

Evasive Malware Campaign Hits in Inbox with Faked HM Revenue and Customs Attachment

INTRODUCTION

On **December 12, 2016** Morphisec identified and monitored a new wave of sophisticated malware delivered via targeted phishing emails with malicious macro-based documents attached. The malicious documents themselves use a clever, new social engineering technique to convince the target to enable macros. Once enabled, the document calls an unknown downloader whose evasion techniques has resemblance to the Cerber downloader, but employs new obfuscation techniques.

This report analyzes the document attack chain. It is part of a [series](#) of reports produced by Morphisec Lab focusing on the most evasive and sophisticated in-memory malware.

Still Can't Evade Morphisec

The campaign recently identified by Morphisec uses several interesting modified evasive techniques and had an extremely low detection ratio on VirusTotal of only 1/ 54.

Despite the brand-new elements and variations, Morphisec's Endpoint Threat Prevention stops this sophisticated attack at its first encounter without the need of changing any rules.



SHA256: d526f134d2fd0bc1bf29b8c0f684a3c518a71c0f3066a85bf54fe9c9457bf39d
File name: Statement of Liabilities.doc
Detection ratio: 1 / 54
Analysis date: 2016-12-12 14:38:27 UTC (1 day, 22 hours ago) [View latest](#)



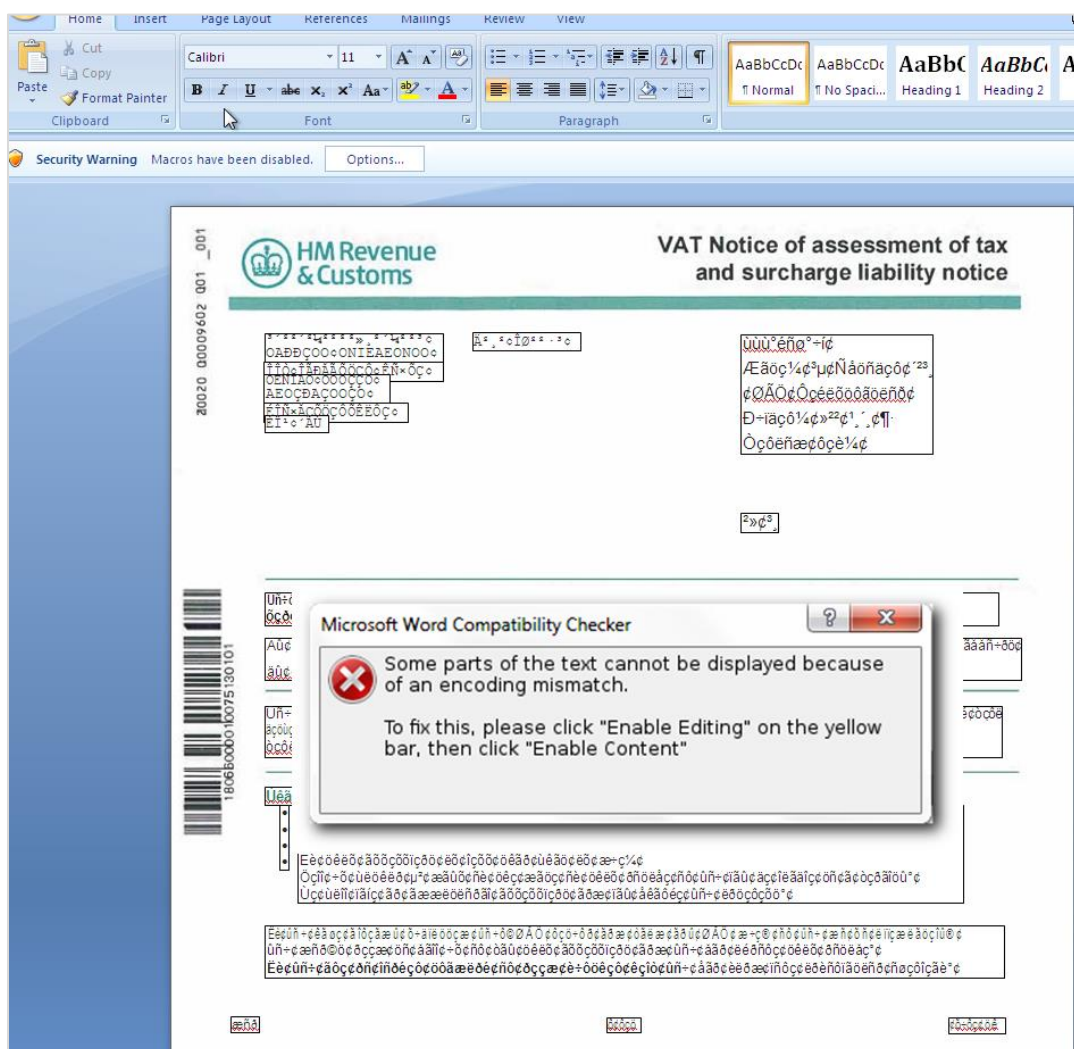
Core Characteristics

- Attempts to detect and bypass traditional defenses.
- Runs interesting EXE version downloaded from a URL and executes it
 - Resources partially taken from the AIMP audio player
- Uses Madshi (a popular productional injection and hooking library used by many security vendors – very similar to Microsoft Detours)
- Injects malicious code into explorer.exe

TECHNICAL ANALYSIS

1. Macro Documents

- 1.1. **Malicious macro WORD documents sent to the targeted victims.** Note the clever new social engineering method to convince people to enable the macro.



1.2. **Machine Check Anti-Sandboxing:** The malicious macro WORD document checks count of recent documents > 0 to verify that this isn't a sandbox environment.

```

ogvatyhl = Application.Documents.Count > 0
kbcumulad9 = "Application.Documents.Count = 1
aldily = "ILOE "
    
```

1.3. **Core capabilities in the macro:**

- First, it makes sure that it isn't in a sandbox (count of recent documents > 0)
- Second, runs the obfuscated string with special character "^" between the chars – PowerShell code downloads executable payload and runs it.
- To remove suspicious activity, the document replaces the original intro into "Microsoft Word has encountered a problem and needs to close"

The screenshot shows the macro editor for a document. The macro code is as follows:

```

yrcaxve = "xcu"
bovyzo = ltyce9 & umytt3 & ibujru3 & edow & tqokwewq9 & egmeni7 & ngezupgogn5 & qudyrk6 & yrcaxve & anviba & awifte & romyxy & yvaxibu & itoero & wycuzco & inda
ihyosymr = ytcoodexa & axysynti & ftyny7 & zbehjyl & tigga & ombys9 & wufokwi & xarfylf2 & rqumafqa & xdosje & ekvumel & nuqctynf & azlyvt4 & novdosj & kbcumulad

If ogvatyhl Then
  Shell bovyzo, 0
  ActiveDocument.Content.Text = ihyosymr
End If
End Sub
Sub AutoOpen()
  ydcahogr
End Sub
    
```

Below the code, a message box is visible with the text: "Microsoft Word has encountered a problem and needs to close".

- Finally, the downloaded executable

#	Result	Protocol	Host	URL	Body	Caching	Content-Type	Process
1	200	HTTP	santiagoveraguas.com	/doc.exe	193,536	max-ag...	application/x-msdownload	powershell:3524

The screenshot shows the response details for the downloaded executable. The headers are as follows:

```

HTTP/1.1 200 OK..Server: ngi
nx/1.10.2..Date: Thu, 15 Dec
2016 12:20:56 GMT..Content-
Type: application/x-msdownlo
ad..Content-Length: 193536..
Connection: keep-alive..Last
-Modified: Tue, 13 Dec 2016
11:28:52 GMT..Accept-Ranges:
bytes..Cache-Control: max-a
ge=0..Expires: Thu, 15 Dec 2
016 12:20:56 GMT..Vary: Acce
pt-Encoding, User-Agent...M2
.....@
.....Í!..L!This pro
gram cannot be run in DOS mo
de.....+iã+xiã+xiã
+xã.bkÿã+xã.tx.ã+xãE".xõã+xiã
õx}ã+xã.sxiã+xõfciã+xã.fxã
+xRichiã+x.....PE..L..YÓ
OX.....à.....
.....s.....@
.....
.....iã
    
```

2. The Executable

Morphisec observed different payloads downloaded by similar documents within a very short time period. We focus here on the most interesting executable malware.

2.1. **IMPORTS: Implicit imports** reminding methods to bypass sandboxes, debugging or reversing:

- Anti-debug (Kernel32.dll):
 1. GetTickCount
 2. TerminateProcess
 3. UnhandledExceptionFilter
 4. IsDebuggerPresent
- Anonymous(Ws2_32.dll,odbc32.dll):
 1. 167
 2. 170
 3. 166
 4. 115
- Others (kernel32.dll,user32.dll,ole32.dll,secur32.dll,rpcrt4.dll):
 1. LoadLibraryW
 2. LoadLibraryA
 3. GetProcAddress
 4. CreateToohelp32Snapshot
 5. Process32First
 6. Process32Next
 7. GetEnvironmentStringsW
 8. GetModuleFileNameA
 9. VirtualAlloc
 10. Sleep
 11. HeapCreate
 12. WriteFile
 13. SendMessageA
 14. CoInitialize – **use of a com objects to run a shellcode**
 15. CoCreateInstance
 16. QueryCredentialsAttributesA
 17. EnumerateSecurityPackagesA
 18. AcquireCredentialsHandleA

2.2. RESOURCES:

Delphi file:

1. Type: RCData
2. Name: **TAIMP****OPTIONS****SPAMEPLAYLIST****BH****HA****VIOR**
3. Size: 2976 byte


This is part of Aimp3.exe of AIMP3 developed by AIMP DevTeam. AIMP is a freeware audio and video player for windows and Android, originally developed by a Russian developer.

What makes AIMP unique is its ability to load the entire media file into the RAM of the computer.

Strings:

- GetLastActivePopup
- &Save capture
- &open
- &Exit
- &File
- Could not initialize winsock
- Unable to initialize heap
- Attempt to initialize the CRE more than once
- www.madshi.net – (open source to inject or hook)

On uploading the malicious executable to Hybrid-analysis.com, we received following risk assessment:

 **Risk Assessment**

Spyware	POSTs files to a webserver
Persistence	Injects into explorer Modifies System Certificates Settings Persists itself using auto-execute at a hidden registry location
Fingerprint	Reads the active computer name Reads the cryptographic machine GUID
Evasive	Tries to sleep for a long time (more than two minutes)
Network Behavior	Contacts 2 domains and 3 hosts. View the network section for more details.

File details: Based on Kerish Products [Kerish Doctor 2015, a Windows maintenance tool]

property	value
file-type	executable
date	n/a
language	English United States
code-page	Unicode UTF-16, little endian
CompanyName	Kerish Products
FileDescription	Kerish Doctor 2015 v4.60
FileVersion	4.60.0.0
LegalCopyright	◆ Kerish Products
ProductName	Kerish Doctor 2015 v4.60
ProductVersion	4.60.0.0

Summary of malicious and suspicious indicators taken from Hybrid-analysis.com and ThreatInfo (form December 15, 2016):

- Tries to locate where the browsers are installed
- One or more potentially interesting buffers were extracted, these generally contain injected code, configuration data, etc.
- Performs some HTTP requests
- Allocates read-write-execute memory (usually to unpack itself)
- Creates executable files on the filesystem
- One or more of the buffers contains an embedded PE file
- **Creates an Alternate Data Stream (ADS)**
- Detects virtualization software with SCSI Disk Identifier trick(s)
- Executed a process and injected code into it, probably while unpacking
- Connects to IP addresses that are no longer responding to requests (legitimate services will remain up-and-running usually)

3. The Shellcode

```

loc_4031FF:
mov     eax, [ebp+var_48]
cmp     dword_414258, eax
jbe     short loc_40320F

loc_403200:
mov     ebx, [ebp+nShowCmd]
call    ebx

loc_40320F:
mov     eax, [ebp+var_40]
mov     eax, [ebp+eax*4+var_22C]
pop     edi
nop     esi

0002600 0040320D: WinMain(x,x,x,x)+2154 (Synchronized with
Hex View-1
004031E0 35 40 42 41 00 8D 85 68 F5 FF FF 50 E8 5F 00 00 50BA..àh)..PF..
004031F0 00 C7 45 14 00 10 40 00 85 C0 75 03 89 7D 14 8B .!E...@.à+u.ë}.ÿ
00403200 45 B8 39 05 58 42 41 00 76 05 8B 5D 14 FF D3 8B E+9.XBA.v.ÿ]..+ÿ
00403210 45 C0 8B 84 85 D4 FD FF FF 5F 5E A3 40 42 41 00 E+ÿÿà+²..^ú@BA.
00403220 5B C9 C2 10 00 CC FF 25 94 F0 40 00 FF 25 90 F0 [+-.!·%ö=@.·%.=

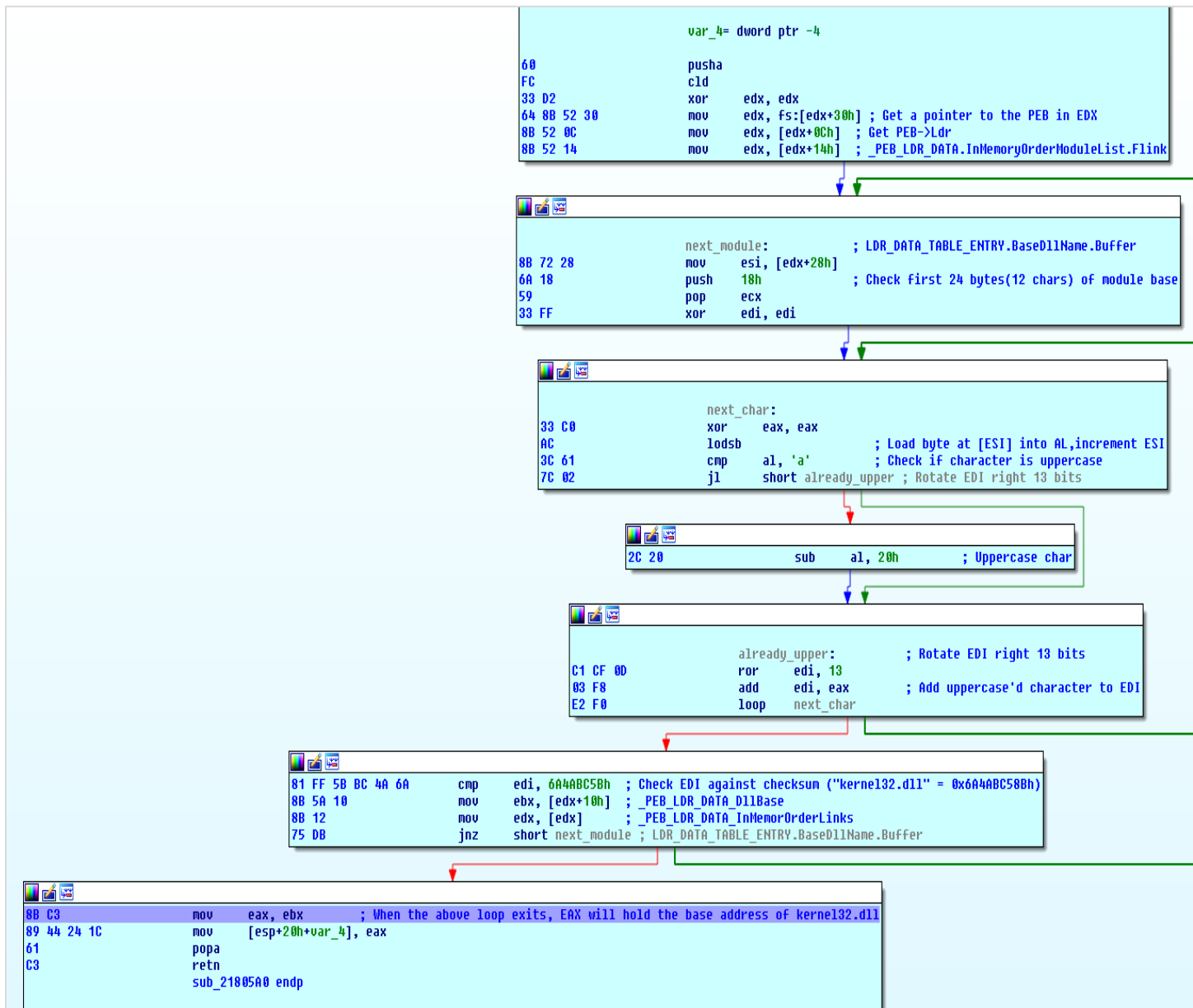
00002600 00403200: WinMain(x,x,x,x):loc_4031FF+1

Output window
Unloaded C:\Windows\system32\COMDLG32.DLL
Unloaded C:\Windows\system32\COMDLG32.DLL
PDBSRC: loading symbols for 'C:\Users\Roy\Desktop\13.12\doc_third.exe'...
Expected data back.
WINDBG>u ebx
020e0590 b800ff1200 mov     eax,12FF00h
020e0595 e976140000 jmp     020e1a10
020e059a cc      int     3
020e059b cc      int     3
020e059c cc      int     3
020e059d cc      int     3
020e059e cc      int     3
020e059f cc      int     3
  
```

Ndll and adavapi Libraries:

```
var_0 byte ptr 0
mov     ebp, esp
sub     esp, 4Ch
call   near ptr unk_20E07D0
mov     [ebp+var_2C], eax
call   near ptr unk_20E05A0
mov     [ebp+var_18], eax
push   0B1866570h
mov     eax, [ebp+var_18]
push   eax
call   near ptr unk_20E0670
mov     [ebp+var_20], eax
mov     ecx, [ebp+var_2C]
mov     edx, [ebp+var_20]
mov     [ecx], edx
mov     [ebp+var_8], 'n'
mov     [ebp+var_7], 't'
mov     [ebp+var_6], 'd'
mov     [ebp+var_5], 'l'
mov     [ebp+var_4], 'l'
mov     [ebp+var_3], 0
mov     [ebp+var_14], 'a'
mov     [ebp+var_13], 'd'
mov     [ebp+var_12], 'u'
mov     [ebp+var_11], 'a'
mov     [ebp+var_10], 'p'
```


Main Sophisticated Shellcode Technique



This function uses the PEB (Process Environment Block) of the current process (stored at fs:[30h]) to locate a linked list of loaded modules. The PEB contains a member called Ldr that is a pointer to a PEB_LDR_DATA structure.

The InMemoryOrderModuleList linked list is used to enumerate the loaded modules by the order that they're loaded in memory.

The technique of enumerating the loaded modules using the PEB is commonly used by malware to locate the base address (which is also the module handle) of certain DLL files to avoid declaring and using GetModuleHandle/LoadLibrary API call implicitly.

There are a number of reasons for the attackers to apply such a bypass:

- Security solutions and generic unpackers hook calls to GetModuleHandle/LoadLibrary
- The address of GetModuleHandle/LoadLibrary is unknown unless declared implicitly - this usually happen if the code doesn't know what context it's running in, for example in **shellcode and injected threads**
- To make static analysis of the code more difficult for researcher.

CONCLUSION:

The macro-based evasion techniques used in this document have a very low detection ratio, demonstrating how most security solutions are always a few steps behind the attacker and how static based solutions cannot cope with today's rapid changes in attack scenarios. Social engineering methods are also constantly evolving, as we see in the new technique used to lure victims into activating the attack. This makes it difficult to educate users to identify phishing mail. More and more file-less or partially file-less attacks like this one are cropping up, which easily evade file-based solutions by persisting in memory.

HASHs:

Document "Statement of Liabilities.doc" –
d526f134d2fd0bc1bf29b8c0f684a3c518a71c0f3066a85bf54fe9c9457bf39d

Downloader "doc.exe"- 3128b0461d4d36a8763da1685d226fae5594364947c6bdff026eebcbde97a1cc