Characteristics and Detectability of Windows Auto-Start Extensibility Points in Memory Forensics

Daniel Uroz, Ricardo J. Rodríguez

duroz@unizar.es, rjrodriguez@unizar.es

Centro Universitario de la Defensa Zaragoza

April 25, 2019

DFRWS EU 2019
Oslo, Norway

Agenda

1. Introduction
2. Windows ASEPs taxonomy
3. Experimental evaluation
4. Related work
5. Conclusions

Agenda

1 Introduction

2 Windows ASEPs taxonomy

3 Experimental evaluation

4 Related work

5 Conclusions

Introduction

Incident response

- **Presence of *malware* in most of security incidents**
  1. Compromise
  2. **Persistence**
  3. Malicious activity
- Persistence on the system using ***Auto-Start Extensibility Points*** **(ASEPs**)
  - Subset of OS and application extensibility points that allow a program to auto-start without any explicit user invocation
- Sometimes, access to device drives are difficult or we need a quickly response incident
  - **Memory forensics focused on computer *memory dumps***

Introduction

Investigation

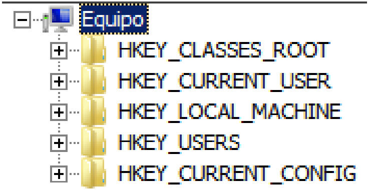
- Memory forensics → **The Volatility Framework**
  - Collection of tools to extract **digital artifacts** from memory dumps
    - TCP connections
    - Drivers
    - Processes
    - **Registry hives**
    - . . .
- **Most of ASEPs relies on *the Windows Registry***

Introduction

The Windows Registry

- **Hierarchical database** that contains critical data for the normal operation of Windows and other applications
  - Database is divided into files called **hives**
- Then, there is an **in-memory representation** of the on-disk hives
- Root keys of Windows Registry:

- HKCR: associations of file extensions and COM class registration information
- HKCU: information of the current signed-in user
- HKLM: configuration of OS and software installed
- HKU: subkeys which correspond to each of the user profile
- HKCC: hardware profile currently being used
- HKPD: ***volatile*** information about performance counters



Introduction

Windows Registry in-memory

- Current system configuration is stored in `CurrentControlSet` registry
  - Link to `HKLM\SYSTEM\ControlSet00[1,2]`
- `SOFTWARE` disk hive memory mapped to `HKLM\SOFTWARE`
- Per-user configuration retrieved by `Ntuser.dat`

Agenda

1 Introduction

2 Windows ASEPs taxonomy

3 Experimental evaluation

4 Related work

5 Conclusions

ASEP categories

ASEP is a place which allows a particular program to run automatically in the system without any explicit user interaction

Four categories:

- System persistence mechanisms
- Program loader abuse
- Application abuse
- System behavior abuse

Characteristics

- Write permissions
  - User privileges, e.g: `HKCU`
  - Elevated privileges, e.g: `HKLM`
- Execution privilege
  - Elevated privileges
  - Inherent to signed-in user
- All tracked by disk forensics, but may not by memory forensics
  - Memory paging could be a problem

- *Freshness of the system*
  - System reboot
  - User sign out and sign in
- Execution scope
  - System-wide
  - Application-wide
- Configuration scope
  - System-level
  - User-level

System persistence mechanisms

Well-known mechanisms provided by Windows to run user programs, privileged tasks, or system services

System persistence mechanisms

Well-known mechanisms provided by Windows to run user programs, privileged tasks, or system services

1. **Run keys**: run programs every time that a new user signs in the system
2. **Startup folder**: every program or application shortcut contained in the folder is launched during system start-up
3. **Scheduled tasks**: execute periodically when certain conditions are met (known as *trigger conditions*)
4. **Services**: background programs that have no user interaction

System persistence mechanisms

Well-known mechanisms provided by Windows to run user programs, privileged tasks, or system services

1. **Run keys**: run programs every time that a new user signs in the system
2. **Startup folder**: every program or application shortcut contained in the folder is launched during system start-up
3. **Scheduled tasks**: execute periodically when certain conditions are met (known as *trigger conditions*)
4. **Services**: background programs that have no user interaction

	Characteristics					
Windows Auto-Start Extensibility Points	**Write permissions**	**Execution privileges**	**Tracked down in memory forensics**[†]	**Freshness of system**	**Execution scope**	**Configuration scope**
System persistence mechanisms						
Run keys (HKLM root key)	yes	user	yes	user session	application	system
Run keys (HKCU root key)	no	user	yes	user session	application	user
Startup folder (%ALLUSERSPROFILE%)	yes	user	no	user session	application	system
Startup folder (%APPDATA%)	no	user	no	user session	application	user
Scheduled tasks	yes	any	no	no needed[‡]	application	system
Services	yes	system	yes	no needed[‡]	application	system

[†] *If the memory is paging to disk, those ASEPs would be not possible to track it down in memory forensics.*

[‡] *It depends on the trigger conditions defined to launch the program.*

Program loader abuse

Techniques based on the abuse of the Windows program loader process

Program loader abuse

Techniques based on the abuse of the Windows program loader process

1. **Image File Execution Options**: launch programs under a debugger
2. **Extension hijacking**: program associated with file extensions
3. **Shortcuts manipulation**: manipulate already existing shortcuts
4. **COM hijacking**: software components that can interact with others
5. **Shim databases**: apply patches prior to program execution

Program loader abuse

Techniques based on the abuse of the Windows program loader process

1. **Image File Execution Options**: launch programs under a debugger
2. **Extension hijacking**: program associated with file extensions
3. **Shortcuts manipulation**: manipulate already existing shortcuts
4. **COM hijacking**: software components that can interact with others
5. **Shim databases**: apply patches prior to program execution

	Characteristics					
Windows Auto-Start Extensibility Points	**Write permissions**	**Execution privileges**	**Tracked down in memory forensics**[†]	**Freshness of system**	**Execution scope**	**Configuration scope**
Program loader abuse						
Image File Execution Options	yes	user	yes	no needed	application	system
Extension hijacking (HKLM root key)	yes	user	yes	no needed	application	system
Extension hijacking (HKCU root key)	no	user	yes	no needed	application	user
Shortcut manipulation	no	user	no	no needed	application	user
COM hijacking (HKLM root key)	yes	any	yes	no needed	system	system
COM hijacking (HKCU root key)	no	user	yes	no needed	system	user
Shim databases	yes	any	yes	no needed	application	system

[†] *If the memory is paging to disk, those ASEPs would be not possible to track it down in memory forensics.*

Application abuse

Extensions of legitimate programs that are abused to persist in the system

Application abuse

Extensions of legitimate programs that are abused to persist in the system

1. **Trojanized system binaries**: modifying a legitimate, system binary program that is patched to load another unintended program
2. **Office add-ins**: till 2013, they act like COM objects and are implemented as a DLL
3. **Browser helper objects (BHO)**: DLL files that work as plugins for the Internet Explorer browser (up to Internet Explorer 11)

Application abuse

Extensions of legitimate programs that are abused to persist in the system

1. **Trojanized system binaries**: modifying a legitimate, system binary program that is patched to load another unintended program
2. **Office add-ins**: till 2013, they act like COM objects and are implemented as a DLL
3. **Browser helper objects (BHO)**: DLL files that work as plugins for the Internet Explorer browser (up to Internet Explorer 11)

	Characteristics					
Windows Auto-Start Extensibility Points	**Write permissions**	**Execution privileges**	**Tracked down in memory forensics**[†]	**Freshness of system**	**Execution scope**	**Configuration scope**
Application abuse						
Trojanized system binaries	yes	any	no	no needed	system	system
Office add-ins	yes	user	yes	no needed	application	user
Browser helper objects	yes	user	yes	no needed	application	system

[†] *If the memory is paging to disk, those ASEPs would be not possible to track it down in memory forensics.*

System behavior abuse

Take advantage of the Windows features to launch programs

System behavior abuse

Take advantage of the Windows features to launch programs

1. **Winlogon**: launch certain programs every time that a user signs into the system
2. **DLL hijacking**: abusing the DLL search order done by Windows
3. **AppInit DLLs**: allows any DLL to be loaded into the address space of every application with a user interface (`user32.dll`)
4. **Active Setup**: enables programs to be launched when a user signs in the system

System behavior abuse

Take advantage of the Windows features to launch programs

1. **Winlogon**: launch certain programs every time that a user signs into the system
2. **DLL hijacking**: abusing the DLL search order done by Windows
3. **AppInit DLLs**: allows any DLL to be loaded into the address space of every application with a user interface (`user32.dll`)
4. **Active Setup**: enables programs to be launched when a user signs in the system

	Characteristics					
Windows Auto-Start Extensibility Points	**Write permissions**	**Execution privileges**	**Tracked down in memory forensics**†	**Freshness of system**	**Execution scope**	**Configuration scope**
System behavior abuse						
Winlogon	yes	user	yes	user session	application	system
DLL hijacking	yes	any	no	no needed	system	system
AppInit DLLs	yes	any	yes	no needed	system	system
Active setup (HKML root key)	yes	user	yes	user session	application	system
Active setup (HKCU root key)	no	user	yes	user session	application	application

† *If the memory is paging to disk, those ASEPs would be not possible to track it down in memory forensics.*

Agenda

1 Introduction

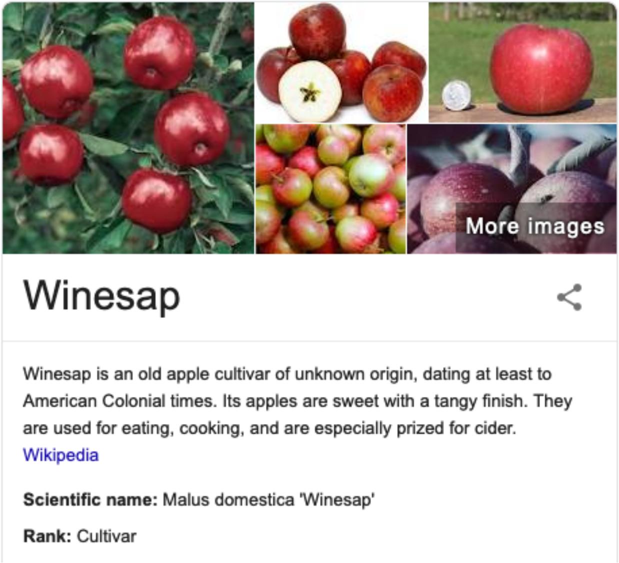
2 Windows ASEPs taxonomy

3 Experimental evaluation

4 Related work

5 Conclusions

Winesap



Winesap

Winesap is an old apple cultivar of unknown origin, dating at least to American Colonial times. Its apples are sweet with a tangy finish. They are used for eating, cooking, and are especially prized for cider.
Wikipedia

Scientific name: Malus domestica 'Winesap'

Rank: Cultivar

Winesap

- **Volatility plugin Winesap, available under GNU GPLv3**:

  `https://gitlab.unizar.es/rrodrigu/winesap`
- Marks different suspicious activity depending on Windows value of registries:
  - `REG_BINARY` or `REG_NONE`
    - PE header
  - `REG_SZ`, `REG_EXPAND_SZ`, or `REG_LINK`
    - Suspicious paths
    - Well-known shell commands that indirectly launch programs (e.g: `rundll32.exe shell32.dll,ShellExecute_RunDLL <filepath>`)

Experiments

- Custom Python scripts that install ASEPs previously described
- Empirically found the Windows sequence order of ASEP program launch:
  1. Winlogon (`Userinit`)
  2. Winlogon (`Shell`)
  3. Run keys (`HKLM/RunOnceEx`)
  4. Run keys (`HKCU/RunOnceEx`)
  5. Run keys (`HKLM/RunOnce`)
  6. Active Setup (`HKLM`)
  7. Active Setup (`HKCU`)
  8. Startup folder (`%ALLUSERSPROFILE%`)
  9. Startup folder (`%APPDATA%`)
  10. Run keys (`HKCU/Run`)
  11. Run keys (`HKLM/Run`)
  12. Run keys (`HKCU/RunOnce`)
- Use of `Winesap` to analyze infected computer memory dumps

Results

- **All Windows ASEPs that relies on Windows Registry can be retrieved by memory forensics regardless of configuration scope**
- Not fully detected unless file carving:
  - Startup folder
  - Shortcut manipulation
  - Scheduled tasks
- Need to use traditional memory forensics analysis focused on running processes
  - Trojanized system binaries
  - DLL hijacking
  - Office add-ins

Agenda

1 Introduction

2 Windows ASEPs taxonomy

3 Experimental evaluation

4 Related work

5 Conclusions

Related work

- **ASEP concept** introduced by Wang et al. (2004)
  1. Start new processes
  2. Hook system processes
  3. Load drivers
  4. Hook multiple processes
- **Study of spyweare abuse of BHO** by Kirda et al. (2006)
- **Study of malware persistence mechanisms** by different authors (Sikorski and Honig, 2012; Russinovich and Margosis, 2016; Hasherezade, 2017; Monnappa, 2018)
- **Tools focused on spyware** like *Gatekeeper* (Wang et al., 2004) or STARS (Wu et al., 2007)
- **Volatility plugin** Autoruns to find malware persistence (Chopitea, 2014)
- Autoruns for Windows to **analyze live systems** (Russinovich, 2017)

Agenda

1 Introduction

2 Windows ASEPs taxonomy

3 Experimental evaluation

4 Related work

5 Conclusions

Conclusions

- **Windows ASEP taxonomy** is independent of the type of forensics analysis (disk or memory)
  - **Four categories** with their different extensibility points
  - **Characterization** according to: write permissions, execution privileges, **detectability in memory forensics**, freshness of system requirements, and execution and configuration scopes
- **Winesap Volatility plugin for analyze ASEPs in memory dumps**

Characteristics and Detectability of Windows Auto-Start Extensibility Points in Memory Forensics

Daniel Uroz, Ricardo J. Rodríguez

duroz@unizar.es, rjrodriguez@unizar.es

Centro Universitario de la Defensa Zaragoza

April 25, 2019

DFRWS EU 2019
Oslo, Norway