is deprecated.
- schtasks.exe
  - Currently used command for managing task scheduler.
  - Can be used to show and modify the scheduled tasks.



# Task Scheduler Artifacts

- When tasks are created, one of the following data is created to store the tasks.
  - Windows Task Scheduler or “schtasks” command: **HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft \Windows NT\CurrentVersion\Schedule\TaskCache\Tasks**



| Key name | # values | # subkeys | Last write ti |
|---|---|---|---|
| Tasks | 0 | 168 | 2018-03 |
| {002E612E-385D-4E40-8F13-204610C5F922} | 9 | 0 | 2018-04 |
| {014833AD-91B1-4001-B63A-A4F945E2BDFB} | 7 | 0 | 2018-04 |
| {026AAA4B-0BE6-4B67-ACF6-533D4DE74E02} | 10 | 0 | 2018-03 |
| {0550484F-2C2D-4AAC-9644-AC8ABD0D33BD} | 10 | 0 | 2018-01 |
| {06272A52-C980-4EFD-BA90-F4226631E733} | 9 | 0 | 2018-04 |
| {06BB1EBB-0053-418A-AD00-B8F42B3DF200} | 10 | 0 | 2018-01 |
| {0858DFA0-D044-429F-B65C-DAF1567D71D4} | 10 | 0 | 2018-01 |
| {0CDFC8C3-7F28-4187-81A6-32023099C6C1} | 7 | 0 | 2018-04 |
| {11EF8237-224D-4CF9-9039-61D08754EA5D} | 12 | 0 | 2018-04 |
| {14383DBE-04CD-485C-98E2-9D909989819D} | 10 | 0 | 2018-04 |
| {15E8268D-E33F-45E2-8570-72B03D10E669} | 7 | 0 | 2018-04 |

Drag a column header here to group by that column

| Value Na... | Value Ty... | Data |
|---|---|---|
| Path | RegSz | \SxS |
| Hash | RegBinary | FD-6A-5F-C3-27-D2-23-C1-CD-22-85-23-19-83-3... |
| Schema | RegDword | 65538 |
| Date | RegSz | 2018-03-14T22:50:28 |
| Author | RegSz | NINJA-MOTORS\CLIENT-WIN10-1$ |
| URI | RegSz | \SxS |
| Triggers | RegBinary | 17-00-00-00-00-00-00-00-01-07-03-00-00-00-0E-... |
| Actions | RegBinary | 02-00-0C-00-00-00-41-00-75-00-74-00-68-00-6F-... |
| DynamicI... | RegBinary | 03-00-00-00-C0-56-CD-69-9B-BB-D3-01-9A-7D-D... |

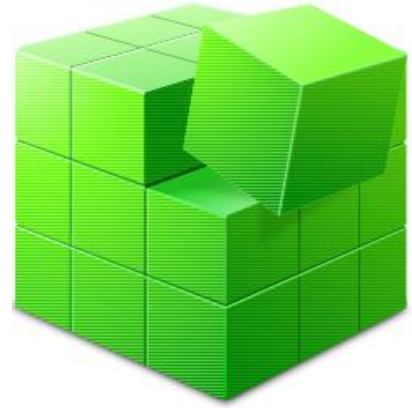
# Practice Exercise: Looking into Task Schedules

# Looking into Task Schedules

- Objectives of the exercise is to look into registry entry of task schedules.

# Looking into Registry

- In this exercise, we will use Registry Explorer by Eric Zimmerman.
  - Open RegistryExplorer.exe in the tools folder.
- We will continue using **Client-Win10-2_honda.E01** image.
  - Please mount it if you have unmounted it.
- When Registry Explorer is opened, select "**Load offline hive**" under "File" menu.
- Open **G:\Windows\System32\config\SOFTWARE** hive.
  - If the disk image is not mounted as "G", modify the drive letter.
  - When you open "config" folder, you will be asked to obtain permission.

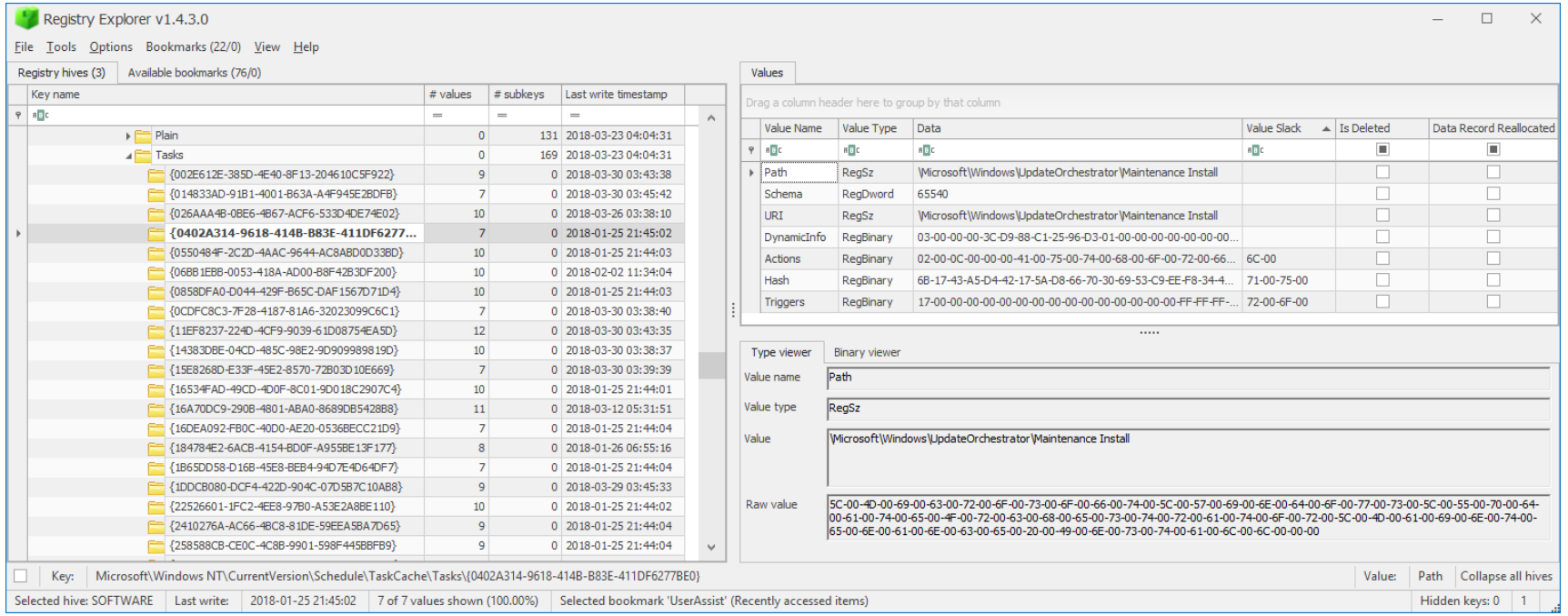


# Navigating to Tasks Registry Key

- Tasks registry key is not in the bookmark.
- To see the Tasks registry key, navigate:
  **Microsoft\Windows NT\CurrentVersion\Schedule\TaskCache\Tasks**
  - There are “Windows” and “Windows NT” subkeys under Microsoft. Be sure to open the right one.

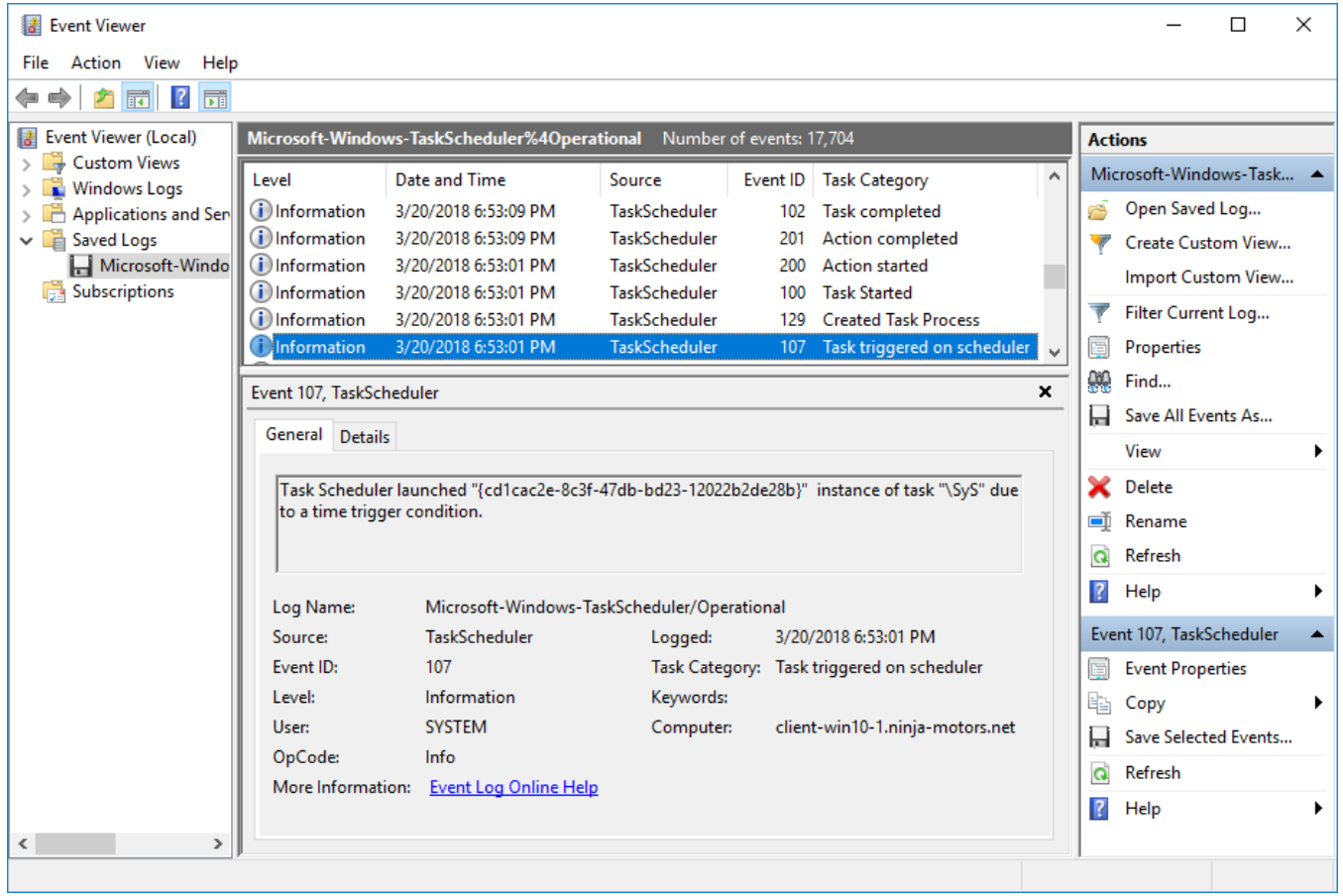
# Inspecting Tasks Registry Values

- You should be able to see the registered tasks.
  - Some values are not human-readable, but you should be able to see the task names and paths.



# Tasks Execution Histories

- When a task was executed:
  - Record remains in the Event Log.
  - Registry value of “DynamicInfo” will be updated.



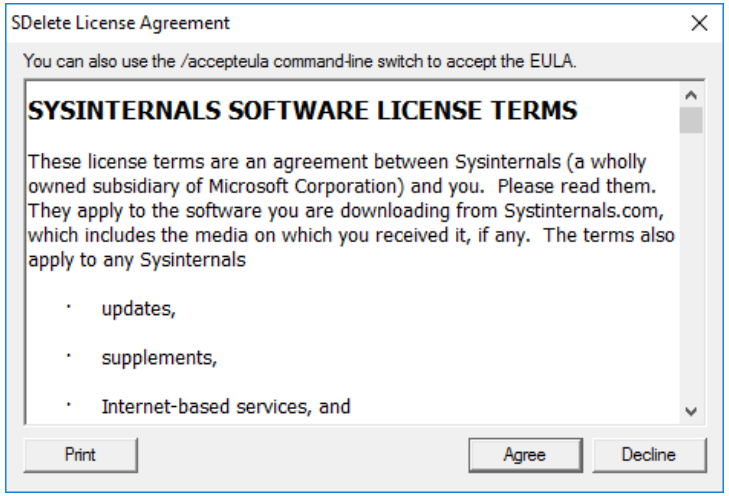
# Sysinternals Tools

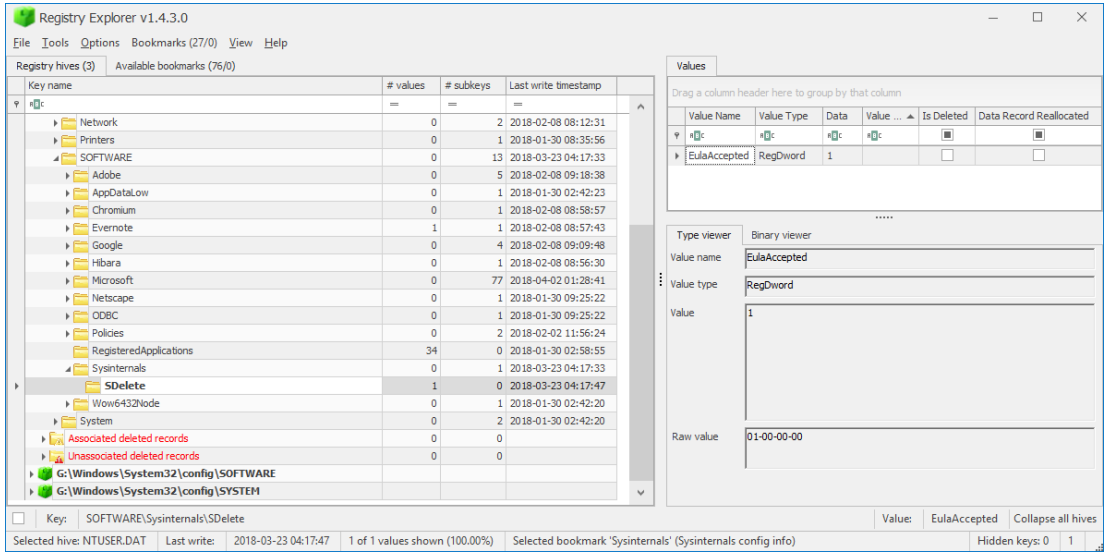
# Sysinternals Tools

- Sysinternals Tools are legitimate tools hosted on Microsoft website.
  - https://docs.microsoft.com/en-us/sysinternals/
- Example tools include:
  - **Autoruns**: shows startup applications.
  - **Process Explorer**: shows information about running processes.
  - **Process Monitor**: monitors process activities.
  - **PsExec**: execute specified commands on a remote host.
  - **SDelete**: securely erase file by overwriting its contents for multiple times.
  - **Sysmon**: provides additional monitors to the Windows system activities.
- They are legitimate, but sometimes used by attackers in their campaigns.

# Sysinternals End-User License Agreement (EULA)

- When one of the Sysinternals tools is executed, if the user has never agreed to the license agreement, the EULA dialog will be displayed.
- Agreeing to the agreement will create a value in the per-user registry as "**SOFTWARE\Sysinternals\<tool name>\EulaAccepted**"
  - If this value exists, this means that the Sysinternals tool was executed by the user.

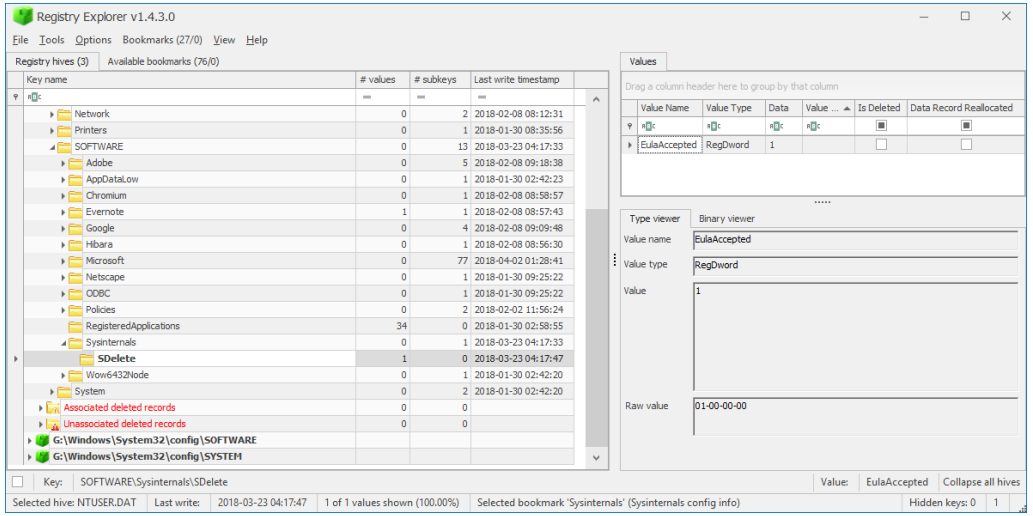


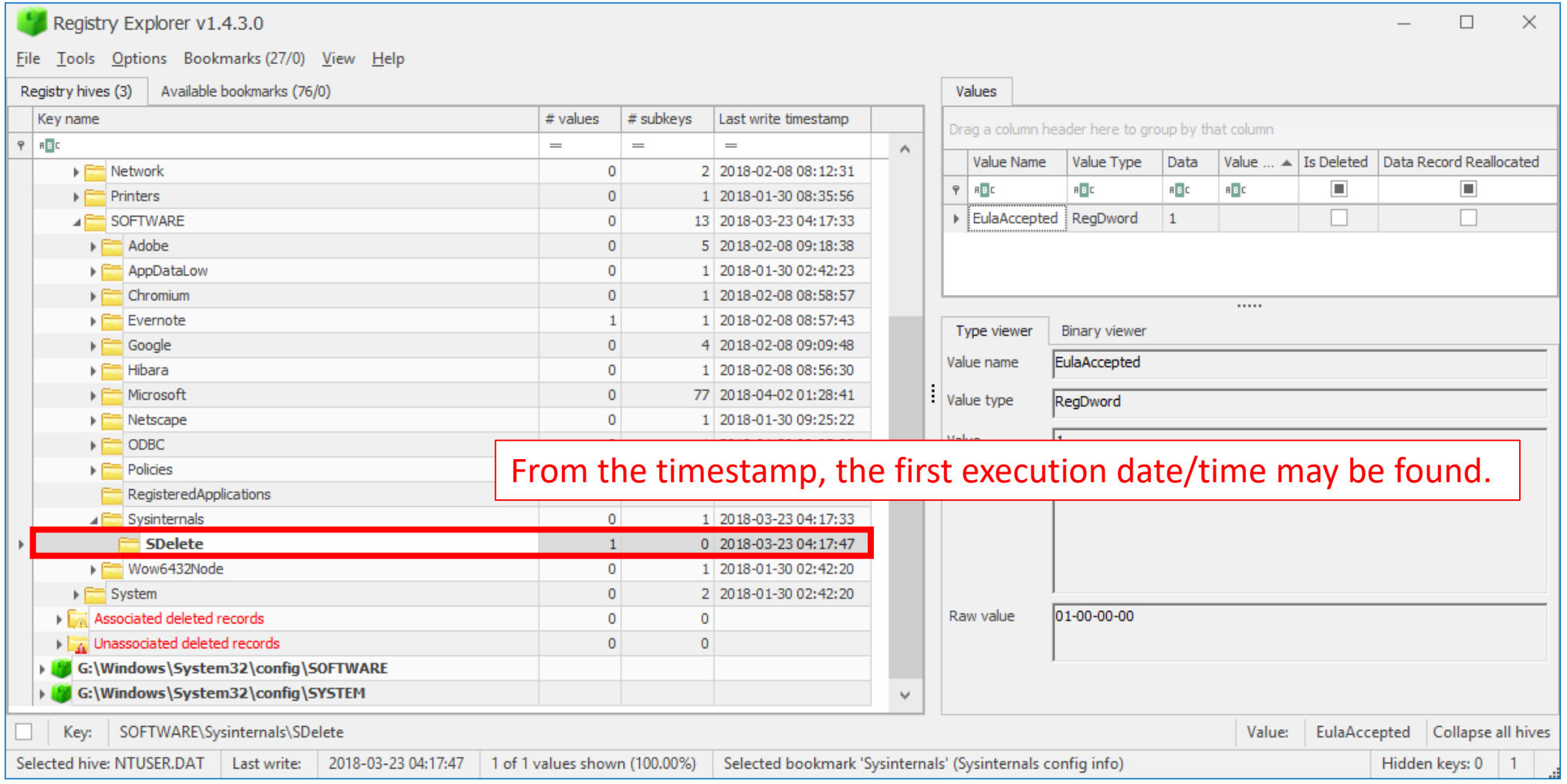
# Scenario 1 Labs:
# Lab 7: Looking into Sysinternals Registry Keys for Client-Win10-2

# Lab 7: Locating Sysinternals EULA (1/3)

- Using Registry Explorer, open per-user hive from G:\Users\honda\NTUSER.DAT
- Navigate to the Sysinternals registry key.
  - SOFTWARE\Sysinternals\<app name>
- Were Sysinternals suite executed on the macine? If so, which tool?
- When was the first time the tool was executed?

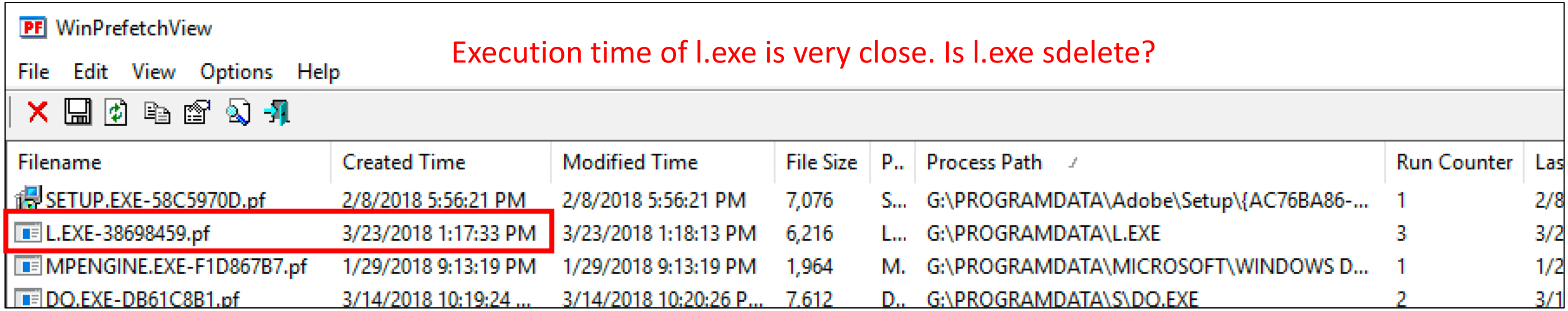


# Lab 7: Locating Sysinternals EULA (2/3)



# Lab 7: Locating Sysinternals EULA (3/3)

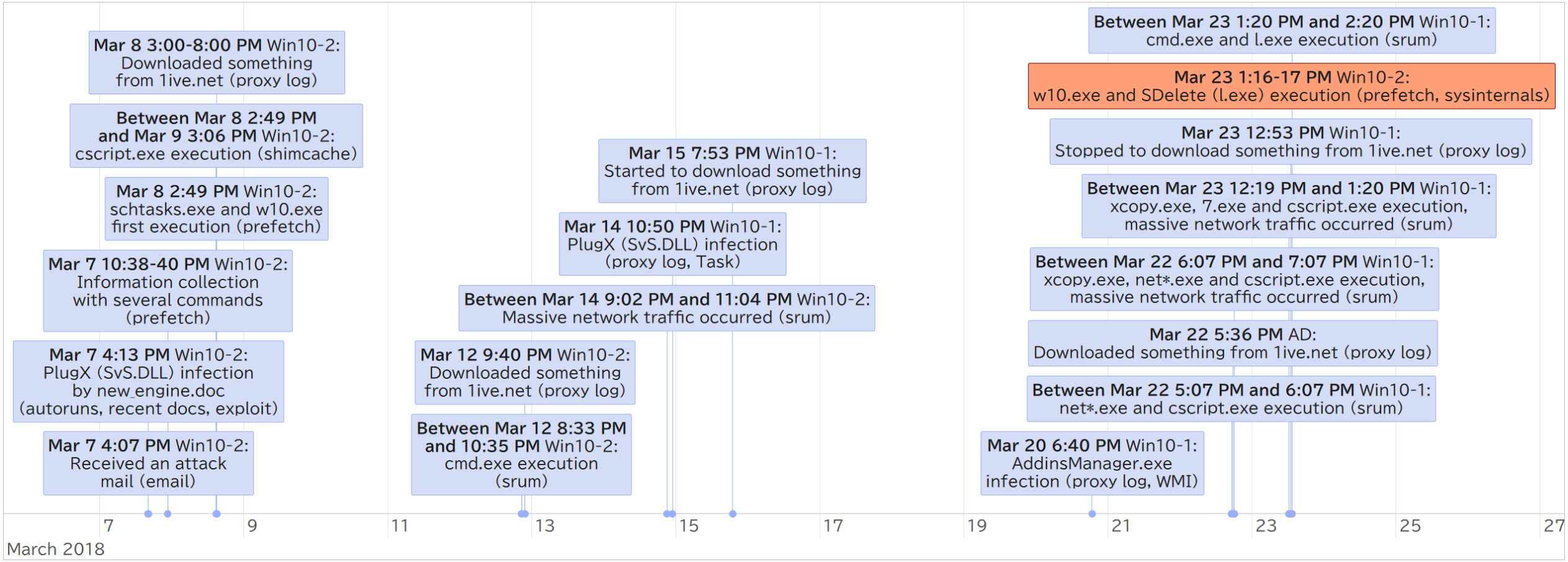
- From the registry keys, the following facts are observed:
  - SDelete was executed on March 23 at 1:17:47 PM (JST).
    - Can we find sdelete from the HDD?
    - The registry keys will remain the same even when attackers modified its filenames.



# Sysinternals EULA and Timestamps

- SDelete is one of the Sysinternals tools that are used in attacks.
  - Overwrites the file with random data for multiple times to avoid files being carved with the forensics procedures.
- If the user has never used the Sysinternals tool and the registry value exists, it is likely that the tool was used during the attack.
  - The EULA is managed for each tool in the Sysinternals. Even if the user has used other Sysinternals tools, if the user has never used SDelete before, then the EulaAccepted value for the SDelete will not exist.
- Registry keys have timestamps. Timestamps of registry keys for the application might be a suggestion for program execution time.

# Scenario 1 Incident Timeline After Sysinternals Registry Analysis



# Summary

# Summary (1)

- When programs are executed on Windows, several “records” of execution remains on the computer.
- By examining the “records”, it may be possible to track the programs that were executed during the intrusion.
- The artifacts that remains depend on “how” the programs were executed.

# Summary (2)

- Prefetch, Shimcache and SRUM are powerful for finding attackers' activities and SRUM can be also used to find malware infection.
- UserAssist can be used to know user's activities.
- You can use many other kinds of program execution artifacts finding deleted executable files on the current file system.

# Notes About Timestamps

- When looking the program execution artifacts, knowing about “when” the artifacts get refreshed is important.
- Some artifacts, such as Prefetch and SRUM, is not updated frequently.
  - File system metadata of a prefetch file is created or modified at the earlier one of: the program being closed, or 10 seconds elapse since the program was executed.
  - SRUM is updated once an hour or at the system shutdown.

# The Scenario 1 Incident Timeline So Far