Bashfuscator Documentation

Release 0.0.1

Bashfuscator Team

Jan 27, 2019

Contents

Contents:

CHAPTER 1

Introduction

Bashfuscator is built to be a modular, flexible Bash obfuscation framework. It achieves this by organizing different obfuscation techniques and methods into modules within the framework, called Mutators. Different obfuscation 'recipes' can be created by stacking different Mutators.

1.1 Mutator types

There are 5 types of Mutators:

1. **Command Obfuscators**
    - Simple obfuscators that leverage behavior of commands or binaries present in a Linux environment
    - Obfuscates entire input in one chunk
2. **String Obfuscators**
    - Obfuscators that use more advanced features/binaries
    - Breaks input into chunks, obfuscates those chunks, then builds up input by concatenating standard output of all of the different obfuscated chunks
3. **Token Obfuscators**
    - Leverages Bash functionality or behavior to obfuscate commands
    - Typically don't use any external binaries
    - Obfuscates entire input in one chunk
4. **Encoders**
    - Encodes the entire input and decodes it using a stub.
5. **Compressors**
    - Compresses the input and decompresses it using a stub, using various compressors typically available in a Linux environment

CHAPTER 2

Usage

2.1 Basic CLI Usage

Before starting to obfuscate for the first time, it may be useful to have a list of all of the available Mutators contained in the Bashfuscator framework. The *-l* option will do just that, and give size and time ratings, descriptions, and more.

```
$ bashfuscator -l
Command Obfuscators:

Name: Case Swapper
Description: Flips the case of all alpha chars
Size Rating: 1
Time Rating: 1
File write: False
Author: capnspacehook

Stubs:

    Name: bash case swap expansion
    Binaries Used: None
    Size Rating: 1
    Time Rating: 1
[snip]

Token Obfuscators:

Name: ANSI-C Quote
Description: ANSI-C quotes a string
Size Rating: 3
Time Rating: 1
Binaries used: None
File write: False
Notes: Requires Bash 4.2 or above
Author: capnspacehook
```

(continues on next page)

(continued from previous page)

```
Credits: DissectMalware, https://twitter.com/DissectMalware/status/1023682809368653826

Name: Special Char Only
Description: Converts commands to only use special characters
Size Rating: 4
Time Rating: 2
Binaries used: cat
File write: False
Notes: Will break when run in Bash's debug mode. Also compresses extremely well
Author: capnspacehook
Credits: danielbohannon, https://github.com/danielbohannon/Invoke-Obfuscation
        Digital Trauma, https://codegolf.stackexchange.com/questions/22533/weirdest-
↪obfuscated-hello-world
[snip]
```

When you're ready to start obfuscating, use the *-c* or *-f* options to specify a one-liner or script file to obfuscate, and Bashfuscator will take care of the rest, randomly choosing Mutators to obfuscate the input with. Bashfuscator only requires one of those two options, although many more are available to fine-tune the obfuscation.

```
$ bashfuscator -c "cat /etc/passwd"
[+] Payload:

eval "$(printf %s '")enod;"}]Y5DFSN$[A55_NI{$" s% ftnirp od;6 3 2 2 7 5 9 8 1 0 9 4 1␣
↪7 8 ni Y5DFSN rof;)/\ c a d p  \ w s t e(=A55_NI($" lave'|rev)"

[+] Payload size: 149 characters
```

The *-s* and *-t* options control the added size and execution time of the obfuscated payload respectively. They both default to 2, but can be set to 1-3 to control the generated payload more closely. The higher the *-s* or *-t* options, the greater the variety of the payload, at the expense of added size. When a low setting for the *-s* or *-t* options is set, Bashfuscator will limit itself to using Mutators that increase the size and execution time of the payload slightly. With high values, Bashfuscator has a chance to pick any combination of it's Mutators!

When you've finally settled on an obfuscation recipe to use, two output options are available: *–clip* and *-o*. *–clip* will automagically copy the obfuscated payload to your clipboard, and *-o* will write the payload to a file that you specify.

You should make sure the obfuscated payload works as expected, and the *–test* option will make that easier. When used, *–test* will invoke the obfuscated payload in memory, and show the output. When *-o* is used to specify an output file to write to, the output file will be run after the payload is written to it.

```
$ bashfuscator -c "cat /etc/passwd" --test
[+] Payload:

eval "$(ijmduN3D=(\[ r f 5 4 G U \" a i s p 1 t \% \} \  e \) \/ \\ 0 b J k z 7 \] \;␣
↪\{ \| D \( X 2 h 3 \= 9 V 8 w n \$ B c 6 d o);for s7SQJyu8 in 11 1 9 42 13 2 16 14␣
↪10 16 7 43 32 24 44 39 8 6 33 37 32 20 19 16 45 16 10 16 47 16 41 16 13 16 11 16 17␣
↪16 8 16 20 16 18 28 2 48 1 16 31 25 35 24 23 36 41 5 16 9 42 16 12 16 40 16 3 16 38␣
↪16 21 16 26 16 3 16 12 16 21 16 46 16 40 16 34 16 34 16 4 16 36 28 47 48 16 11 1 9␣
↪42 13 2 16 14 10 16 7 43 29 24 44 39 8 6 33 0 43 31 25 35 24 23 36 41 5 27 15 7 28␣
↪47 48 42 17 18 7 30 22 8 10 35;do printf %s "${ijmduN3D[$s7SQJyu8]}";done)"

[+] Payload size: 578 characters
[+] Testing payload:

root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
```

(continues on next page)

(continued from previous page)

```
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
[snip]
```

2.2 Advanced CLI Usage

Note: All options are tab-completable! This makes CLI usage sooo much easier, especially when using long options.

The *–layers* option will control the amount of obfuscation layers Bashfuscator will apply to the input. The default is 2 layers. This is useful to control the amount of obfuscation applied to the input.

The *–full-ascii-strings* option is an interesting one. When used, the full ASCII character set is used when randomly generating strings to be used within the final payload. This means non-printable characters can possibly exist within the obfuscated payload, potentially (hopefully) messing with tools and regex used to examine your payload.

You can further fine-tune the obfuscation process by using the *–choose-mutators* option. This option allows you to manually select which Mutators Bashfuscator will use, and in what order. You can create some pretty creative obfuscation recipes using this option. If used with the *–layer* option, the obfuscation recipe can be layered upon itself multiple times.

```
$ bashfuscator -c "cat /etc/passwd" --choose-mutators token/special_char_only␣
↪compress/bzip2 command/case_swapper command/reverse --layers 2 --test
[+] Payload:

printf %s "$(rev <<<'hsab|")"}~~BiT8Vs5Q{$" s% ftnirp;'"'"'HSAB|")C- 2PIZNUB|D-␣
↪46ESAB|GPgu/3qcfofj3f/
↪NZi1eLAfCjO8uB+++Xvp8RH+9+FFgauqrDLKMXJsr4xvIOpVD++89yEDpN1j9tHK5oGIcIoOgkisz1ImrbrWbaQikXLOGO0SMKo
↪wjFHVBHpNqx4JH3qvSeXHDNvtZ5YJ2YGRrtZQ1YO5YMY9Eo2c7z3UzYNbh3zUt62yoFLLwc9kcTS+GToSKM1R7MOvA6IOtXsDcl
↪VV+eFCYuYfidsiHLmvvPQkkvLkVeUV99/4t/HIO6+JF+Tpuev5/
↪051COOG1rvGtprv1WdKuh8nKaWOeaiIaXf7beyGyfsei9G22AZcEmUmo+iNF1+e+rLA/
↪3Zt7FV8P9UN9Q73SN5NdFidLYRjELTp5yOsO2VsqMqNVgXQ1YqruWDR/0dF6tOxzRVU+K+IO+GpFeb+O/
↪45jQ91ep9QzIVBwqRKzwMCmuses4HwkwRXwae1DZKB13uhj0mEwxBk78vWIiU8NQ8tc6dLd8jxG5u+axFb/
↪3FjjHKKaFp/N3bqxrHeHKmDw+7PNi5+Ks2NOFJIYfOUikS6HQ6olmpeOExLQWRTl/6pVIuUNEvn8c/
↪1R+Thb7IKrP8EpbF8Dz+4Rz8zTWHiQqj1QgEvBW8UvnIVAfUZ06Liisefxate WsDUT6ljz7tAFtJ3RrTvKapsajlcbacHefutSr
↪fgcJ50S4sxIsdFnbTGpeqLC/
↪s2njiIEd2Q0PVnkuTMBgp7lV6sL9wVvnvblvbbacamQivfdvS133vVRRzY9HasvnGIf32yUoLuxxYLQ4CZ20XjLgxK4CBzgZ2CU
↪xplrvmreHcsbJksvSefBPR5DXzwJJdEKx+glQBd5EMPRdTOGSDo+COOgF067HWBepUnoRorUnChhgnyv5KcjxNqfh1d4BeOdBUl
↪M3ZetPgvYrieVUVdjaHejazKa1tZ46Y1Xn8gwiFKVMNNrQ3HP5SX5dLWm048kdzAtB85kFr1y0JNMSeK1vU5s9IhhEN7DQcwQOu
↪aX8mO8KenP5XdGqaBtObLqigP8dHaVaX8d2gtEGbGFAWKN51emb4NMGjX208OHNbBBmHymqCu+Fm/
↪MFBAOTTXKM0MGTz+16/3qBBJfVx95j9Q5558WN8PtJbufHUJa1OiyAlH/
↪swrhvwswUDlR1HkVUlrZuU+i78SFEGYZ6JQI/
↪XbSV8LFEKp0pFOiQ2ZzMR11CMDorXvnxQVX35BEQCTeBrEm9+WEvu0+1rTR4rIVzXynZmZOPtV09TVUG08SN9+wYDdzbrLUen0X
↪K3Z1SxNS5X3+4ZL3rhikf0k35oJhfLj7QdCWMHJRYqw9wMbNVKczv9CGg7C9EddoyY9Iz60ZPdxu49zJnwyI71vhpKBM9+o1kPr
↪itdwQBokRMkXmJgFYCoL4hUzVloS9Ox3woisp2TqJWSDPKVx6SMqKXZyQgKwnPZW7r9iVv7lCBrzGNvRbUjo50qDMbNkBjES3mC
↪T+y7p+rp6T+6lEgF9RBDs32VYJsr5fHkH/
↪fpQna1daGgqt6bn0KIAQKc0O6hjpPn1JAJYtmfBOYmPeeHeHrtdbmnnIHmHjmrPFQsLG+PgGAd0Kr9qM0gnftEOHg1tKObAAaaA
↪9b+V39gG/XyXayQb1/DwtzswbftoOPLq FTNIRP($" S% FTNIRP'"'"'=BiT8Vs5Q($" s% ftnirp')
↪"|bash

[+] Payload size: 5810 characters
```

(continues on next page)

(continued from previous page)

```
[+] Testing payload:

root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
[snip]
```

Tab-completion is especially helpful when using the *–choose-mutators* option.

```
$ bashfuscator -c "cat /etc/passwd" --choose-mutators
command/case_swapper     compress/bzip2           encode/base64            string/
↪file_glob         string/forcode           token/ansi-c_quote
command/reverse          compress/gzip            encode/urlencode         string/
↪folder_glob       string/hex_hash          token/special_char_only
$ bashfuscator -c "cat /etc/passwd" --choose-mutators string/
string/file_glob    string/folder_glob  string/forcode      string/hex_hash
$ bashfuscator -c "cat /etc/passwd" --choose-mutators string/
```

CHAPTER 3

Module Documentation

3.1 common - Modules used all over the framework

3.1.1 *colors.py*

Provides helper functions for pretty-printing with colors. This file is borrowed from the Cuckoo Sandbox Project: https://github.com/cuckoosandbox/cuckoo

bashfuscator.common.colors.**color**(*text*, *colorCode*)
Colorize input text, on supported terminals.

Parameters

- **text** – text to color
- **color_code** – color to colorize text

Returns colorized text

bashfuscator.common.colors.**black**(*text*)
Returns black text on supported terminals.

bashfuscator.common.colors.**red**(*text*)
Returns red text on supported terminals.

bashfuscator.common.colors.**green**(*text*)
Returns green text on supported terminals.

bashfuscator.common.colors.**yellow**(*text*)
Returns yellow text on supported terminals.

bashfuscator.common.colors.**blue**(*text*)
Returns blue text on supported terminals.

bashfuscator.common.colors.**magenta**(*text*)
Returns magenta text on supported terminals.

bashfuscator.common.colors.**cyan**(*text*)
Returns cyan text on supported terminals.

bashfuscator.common.colors.**white**(*text*)
Returns white text on supported terminals.

bashfuscator.common.colors.**bold**(*text*)
Returns bold text on supported terminals.

3.1.2 *messages.py*

Helper functions to print output to the terminal.

bashfuscator.common.messages.**activateQuietMode**()
Activate quiet mode, only print errors.

bashfuscator.common.messages.**printInfo**(*msg*)
Format and print informational messages to the terminal.

bashfuscator.common.messages.**printWarning**(*msg*)
Format and print warning messages to the terminal.

bashfuscator.common.messages.**printError**(*msg*)
Format and print error messages to the terminal.

bashfuscator.common.messages.**printExitMsg**(*msg*)
Format and print exit messages to the terminal.

3.1.3 *objects.py*

3.1.4 *random.py*

3.2 core - Modules core to the framework's functionality

3.2.1 *mangler.py*

3.2.2 *obfuscation_handler.py*

3.3 lib - Modules used for obfuscation

3.3.1 *command_obfuscators.py*

3.3.2 *compressors.py*

3.3.3 *encoders.py*

3.3.4 *string_obfuscators.py*

3.3.5 *token_obfuscators.py*

genindex

Python Module Index

b

Index