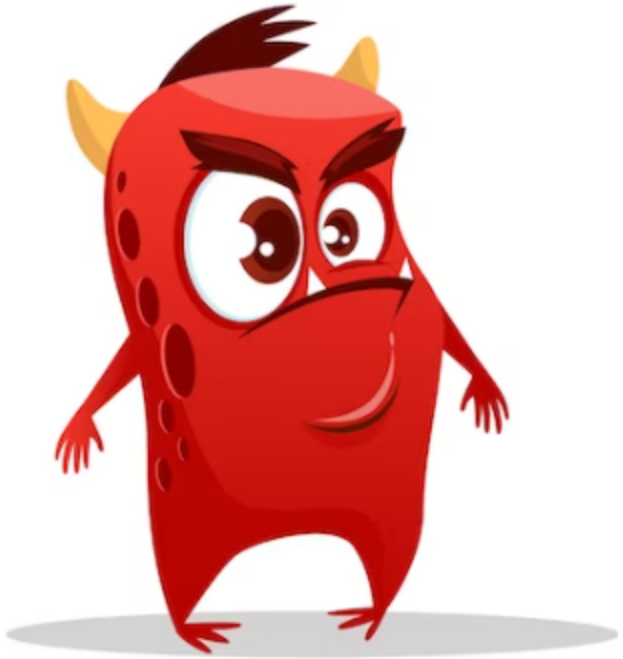
itaymigdal/PichichiH0ll0wer

github.com/itaymigdal/PichichiH0ll0wer

itaymigdal

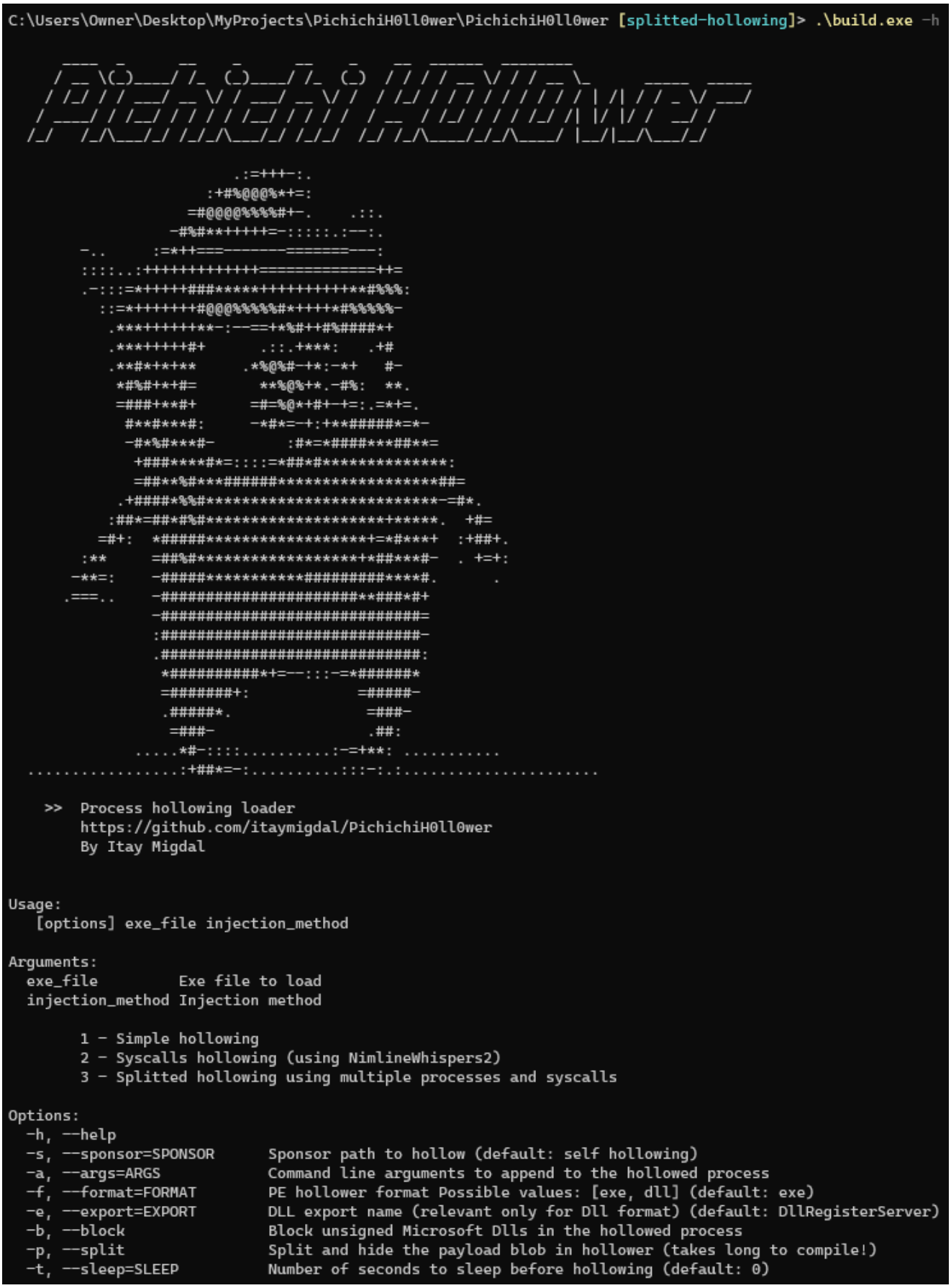
PichichiH0ll0wer



About

--== Process hollowing loader written in Nim for PEs only ==--

I built PichichiH0ll0wer to learn and contribute, sure. but also because I'm quite tired of shellcodes everywhere. Loading PEs might be less evasive, I know, but it's still efficient and more convenient than fighting to turn your PE payload into a shellcode each time (which not always works smoothly). Also, PichichiH0ll0wer has some features to protect your payload. I may add some more injection techniques and features in the future. Supports only x64 EXEs currently.

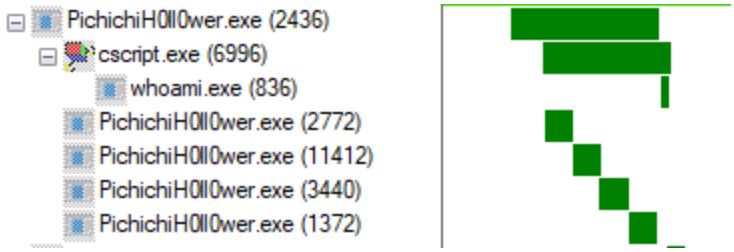

C:\Users\Owner\Desktop\MyProjects\PichichiH0ll0wer\PichichiH0ll0wer [splitted-hollowing]> .\build.exe -h
>> Process hollowing loader
https://github.com/itaymigdal/PichichiH0ll0wer
By Itay Migdal
Usage:
[options] exe_file injection_method
Arguments:
exe_file Exe file to load
injection_method Injection method
1 - Simple hollowing
2 - Syscalls hollowing (using NimlineWhispers2)
3 - Splitted hollowing using multiple processes and syscalls
Options:
-h, --help
-s, --sponsor=SPONSOR Sponsor path to hollow (default: self hollowing)
-a, --args=ARGS Command line arguments to append to the hollowed process
-f, --format=FORMAT PE hollower format Possible values: [exe, dll] (default: exe)
-e, --export=EXPORT DLL export name (relevant only for Dll format) (default: DllRegisterServer)
-b, --block Block unsigned Microsoft Dlls in the hollowed process
-p, --split Split and hide the payload blob in hollower (takes long to compile!)
-t, --sleep=SLEEP Number of seconds to sleep before hollowing (default: 0)

Features

- Configurable builder
- Payload encrypted and compressed (and optionally splitted) in the hollow loader
- Hollower does not using the very suspicious call Nt/ZwUnmapViewOfSection
- Can build EXE / DLL hollow loaders
- Can block unsigned microsoft DLLs from being loaded to the hollowed process
- Obfuscated sleep using useless calculations

Injection methods

1. Simple hollowing: just the usual stuff: VirtualAlloc -> WriteProcessMemory -> GetThreadContext -> SetThreadContext -> ResumeThread.
2. Syscalls hollowing: using the great NimlineWhispers2 direct syscalls.
3. Splitted hollowing: each step of method (2) is occuring in a seperate process with inherited handles, also uses NimlineWhispers2 syscalls. this method is more evasive, and known to bypass some EDR's.



Installation

Built with Nim 1.6.12, should be run on Windows only.

```
nimble install winim ptr_math nimprotect zip argparse
```

Usage

```
Usage:
   [options] exe_file injection_method

Arguments:
  exe_file         Exe file to load
  injection_method Injection method

        1 - Simple hollowing
        2 - Syscalls hollowing (using NimlineWhispers2)
        3 - Splitted hollowing using multiple processes and syscalls

Options:
  -h, --help
  -s, --sponsor=SPONSOR      Sponsor path to hollow (default: self hollowing)
  -a, --args=ARGS            Command line arguments to append to the hollowed process
  -f, --format=FORMAT        PE hollower format Possible values: [exe, dll] (default:
exe)
  -e, --export=EXPORT        DLL export name (relevant only for Dll format) (default:
DllRegisterServer)
  -b, --block                Block unsigned Microsoft Dlls in the hollowed process
  -p, --split                Split and hide the payload blob in hollower (takes long
to compile!)
  -t, --sleep=SLEEP          Number of seconds to sleep before hollowing (default: 0)
  -d, --debug                Compile as debug instead of release (loader is verbose)
```

Also, check the examples.

Credits