Scanning Cobalt Strike in current machine

This repository includes packages and environments for the users to use without installing any additional packages

Once the scan is done, the user is free to delete this from their disk. This is only for Windows OS.

> NOTE: Apr4h's source code only allows Windows x64 Machines. However, I am not sure if the x86 compiled executable will have any problems or not. Please raise an issue if there are any issues.

Preparation

Download this repository

> Make sure you are in the C:\ root folder when performing this git clone. This is because the environments were set up at C:\ folder. `git clone --recursive https://github.com/jeremyng123/scanning_cobalt_strike.git`

Note the use of `--recursive` because of the other submodules. The 2 submodules I have added to this repository are:

1. 1768 from DidierStevensSuite
2. CobaltStrikeScan from Apr4h -> the compiled executables are found in /exe folder.

Scan

`run.bat`

This command would be sufficient. Please note that it will request for administrative privilege on execution.

Understanding the project structure

1. 1768 folder contains the repository from DidierStevensSuite
   This file uses a Python script to scan all mounted volumes in the machine recursively. It scans only static files and look for any likely Cobalt Strike configurations and signatures.
   `pefile` and `peutils` are the 2 required Python libraries to scan
   When it is done scanning, it will create a csv file called `1768_output.csv` in the `/` folder.

2. Apr4h folder contains the repository from Apr4h According to the author, a Windows x64 machine is required for this scan.
   Not sure if there will be any issues, please raise them if any issues were found.
   the -p flag was given to the executable to scan all running processes found in the infected machine using YARA rules.
   When done, the file `Apr4h_out.txt` will be created in the `/` folder. All Cobalt Strike configuration related to the processes will be extracted.

3. `WPy*` folders The `WPy32*` and `WPy64*` are portable Python folders in the event the infected machine does not have these installed.
   As the name of the folder suggests, `WPy32*` is for 32bit machine and `WPy64*` is for 64bit machine.

4. `env*` folders These are virtual environment created using the respective `WPy*` folders.
   When running the Python script for static file recursive scan with `1768.py`, the batch file will first activate the environment with the command `env*\Scripts\activate.bat`, where `*` is either `32` or `64`.

   Whatever libraries are needed to run the Python script `1768.py` has already been installed. no further installation/downloads are required.

5. The `*.bat` files The `run_*.bat` files are there for the purpose of running the correct and specific executables/scripts depending on the type of machine the infected machine is in (32 or 64 bit machine?)

   The `run_*_static.bat` files will activate the Python environment and execute the `1768.py` Python script with the arguments `--recurse-dir %%j --csv 1768_out.csv` where `%%j` contains all the mounted volumes.
   The `run_*.bat` files will execute the `CobaltStrikeScan_x*.exe` file with the `-p` flag to scan all running processes

Credits

If not obvious enough, I would love to credit the 2 amazing submodules that I have added into this repositories:

1. DidierStevensSuite
2. Apr4h

For the amazing work they are doing in this space and detection of Cobalt Strike

Downside

Please note, however, that just using these tools can only detect **vanilla** Cobalt Strike configurations.
If the adversary used additional Aggressor scripts and other configurations (i.e., Memory Stomping), then it is paramount that the user know there are other ways that Cobalt Strike can conntinue to covertly operate in your machine.

To read more on other Cobalt Strike detections methodology, please read this amazing blog written by Nviso (the author is Didier Stevens), here:
https://blog.nviso.eu/series/cobalt-strike-decrypting-traffic/