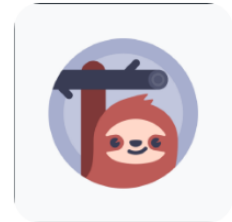


itaymigdal/Nimbo-C2

 github.com/itaymigdal/Nimbo-C2


itaymigdal


itaymigdal/**Nimbo-C2**



Nimbo-C2 is yet another (simple and lightweight) C2 framework

 1
Contributor

 0
Issues

 76
Stars

 5
Forks



Nimbo-C2



- [Nimbo-C2](#)
- [About](#)
- [Features](#)
- [Installation](#)
 - [Easy Way](#)
 - [Easier Way](#)

My work wouldn't be possible without the previous great work done by others, listed under credits.

Features

- Build EXE, DLL payloads.
- Packing payloads using `UPX` and obfuscate the PE section names (`UPX0` , `UPX1`) to make detection and unpacking harder.
- Encrypted HTTP communication (AES in CBC mode, key hardcoded in the agent and configurable by the `config.jsonc`).
- Auto-completion in the C2 Console for convenient interaction.
- In-memory Powershell commands execution.
- File download and upload commands.
- Built-in discovery commands.
- Screenshot taking and clipboard stealing.
- Memory evasion techniques like NTDLL unhooking, ETW & AMSI patching.
- LSASS and SAM hives dumping.
- Shellcode injection.
- Persistence capabilities.
- UAC bypass methods.
- And more !

Installation

Easy Way

1. Clone the repository and `cd` in

```
git clone https://github.com/itaymigdal/Nimbo-C2
cd Nimbo-C2
```

1. Build the docker image

```
docker build -t nimbo-dependencies .
```

1. `cd` again into the source files and run the docker image interactively, expose port 80 and mount Nimbo-C2 directory to the container (so you can easily access all project files, modify `config.jsonc` , download and upload files from agents, etc.). For Linux replace `${pwd}` with `$(pwd)` .

```
cd Nimbo-C2
docker run -it --rm -p 80:80 -v ${pwd}:/Nimbo-C2 -w /Nimbo-C2 nimbo-dependencies
```

Easier Way

```
git clone https://github.com/itaymigdal/Nimbo-C2
cd Nimbo-C2/Nimbo-C2
docker run -it --rm -p 80:80 -v ${pwd}:/Nimbo-C2 -w /Nimbo-C2 itaymigdal/nimbo-
dependencies
```

Usage

First, edit `config.jsonc` for your needs.

Then run with: `python3 Nimbo-C2.py`

Use the `help` command for each screen, and tab completion.

Main Window

```
Nimbo-C2 > help
```

```
--== Agent ==--
```

```
agent list           -> list active agents
agent interact <agent-id> -> interact with the agent
agent remove <agent-id> -> remove agent data
```

```
--== Builder ==--
```

```
build exe           -> build exe agent (-h for help)
build dll           -> build dll agent (-h for help)
```

```
--== Listener ==--
```

```
listener start      -> start the listener
listener stop        -> stop the listener
listener status      -> print the listener status
```

```
--== General ==--
```

```
cls                 -> clear the screen
help                -> print this help message
exit                -> exit Nimbo-C2
```

Agent Window

Nimbo-2 [d337c406] > help

```
--== Send Commands ==--
cmd <shell-command>           -> execute a shell command
iex <powershell-scriptblock> -> execute in-memory powershell command

--== File Stuff ==--
download <remote-file>        -> download a file from the agent (wrap
path with quotes)
upload <local-file> <remote-path> -> upload a file to the agent (wrap paths
with quotes)

--== Discovery Stuff ==--
pstree                         -> show process tree
checksec                       -> check for security products

--== Collection Stuff ==--
clipboard                      -> retrieve clipboard
screenshot                    -> retrieve screenshot

--== Post Exploitation Stuff ==--
lsass <method>                -> dump lsass.exe [methods:
direct,comsvcs] (elevation required)
sam                            -> dump sam,security,system hives using
reg.exe (elevation required)
shellc <raw-shellcode-file> <pid> -> inject shellcode to remote process

--== Evasion Stuff ==--
unhook                        -> unhook ntdll.dll
amsi                          -> patch amsi out of the current process
etw                           -> patch etw out of the current process

--== Persistence Stuff ==--
persist run <command> <key-name> -> set run key (will try first hkln, then
hkcu)
persist spe <command> <process-name> -> persist using silent process exit
technique (elevation required)

--== Privesc Stuff ==--
uac fodhelper <command> <keep/die> -> elevate session using the fodhelper
uac bypass technique
uac sdclt <command> <keep/die> -> elevate session using the sdclt uac
bypass technique

--== Interaction stuff ==--
msgbox <title> <text>          -> pop a message box (blocking! waits for
enter press)
speak <text>                   -> speak using sapi.spvoice com interface

--== Communication Stuff ==--
sleep <sleep-time> <jitter-%> -> change sleep time interval and jitter
clear                          -> clear pending commands
```

collect	-> recollect agent data
kill	-> kill the agent (persistence will still
take place)	
--== General ==--	
show	-> show agent details
back	-> back to main screen
cls	-> clear the screen
help	-> print this help message
exit	-> exit Nimbo-C2

Credits

- [OffensiveNim](#) - Great resource that taught me a lot about leveraging Nim for implant tasks. Some of Nimbo-C2 agent capabilities are basically wrappers around OffensiveNim modified examples.
- [Python-Prompt-Toolkit-3](#) - Awsome library for developing python CLI applications. Developed the Nimbo-C2 interactive console using this.
- [ascii-image-converter](#) - For the awesome Nimbo ascii art.
- All those random people from Github & Stackoverflow that I copy & pasted their code 😊.

TODO

Modules

- Improve Unhooking and patching by using syscalls.
- Run .NET assemblies.
- Migrate to another process (Meterpreter-like).
- Getsystem.
- Collect installed software.
- Zip & exfiltrate folder.
- Find sensitive files by keywords / regex search.

Misc

- Develop Proxy awareness for the agent.
- Add option to sign PE agent with digital signature. Some very nice Github projects do that to lower detection.
- Support shellcode payload type.
- Support more packers.