20TH ANNUAL

安全焦点信息安全技术峰会

XFOCUS INFORMATION SECURITY CONFERENCE

重拾圈点·致敬无限

Print Way To SYSTEM

绿盟科技 天机实验室 张云海

目录 Contents

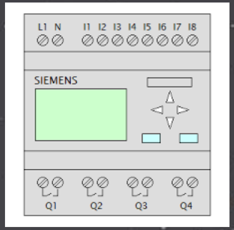
第一部分

回顾

2010年的Stuxnet

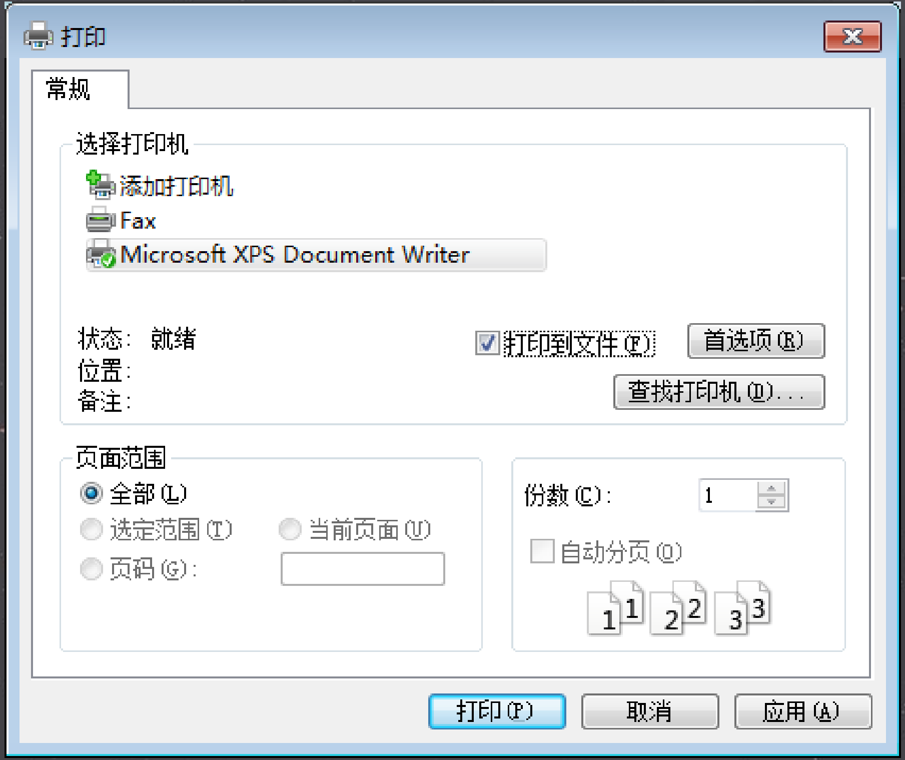
- MS08-067/CVE-2008-4250
- MS10-046/CVE-2010-2568
- **MS10-061/CVE-2010-2729**
- MS10-073/CVE-2010-2743
- MS10-092/CVE-2010-3338

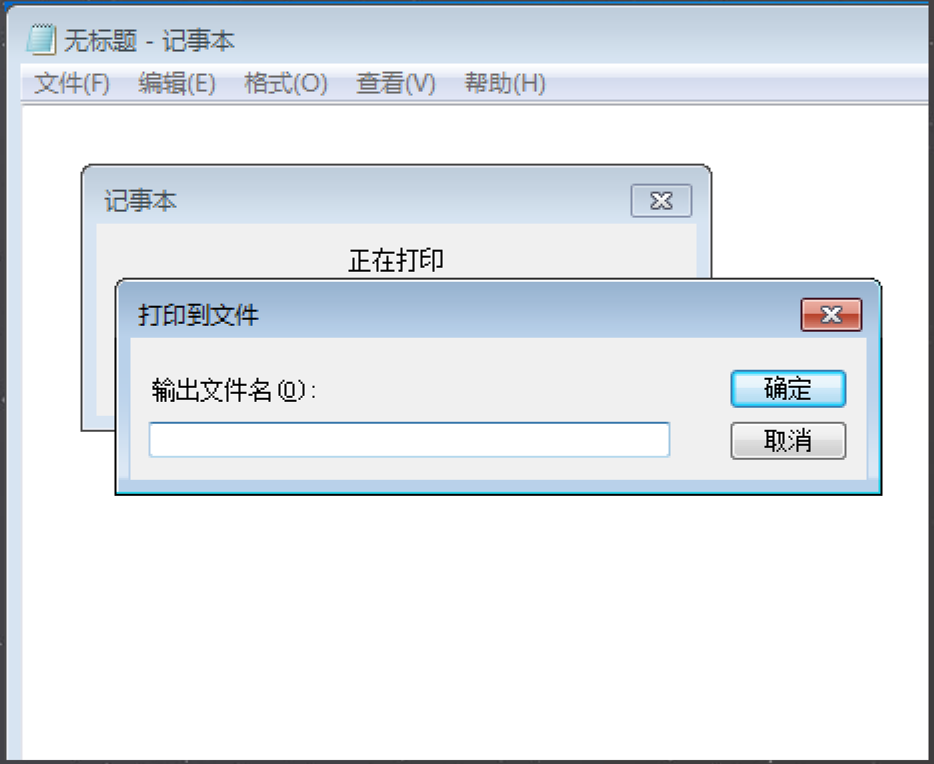
- CVE-2010-2772



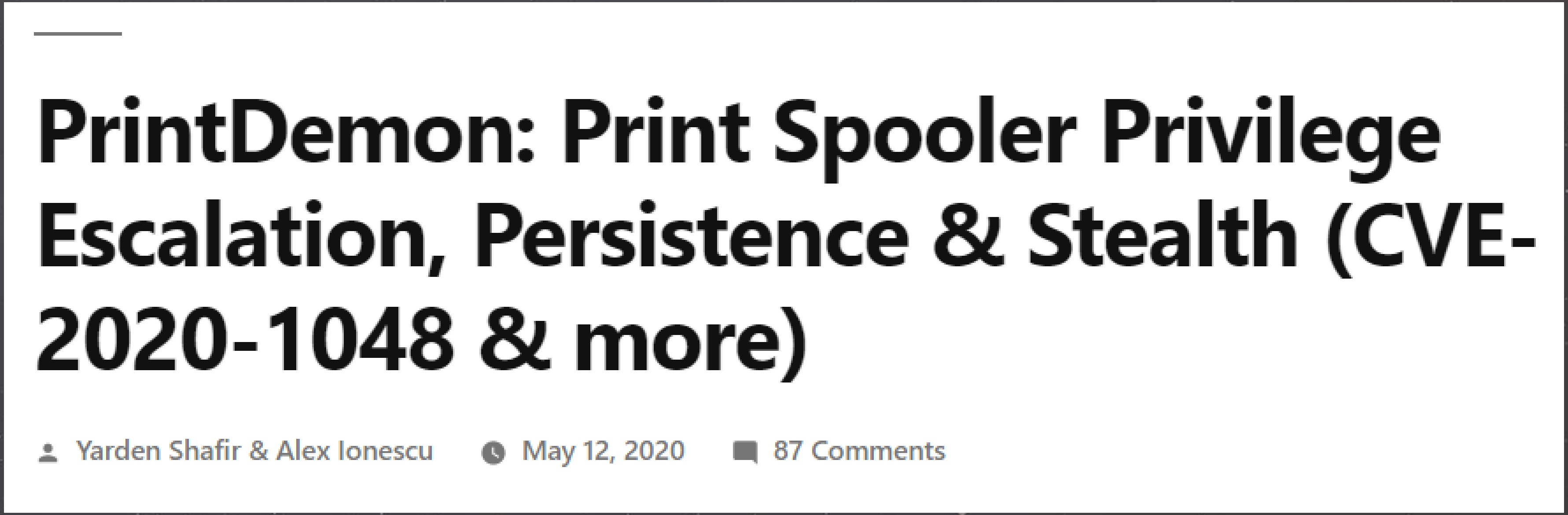
2010年的Stuxnet

MS10-061/CVE-2010-2729 Print Spooler Service Impersonation Vulnerability





2020年的PrintDemon

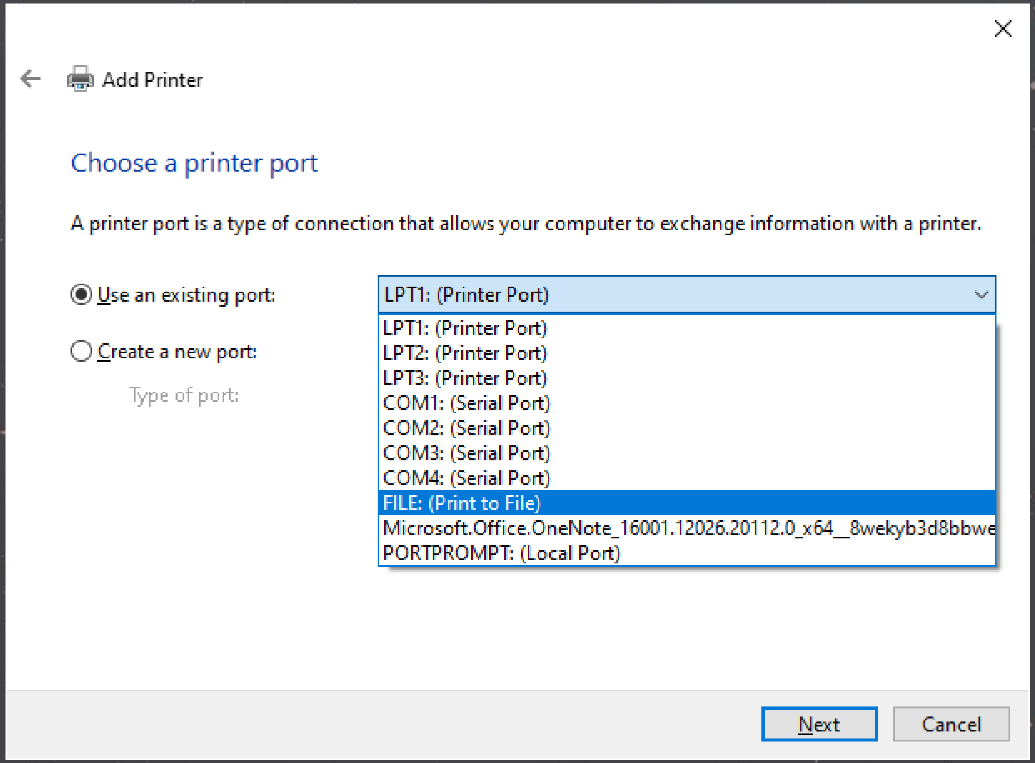


PrintDemon: Print Spooler Privilege Escalation, Persistence & Stealth (CVE-2020-1048 & more)

Yarden Shafir & Alex Ionescu May 12, 2020 87 Comments

2020年的PrintDemon

CVE-2020-1048 Windows Print Spooler Elevation of Privilege Vulnerability



Add Printer

Choose a printer port

A printer port is a type of connection that allows your computer to exchange information with a printer.

(●) Use an existing port: LPT1: (Printer Port)

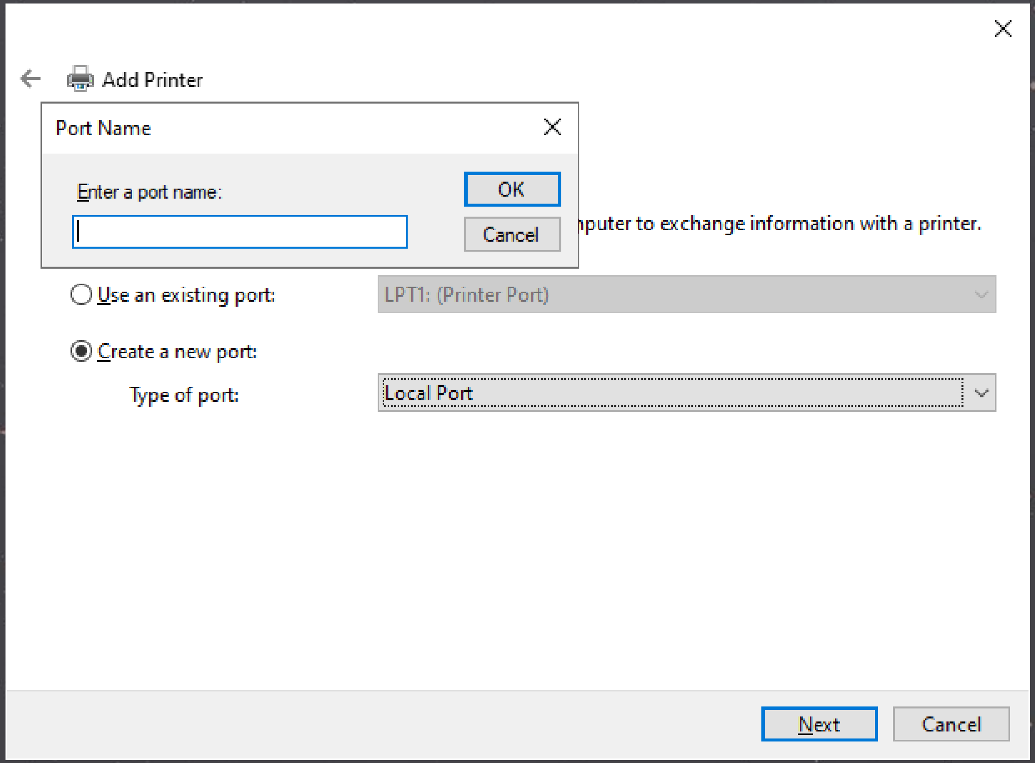
( ) Create a new port:

Type of port:

LPT1: (Printer Port)
LPT2: (Printer Port)
LPT3: (Printer Port)
COM1: (Serial Port)
COM2: (Serial Port)
COM3: (Serial Port)
COM4: (Serial Port)
FILE: (Print to File)
Microsoft.Office.OneNote_16001.12026.20112.0_x64__8wekyb3d8bbwe
PORTPROMPT: (Local Port)

2020年的PrintDemon

CVE-2020-1048 Windows Print Spooler Elevation of Privilege Vulnerability



Add Printer

Port Name

Enter a port name:

OK

Cancel

nputer to exchange information with a printer.

Use an existing port: LPT1: (Printer Port)

[✓ Create a new port:]

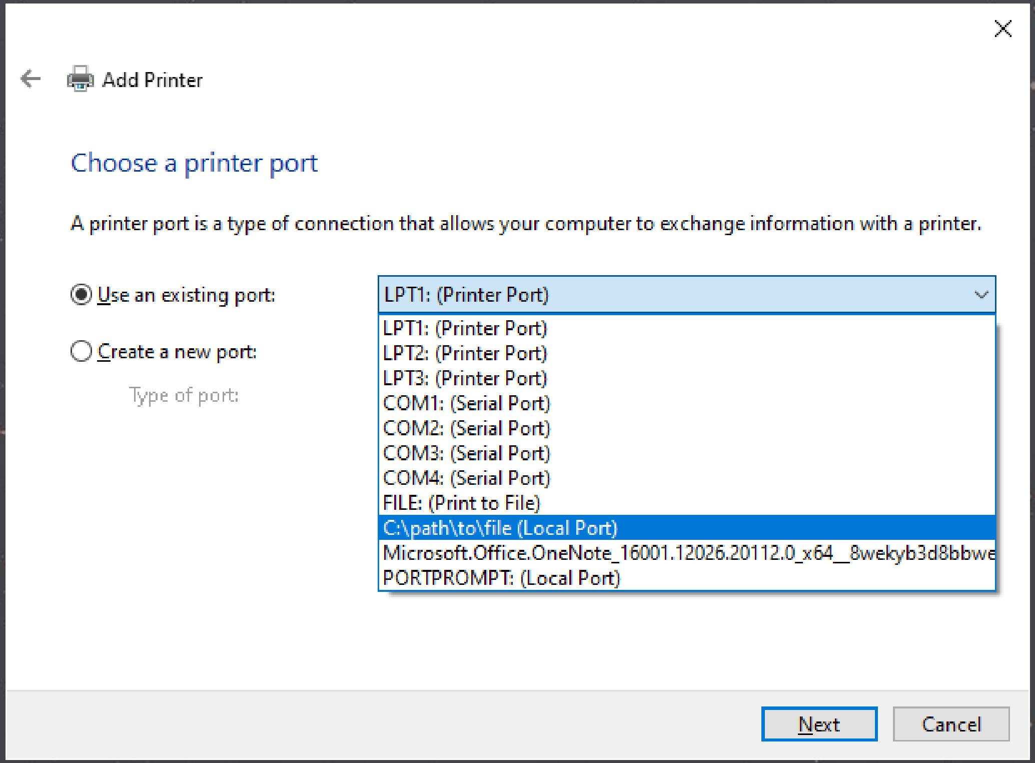
Type of port: Local Port

Next

Cancel

2020年的PrintDemon

CVE-2020-1048 Windows Print Spooler Elevation of Privilege Vulnerability



Add Printer

Choose a printer port

A printer port is a type of connection that allows your computer to exchange information with a printer.

Use an existing port: LPT1: (Printer Port)

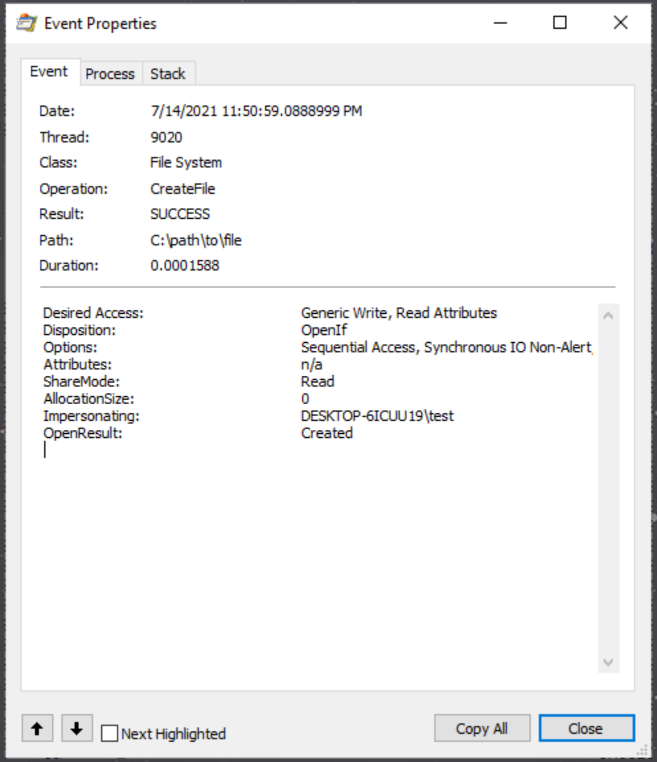
Create a new port:

Type of port:

```
LPT1: (Printer Port)
LPT2: (Printer Port)
LPT3: (Printer Port)
COM1: (Serial Port)
COM2: (Serial Port)
COM3: (Serial Port)
COM4: (Serial Port)
FILE: (Print to File)
C:\path\to\file (Local Port)
Microsoft.Office.OneNote_16001.12026.20112.0_x64__8wekyb3d8bbwe
PORTPROMPT: (Local Port)
```

2020年的PrintDemon

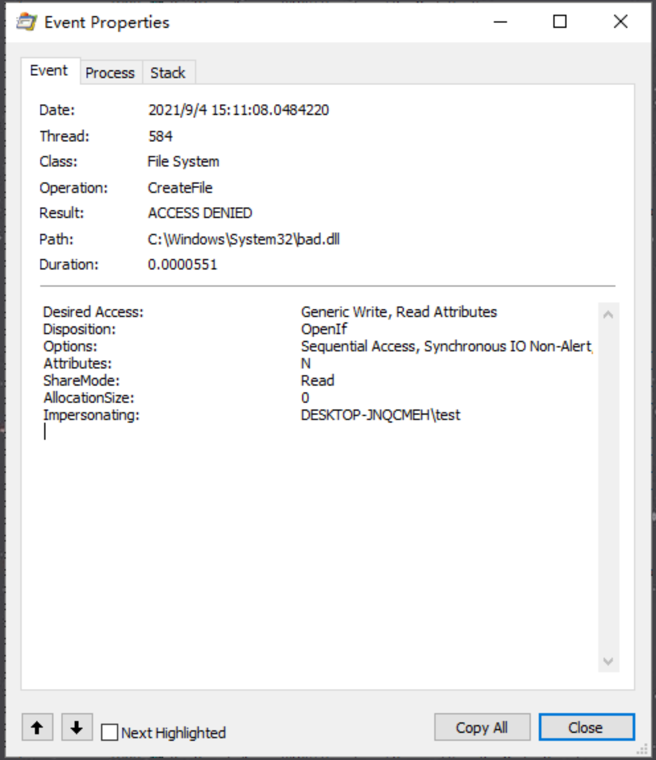
CVE-2020-1048 Windows Print Spooler Elevation of Privilege Vulnerability



```
Date:          7/14/2021 11:50:59.0888999 PM
Thread:        9020
Class:         File System
Operation:     CreateFile
Result:        SUCCESS
Path:          C:\path\to\file
Duration:      0.0001588

Desired Access:     Generic Write, Read Attributes
Disposition:        OpenIf
Options:            Sequential Access, Synchronous IO Non-Alert
Attributes:         n/a
ShareMode:          Read
AllocationSize:     0
Impersonating:      DESKTOP-6ICUU19\test
OpenResult:         Created
```

2020年的PrintDemon

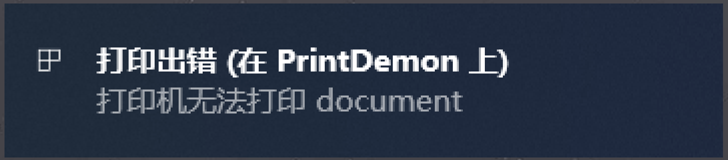
CVE-2020-1048 Windows Print Spooler Elevation of Privilege Vulnerability



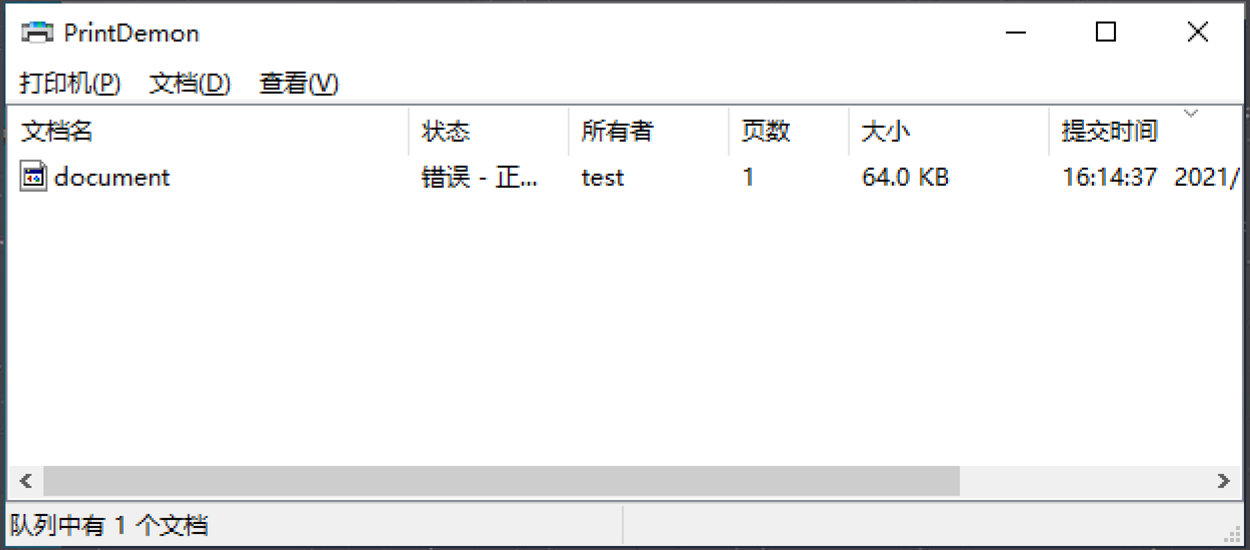
```
Date:        2021/9/4 15:11:08.0484220
Thread:      584
Class:       File System
Operation:   CreateFile
Result:      ACCESS DENIED
Path:        C:\Windows\System32\bad.dll
Duration:    0.0000551

Desired Access:      Generic Write, Read Attributes
Disposition:         OpenIf
Options:             Sequential Access, Synchronous IO Non-Alert,
Attributes:          N
ShareMode:           Read
AllocationSize:      0
Impersonating:       DESKTOP-JNQCMEH\test
```

2020年的PrintDemon

CVE-2020-1048 Windows Print Spooler Elevation of Privilege Vulnerability



打印出错 (在 PrintDemon 上)
打印机无法打印 document



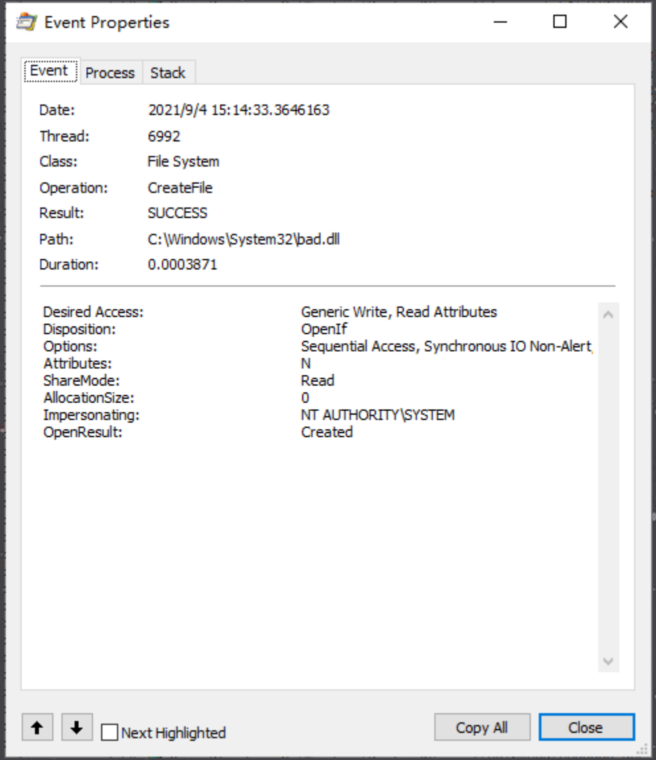
PrintDemon

文档名	状态	所有者	页数	大小	提交时间
document	错误 - 正...	test	1	64.0 KB	16:14:37 2021/

队列中有 1 个文档

2020年的PrintDemon

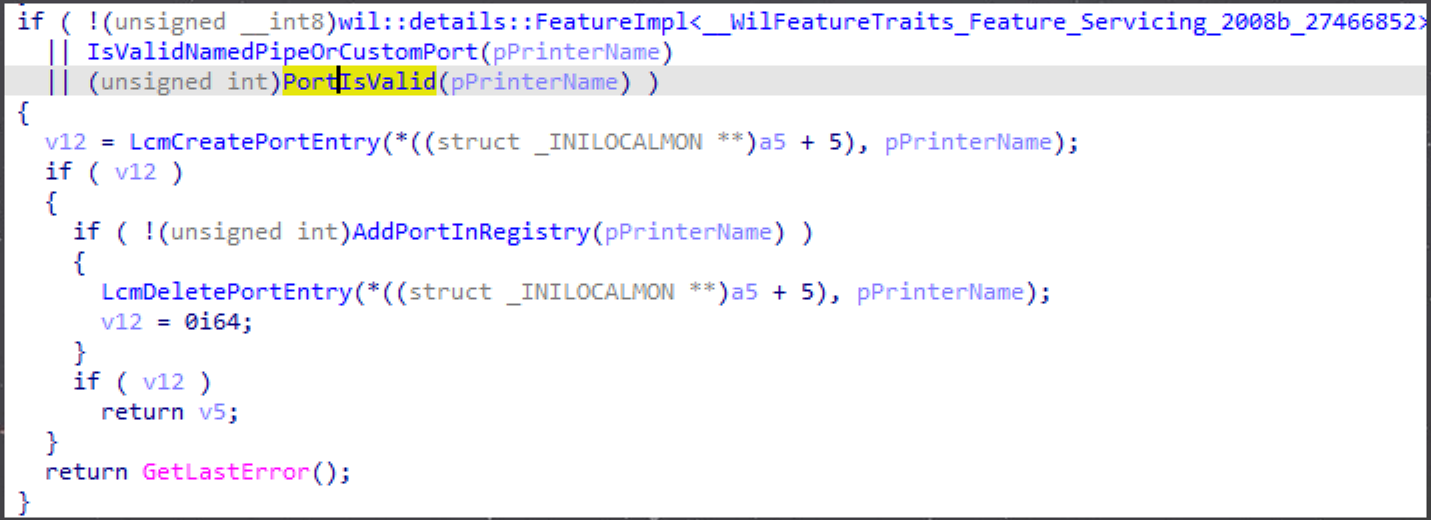
CVE-2020-1048 Windows Print Spooler Elevation of Privilege Vulnerability



```
Date:        2021/9/4 15:14:33.3646163
Thread:      6992
Class:       File System
Operation:   CreateFile
Result:      SUCCESS
Path:        C:\Windows\System32\bad.dll
Duration:    0.0003871

Desired Access:     Generic Write, Read Attributes
Disposition:        OpenIf
Options:            Sequential Access, Synchronous IO Non-Alert,
Attributes:         N
ShareMode:          Read
AllocationSize:     0
Impersonating:      NT AUTHORITY\SYSTEM
OpenResult:         Created
```

2020年的PrintDemon

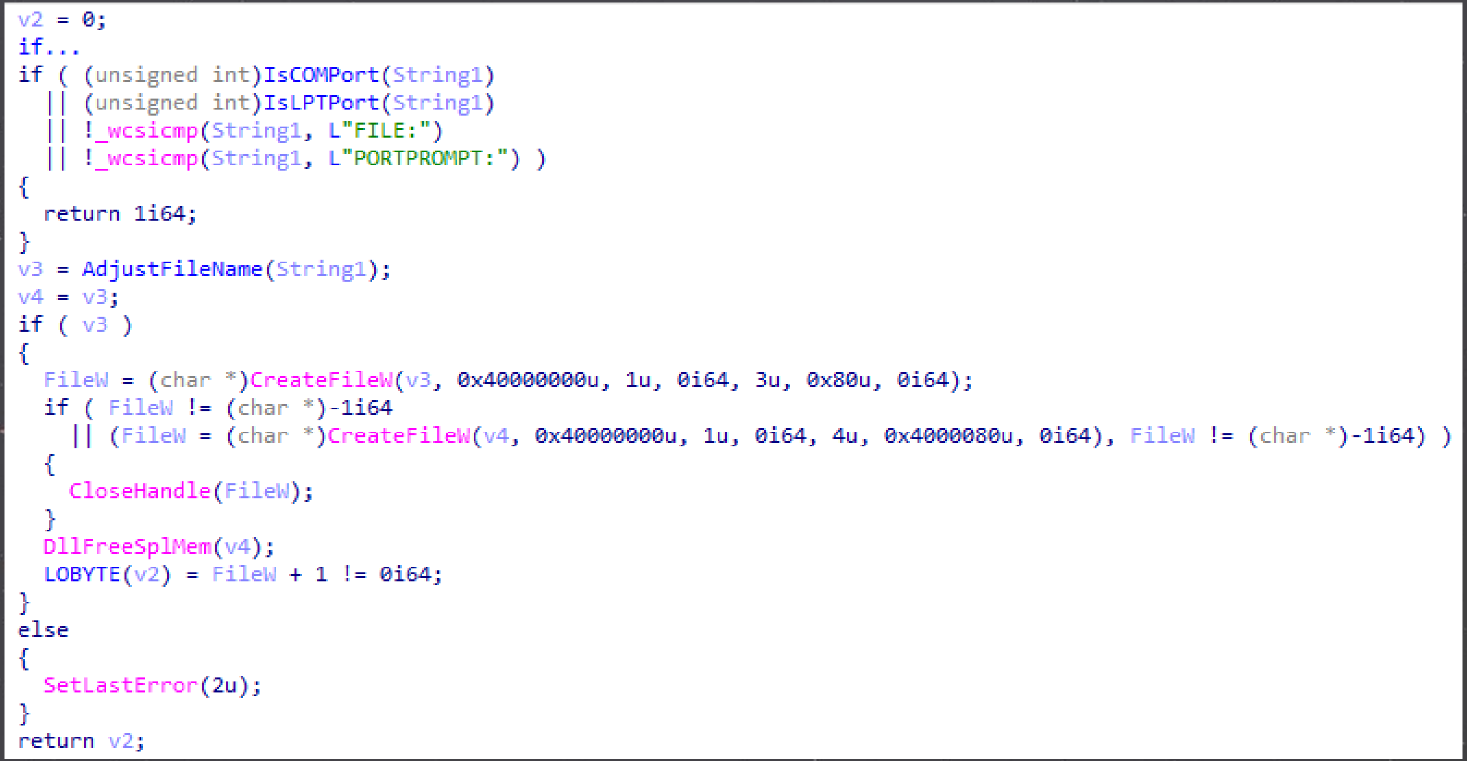
CVE-2020-1048 Windows Print Spooler Elevation of Privilege Vulnerability



```
if ( !(unsigned __int8)wil::details::FeatureImpl<__WilFeatureTraits_Feature_Servicing_2008b_27466852>
  || IsValidNamedPipeOrCustomPort(pPrinterName)
  || (unsigned int)PortIsValid(pPrinterName) )
{
  v12 = LcmCreatePortEntry(*((struct _INILOCALMON **)a5 + 5), pPrinterName);
  if ( v12 )
  {
    if ( !(unsigned int)AddPortInRegistry(pPrinterName) )
    {
      LcmDeletePortEntry(*((struct _INILOCALMON **)a5 + 5), pPrinterName);
      v12 = 0i64;
    }
    if ( v12 )
      return v5;
  }
  return GetLastError();
}
```

2020年的PrintDemon

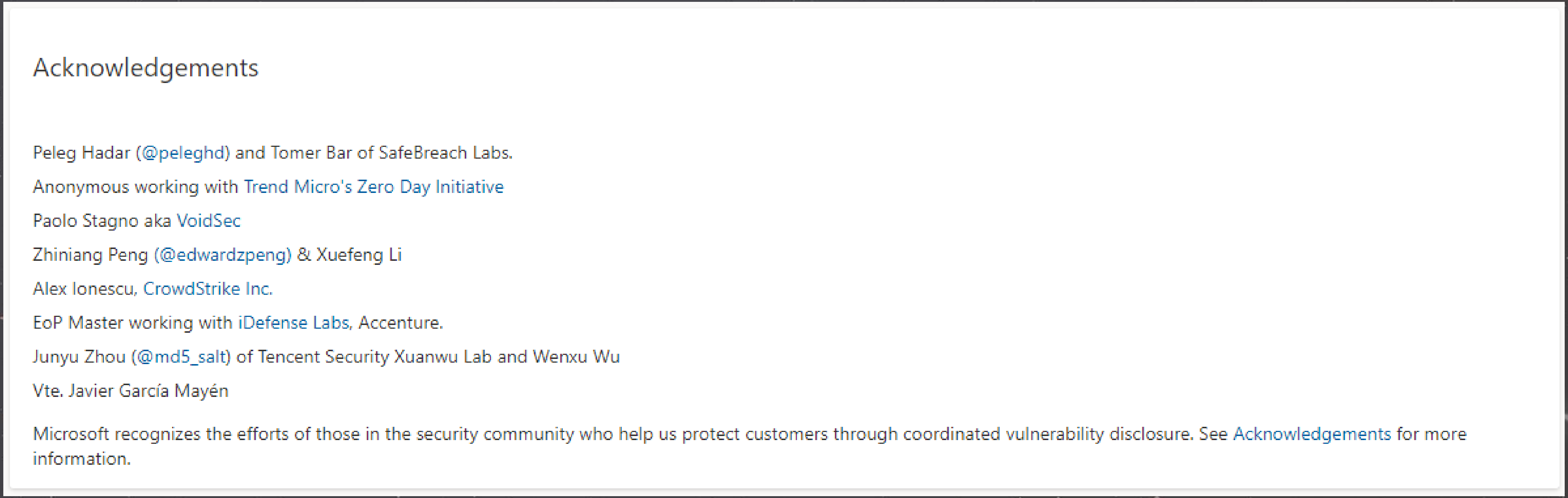
CVE-2020-1048 Windows Print Spooler Elevation of Privilege Vulnerability



```
v2 = 0;
if...
if ( (unsigned int)IsCOMPort(String1)
  || (unsigned int)IsLPTPort(String1)
  || !_wcsicmp(String1, L"FILE:")
  || !_wcsicmp(String1, L"PORTPROMPT:") )
{
  return 1i64;
}
v3 = AdjustFileName(String1);
v4 = v3;
if ( v3 )
{
  FileW = (char *)CreateFileW(v3, 0x40000000u, 1u, 0i64, 3u, 0x80u, 0i64);
  if ( FileW != (char *)-1i64
    || (FileW = (char *)CreateFileW(v4, 0x40000000u, 1u, 0i64, 4u, 0x4000080u, 0i64), FileW != (char *)-1i64) )
  {
    CloseHandle(FileW);
  }
  DllFreeSplMem(v4);
  LOBYTE(v2) = FileW + 1 != 0i64;
}
else
{
  SetLastError(2u);
}
return v2;
```

2020年的PrintDemon

CVE-2020-1337 Windows Print Spooler Elevation of Privilege Vulnerability



Acknowledgements

Peleg Hadar (@peleghd) and Tomer Bar of SafeBreach Labs.

Anonymous working with Trend Micro's Zero Day Initiative

Paolo Stagno aka VoidSec

Zhiniang Peng (@edwardzpeng) & Xuefeng Li

Alex Ionescu, CrowdStrike Inc.

EoP Master working with iDefense Labs, Accenture.

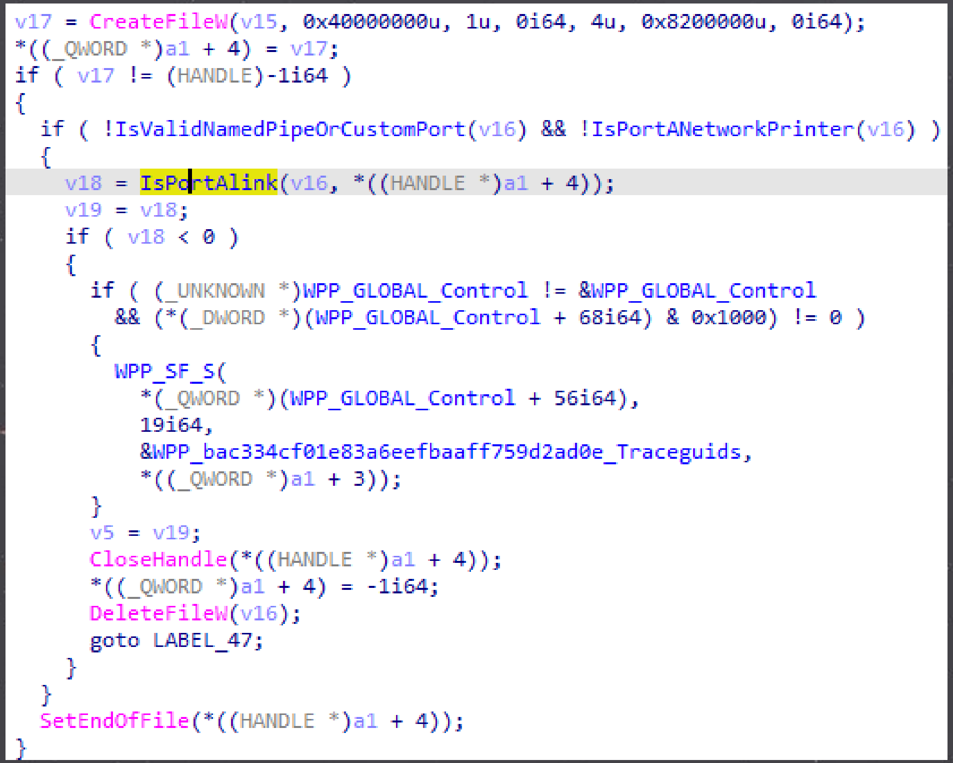
Junyu Zhou (@md5_salt) of Tencent Security Xuanwu Lab and Wenxu Wu

Vte. Javier García Mayén

Microsoft recognizes the efforts of those in the security community who help us protect customers through coordinated vulnerability disclosure. See Acknowledgements for more information.

2020年的PrintDemon

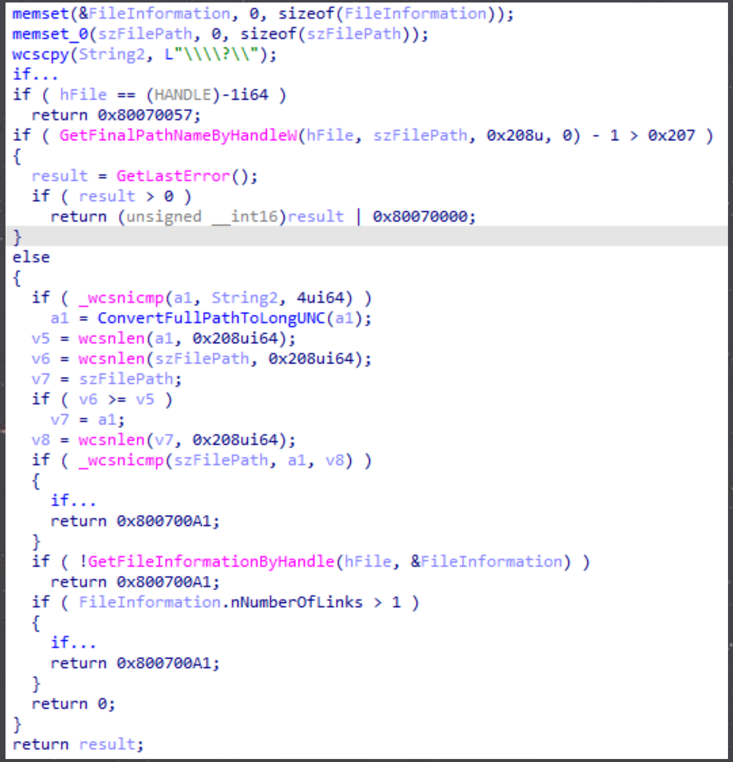
CVE-2020-1337 Windows Print Spooler Elevation of Privilege Vulnerability



```
v17 = CreateFileW(v15, 0x40000000u, 1u, 0i64, 4u, 0x8200000u, 0i64);
*((_QWORD *)a1 + 4) = v17;
if ( v17 != (HANDLE)-1i64 )
{
  if ( !IsValidNamedPipeOrCustomPort(v16) && !IsPortANetworkPrinter(v16) )
  {
    v18 = IsPortAlink(v16, *((HANDLE *)a1 + 4));
    v19 = v18;
    if ( v18 < 0 )
    {
      if ( (_UNKNOWN *)WPP_GLOBAL_Control != &WPP_GLOBAL_Control
        && (*(_DWORD *)(WPP_GLOBAL_Control + 68i64) & 0x1000) != 0 )
      {
        WPP_SF_S(
          *(_QWORD *)(WPP_GLOBAL_Control + 56i64),
          19i64,
          &WPP_bac334cf01e83a6eefbaaff759d2ad0e_Traceguids,
          *((_QWORD *)a1 + 3));
      }
      v5 = v19;
      CloseHandle(*((HANDLE *)a1 + 4));
      *((_QWORD *)a1 + 4) = -1i64;
      DeleteFileW(v16);
      goto LABEL_47;
    }
  }
  SetEndOfFile(*((HANDLE *)a1 + 4));
}
```

2020年的PrintDemon

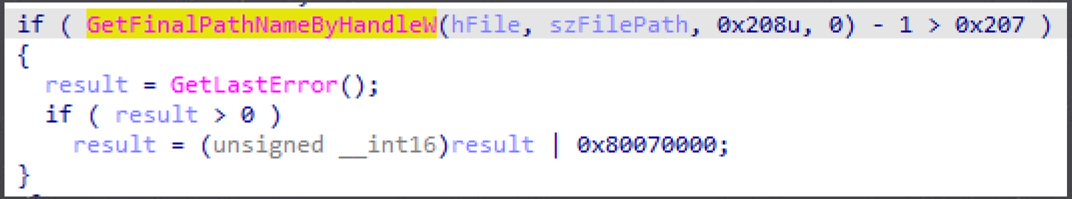
CVE-2020-1337 Windows Print Spooler Elevation of Privilege Vulnerability



```
memset(&FileInformation, 0, sizeof(FileInformation));
memset_0(szFilePath, 0, sizeof(szFilePath));
wcscpy(String2, L"\\\\?\\");
if...
if ( hFile == (HANDLE)-1i64 )
  return 0x80070057;
if ( GetFinalPathNameByHandleW(hFile, szFilePath, 0x208u, 0) - 1 > 0x207 )
{
  result = GetLastError();
  if ( result > 0 )
    return (unsigned __int16)result | 0x80070000;
}
else
{
  if ( _wcsnicmp(a1, String2, 4ui64) )
    a1 = ConvertFullPathToLongUNC(a1);
  v5 = wcsnlen(a1, 0x208ui64);
  v6 = wcsnlen(szFilePath, 0x208ui64);
  v7 = szFilePath;
  if ( v6 >= v5 )
    v7 = a1;
  v8 = wcsnlen(v7, 0x208ui64);
  if ( _wcsnicmp(szFilePath, a1, v8) )
  {
    if...
    return 0x800700A1;
  }
  if ( !GetFileInformationByHandle(hFile, &FileInformation) )
    return 0x800700A1;
  if ( FileInformation.nNumberOfLinks > 1 )
  {
    if...
    return 0x800700A1;
  }
  return 0;
}
return result;
```

2020年的PrintDemon

CVE-2020-17001 Windows Print Spooler Elevation of Privilege Vulnerability

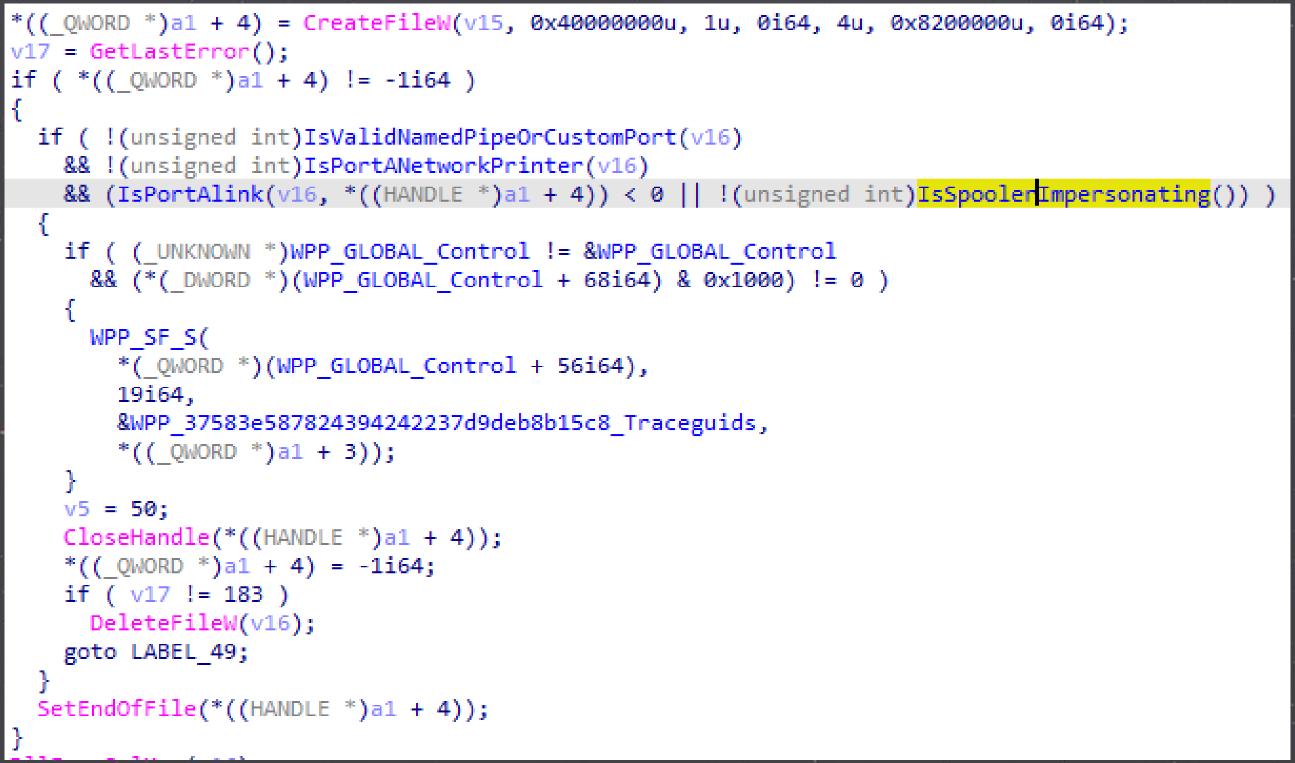


```
if ( GetFinalPathNameByHandleW(hFile, szFilePath, 0x208u, 0) - 1 > 0x207 )
{
  result = GetLastError();
  if ( result > 0 )
    result = (unsigned __int16)result | 0x80070000;
}
```

\\localhost\admin$\system32\tasks\test\test.dll

2020年的PrintDemon

CVE-2020-17001 Windows Print Spooler Elevation of Privilege Vulnerability



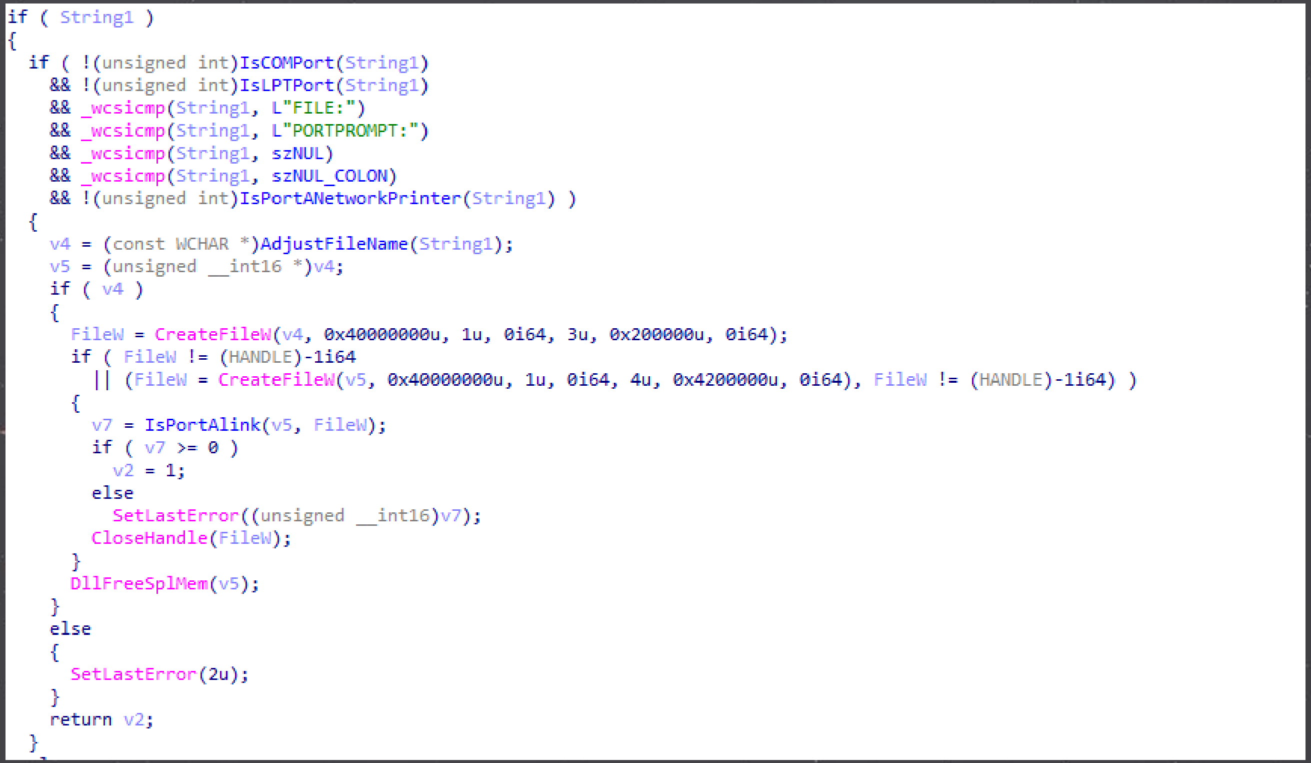
```
*((_QWORD *)a1 + 4) = CreateFileW(v15, 0x40000000u, 1u, 0i64, 4u, 0x8200000u, 0i64);
v17 = GetLastError();
if ( *((_QWORD *)a1 + 4) != -1i64 )
{
  if ( !(unsigned int)IsValidNamedPipeOrCustomPort(v16)
    && !(unsigned int)IsPortANetworkPrinter(v16)
    && (IsPortAlink(v16, *((HANDLE *)a1 + 4)) < 0 || !(unsigned int)IsSpoolerImpersonating()) )
  {
    if ( (_UNKNOWN *)WPP_GLOBAL_Control != &WPP_GLOBAL_Control
      && (*(_DWORD *)(WPP_GLOBAL_Control + 68i64) & 0x1000) != 0 )
    {
      WPP_SF_S(
        *(_QWORD *)(WPP_GLOBAL_Control + 56i64),
        19i64,
        &WPP_37583e587824394242237d9deb8b15c8_Traceguids,
        *((_QWORD *)a1 + 3));
    }
    v5 = 50;
    CloseHandle(*((HANDLE *)a1 + 4));
    *((_QWORD *)a1 + 4) = -1i64;
    if ( v17 != 183 )
      DeleteFileW(v16);
    goto LABEL_49;
  }
  SetEndOfFile(*((HANDLE *)a1 + 4));
}
```

第二部分

问题

漏洞修复的文件名校验

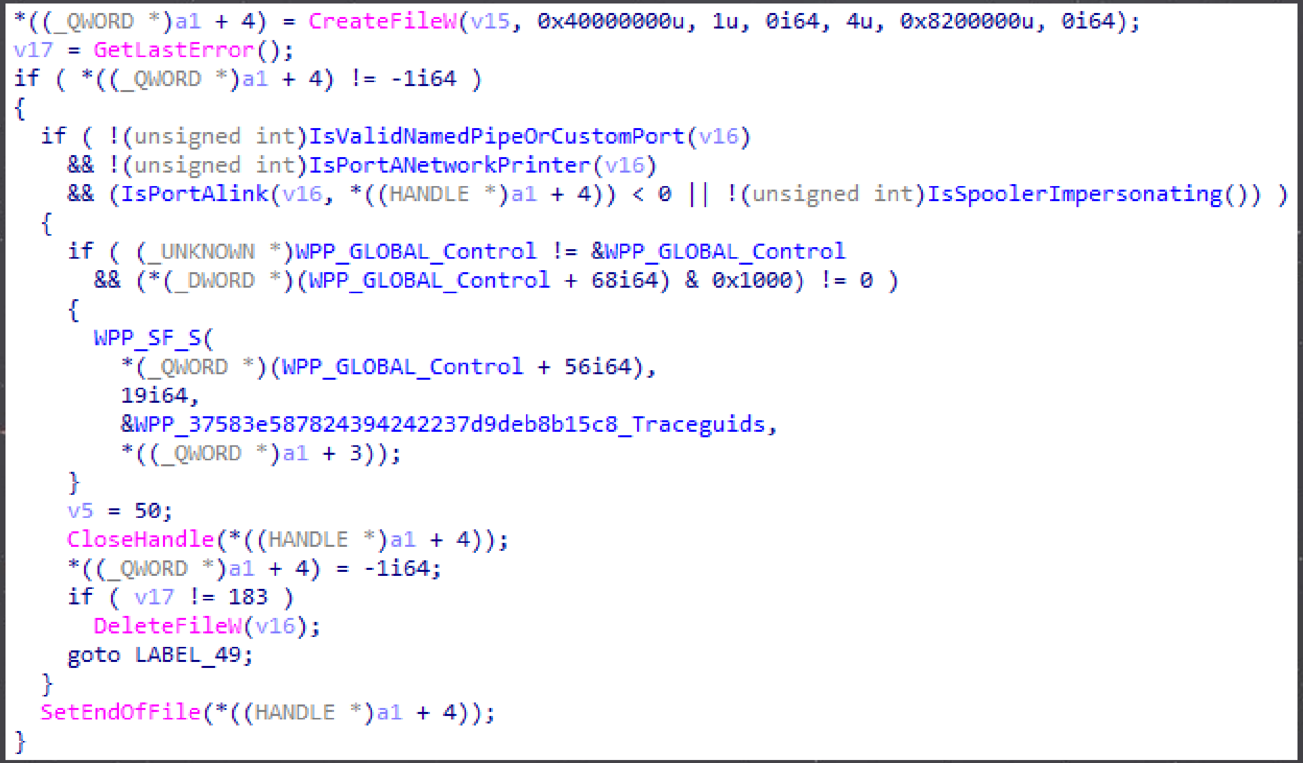
新建端口时的校验



```
if ( String1 )
{
  if ( !(unsigned int)IsCOMPort(String1)
    && !(unsigned int)IsLPTPort(String1)
    && _wcsicmp(String1, L"FILE:")
    && _wcsicmp(String1, L"PORTPROMPT:")
    && _wcsicmp(String1, szNUL)
    && _wcsicmp(String1, szNUL_COLON)
    && !(unsigned int)IsPortANetworkPrinter(String1) )
  {
    v4 = (const WCHAR *)AdjustFileName(String1);
    v5 = (unsigned __int16 *)v4;
    if ( v4 )
    {
      FileW = CreateFileW(v4, 0x40000000u, 1u, 0i64, 3u, 0x200000u, 0i64);
      if ( FileW != (HANDLE)-1i64
        || (FileW = CreateFileW(v5, 0x40000000u, 1u, 0i64, 4u, 0x4200000u, 0i64), FileW != (HANDLE)-1i64) )
      {
        v7 = IsPortAlink(v5, FileW);
        if ( v7 >= 0 )
          v2 = 1;
        else
          SetLastError((unsigned __int16)v7);
        CloseHandle(FileW);
      }
      DllFreeSplMem(v5);
    }
    else
    {
      SetLastError(2u);
    }
    return v2;
  }
```

漏洞修复的文件名校验

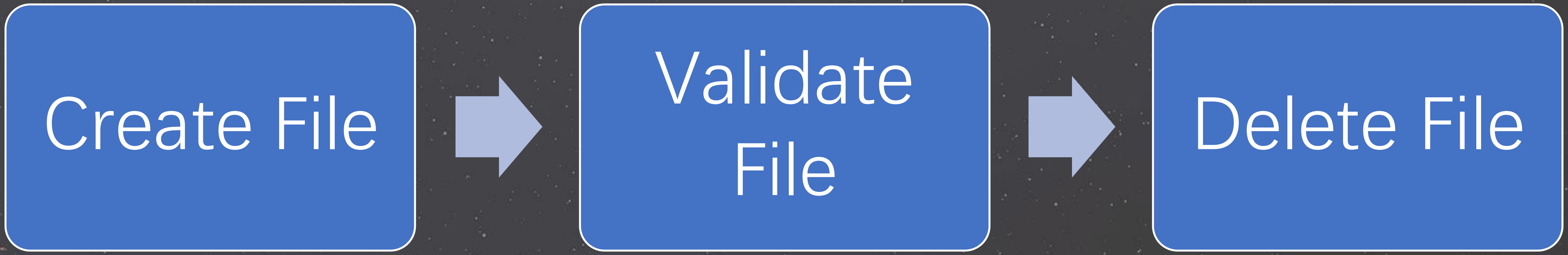
打印时的校验



```
*((_QWORD *)a1 + 4) = CreateFileW(v15, 0x40000000u, 1u, 0i64, 4u, 0x8200000u, 0i64);
v17 = GetLastError();
if ( *((_QWORD *)a1 + 4) != -1i64 )
{
  if ( !(unsigned int)IsValidNamedPipeOrCustomPort(v16)
    && !(unsigned int)IsPortANetworkPrinter(v16)
    && (IsPortAlink(v16, *((HANDLE *)a1 + 4)) < 0 || !(unsigned int)IsSpoolerImpersonating()) )
  {
    if ( (_UNKNOWN *)WPP_GLOBAL_Control != &WPP_GLOBAL_Control
      && (*(_DWORD *)(WPP_GLOBAL_Control + 68i64) & 0x1000) != 0 )
    {
      WPP_SF_S(
        *(_QWORD *)(WPP_GLOBAL_Control + 56i64),
        19i64,
        &WPP_37583e587824394242237d9deb8b15c8_Traceguids,
        *((_QWORD *)a1 + 3));
    }
    v5 = 50;
    CloseHandle(*((HANDLE *)a1 + 4));
    *((_QWORD *)a1 + 4) = -1i64;
    if ( v17 != 183 )
      DeleteFileW(v16);
    goto LABEL_49;
  }
  SetEndOfFile(*((HANDLE *)a1 + 4));
}
```

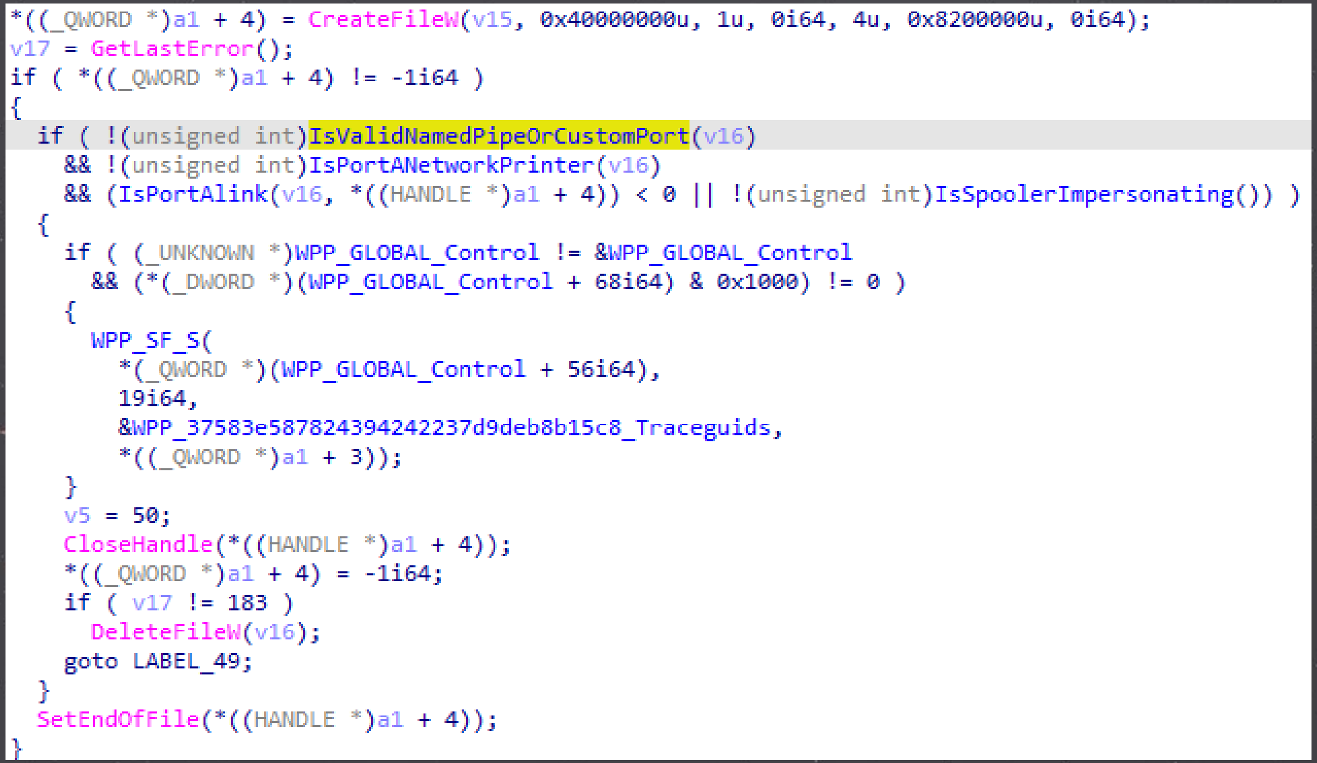
文件名校验的常见问题

问题1：不安全的文件删除



文件名校验的常见问题

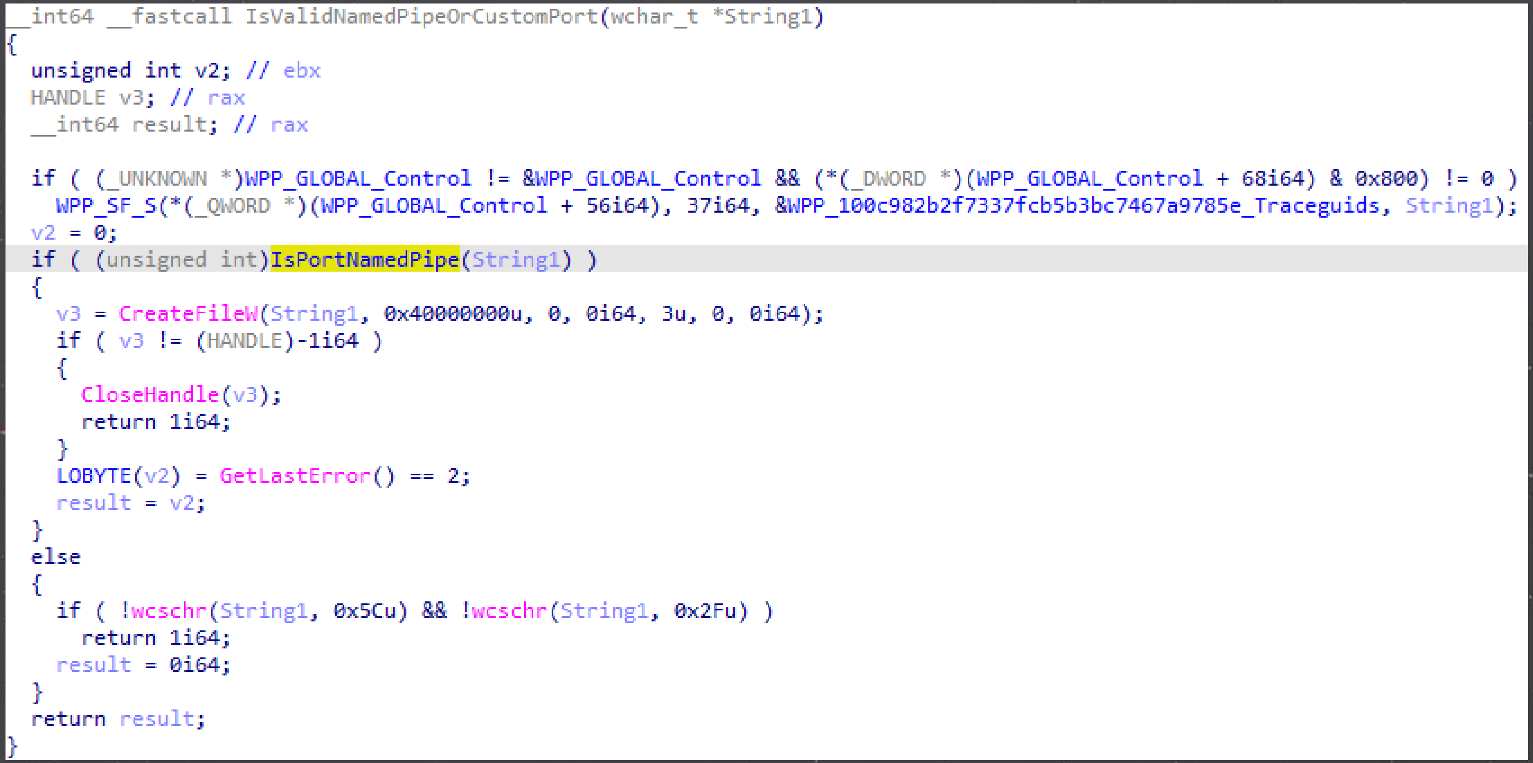
问题2：未覆盖所有的情况



```
*((_QWORD *)a1 + 4) = CreateFileW(v15, 0x40000000u, 1u, 0i64, 4u, 0x8200000u, 0i64);
v17 = GetLastError();
if ( *((_QWORD *)a1 + 4) != -1i64 )
{
  if ( !(unsigned int)IsValidNamedPipeOrCustomPort(v16)
    && !(unsigned int)IsPortANetworkPrinter(v16)
    && (IsPortAlink(v16, *((HANDLE *)a1 + 4)) < 0 || !(unsigned int)IsSpoolerImpersonating()) )
  {
    if ( (_UNKNOWN *)WPP_GLOBAL_Control != &WPP_GLOBAL_Control
      && (*(_DWORD *)(WPP_GLOBAL_Control + 68i64) & 0x1000) != 0 )
    {
      WPP_SF_S(
        *(_QWORD *)(WPP_GLOBAL_Control + 56i64),
        19i64,
        &WPP_37583e587824394242237d9deb8b15c8_Traceguids,
        *((_QWORD *)a1 + 3));
    }
    v5 = 50;
    CloseHandle(*((HANDLE *)a1 + 4));
    *((_QWORD *)a1 + 4) = -1i64;
    if ( v17 != 183 )
      DeleteFileW(v16);
    goto LABEL_49;
  }
  SetEndOfFile(*((HANDLE *)a1 + 4));
}
```

文件名校验的常见问题

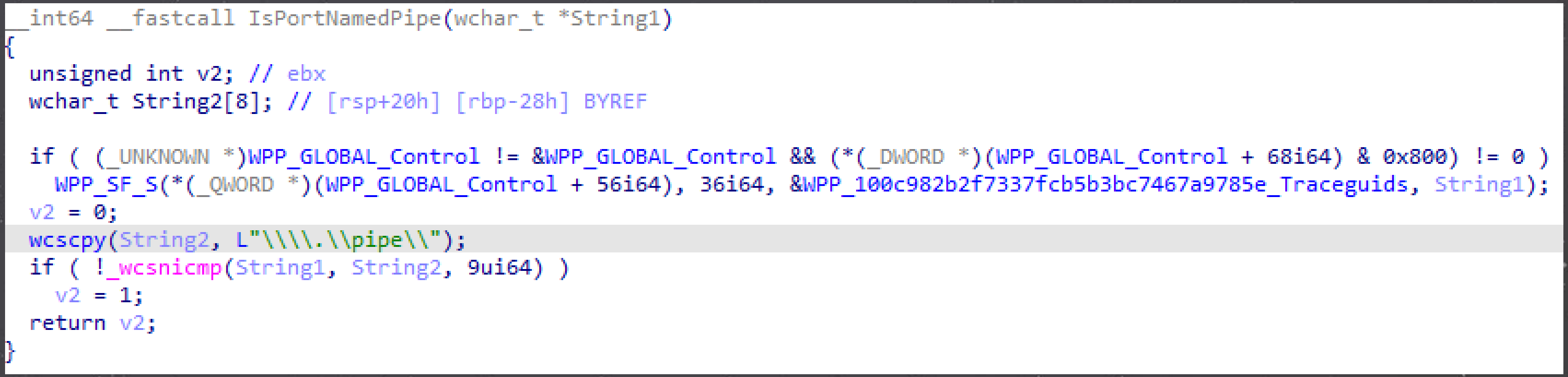
问题2：未覆盖所有的情况



```
__int64 __fastcall IsValidNamedPipeOrCustomPort(wchar_t *String1)
{
  unsigned int v2; // ebx
  HANDLE v3; // rax
  __int64 result; // rax

  if ( (_UNKNOWN *)WPP_GLOBAL_Control != &WPP_GLOBAL_Control && (*(_DWORD *)(WPP_GLOBAL_Control + 68i64) & 0x800) != 0 )
    WPP_SF_S(*(_QWORD *)(WPP_GLOBAL_Control + 56i64), 37i64, &WPP_100c982b2f7337fcb5b3bc7467a9785e_Traceguids, String1);
  v2 = 0;
  if ( (unsigned int)IsPortNamedPipe(String1) )
  {
    v3 = CreateFileW(String1, 0x40000000u, 0, 0i64, 3u, 0, 0i64);
    if ( v3 != (HANDLE)-1i64 )
    {
      CloseHandle(v3);
      return 1i64;
    }
    LOBYTE(v2) = GetLastError() == 2;
    result = v2;
  }
  else
  {
    if ( !wcschr(String1, 0x5Cu) && !wcschr(String1, 0x2Fu) )
      return 1i64;
    result = 0i64;
  }
  return result;
}
```

文件名校验的常见问题

问题2：未覆盖所有的情况



```
__int64 __fastcall IsPortNamedPipe(wchar_t *String1)
{
  unsigned int v2; // ebx
  wchar_t String2[8]; // [rsp+20h] [rbp-28h] BYREF

  if ( (_UNKNOWN *)WPP_GLOBAL_Control != &WPP_GLOBAL_Control && (*(_DWORD *)(WPP_GLOBAL_Control + 68i64) & 0x800) != 0 )
    WPP_SF_S(*(_QWORD *)(WPP_GLOBAL_Control + 56i64), 36i64, &WPP_100c982b2f7337fcb5b3bc7467a9785e_Traceguids, String1);
  v2 = 0;
  wcscpy(String2, L"\\\\.\\pipe\\");
  if ( !_wcsnicmp(String1, String2, 9ui64) )
    v2 = 1;
  return v2;
}
```

文件名校验的常见问题

问题3：基于输入的文件名做判断

```
\\.\pipe\ABC
\\.\pipe\ABC\XYZ
\\.\pipe\ABC\..\XYZ
\\.\pipe\ABC\..\..\XYZ
\\.\pipe\ABC\..\..\C:\XYZ
```

第三部分

修复

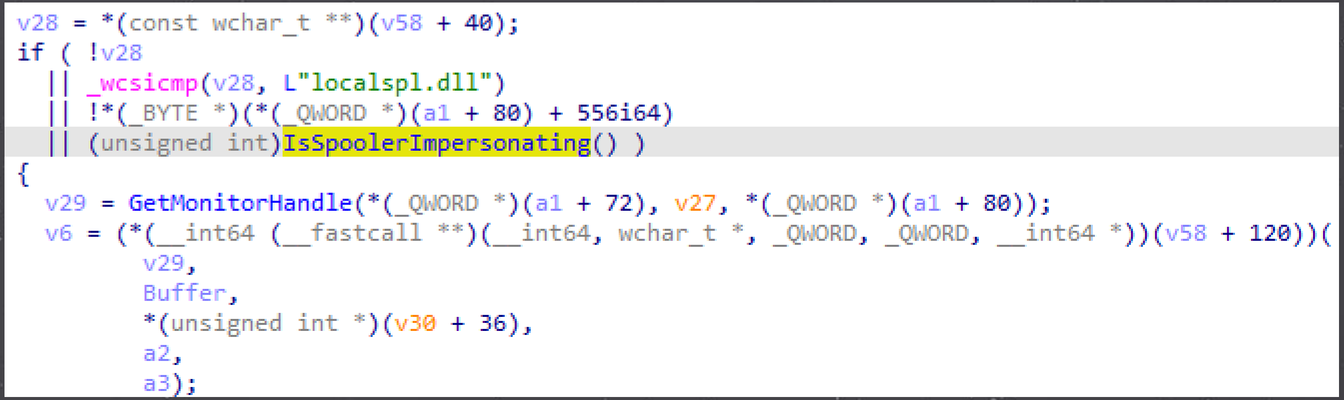
完善的修复方案

PrintDemon系列漏洞的本质问题

模拟SYSTEM用户进行文件操作

完善的修复方案

在PrintingDirectlyToPort 函数中对模拟状态进行检查

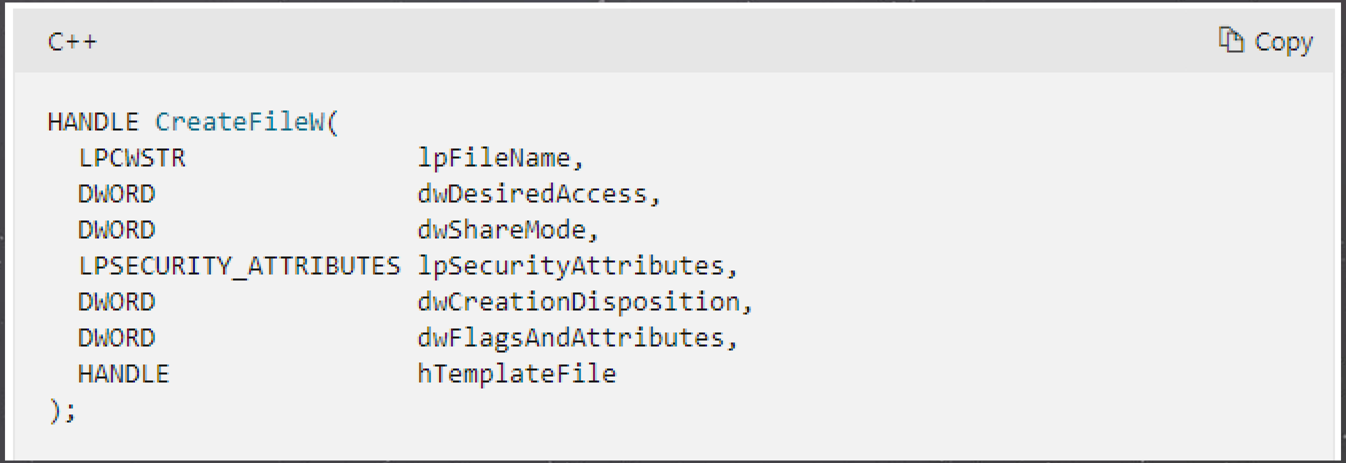


```
v28 = *(const wchar_t **)(v58 + 40);
if ( !v28
  || _wcsicmp(v28, L"localspl.dll")
  || !*(_BYTE *)(*(_QWORD *)(a1 + 80) + 556i64)
  || (unsigned int)IsSpoolerImpersonating() )
{
  v29 = GetMonitorHandle(*(_QWORD *)(a1 + 72), v27, *(_QWORD *)(a1 + 80));
  v6 = (*(__int64 (__fastcall **)(__int64, wchar_t *, _QWORD, _QWORD, __int64 *))(v58 + 120))(
         v29,
         Buffer,
         *(unsigned int *)(v30 + 36),
         a2,
         a3);
```

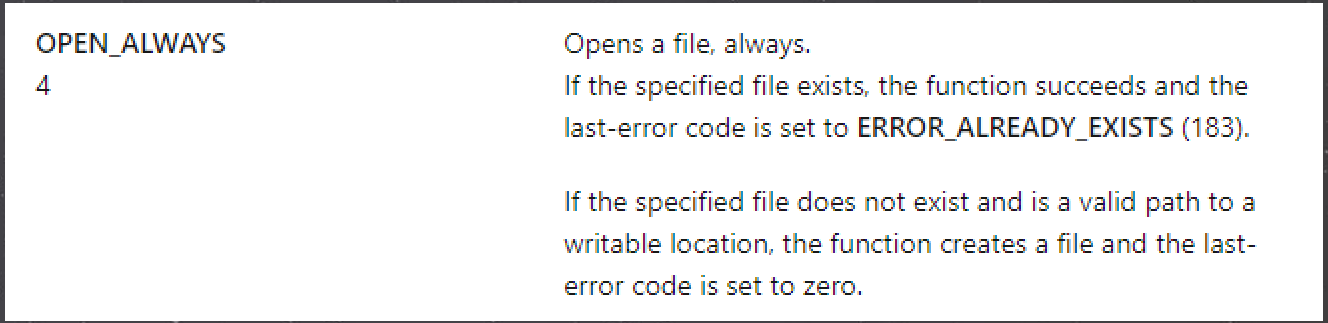
第四部分

建议

如何正确的校验文件名

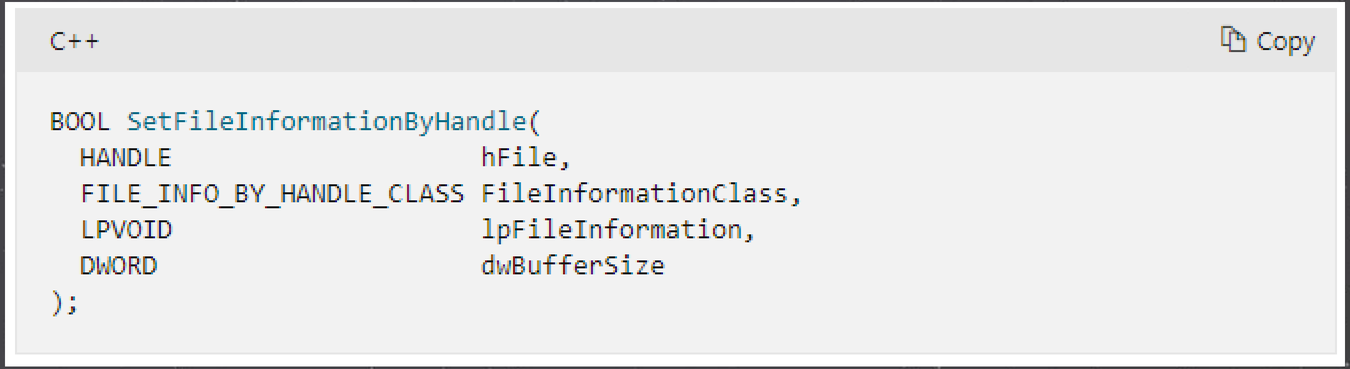

C++ Copy

```
HANDLE CreateFileW(
  LPCWSTR               lpFileName,
  DWORD                 dwDesiredAccess,
  DWORD                 dwShareMode,
  LPSECURITY_ATTRIBUTES lpSecurityAttributes,
  DWORD                 dwCreationDisposition,
  DWORD                 dwFlagsAndAttributes,
  HANDLE                hTemplateFile
);
```



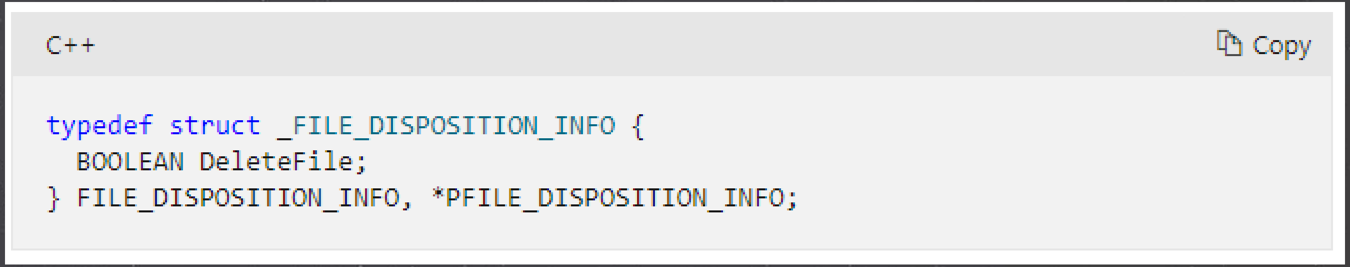
OPEN_ALWAYS 4	Opens a file, always. If the specified file exists, the function succeeds and the last-error code is set to **ERROR_ALREADY_EXISTS** (183). If the specified file does not exist and is a valid path to a writable location, the function creates a file and the last-error code is set to zero.

如何正确的校验文件名



C++ Copy

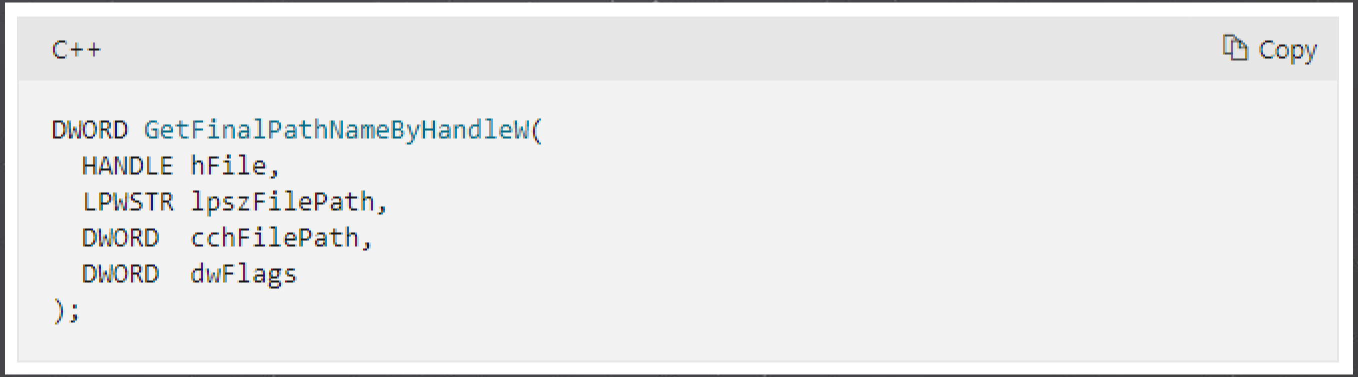
```cpp
BOOL SetFileInformationByHandle(
  HANDLE                    hFile,
  FILE_INFO_BY_HANDLE_CLASS FileInformationClass,
  LPVOID                    lpFileInformation,
  DWORD                     dwBufferSize
);
```



C++ Copy

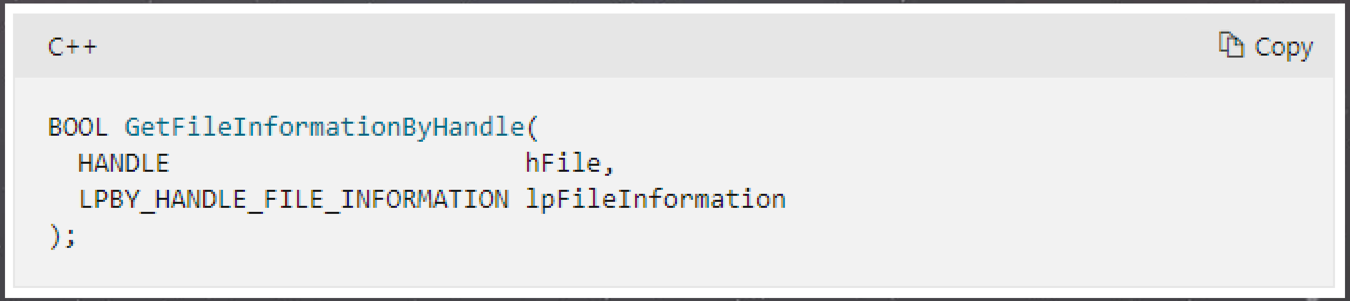
```cpp
typedef struct _FILE_DISPOSITION_INFO {
  BOOLEAN DeleteFile;
} FILE_DISPOSITION_INFO, *PFILE_DISPOSITION_INFO;
```

如何正确的校验文件名

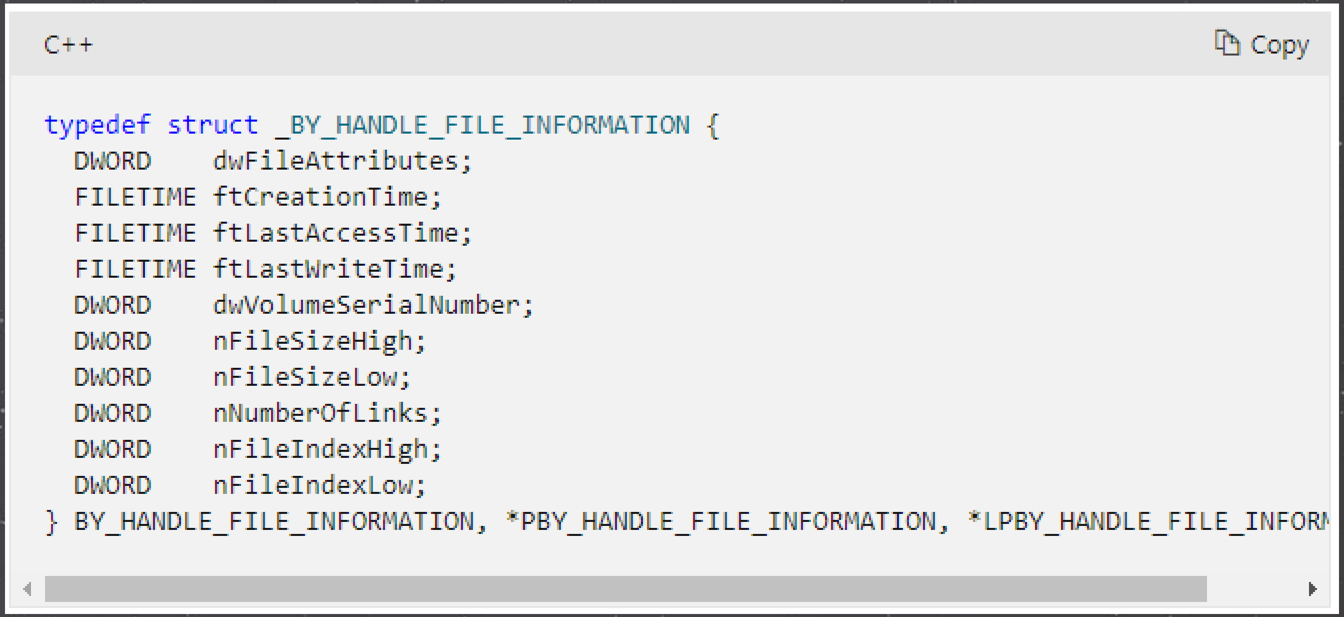

C++ Copy

```cpp
DWORD GetFinalPathNameByHandleW(
  HANDLE hFile,
  LPWSTR lpszFilePath,
  DWORD  cchFilePath,
  DWORD  dwFlags
);
```

如何正确的校验文件名

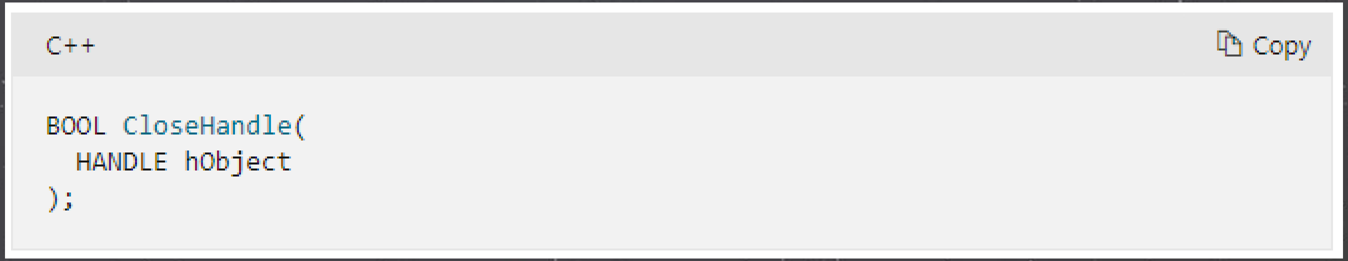


```cpp
BOOL GetFileInformationByHandle(
  HANDLE                       hFile,
  LPBY_HANDLE_FILE_INFORMATION lpFileInformation
);
```



```cpp
typedef struct _BY_HANDLE_FILE_INFORMATION {
  DWORD    dwFileAttributes;
  FILETIME ftCreationTime;
  FILETIME ftLastAccessTime;
  FILETIME ftLastWriteTime;
  DWORD    dwVolumeSerialNumber;
  DWORD    nFileSizeHigh;
  DWORD    nFileSizeLow;
  DWORD    nNumberOfLinks;
  DWORD    nFileIndexHigh;
  DWORD    nFileIndexLow;
} BY_HANDLE_FILE_INFORMATION, *PBY_HANDLE_FILE_INFORMATION, *LPBY_HANDLE_FILE_INFOR
```

如何正确的校验文件名


C++

Copy

```
BOOL CloseHandle(
  HANDLE hObject
);
```

20TH ANNUAL

安全焦点信息安全技术峰会

XFOCUS INFORMATION SECURITY CONFERENCE

重拾圈点·致敬无限

THANKS