Samuel Tracz

Simple Malware Code

Intro

My malware was inspired by an article I have read where attackers are concatenating two zip files where the extracting software is only able to see the contents of the first zip contents (https://www.bleepingcomputer.com/news/security/hackers-now-use-zip-file-concatenation-to-evade-detection/amp/)

The malware consists of the initial payload, the attacker server, and the keylogger.

Initial Payload:

My version of the initial malicious payload is a concatenation of a jpeg file and a malicious executable. In this case the malicious file is a reverse shell coded in C and compiled using Visual Studio compiler. This was compiled using the Visual Studio compiler due to certain C libraries that are on Windows that are not available by default on Linux using gcc. My assumption is that this payload would only work on Windows systems.

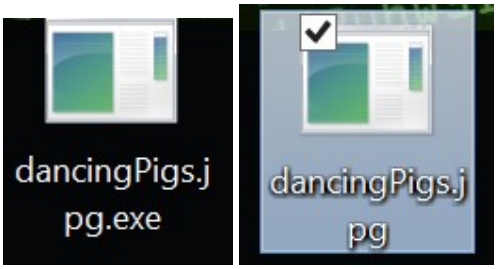
```
C:\Users\Sam\Desktop>cl reverse_shell.c
Microsoft (R) C/C++ Optimizing Compiler Version 19.29.30151 for x86
Copyright (C) Microsoft Corporation.  All rights reserved.

reverse_shell.c
Microsoft (R) Incremental Linker Version 14.29.30151.0
Copyright (C) Microsoft Corporation.  All rights reserved.

/out:reverse_shell.exe
reverse_shell.obj
```

I transferred the executable to Linux, where I concatenated the malicious executable with the JPG file. The filename contains a false extension so that when Windows hides the real extension, it would look like a jpg file.

```
root@kali:~/Desktop# cat reverse_shell.exe dancing-pigs.jpg > dancing-pig.jpg.exe
```



Server:

The server was tested on a Kali linux machine. It was coded using python. It assumes that both ports 4444 and 8000 are not in use prior to running the program. The program has a few

key features. First it listens on port 4444 and listens for a socket connection and hosts the keylogger executable on a web server, then when a connection is established the reverse shell is created and socket commands for the client to pull the keylogger are sent. This is followed up by another command to run the keylogger. The server has shell access to the victim computer, but the commands are seemingly 1-off in a queue and I can't seem to fix it. I have added a custom 'pull_keylog' command which pulls the keylogged output file back onto the server.

Keylogger:

This keylogger assumes that it is a windows machine due to the use of Windows api. It saves the keystrokes to a local file.

Future Additions

To improve on this malware, I would like to initially figure out how to fix the reverse shell data mis-synchronization between data sent and received. I would then either like to figure out how to execute the jpeg file directly from memory so that I wouldn't have to create a new file to view it, or leave it that way and have my malicious payload uninstall itself so the victim is left with a normal picture and wouldn't notice that there was malicious code on the machine.

An initial goal for this project was to try to avoid antivirus detection. The test was to put the executable onto my local machine and see if SentinelOne detected it. Initially, it did not detect the program but after a short period (<24 hours) the antivirus deleted the program. In the future, I would probably have to obscure the code more and use more sophisticated methods such as embedding the malicious payload into the jpeg compared to just concatenating.