Practical Malware Analysis & Triage

Malware Analysis Report

WannaHusky.exe Ransomware

Jan 2022 | Crypt0ace | v1.0

Table of Contents

Executive Summary

SHA256 hash	3d35cebcf40705c23124fdc4656a7f400a316b8e96f1f9e0c187e82a9d17dca3
MD5 hash	0287b38f8240a025b30c0a231ea403fc

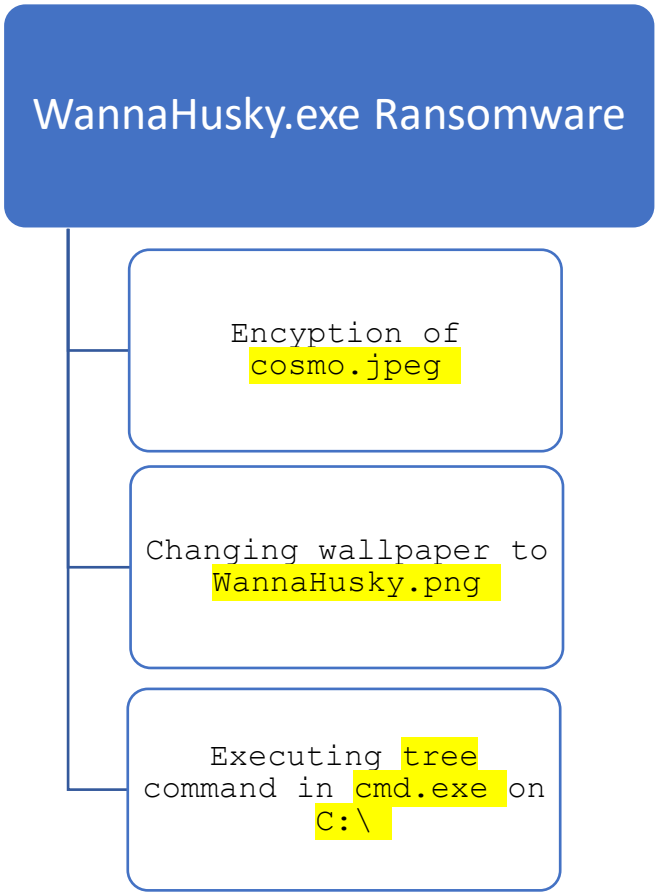
WannaHusky.exe Ransomware is a ransomware sample that was provided for analysis and triage. It has 3 stages:

1. Looks for cosmo.jpeg in the user's Desktop folder. Encrypts it and deletes the original file.
2. Creates a ps1.ps1 powershell script that is used to change the background to WannaHusky.png.
3. Runs the tree command on a command prompt.

YARA signature rules are attached in Appendix A. Malware sample and hashes have been submitted to VirusTotal for further examination. Results of which can be seen on Page 5.

High-Level Technical Summary

As stated above the WannaHusky.exe Ransomware runs in 3 stages, encryption of cosmo.jpeg, changing wallpaper and running the tree command.



Malware Composition

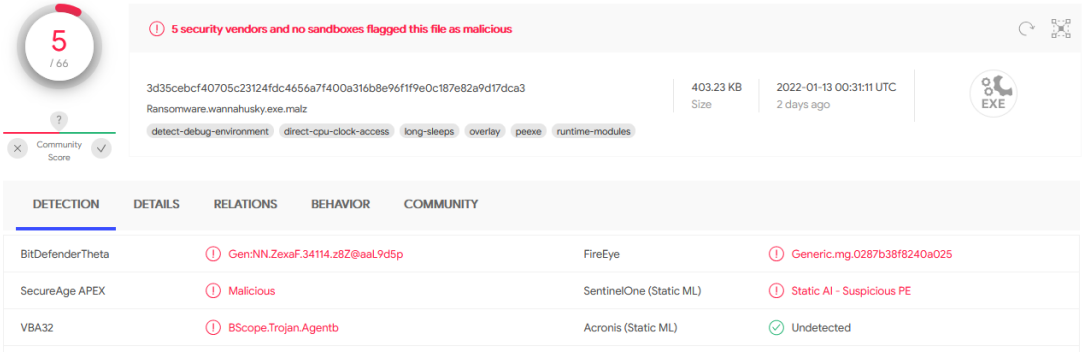
WannaHusky.exe Ransomware consisted of the only one component:

File Name	SHA256 Hash
WannaHusky.exe	3d35cebcf40705c23124fdc4656a7f400a316b8e96f1f9e0c187e82a9d17dca3

Basic Static Analysis

VirusTotal:

Submitting the binary to VirusTotal shows 5 detections out of 66



Strings:

Some interesting strings we found were classified as:

- Several mentions of nim indicate that this binary was written in nim language.
- The binary uses NimCrypto library to encrypt the cosmo.jpeg file, saves it as cosmo.WANNAHUSKY.
- Saves a WANNAHUSKY.png file on Desktop (Later seen as wallpaper as per the ps1.ps1 powershell script)
- Retrieved a powershell script that comes into play later on. (See page 11 to read the script).
- The command tree is executed on C:\

Basic Dynamic Analysis

Initial Detonation:

The initial detonation of the binary shows that upon execution, it encrypts the cosmo.jpeg file from Desktop, changes it's extension to cosmo.WANNAHUSKY and deletes the original file. It then proceeds to change the wallpaper to WANNAHUSKY.png, which mentions that this is a ransomware and demands 100 husky coins in 24 hours at "hxxps[://]huskyhacks[.]dev" to decrypt the cosmo.jpeg file. It also executes a command prompt with tree command on C:\.

Conditions required for it to run successfully:

In order for the binary to run successfully, the cosmo.jpeg file needs to be on the user's desktop.

Advanced Static Analysis

Opening the binary in a disassembler, we can see 3 main functions in the NimMainModule:

- wannaHusky
- changeBackground
- nosexecShellCmd

The first function wannaHusky is responsible of encrypting the cosmo.jpeg, also can be classified as the primary function of the binary.

The second function is responsible for changing the background to WANNAHUSKY.png.

The third function is responsible of running the tree command on C:\.

i: Important Functions

With controlling the execution in x32dbg, we can see the process of opening and reading on cosmo.jpeg in Procmon.

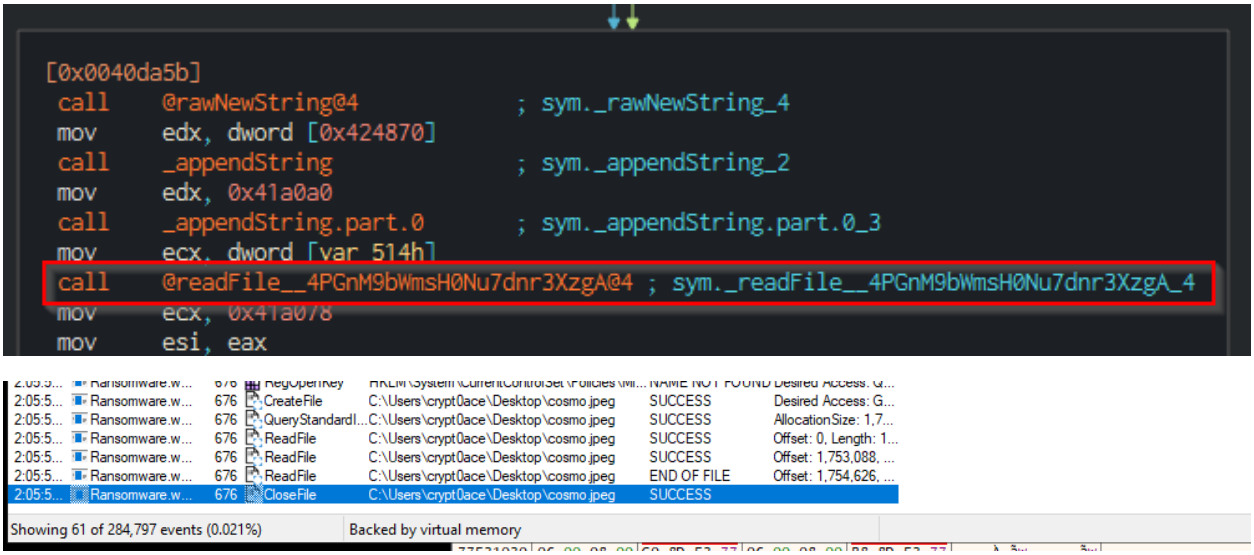
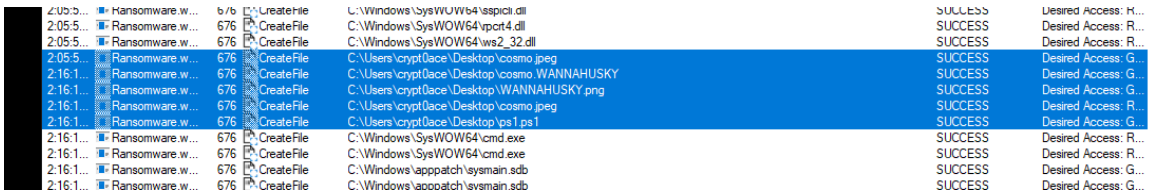


Figure ii: Reading the cosmo.jpeg file

Then it writes to cosmo.WANNAHUSKY file on desktop. It also creates 2 more files, WANNAHUSKY.png and ps1.ps1 on desktop. We can see these processes in Procmon.



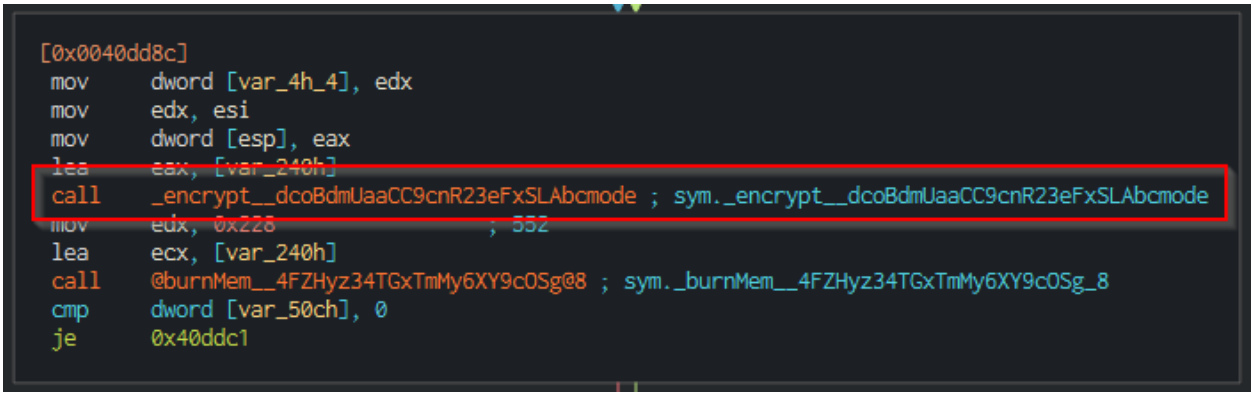


Figure iii: Possible encryption of cosmo.jpeg

After the creation of WANNAHUSKY.png file, it gets used by the ps1.ps1 powershell script.

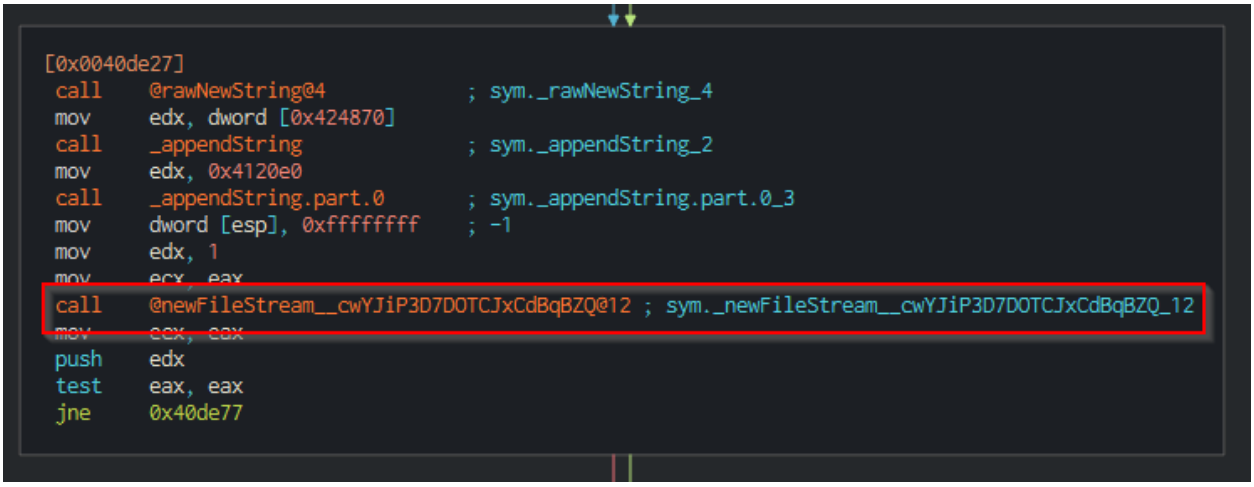


Figure iv: Creation of WANNAHUSKY.png

It then deletes the original cosmo.png file.

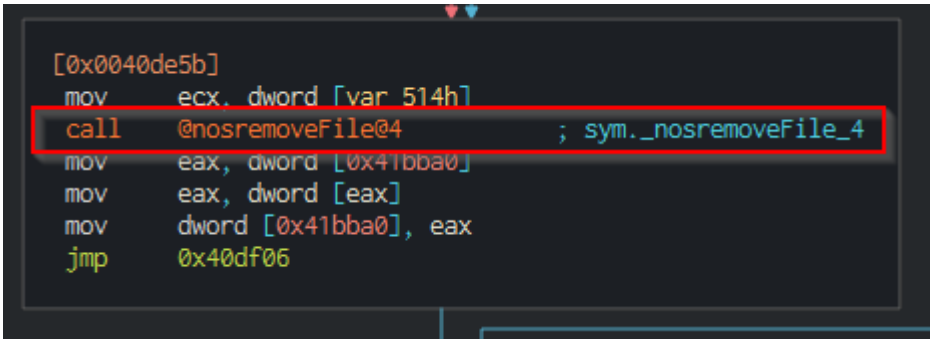


Figure v: Deletion of cosmo.jpeg file

We can see the execution of ps1.ps1 from here

Figure vi: Running powershell on ps1.ps1 and deleting it afterwards

And lastly, it executes the tree command on C:\.

Advanced Dynamic Analysis

The only thing required to analyze through advanced dynamic analysis was the ps1.ps1 powershell script that gets placed in the user's desktop folder. This file is responsible for changing of wallpaper. We can control the flow of execution using x32dbg and pause the execution right before the file gets deleted.

```
$code = @'
using System.Runtime.InteropServices;

namespace Win32{

    public class Wallpaper{

      [DllImport("user32.dll", CharSet=CharSet.Auto)]
      static  extern int SystemParametersInfo (int uAction , int uParam , string
lpvParam , int fuWinIni) ;

      public static void SetWallpaper(string thePath){
         SystemParametersInfo(20,0,thePath,3);
      }
    }
}
'@
add-type $code

$currDir = Get-Location
$wallpaper = ".\WANNAHUSKY.PNG"
$fullpath = Join-Path -path $currDir -ChildPath $wallpaper

[Win32.Wallpaper]::SetWallpaper($fullpath)
```

We can understand that user32.dll is imported and SetWallpaper is used to change the wallpaper to WANNAHUSKY.png. This was also recovered from strings on the binary.

Indicators of Compromise

Host Based Indicators:

A file ps1.ps1 appears on the desktop for a short interval before deleting itself. The cosmo.jpeg file is encrypted with the .WANNAHUSKY file extension. The wallpaper is changed to WANNAHUSKY.png and a command prompt is spawned with tree command running on “C:\”.

Network Based Indicators:

No network-based indicators were found.

Rules & Signatures

A full set of YARA rules is included in Appendix A.

Strings:

The strings added for YARA Rules:

```
@tree C:\
@Desktop\ps1.ps1
@powershell
@Desktop\ps1.ps1
@$code = @'
@Desktop\WANNAHUSKY.png
@Desktop\cosmo.WANNAHUSKY
@COSMO
@Desktop\target\cosmo.WANNAHUSKY
@Desktop\cosmo.jpeg
```

Magic Bytes:

The magic bytes found was “MZ” that indicates it as a Portable Executable (PE).

Appendices

A. Yara Rules:

```
rule WANNAHUSKY {

    meta:
        last_updated = "2022-01-15"
        author = "Crypt0ace"
        description = "YARA Rules for WannaHusky Ransomware"

    strings:
        $string1 = "tree" ascii
        $string2 = "ps1.ps1" ascii
        $string3 = "powershell"  ascii
        $string4 = "WANNAHUSKY.png" ascii
        $string5 = "cosmo.WANNAHUSKY" ascii
        $string6 = "cosmo.jpeg" ascii
        $PE_magic_byte = "MZ"

    condition:
        $PE_magic_byte at 0 and
        ($string1 and $string2 and $string3 and $string4 and $string5 and
$string6)
}
```