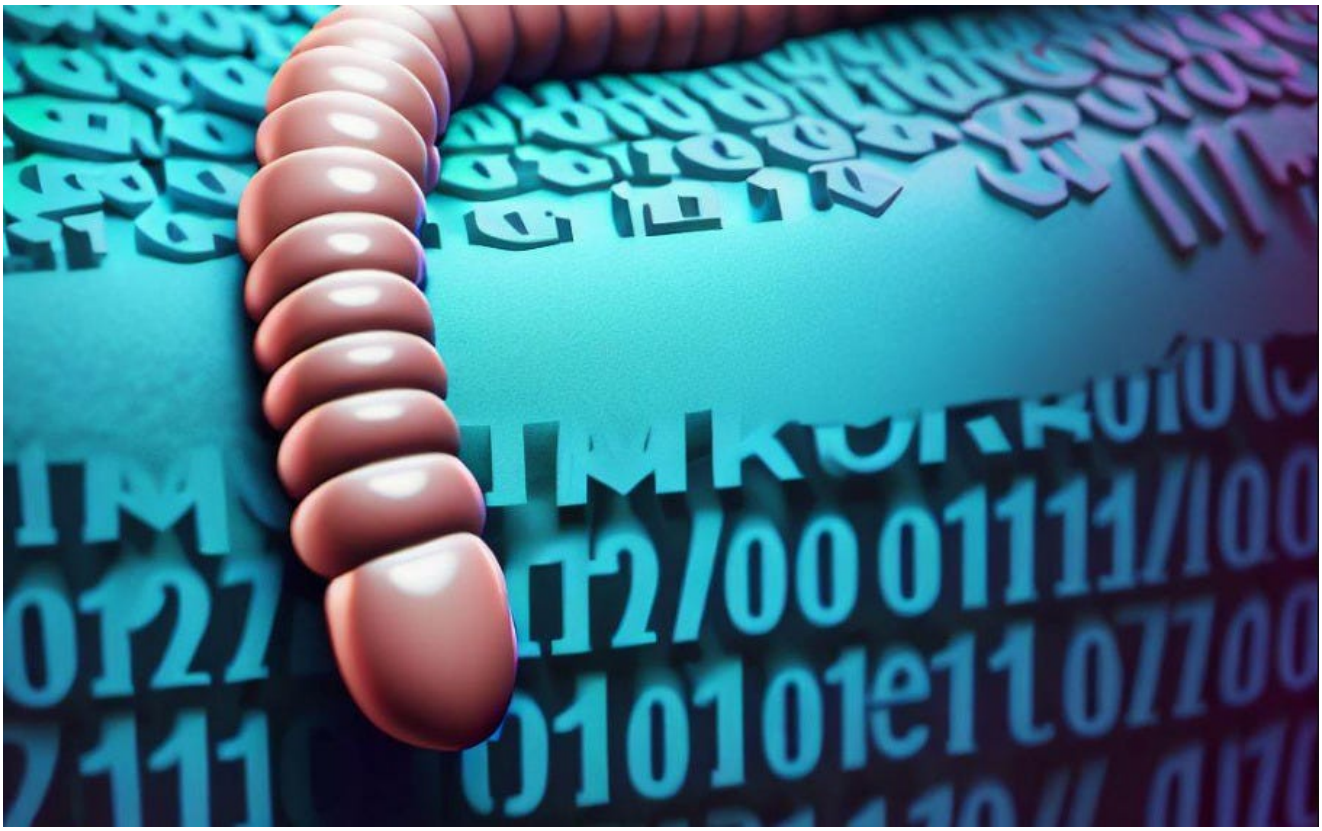
Malware Analysis — Xworm

medium.com/@b.magnezi/malware-analysis-xworm-80b3bbb072fb

0xMrMagnezi February 22, 2024



0xMrMagnezi

--

Xworm is a type of malware that spreads through removable drives , such as USB flash drives , infecting Windows systems and potentially stealing information or allowing remote access.

Malware Bazaar Sample

As always, this sample has been downloaded and extracted using the password ‘infected.’ At first glance, it seemed like a simple deobfuscation task, but I later found out that it is a multi-stage malware.

Stage1 — Obfuscated VBS code

This seemed like a simple deobfuscation where the first function returns the character associated with the specified ASCII code. In this example, it was the letter ‘e’, which is used to build the final URL and execute the context of that URL.

Stage1 — Deobfuscated VBS code

CURL to get that Stage 2

The second stage appeared more obfuscated than the first. I noticed keywords such as “WScript” , “-window” , and what seemed like “execution policy bypass”.

Stage 2 — Obfuscated

Given the length of that file, I have decided that it will be more effective to launch the code and check for new processes.

ProcMon — PowerShell was executed under the VBS

First PowerShell code

Second PowerShell code

It seemed like the second PowerShell script is directly related to the first one , essentially being the decoded version of the first. After some cleaning of the code it was pretty clear what this code does.

Clean PowerShell code

This PowerShell script downloads data from a list of URLs. It searches for a section in the downloaded data enclosed by <<BASE64_START>> and <<BASE64_END>> flags , assuming this section is base64-encoded data. The script decodes and loads this data as a .NET assembly , expecting it to contain a method named “VAI” that is then invoked with specific parameters. I once again used curl to access these URLs and observe their content.

CURL to the given URLs

`"FF D8"` indicates the start of a JPEG file

Based on its content and the file header, I assumed that it’s a genuine JPEG. I then started looking for the flags <<BASE64_START>> and <<BASE64_END>> , and indeed , they were found at the end of the file. This technique is called **steganography** , which involves embedding content within a file.

Finding the Base64 embedded within the JPEG

Decoding the Base64 using CyberChef

The decoding was successful , resulting in a new EXE file. This file , written in .NET , was then opened in DNSPY for debugging. Inside , a persistence technique was discovered , where the program adds itself to a registry key that runs it every time the system starts up.

Persistence technique

The main Function — “VAI” , that is being invoked by the previous PS

It’s the downloader function. It reverses the first string it receives and attempts to download its content.

Downloading from the URL

Reversing and decoding From Base64 — Output is EXE once again

This is the actual Xworm. Inside , there were some functions related to keyloggers , persistence , sockets etc.

Key Logger functions (GetKeyState / GetKeyboardLayout etc..)

Adding itself to AppData and Startup folder

Decrypting its doing

Information about the system being sent over Socket

IOCs:

- hxxps://uploaddeimagens[.]com[.]br/images/004/739/227/original/new_image.jpg?1707826222
- hxxp://45[.]74[.]19[.]84/xampp/bkp/vbs_novo_new_image[.]jpg
- hxxps://wayoutkwt[.]com/bk/fexrw[.]txt
- hxxps://pastebin[.]com/raw/G9zY5tnh
- febxworm39090[.]duckdns[.]org
- 45[.]74[.]19[.]84
- 85673635.vbs — 032c2cc1862303d06832c0ebe34b9dae
- PROJETOAUTOMACAO.VB1.dll — ce91eb459e4f6a9e2871088d855cd211
- febxworm39090.exe — 1a2b23fd06525561826e61fc104b66d0
- vbs_novo_new_image.jpg — ddb09774c5a870c73cf0cf71e6d97d3e
- output.273230999.txt — 983865b130c91b9b3a36d488afbdd1bd

In conclusion , Xworm is a sophisticated malware targeting Windows systems. It employs advanced evasion techniques like polymorphism , multi-staging and anti-debugging mechanisms , making detection and analysis challenging. Xworm also uses steganography to conceal its malicious code within innocuous files , further complicating its identification.