
CobaltStrikeScan

Scan files or process memory for Cobalt Strike beacons and parse their configuration.

CobaltStrikeScan scans Windows process memory for evidence of DLL injection (classic or reflective injection) and/or performs a YARA scan on the target process' memory for Cobalt Strike v3 and v4 beacon signatures.

Alternatively, CobaltStrikeScan can perform the same YARA scan on a file supplied by absolute or relative path as a command-line argument.

If a Cobalt Strike beacon is detected in the file or process, the beacon's configuration will be parsed and displayed to the console.

Cloning This Repo

CobaltStrikeScan contains GetInjectedThreads as a submodule. Ensure you use `git clone --recursive https://github.com/Apr4h/CobaltStrikeScan.git` when cloning CobaltStrikeScan so that the submodule's code is also downloaded/cloned.

Building the Solution

Costura.Fody is configured to embed CommandLine.dll and libyara.NET.dll in the compiled CobaltStrikeScan.exe assembly. CobaltStrikeScan.exe should then serve as a static, portable version of CobaltStrikeScan. For this to occur, ensure that the "Active Solution Platform" is set to x64 when building.

Acknowledgements

This project is inspired by the following research / articles: - SpecterOps - Defenders Think in Graphs Too - JPCert - Volatility Plugin for Detecting Cobalt Strike - SentinelLabs - The Anatomy of an APT Attack and CobaltStrike Beacon's Encoded Configuration - Neo23x0's Signature Base for high-quality YARA signatures used to detect Cobalt Strike's encoded configuration block.

Requirements

- 64-bit Windows OS
- .NET Framework 4.6
- Administrator or SeDebugPrivilege is required to scan process memory for injected threads

Usage

```
1  -d, --directory-scan      Scan all process/memory dump files in a
    directory for Cobalt Strike beacons
2
3  -f, --scan-file           Scan a process/memory dump for Cobalt
    Strike beacons
4
5  -i, --injected-threads    Scan running (64-bit) processes for
    injected threads and Cobalt Strike beacons
6
7  -p, --scan-processes      Scan running processes for Cobalt
    Strike beacons
8
9  -v, --verbose             Write verbose output
10
11 -w, --write-process-memory Write process memory to file when
    injected threads are detected
12
13 -h, --help                Display Help Message
14
15 --help                   Display this help screen.
16
17 --version                 Display version information.
```

Example

```
Administrator: Command Prompt

c:\Users\Apr4h\Desktop>CobaltStrikeScan.exe -p -d
Scanning processes for injected threads
Found injected thread
Process           : artifact
Process ID        : 2984
Thread ID         : 5544

Writing injected thread bytes to file: 2020-08-9--00-12-02-proc2984-thread5544.dmp

Writing process bytes to file: 2020-08-9--00-12-02-proc2984.dmp

Scanning injected thread for CobaltStrike beacon
CobaltStrike Beacon Configuration:

BeaconType:           : 0 (HTTP)
Port:                 : 8000
Polling(ms):          : 60000
MaxGetSize:           : 1048576
Jitter:               : 20
Maxdns:               : 235
C2Server:              : stygga.castle.local,/c/msdownload/update/others/2016/12/29136388_
UserAgent:             : Windows-Update-Agent/10.0.10011.16384 Client-Protocol/1.40
HTTP_Post_URI:         : /c/msdownload/update/others/2016/12/3215234_
HTTP_Method1_Header:   : Accept: /*/*
                        : Host: download.windowsupdate.com
                        : .cab
HTTP_Method2_Header:   : Accept: /*/*
                        : download.windowsupdate.com/c/
                        : Host
                        : .cab
Injection_Process:      : ^0[?5?i?o?q?JD
Spawnto_x86:           : %windir%\syswow64\rundll32.exe
Spawnto_x64:           : %windir%\sysnative\rundll32.exe
PipeName:              :
CryptoScheme:          : 1
DNS_idle:              : 0
DNS_sleep(ms):         : 0
HTTP_Method1:          : GET
HTTP_Method2:          : GET
HttpPostChunk:         : 96
Watermark:             : 1873433027
StageCleanup:          : False
CfgCaution:           : False
UsesCookies:           : False
KillDate:              : 0
ProcInject_StartRWX:    : True
ProcInject_UserRWX:    : True
ProcInject_MinAllocSize: : 0
ProcInject_PrependedAppend_x86: :
ProcInject_PrependedAppend_x64: :
ProcInject_AllocationMethod: : VirtualAllocEx
```