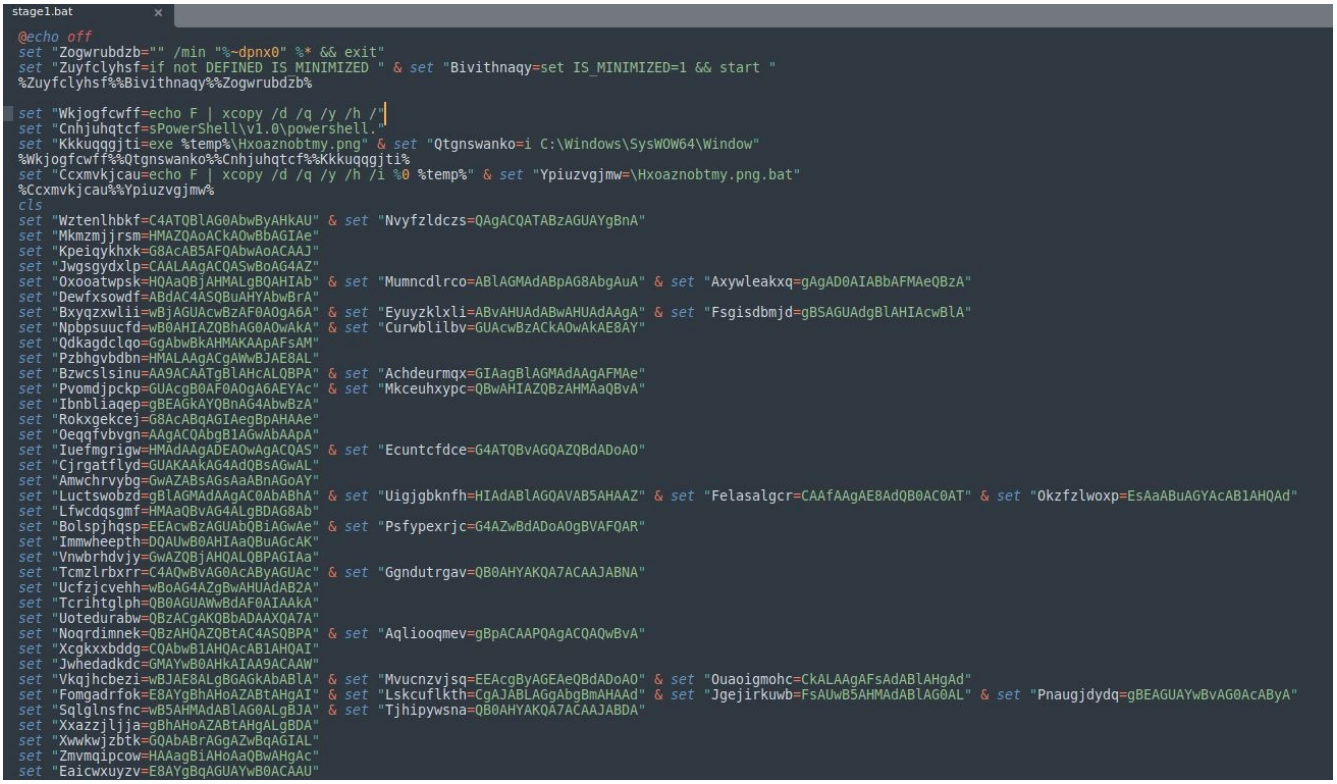
Malware Analysis October 04, 2023
File: client.exe
Source: Malware Bazaar (abuse.ch)
MD5: 4d1fe0a1752f2405de5802aef37be7e4
Sha1: 7b0f523763798939b510f29437aec76e8bf7388c
Virustotal: 26/60
Malicious batch file
Originally detected Oct 2, 2023

This file is a windows bat file. These are simple script files used since way back with MS-DOS. I began analysis in Remnux. Open the file in Sublime Text. First lines of the of the try and hide the window or minimize it, so the user won't see a command prompt window. It also runs Echo off, meaning, don't print the commands in the command prompt window as they are run.



The next section calls lots of variables with the **set** command. These combine to make two xcopy commands:

echo F | xcopy /d /q /y /h /i c:\Windows\SysWOW64\WindowsPowerShell\v1.0\powershell.exe %temp%\Hxoaznobtmy.png
echo F | xcopy /d /q /y /7h /7i 0 %tempt%\Hxoaznobtmy.png.bat

These lines copy two files. One goes to two places, the second goes just into the %temp% directory. One has an extension as a .png file, and the second adds a .bat extension. The next command contains all the remaining variables. The beginning of the command is:

%temp%\Hxoaznobtmy.png -win 1 -enc

This points to the .png file in the %temp% directory. Most likely a bat file that decodes the values of each of the remaining variables. Unable to continue on this path. I can continue looking at the remaining and very large blob of nonsense text.

It appears to be base64 encoded. Based on the equals sign at the end, I can assume it is. The entire blob is formatted into one line. I copied the text into a new file. Ran File on it to see it's a gzip file. I quickly unzipped it and ran File again. This reported as just data.

To better understand what this is, I ran strings on it. At first I did not see anything in the beginning that would help me identify the file. It wasn't until I got to end and saw it.

The bytes in the file were flipped and mirrored. I needed to reverse all this to begin further analysis of this PE file.

```
crsr.
txet.
.edom SOD ni nur eb tonnac margorp sihT!
```

Created a python script that would do the trick. I ran it once and got only a partial of the data. The problem with this line **b = b.to_bytes(2, 'big')**. The 2 was wrong. Changed this to 1 and re-ran the script on my mirrored file. This worked. File command reported it as *PE32 executable (DLL) (console) Intel 80386 Mono/.Net assembly, for MS Windows*.

stage1.bat | reverse.py | decoded

```
#reverse and invert bytes in file


with open('decoded', 'rb') as reverseme, open('stage2.exe', 'wb') as outfile:
    bytes_read = reverseme.read()
    bytes_reversed = bytes_read[::-1]

    for b in bytes_reversed:
        b = b.to_bytes(1, 'big')
        outfile.write(b)
outfile.close()

```

This file had the following information about it:
MD5: 5ffb962964a1cba905dd59e51b565e92
SHA1: b299bdbce9dc21e5e56a5a5dd180a1f62a9a0402
DLL is packed. Entropy: 7.97057(packed)
Size: 800256(781.50 kB)

I ran Virustotal on it. The scans reported 23/70 vendors flagging it as a Trojan.msiheracles/msi

Attempted to continue analysis of the file. Saw some strings that raised some concerns: CryptoStreamMode, FromBase64String, ToBase64String, GetTempPath, kernel32.dll, Gzipstream, CryptoStream, set_DisallowStartIfOnBatteries, set_StopIfGoingOnBatteries.

I can tell the DLL appears to be packed. Has a high entropy based on a quick analysis. Unable to identify what packer was used. Not UPX. Attempted to run de4dot on the dll. This resulted in some changes to strings, but nothing that indicated the packer, or what this dll does. Not able to dissect via assembly code. Haven't learned enough about that yet.

Based on the previous line in the original bat file, I do not have the .png file that to further decode what the bat file is doing. This appears to be a second or third stage of an attack. Something before this bat file downloaded this file and the other two files that get copied to the Windows\ syswow64\powershell1.0\ and the %temp% directories.

Indicators that this bat file is malicious are:
1. Obfuscation of commands – broken up in confusing ways into many variables
2. Minimizing or hiding the command prompt window as the bat file runs
3. Long line of text that became the zipped dll
4. Variables with obfuscated values.
5. Copying files into the Windows\sysWOW64\PowerShell1.0 directory
6. Copying a file with a .png.bat name (tricking the user to think this is an image file)
7. A line that calls to a png to decode remaining variables