Objective--Stop the Malware (Part 1)

# What you can learn from this

The WannaCookie malware in this year’s challenge is patterned after the famous WannaCry ransomware. A young security person stumbled into a DNS domain name that was a kill switch for WannaCry and stopped the malware in its tracks. [This article](http://nymag.com/intelligencer/2018/03/marcus-hutchins-hacker.html) talks about the young man and his fate since then. Young hackers need to be careful.

If you want more than history, you can learn that in this challenge as well. This one is all about reverse-engineering malware written in PowerShell.

## Malware functions list

The list we generated as homework will help us begin to understand this malware.

|  |  |  |
| --- | --- | --- |
| **Function** | **Purpose** | **Notes** |
| Enc\_Dec-File | encrypts or decrypts files | uses AES 256 |
| H2B | converts hex string to byte array | "-split '(..)' is regex for any two characters |
| A2H | converts ascii string to hex | "{0:X}" is a format operator, converts to hex |
| H2A | converts hex string to ascii | ?{$\_} seems to strip extra lines |
| B2H | converts byte array to hex |  |
| ti\_rox | bitwise XOR |  |
| B2G | compresses byte array with gzip |  |
| G2B | uncompresses byte array |  |
| sha1 | computes SHA-1 hash |  |
| Pub\_Key\_Enc | encrypts a byte array with pub key | |  | | --- | | $key\_bytes to be encrypted, byte array | | $pub\_bytes is public key, byte array | | output is hex of encrypted key | |
|  |  |  |
| enc\_dec | calls Enc\_Dec-File to encrypt/decrypt | runs 12 jobs at a time |
| get\_over\_dns | receives files from DNS server | $f.erohetfanu.com returns # of blocks |
|  |  | $i.$f.ero…. Is an individual block |
| split\_into\_chunks | breaks string into 32 byte chunks | used by send\_key |
| send\_key | sends encrypted key to server | first time, gets botid |
|  |  | after, prepends botid to chunk |

Many of the functions are simple conversion routines. The evil deed in encrypting the file is done by Enc\_Dec-File, using a key and AES. The actual control happens in the function wannacookie (or wanc in the minimized version of the script.

# Caution

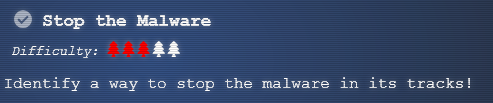
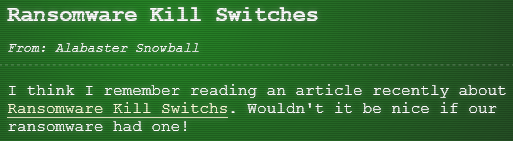
Remember to work on malware in a protected VM. Also, the end of the wannacookie function has code that downloads a large file via the DNS mechanism. That is really slow, so if it runs it will appear that your ISE has hung, and the malware DNS server has stopped. If you open Wireshark and see loads of DNS traffic to your machine, that is what happened.

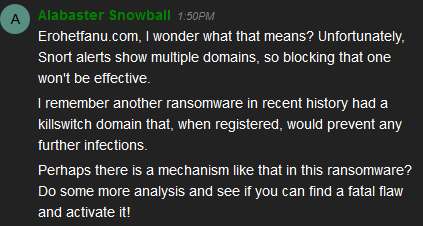
# Dot sourcing

In his talk, Chris showed how to set a breakpoint in ISE and then step through the code. While the script is running in Debug mode, all its functions are available for your use on the command line. I found I wanted to use the functions even when the script was not running, so I resorted to dot sourcing. That just loads the functions into memory. The steps are simple:

1. Copy the functions you want into a separate file, let’s say malware-functions.ps1. I left the main wannacookie function out of my file, as well as enc\_dec and Enc\_Dec-File since I didn’t expect to need them.
2. From your ISE command prompt, enter . path/to/malware-functions.ps1
3. There is a period (dot) followed by a space at the beginning of the command, the reason it is called dot sourcing. Once it runs the functions will be loaded into memory
4. Now, you don’t have to be in Debug mode to use the script’s functions. You can type something like H2A "77616E6E61636F6F6B69652E6D696E2E707331" and it will run.

# Get to work

Like WannaCry, WannaCookie also has a kill switch. Our job is to find it. Since wannacookie is the primary function, look for things that end it prematurely. It would also be a good idea to translate any hex strings you find into ASCII. Alabaster’s hint about the kill switch [points here](https://www.wired.com/2017/05/accidental-kill-switch-slowed-fridays-massive-ransomware-attack/).  
 



Hand in

1. What is the domain that kills WannaCookie?