Information
Security

Abusing privileged file manipulation

Privilege escalation low-hanging fruits

Clément Lavoillotte (@clavoillotte)
Provadys

GreHack 2018

GreHack
New is not always better

Agenda

- **Looking at privileged file manipulation and its attack surface on Windows**
- **Turning arbitrary file R/W bugs into privesc**
  - Focus on file deletion / AV quarantine bugs
- **Showing some example (actual) bugs found in popular products**
  - Showcasing some techniques and tools from @tiraniddo that pentesters / defenders / vendors need to use more
  - Old-school bugs are hot again
- **~~Bashing~~ Helpfully criticizing AV software**

Introductory demo

[1]

Privileged file manipulation bugs

Privileged file manipulation bugs

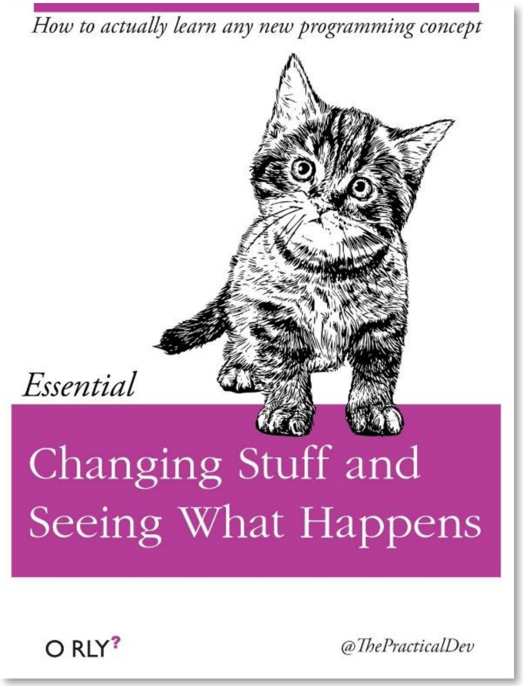
- **File operations by a privileged process (service, driver, SYSTEM process, etc.)**
  - Problems occur when an unprivileged user/process has some control over that file
  - Works with all kinds of resources, files are just an easy target
- **Examples**
  - Service started from a user-writable EXE file
  - DLL loaded in a privileged process from a user-writable location
  - Sensitive files in a user-readable location
- **Quite common in (security) software**
  - Access rights misconfiguration
  - Access to user-owned files without impersonation or restrictions
  - Time Of Check vs. Time Of Use (TOCTOU)
- **Logic bugs**
  - Very stable (no memory corruption)
  - Can survive code refactoring

How to find these bugs

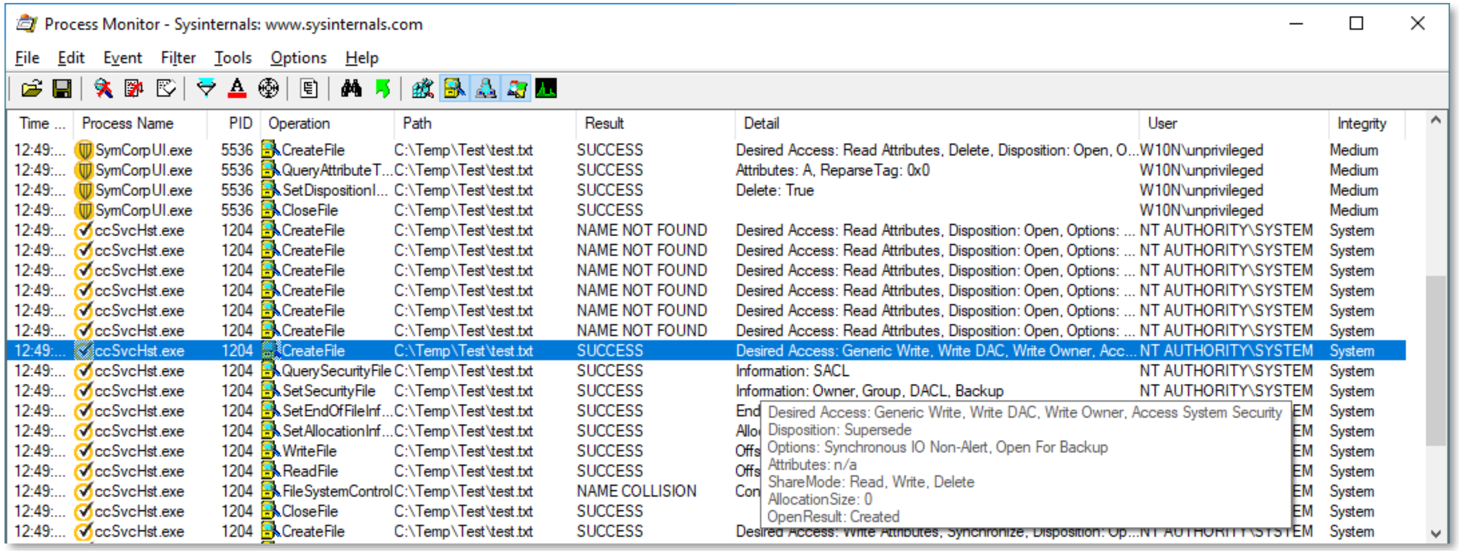
- **No assembly required (for the low-hanging ones)**

- **Process Monitor**
  - Filters on the product’s privileged processes
  - Perform actions as unprivileged user, look at the effects
    - Files & registry keys accessed, DLL loaded, processes created
    - Infer theories on the how/why the product does this
  - Fast and effective
  - Userland only

- **Explorer**
  - Or any way to view ACLs on files / folders
    - icacls
    - accesschk
    - PowerShell


How to actually learn any new programming concept
Essential
Changing Stuff and Seeing What Happens
O RLY?
@ThePracticalDev

@ThePracticalDev, CC BY-NC


Process Monitor - Sysinternals: www.sysinternals.com
File Edit Event Filter Tools Options Help

Time ...	Process Name	PID	Operation	Path	Result	Detail	User	Integrity
12:49:...	SymCorpUI.exe	5536	CreateFile	C:\Temp\Test\test.txt	SUCCESS	Desired Access: Read Attributes, Delete, Disposition: Open, O...	W10N\unprivileged	Medium
12:49:...	SymCorpUI.exe	5536	QueryAttributeT...	C:\Temp\Test\test.txt	SUCCESS	Attributes: A, ReparseTag: 0x0	W10N\unprivileged	Medium
12:49:...	SymCorpUI.exe	5536	SetDispositionI...	C:\Temp\Test\test.txt	SUCCESS	Delete: True	W10N\unprivileged	Medium
12:49:...	SymCorpUI.exe	5536	CloseFile	C:\Temp\Test\test.txt	SUCCESS		W10N\unprivileged	Medium
12:49:...	ccSvcHst.exe	1204	CreateFile	C:\Temp\Test\test.txt	NAME NOT FOUND	Desired Access: Read Attributes, Disposition: Open, Options: ...	NT AUTHORITY\SYSTEM	System
12:49:...	ccSvcHst.exe	1204	CreateFile	C:\Temp\Test\test.txt	NAME NOT FOUND	Desired Access: Read Attributes, Disposition: Open, Options: ...	NT AUTHORITY\SYSTEM	System
12:49:...	ccSvcHst.exe	1204	CreateFile	C:\Temp\Test\test.txt	NAME NOT FOUND	Desired Access: Read Attributes, Disposition: Open, Options: ...	NT AUTHORITY\SYSTEM	System
12:49:...	ccSvcHst.exe	1204	CreateFile	C:\Temp\Test\test.txt	NAME NOT FOUND	Desired Access: Read Attributes, Disposition: Open, Options: ...	NT AUTHORITY\SYSTEM	System
12:49:...	ccSvcHst.exe	1204	CreateFile	C:\Temp\Test\test.txt	NAME NOT FOUND	Desired Access: Read Attributes, Disposition: Open, Options: ...	NT AUTHORITY\SYSTEM	System
12:49:...	ccSvcHst.exe	1204	CreateFile	C:\Temp\Test\test.txt	NAME NOT FOUND	Desired Access: Read Attributes, Disposition: Open, Options: ...	NT AUTHORITY\SYSTEM	System
12:49:...	ccSvcHst.exe	1204	CreateFile	C:\Temp\Test\test.txt	SUCCESS	Desired Access: Generic Write, Write DAC, Write Owner, Acc...	NT AUTHORITY\SYSTEM	System
12:49:...	ccSvcHst.exe	1204	QuerySecurityFile	C:\Temp\Test\test.txt	SUCCESS	Information: SACL	NT AUTHORITY\SYSTEM	System
12:49:...	ccSvcHst.exe	1204	SetSecurityFile	C:\Temp\Test\test.txt	SUCCESS	Information: Owner, Group, DACL, Backup	NT AUTHORITY\SYSTEM	System
12:49:...	ccSvcHst.exe	1204	SetEndOfFileInf...	C:\Temp\Test\test.txt	SUCCESS	End	EM	System
12:49:...	ccSvcHst.exe	1204	SetAllocationInf...	C:\Temp\Test\test.txt	SUCCESS	Allo	EM	System
12:49:...	ccSvcHst.exe	1204	WriteFile	C:\Temp\Test\test.txt	SUCCESS	Offs	EM	System
12:49:...	ccSvcHst.exe	1204	ReadFile	C:\Temp\Test\test.txt	SUCCESS	Offs	EM	System
12:49:...	ccSvcHst.exe	1204	FileSystemControl	C:\Temp\Test\test.txt	NAME COLLISION	Con	EM	System
12:49:...	ccSvcHst.exe	1204	CloseFile	C:\Temp\Test\test.txt	SUCCESS		EM	System
12:49:...	ccSvcHst.exe	1204	CreateFile	C:\Temp\Test\test.txt	SUCCESS	Desired Access: Write Attributes, Synchronize, Disposition: Op...	NT AUTHORITY\SYSTEM	System

Desired Access: Generic Write, Write DAC, Write Owner, Access System Security
Disposition: Supersede
Options: Synchronous IO Non-Alert, Open For Backup
Attributes: n/a
ShareMode: Read, Write, Delete
AllocationSize: 0
OpenResult: Created

How to exploit these bugs

- **Make the privileged process do something helpful**

- **Arbitrary file read**
  - Read `SAM` / `SECURITY` / `SYSTEM` hives to grab local hashes, caches, LSA secrets, etc.
  - Must give a way to access the content (e.g. copy in a user-readable location)

- **Arbitrary file write**
  - Replace an existing binary if overwrite is possible
  - Drop a DLL somewhere in the PATH
  - Drop/replace a file in System32
    - Helpful trick by James Forshaw https://googleprojectzero.blogspot.com/2018/04/windows-exploitation-tricks-exploiting.html
    - The privileged DiagHub service can be made to load from System32 a file with any extension as a DLL
  - Must have a way to control the content

- **Arbitrary file delete**

Techniques & tools

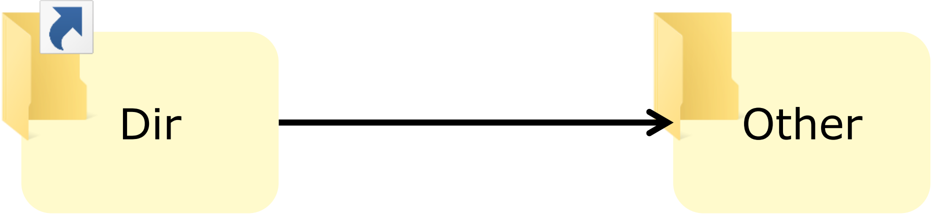
- **Useful techniques as an unprivileged user**
  - NTFS mount points (junctions)
  - Object manager symbolic links
  - Opportunistic Locks
  - Combinations
  - Courtesy of James Forshaw (@tiraniddo)
    - "A Link to the Past - Abusing Symbolic Links on Windows" at SyScan & Infiltrate 2015 (must watch!)
    - Following descriptions are shameless over-simplifications

- **Tools**
  - James' purpose-built tools & libraries
    - https://github.com/googleprojectzero/symboliclink-testing-tools
    - https://github.com/googleprojectzero/sandbox-attacksurface-analysis-tools
  - Windows built-in tools (powershell, cmd, filesystem utilities)
  - SysInternals

- **Many filesystem-level attacks are now low-hanging fruits**

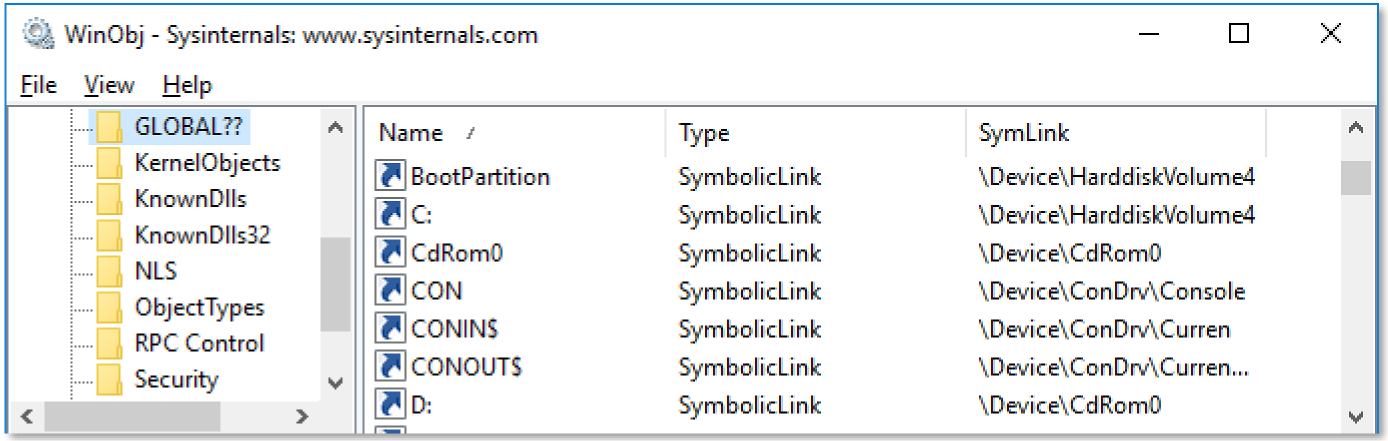
Techniques & tools (cont.)

- **NTFS mount points (junctions)**
  - Redirects a directory to another directory
    - CreateMountPoint.exe, junction.exe, mklink /j, New-Item –Type Junction



`C:\Dir\file.exe` is reparsed as `C:\Other\file.exe`

- **Object manager symbolic links**
  - Links in the object manager namespace that can point to files (and other stuff), even if the file does not exist
    - CreateDosDeviceSymlink.exe, WinObj.exe

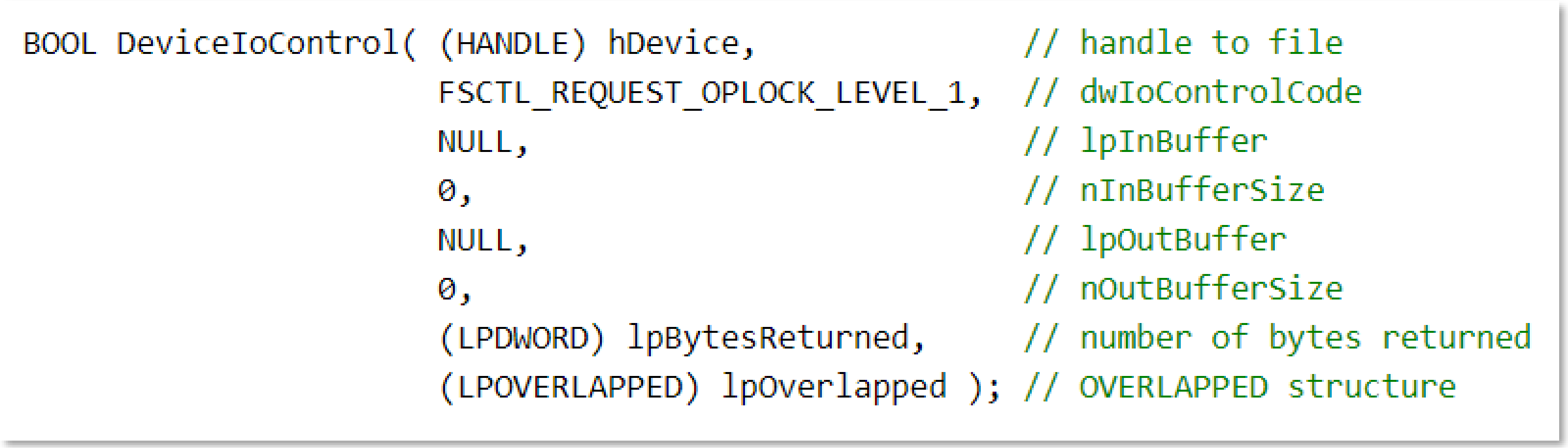


- GLOBAL??
- KernelObjects
- KnownDlls
- KnownDlls32
- NLS
- ObjectTypes
- RPC Control
- Security

Name	Type	SymLink
BootPartition	SymbolicLink	\Device\HarddiskVolume4
C:	SymbolicLink	\Device\HarddiskVolume4
CdRom0	SymbolicLink	\Device\CdRom0
CON	SymbolicLink	\Device\ConDrv\Console
CONIN$	SymbolicLink	\Device\ConDrv\Curren
CONOUT$	SymbolicLink	\Device\ConDrv\Curren...
D:	SymbolicLink	\Device\CdRom0

Techniques & tools (cont.)

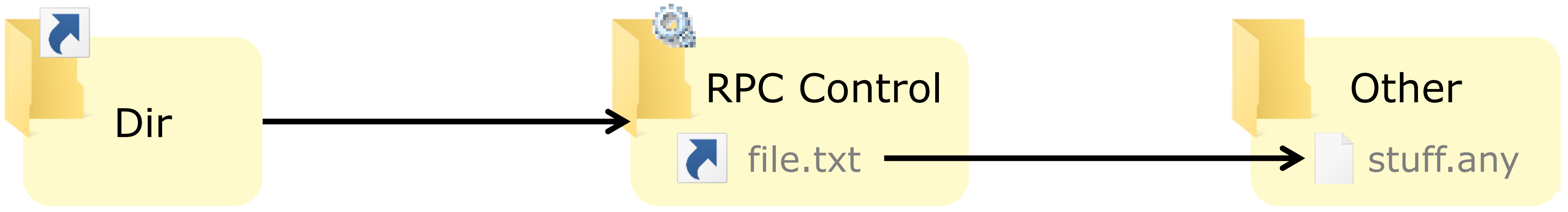
- **Opportunistic Locks (OpLocks)**
  - Placed on a file/directory to trigger an action (callback) when it is accessed
    - SetOpLock.exe
  - Can turn some race conditions into reliable exploit
  - Some limitations
    - One-shot
    - Does not work with all types of access



```
BOOL DeviceIoControl( (HANDLE) hDevice,              // handle to file
                      FSCTL_REQUEST_OPLOCK_LEVEL_1,  // dwIoControlCode
                      NULL,                          // lpInBuffer
                      0,                             // nInBufferSize
                      NULL,                          // lpOutBuffer
                      0,                             // nOutBufferSize
                      (LPDWORD) lpBytesReturned,     // number of bytes returned
                      (LPOVERLAPPED) lpOverlapped ); // OVERLAPPED structure
```

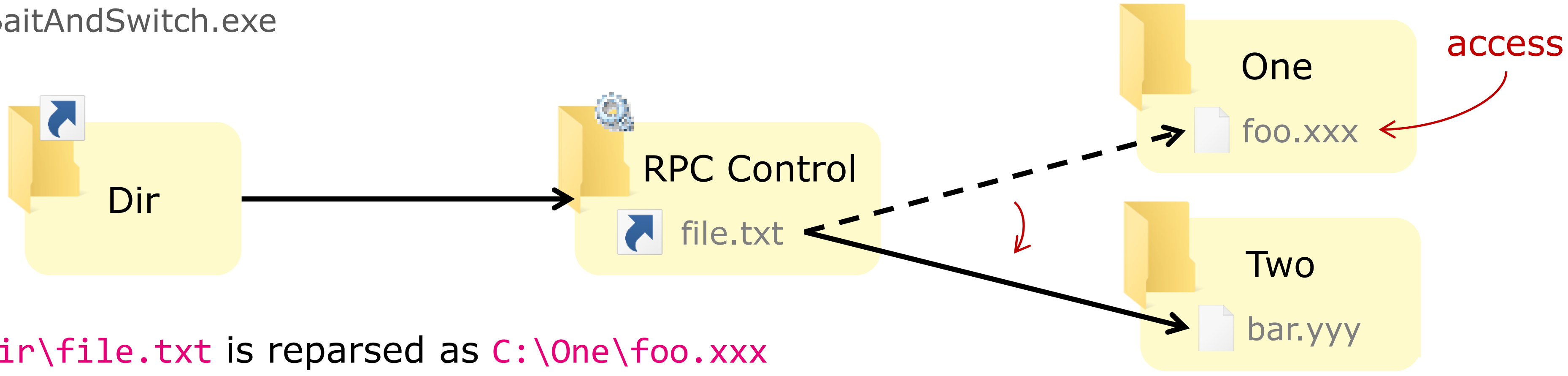
Techniques & tools (cont.)

- **Combinations**
  - Junction + Object Manager symbolic link = pseudo-symlink
    - CreateSymlink.exe



C:\Dir\file.txt is reparsed as C:\Other\stuff.any

  - Pseudo-symlink + OpLock = “BaitAndSwitch”
    - BaitAndSwitch.exe



C:\Dir\file.txt is reparsed as C:\One\foo.xxx

then as C:\Two\bar.yyy

Bug: Arbitrary file write

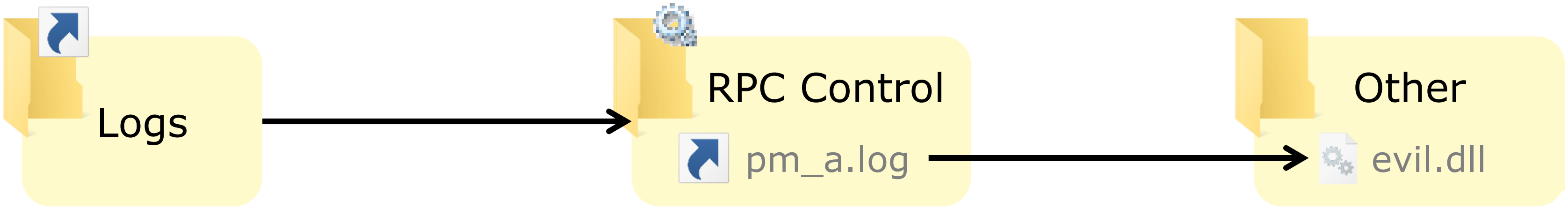
- **Log file with over-permissive ACL**
  - The “Everyone” group has full control over the log file
  - Can also add files in / set properties of its parent folder

- **Write from a privileged service/process**
  - Without impersonation

- **Somewhat common bug**
  - Similar bugs found in multiple products
    - In Windows components (e.g. P0 bug #1492) by James Forshaw (@tiraniddo)
    - In Cylance and in the Windows Standard Collector Service by Ryan Hanson (@ryhanson)
    - In Symantec / Altiris agent by Ben Turner (@benpturner)
    - In McAfee ES (patched) and several other products (one with a bug collision)
  - Variants (e.g. DACL change instead of write)
    - In other Windows components (e.g. P0 bug #1428) by James Forshaw (@tiraniddo)
    - In Task Scheduler ALPC (CVE-2018–8440), by @SanboxEscaper
    - Harder to find/exploit
    - Also some low-hanging ones (one with a bug collision)

Bug: Arbitrary file write in McAfee Endpoint Security

- **Exploitation**
  - Delete log files in `C:\ProgramData\McAfee\Endpoint Security\Logs\`
  - Replace `C:\ProgramData\McAfee\Endpoint Security\Logs` by a junction to the `\RPC Control\` object directory
  - Create a `PackageManager_Activity.log` symlink in `\RPC Control\` that points to the target path `C:\Other\evil.dll`
  - Trigger the log creation (e.g. update)
    - evil.dll is created in the target folder with the same permissive ACL
  - Use the DLL to hijack a privileged service/app



- **With the diaghub service**
  - Can be used to load the DLL written to `C:\Windows\System32\`
  - The "\RPC Control\" redirection is not needed then
    - DiagHub service can load files with arbitrary names
    - We can use a simple Logs → System32 junction

[2]

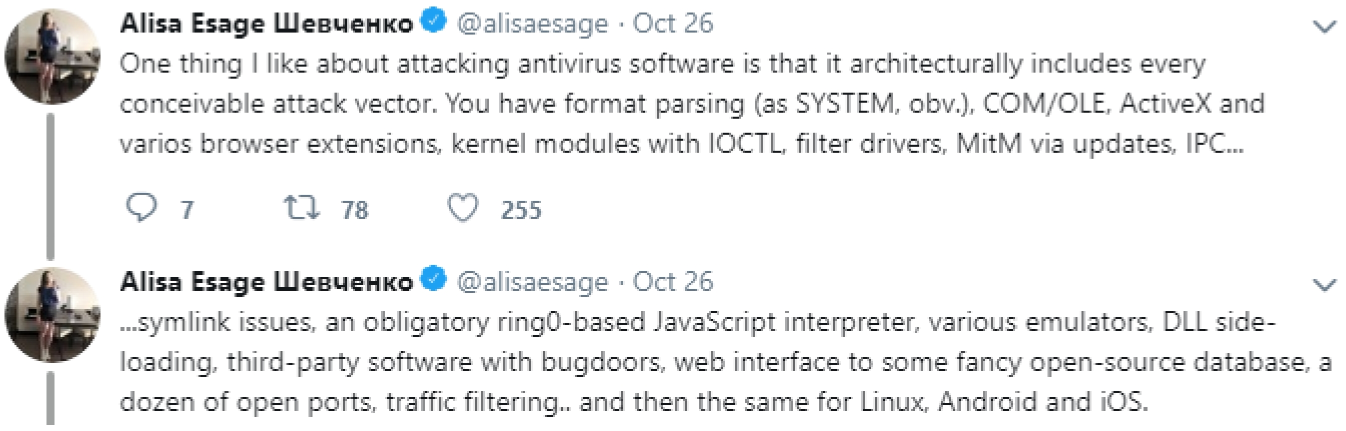
AV file scanning

AV file scanning

- **Files are usually scanned / removed / restored by a privileged process**
  - Can often be triggered by unprivileged users
    - Even restore in many AVs’ default configuration
    - Or disabled in the UI but accessible via COM hijacking, as shown by Bálint Varga-Perke (@buherator)
  - Cool kids use impersonation on restore

- **Abuse potential**
  - Scanning a file → privileged file read
  - Putting a file in Quarantine → privileged file read/copy
  - Deleting the original file → privileged file delete
  - Restoring a file → privileged file write

- **Also a lot of other attack surface**



Alisa Esage Шевченко @alisaesage · Oct 26

One thing I like about attacking antivirus software is that it architecturally includes every conceivable attack vector. You have format parsing (as SYSTEM, obv.), COM/OLE, ActiveX and varios browser extensions, kernel modules with IOCTL, filter drivers, MitM via updates, IPC...

7 78 255

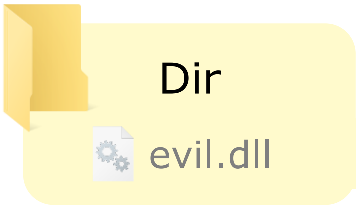
Alisa Esage Шевченко @alisaesage · Oct 26

...symlink issues, an obligatory ring0-based JavaScript interpreter, various emulators, DLL side-loading, third-party software with bugdoors, web interface to some fancy open-source database, a dozen of open ports, traffic filtering.. and then the same for Linux, Android and iOS.

Abusing file restore: AVgater

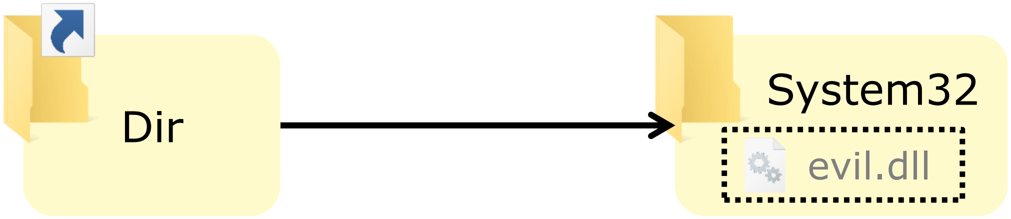
- **Logic bug found on multiple AVs by Florian Bogner**
- **Abuses the file restore process**
  - Put `C:\Dir\evil.dll` into Quarantine (manually or via detection)
  - Replace `C:\Dir` by a junction to `C:\Windows\System32`
  - Restore `C:\Dir\evil.dll`
    which reparses to `C:\Windows\System32\evil.dll`
  - The target file is created by the privileged component

Before Quarantine



`C:\Dir\evil.dll` is a standard file

Before Restore

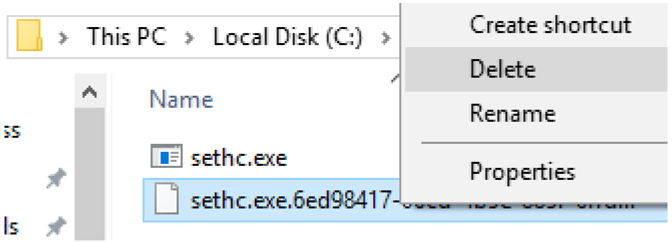


`C:\Dir\evil.dll` will be reparsed as
`C:\Windows\System32\evil.dll`

Bug: Privilege escalation via file restore in Symantec Endpoint Protection (CVE-2018-5237)

provadys

- **AVgater-style**
  - Access rights check before the file is restored
  - But no impersonation on the actual file write → TOCTOU
  - Race is easy to win when overwriting an existing file (prompt after the access check)
  - "Fixed" around 10/2017
    - SEP creates a temporary file to prevent deleting / renaming / redirecting the parent folder
  - Some l33t bypass :



- **Exploitation**
  - Create file `C:\Temp\Dir\sethc.exe` with the desired content
  - Manually add it to quarantine
  - Restore; delete the temporary file when the overwrite confirmation prompt pops up
  - Replace `C:\Temp\Dir` by a junction to `C:\Windows\System32`
  - Click yes on the overwrite prompt
  - `C:\Windows\System32\sethc.exe` is overwritten (and the user is owner)

Demo

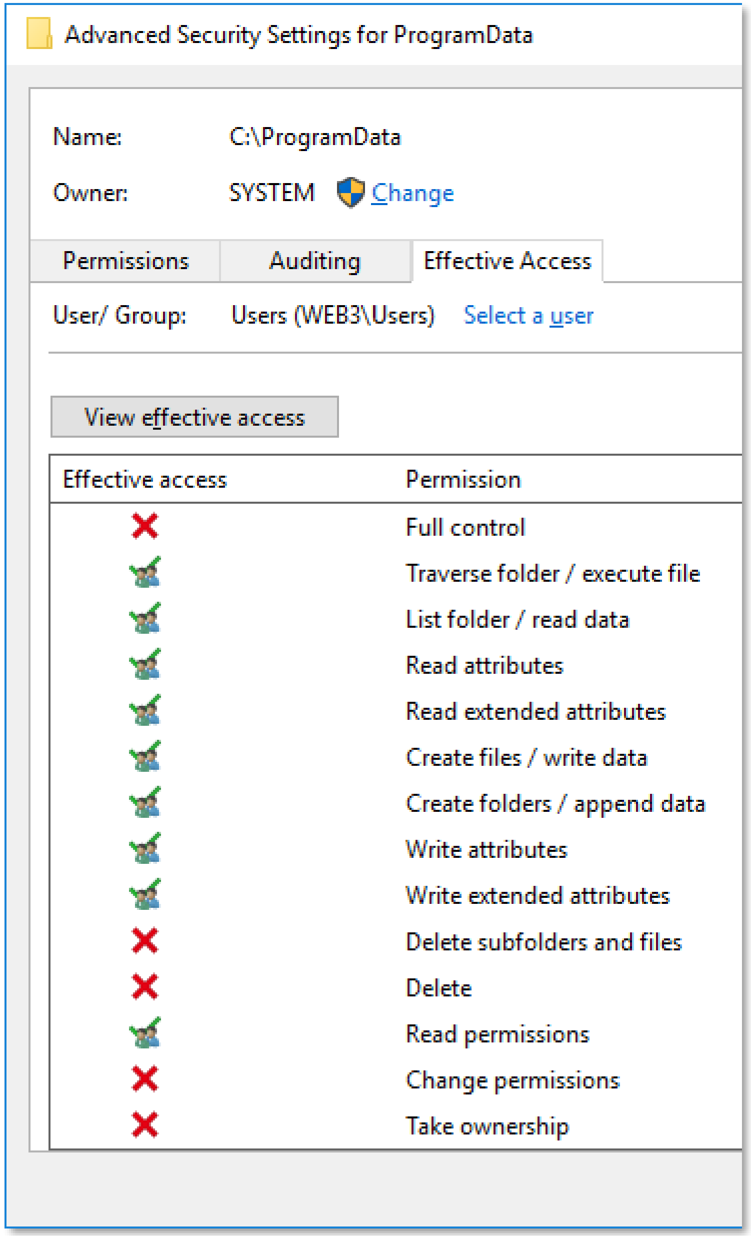
Bugs: Privilege escalation via file restore (multiple products)

- **Quarantine access rights misconfiguration**
  - User can modify, move, rename or replace quarantine files
- **Over-privileged restore**
  - Restore without impersonation
  - Restore creates/overwrites the target file
    - Also registry keys
- **Exploitation**
  - Change content of the quarantine file (often XOR-ed) or metadata
  - File : Change the path in metadata or use a junction/symlink
  - Registry : add an entry with the keys we want to create
    - Create a new service, adjust Image Execution Options, etc.
- **Found in multiple AV products**

[3] Abusing file deletion

Abusing file deletion

- **Files are removed when deemed malicious**
  - Manipulate the file and/or the deletion process
  - Remove arbitrary files
  - So… what?
- **Exploiting arbitrary delete**
  - Remove files that we can replace
    - C:\ProgramData
    - C:\Windows\Temp
  - Default rights allow unprivileged users to create files & directories
    - But not to modify existing files (owner-locked)
- **AV software is an obvious target for these**
  - Similar technique to exploit installers (and others programs) that e.g. do not check for preexisting files



Advanced Security Settings for ProgramData

Name: C:\ProgramData

Owner: SYSTEM Change

User/ Group: Users (WEB3\Users) Select a user

Effective access	Permission
✗	Full control
✓	Traverse folder / execute file
✓	List folder / read data
✓	Read attributes
✓	Read extended attributes
✓	Create files / write data
✓	Create folders / append data
✓	Write attributes
✓	Write extended attributes
✗	Delete subfolders and files
✗	Delete
✓	Read permissions
✗	Change permissions
✗	Take ownership

Abusing file deletion on AV

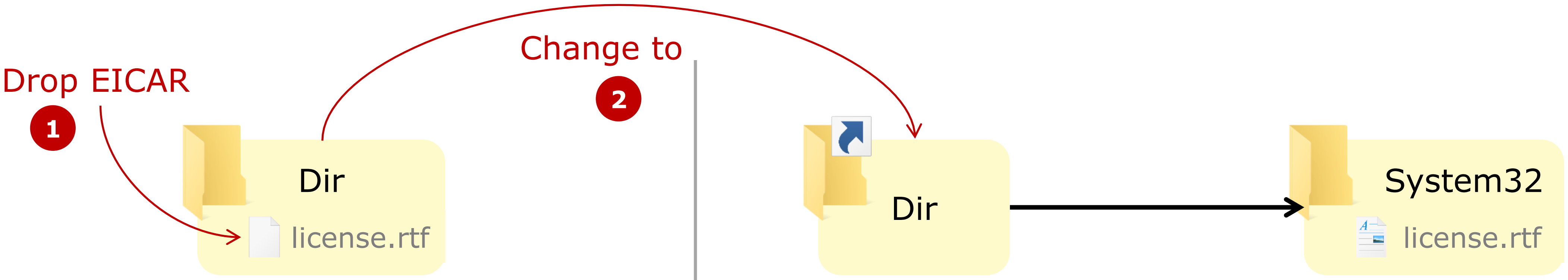
- **Vectors & triggers**
  - Manually put file into quarantine
  - Use auxiliary tools (e.g. file shredder)
  - Make the AV believe the file is malicious
  - Bypass access checks: TOCTOU / oplocks, process injection (checks in GUI), path confusion

- **Exploitation strategies**
  - DLL planting with default rights and a DLL in `C:\ProgramData\AV`
    - Trigger the deletion of `C:\ProgramData\AV\some.dll`
    - Replace it with a malicious DLL
    - DLL is loaded in the privileged AV process
  - DLL planting without add/write access to subdirs of `C:\ProgramData\AV`
    - Bug must allow recursive deletion of directories
    - Same method, but on the whole `C:\ProgramData\AV` directory structure (make a copy beforehand)
  - Without a DLL loaded from `C:\ProgramData`
    - Target subsequent file write/copy/move operations on other files in `C:\ProgramData` or `C:\Windows\Temp`

Redirected file deletion

- **What if the file AV wants to remove is no longer there?**

- **“Redirect” a file deletion via TOCTOU**
  - Drop EICAR in `C:\Dir\license.rtf`
  - Wait for it to be detected
  - Replace `C:\Dir` by a junction to `C:\Windows\System32`
  - AV deletes `C:\Dir\license.rtf` which reparses to `C:\Windows\System32\license.rtf`



`C:\Dir\license.rtf` is a standard file

`C:\Dir\license.rtf` will be reparsed as `C:\Windows\System32\license.rtf`

Redirected file deletion (cont.)

- **The proper way: oplocks, BaitAndSwitch style**
  - Place an oplock on the file to replace it after its scan but before its removal
  - Does not always work
    - File can be accessed a (variable) number of times before it is removed
    - No granular control: once the lock is released, multiple accesses can occur before the next lock

- **The quick & dirty way**
  - Create directory & junction
  - Drop EICAR in directory
  - Exchange dir & junction
  - Loop
    - ~1/3 chance of hit
  - Retry as needed
    - Add milliseconds of Sleep
  - Works surprisingly well (in some cases)

```
New-Item -ItemType Directory C:\Temp\Dir
New-Item -ItemType Junction C:\Temp\Dir2 -Value $target_dir

'X5O!P%@AP[4\PZX54(P^)7CC)7}$EICAR-STANDARD-ANTIVIRUS-TEST-
FILE!$H+H*' | Out-File -NoNewline -FilePath C:\Temp\Dir\decoy.exe

While ($True) {
    Rename-Item C:\Temp\Dir C:\Temp\DirX
    Rename-Item C:\Temp\Dir2 C:\Temp\Dir
    Rename-Item C:\Temp\DirX C:\Temp\Dir2
}
```

Exploiting AV file quarantine

- **Some AVs perform operations before removing an infected file**
  - Create/delete temporary files in the same directory
  - Copy or move/rename the infected file in a user-writable location
  - Copy or move the infected file to a user-readable quarantine location

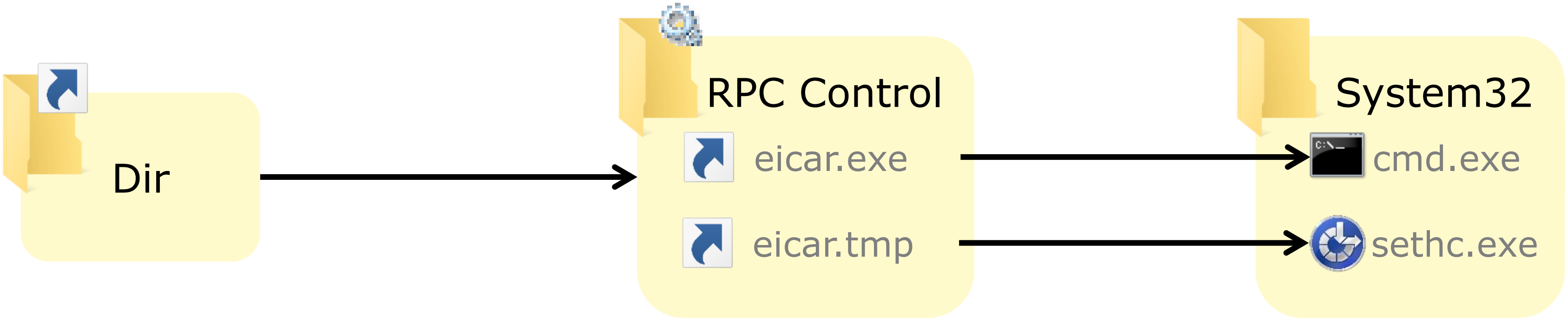
- **Exploiting for arbitrary write**
  - Target the file copy/move/rename operation
  - Use oplock to lock the thread at the right time
  - Use junction, pseudo-symlinks & hardlinks to replace files as needed

- **Exploiting for arbitrary read**
  - Redirected (or manual) quarantine of `C:\Windows\System32\config\SAM`
  - Read the content from the quarantine file
  - Undo to prevent removal on reboot (bricks the OS)
  - Same with a temporary copy of the file (+ oplocks to have the time to read it)
  - Used to exploit e.g. CVE-2017-13680 (TOCTOU in Symantec Endpoint Protection)

Bug: TOCTOU during file quarantine in multiple products

- **Temporary copy/rename of the infected file**
  - AV copies or renames the infected file in the same (user-writable) directory
    - File `eicar.exe` copied or renamed to `eicar.tmp` before removal
  - Found in multiple AVs (most already patched)

- **Exploitation**
  - OpLock the process before the copy/rename operation
  - Replace the file and its copy/rename target by pseudo-symlinks
    - File `eicar.exe` symlink → the file we want to copy (`cmd.exe`)
    - Target file `eicar.tmp` symlink → the file we want to replace (`sethc.exe`)
  - Trigger the copy/rename
    - `sethc.exe` is replaced by `cmd.exe`
    - For copy/rename without the overwrite flag, use DLL planting or DiagHub trick



File layout before copy/rename of `C:\Dir\eicar.exe` to `C:\Dir\eicar.tmp`

Demo

[4]

Conclusion

Vendor responses

Product	ID	Vulnerablity	Arbitrary file	Reported	Fix
Symantec Endpoint Protection 12 & 14	CVE-2017-13680	TOCTOU in the quarantine GUI	Deletion Read	09/2017	Available 11/2017
	CVE-2018-5236	TOCTOU during file deletion	Deletion	11/2017	Available 06/2018
	CVE-2018-5237	Check bypass in file restore	Write	11/2017	Available 06/2018
AV product A	TBD	Over-privileged file deletion	Deletion	03/2018	In progress
AV product B	TBD	Over-privileged file restore	Write	05/2018	In progress
McAfee Endpoint Security 10	TBD	Overpermissive access rights Over-privileged file creation	Write Deletion	05/2018	Available 10/2018
AV product C	TBD	TOCTOU during file deletion	Deletion	05/2018	In progress
AV product D	TBD	TOCTOU during file deletion	Deletion	05/2018	In progress
F-Secure SAFE/CS/CP	(none)	Over-privileged file copy	Write, Read, Delete	07/2018	Available 08/2018
Product E	TBD	Overpermissive access rights Over-privileged file creation	Write	06/2018	In progress
Product F	TBD	Over-privileged file creation	Write	07/2018	In progress
Product G	TBD	Over-privileged file creation	Write	07/2018	In progress
Product H	TBD	Over-privileged file creation	Write	08/2018	In progress
Product I	TBD	Overpermissive access rights	DACL set	08/2018	In progress

Product names & additional details will be published as fixes become available

Preventing / fixing these bugs

- **Least privilege**
  - Do not break the security boundary in the first place
  - Impersonate and/or use restricted tokens when possible
    - Difficult tradeoff, depending on the existing software architecture / design
    - Malware could also abuse the lack of privileges on deletion, e.g. with deny ACLs
- **Harden the product**
  - Work on fully resolved paths
  - Lock before check, release lock after use
  - Harden all “secondary” tools, not just the main product
- **Defense in depth**
  - Restrict access rights
    - Remove write permission to your ProgramData & Windows\Temp subfolders
    - Also remove read permissions when possible
  - Break exploitation avenues
    - Do not load code from ProgramData
    - Do not use user-accessible files / directories when it can be avoided

Detection

- **Most of these attempts will generate logs**
  - But not necessarily alarming ones
    - EICAR, Low risk, Threat mitigated, etc.
  - Correlate with filesystem changes and privileged process creation
  - Real-time log forwarding when online
    - The first file getting deleted could be the log file

- **Behavioral analysis**
  - Detected files that are no longer there
  - Loops exchanging directories
  - OpLocks outside of the usual processes
  - Processes (even your own) replacing system files

Takeways

- **Balance AV risk-benefit**
  - Most AVs have a huge attack surface
  - Depend on your use case & threat model
    - E.g. personal computer of non-tech people vs. multi-user sensitive VDI server
  - Too many easily exploitable bugs?
    The tradeoff might not be worth it (vs strong app whitelising etc.)

- **Vendors**
  - Identify and reduce the attack surface of your own software
  - Kill the low-hanging fruits

- **Defenders**
  - Test your security software
  - Watch those “low risk” / “remediated” AV entries in the logs

- **Pentesters**
  - Privileged software is attack surface, pwn it like no AV’s watching

Questions

PoCs and additional details to be published at:
https://offsec.provadys.com/

References

- **Research by James Forshaw / Google Project Zero**
  - https://googleprojectzero.blogspot.com/2015/08/windows-10hh-symbolic-link-mitigations.html
  - https://googleprojectzero.blogspot.com/2016/02/the-definitive-guide-on-win32-to-nt.html
  - https://googleprojectzero.blogspot.com/2017/08/windows-exploitation-tricks-arbitrary.html
  - https://googleprojectzero.blogspot.com/2018/04/windows-exploitation-tricks-exploiting.html
  - https://infocon.org/cons/SyScan/SyScan%202015%20Singapore/SyScan%202015%20Singapore%20presentations/SyScan15%20James%20Forshaw%20-%20A%20Link%20to%20the%20Past.pdf
  - https://vimeo.com/133002251

- **Vulnerabilities & other research**
  - https://bogner.sh/2017/11/avgater-getting-local-admin-by-abusing-the-anti-virus-quarantine/, Florian Bogner
  - https://blog.silentsignal.eu/2018/01/08/bare-knuckled-antivirus-breaking/, Bálint Varga-Perke / Silent Signal
  - https://www.atredis.com/blog/cylance-privilege-escalation-vulnerability and https://www.atredis.com/blog/cve-2018-0952-privilege-escalation-vulnerability-in-windows-standard-collector-service, Ryan Hanson / Atredis Partners
  - https://labs.nettitude.com/blog/cve-2018-5240-symantec-management-agent-altiris-privilege-escalation/, Ben Turner / Nettitude Labs
  - https://github.com/SandboxEscaper/randomrepo and https://twitter.com/SandboxEscaper and http://sandboxescaper.blogspot.com/2018/10/reversing-alpc-where-are-your-windows.html