Wifiphisher Documentation

Release 1.2

George Chatzisofroniou

Aug 15, 2018

Contents

1 Table Of Contents **1**

1.1 Getting Started 1
1.2 User's guide 2
1.3 API Refrence 3
1.4 Frequently Asked Questions 5
1.5 Phishing Scenarios 5
1.6 Module Documentaion 8

2 Indices and tables **9**

Python Module Index **11**

CHAPTER 1

Table Of Contents

1.1 Getting Started

1.1.1 What is Wifiphisher

Wifiphisher is a security tool that mounts automated victim-customized phishing attacks against WiFi clients in order to obtain credentials or infect the victims with malwares. It is primarily a social engineering attack that unlike other methods it does not include any brute forcing. It is an easy way for obtaining credentials from captive portals and third party login pages (e.g. in social networks) or WPA/WPA2 passwords.

1.1.2 How to get Wifiphisher

There are three ways to obtain Wifiphisher

Using Pip

This method is the simplest and easiest way to obtain Wifiphisher. All you have to do is Simply run the following command in the terminal

```
[sudo] pip install --upgrade wifiphisher
```

Note: Using this method is recommended

Warning: `pip` must be installed on your system.

See also:

pip and `pip --help` for more information.

Using git

This method involves using the `git clone` command. Simply run the following command from the terminal

```
git clone https://github.com/wifiphisher/wifiphisher.git
```

Warning: `git` must be installed on your system.

See also:

Git and `git clone --help` for more information.

Using browser

This method involves downloading all the files using a browser.

1. Visit the projects page on Github
2. Press the `clone or download` button
3. Click the `Download ZIP` button

1.2 User's guide

1.2.1 Dependencies

RoboPhisher currently has the following dependencies:

- PyRIC
- Blessings
- Tornado
- Dnsmasq
- Hostapd

1.2.2 Requirements

Hardware

Frequently Asked Questions

Software

Python 2.7

1.2.3 Supported Platforms

Currently our only supported platform is Kali Linux. However the plan to add android and other Linux is possible.

1.3 API Refrence

1.3.1 Command Line Arguments

-h
--help

Show this help message and exit

Hint:

```
[sudo] RoboPhisher -h
```

```
[sudo] RoboPhisher --help
```

-jI
--jamminginterface

Manually choose an interface that supports monitor mode for deauthenticating the victims.

Hint:

```
[sudo] RoboPhisher -jI wlan1
```

```
[sudo] RoboPhisher --jamminginterface wlan1
```

Warning: The same interface can **not** be used for both *jamming interface* and *ap interface* .

-aI
--apinterface

Manually choose an interface that supports AP mode for spawning an AP.

Hint:

```
[sudo] RoboPhisher -aI wlan1
```

```
[sudo] RoboPhisher --apinterface wlan1
```

> **Warning:** The same interface can **not** be used for both *jamming interface* and *ap interface* .

-nJ
--nojamming

Skip the deauthentication phase. When this option is used, only one wireless interface is required.

Hint:

```
[sudo] RoboPhisher -nJ
```

```
[sudo] RoboPhisher -nojamming
```

-e
--essid

Enter the ESSID of the rogue access point you would like to create.

Hint:

```
[sudo] RoboPhisher -e "FREE WIFI"
```

```
[sudo] RoboPhisher --essid "FREE WIFI"
```

> **Warning:** This option will skip access point selection phase.

-p
--phishingscenario

Choose the phishing scenario to run.This option will skip the scenario selection phase.

Hint:

```
[sudo] RoboPhisher -p firmware_upgrade
```

```
[sudo] RoboPhisher --phishingscenario firmware_upgrade
```

Note: The name of the phishing scenario you specify here must match the folder name of the phishing scenario not it's actual name.

> **Warning:** This option will skip phishing scenario selection phase. This option will also raise an error if the specified phishing scenario is not found.

-pk
--presharedkey

Add WPA/WPA2 protection on the rogue access point.

Hint:

```
[sudo] RoboPhisher -pk s3cr3tp4ssw0rd
```

```
[sudo] RoboPhisher --presharedkey s3cr3tp4ssw0rd
```

--log-file

Enable logging information to a file.

Hint:

```
[sudo] RoboPhisher --log-file
```

Warning: This argument will only keep the three most recent logs. This means that after the fifth execution with the logging option it will overwrite the oldest log.

1.4 Frequently Asked Questions

1.4.1 How to check wireless card compatibility with RoboPhisher?

1.4.2 What are some effective wireless adapters?

Name	AP Support	Monitor Support
TP-LINK TL-WN722N	✓	✓

1.5 Phishing Scenarios

1.5.1 Templates

RoboPhisher supports community-built templates for different phishing scenarios. Currently, the following phishing scenarios are in place

Firmware Upgrade Page

A router configuration page without logos or brands asking for WPA/WPA2 password due to a firmware upgrade.

OAuth Login Page

A free Wi-Fi Service asking for Facebook credentials to authenticate using OAuth.

Warning: The template is **not** mobile friendly.

Browser Plugin Update

A generic browser plugin update page that can be used to serve payloads to the victims.

Warning: The template is **not** mobile friendly.

Note: The template support payloads.

Network Manager Connect

Imitates the behaviour of the network manager. This template shows Chrome's "Connection Failed" page and displays a network manager window through the page asking for the pre-shared key. Currently, the network managers of Windows and MAC OS are supported.

Warning: The template is **not** mobile friendly. Also this template only imitates Windows or MAC OS.

1.5.2 Creating a custom phishing scenario

For specific target-oriented attacks, custom scenarios may be necessary. Creating a phishing scenario is easy and consists of two steps

Create the `config.ini`

A config.ini file lies in template's root directory and its contents can be divided into two sections

1. Info: This section defines the scenario's characteristics.

- **Name** (mandatory): The name of the phishing scenario
- **Description** (mandatory): A quick description (<50 words) of the scenario
- **PayloadPath** (optional): If the phishing scenario pushes malwares to victims, users can insert the absolute path of the malicious executable here

2. Context: This section is optional and holds user-defined variables that may be later injected to the template.

Here's an example of a config.ini file

```
# This is a comment
[info]
Name: ISP warning page
Description: A warning page from victim's ISP asking for DSL credentials

[context]
victim_name: John Phisher
victim_ISP: Interwebz
```

Create the template files

A template contains the static parts of the desired HTML output and may consist of several static `HTML` files, images, `CSS` or `Javascript` files. Dynamic languages (e.g. `PHP`) are not supported.

Placeholders

The HTML files may also contain some special syntax (think placeholders) describing how dynamic content will be inserted. The dynamic content may originate from two sources

Beacon frames

Beacon frames contain all the information about the target network and can be used for information gathering. The main process gathers all the interesting information and passes them to the chosen template on the runtime.

At the time of writing, the main process passes the following data

- `target_ap_essid` <`str`>: The ESSID of the target Access Point
- `target_ap_bssid` <`str`>: The BSSID (MAC) address of the target Access Point
- `target_ap_channel` <`str`>: The channel of the target Access Point
- `target_ap_vendor` <`str`>: The vendor's name of the target Access Point
- `target_ap_logo_path` <`str`>: The relative path of the target Access Point vendor's logo in the filesystem
- `APs_context` <`list`>: A list containing dictionaries of the Access Points captured during the AP selection phase
- `AP` <`dict`>: A dictionary holding the following information regarding an Access Point
  - `channel` <`str`> The channel of the Access Point
  - `essid` <`str`> The ESSID of the Access Point
  - `bssid` <`str`> The BSSID (MAC) address of the Access Point
  - `vendor` <`str`> The vendor's name of the Access Point

Note that the above values may be 'None' accordingly. For example, all the target_* values will be None if there user did not target an Access Point (by using –essid option). The `target_ap_logo_path` will be None if the logo of the specific vendor does not exist in the repository.

`config.ini` file

All the variables defined in the *Create the config.ini* section may be used from within the template files. In case of naming conflicts, the variables from the configuration file will override those coming from the beacon frames.

1.6 Module Documentaion

1.6.1 Interfaces Module

1.6.2 Firewall Module

This module handles all the routing and firewall related tasks

`robophisher.common.firewall.`**`clear_rules`**`()`

Clear(reset) all the firewall rules back to default state and return a tuple containing completion status followed by the first error that occurred or None

Returns A tuple containing completion status followed by an error or None

Return type namedtuple(status=bool, error_message=None or str)

Example

```
>>> clear_rules()
Result(status=True, error_message=None)
```

```
>>> clear_rules()
Result(status=False, error_message="SOME ERROR HAPPENED")
```

`robophisher.common.firewall.`**`redirect_to_localhost`**`()`

Configure firewall such that all request are redirected to local host

Returns A namedtuple containing completion status followed by an error or None

Return type Result(status=bool, error_message=None or str)

Example

```
>>> redirect_to_localhost()
Result(status=True, error_message=None)
```

```
>>> redirect_to_localhost()
Result(status=False, error_message="SOME ERROR HAPPNED")
```

CHAPTER 2

Indices and tables

- genindex
- modindex
- search

Python Module Index

r

robophisher.common.firewall, 8

Index

C

clear_rules() (in module robophisher.common.firewall), 8

R

redirect_to_localhost() (in module robophisher.common.firewall), 8

robophisher.common.firewall (module), 8