Practical 2

Antoine Ferguson
3/21/2022
a.ferguson@ufl.edu

Practical 2 - CAP4136

Executive Summary

This malware creates a task with the task schedule that enables the x64-bot file to run constantly. It's scheduled to run the x64-bot file daily at 1-minute intervals. This is confirmed by the Autoruns application, which shows the newly created task named "\Bot" and the TaskCache\Tasks registry key with the value 'URI: "\Bot"'. This malware is obfuscated. Its source code is encrypted and stored in its resource section. It's true nature is revealed by dumping the decrypted source code from where the malware stored it in the current running process.

Static Analysis

Using PEStudio the malware's compile date was revealed as August 19, 2016. Imported functions included GetProcessAddress, LoadLibrary, HeapAlloc, VirtualProtect, and LoadResource and no suspicious strings were found. However, the malware's real source code was embedded in the resource section. After getting the decrypted x64-bot and x64-loader files, PEStudio revealed more information.

PEStudio revealed some of the libraries used by the x64-bot and x64-loader files, including: crypt32.dll, winhttp.dll, iphlpapi.dll, ws2_32.dll, ole32.dll, and oleaut32.dll. Some suspicious strings include: "zen.spamhaus.org", "wtfismyip.com", "spam.dnsbl.sorbs.net", "myexternalip.com", and "client is not behind NAT". I've previously stated a x64-bot and x64-loader file were stored in the resource section, but a x86-bot file was stored as well.

I didn't discover any major anti-disassembly techniques. There were one sequence of instructions Ghirda misinterpreted as a date, however I'm unsure if that was maliciously crafted or just a misinterpretation. However, the malware obfuscated its source code.

When I first analyzed the malware, I got suspicious when very few strings and imports were found. I first checked if the malware was packed. The malware's virtual size was nearly equal to its raw size. Also, the UPX packer determined it wasn't packed by UPX. Then I reviewed the imports that were discovered, particularly the heap and resource related functions. This convinced me that the malware was obfuscated and not packed. Using Ghirda and x32dbg, I determined the malware stored the x64-bot and x64-loader files in the process' heap memory. The malware copied data from the resource section into the space it allocated within the process's heap and decrypted the data using a custom xor/multiplication equation. Malware allocated space for the x64-bot file at address 0x5D3E50 and the x64-loader file at 0x5Ed468.

Dynamic Analysis

After running the x64-bot and x64-loader files, separately, with x64dgb, a prompt to open the x64-bot file with an application persists. This prompt constantly reoccurs, regardless if the file is open with an application or not. This occurs when the file is stored as a .bin, but not when it's a .exe file. Autoruns shows a scheduled task named "\Bot" added by the malware. This task starts daily at 12 AM and repeats after a minute for the duration of the day. Even if the computer restarts or if the process is killed, the task will continue to act.

Some keys added to …Schedule\TaskCache, Plain, Tasks, and Tree\Bot. Values added to …\Schedule\TaskCache\Tasks were '\Path: "\Bot"', a hash value, a schema, 'Author: "Author Name"', and 'URI: "\Bot"'. Both the x64-bot and the x64-loader files were moved to the Malware\AppData\Roaming directory. Besides the two processes for x64-bot and x64-loader running in the Roaming directory, there were no other processes created.

Indicators of Compromise

To verify the malware, a Yara rule was applied to the Remnux host. My rule checked for the kernel32.dll from the imports section of files and checked if the file extension was an exe file. This rule was effective as the Remnux host has no other Windows applications installed. The main indicator of infection was the newly created task named Bot. This task scheduled the x64-bot file, stored in the Malware\AppData\Roaming directory, to run daily.