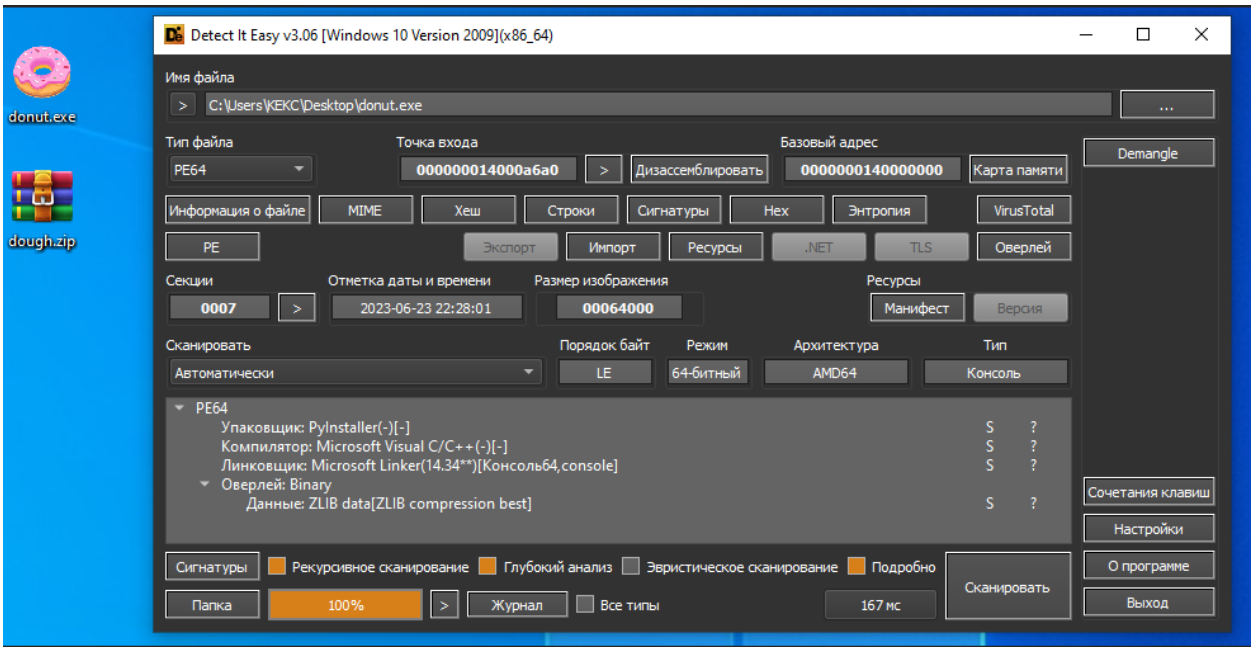
Название:	`Пончик`
Категория:	`Реверс-инжиниринг`
Уровень:	`Легко`
Очки:	`300`

Описание:	`Я вкус пончиков обожаю,`
	`Но безопасность мне нужна.`
	`EXE, как тайный лабиринт воспринимаю,`
	`И точно знаю, что могу ревёрсить я без сна.`

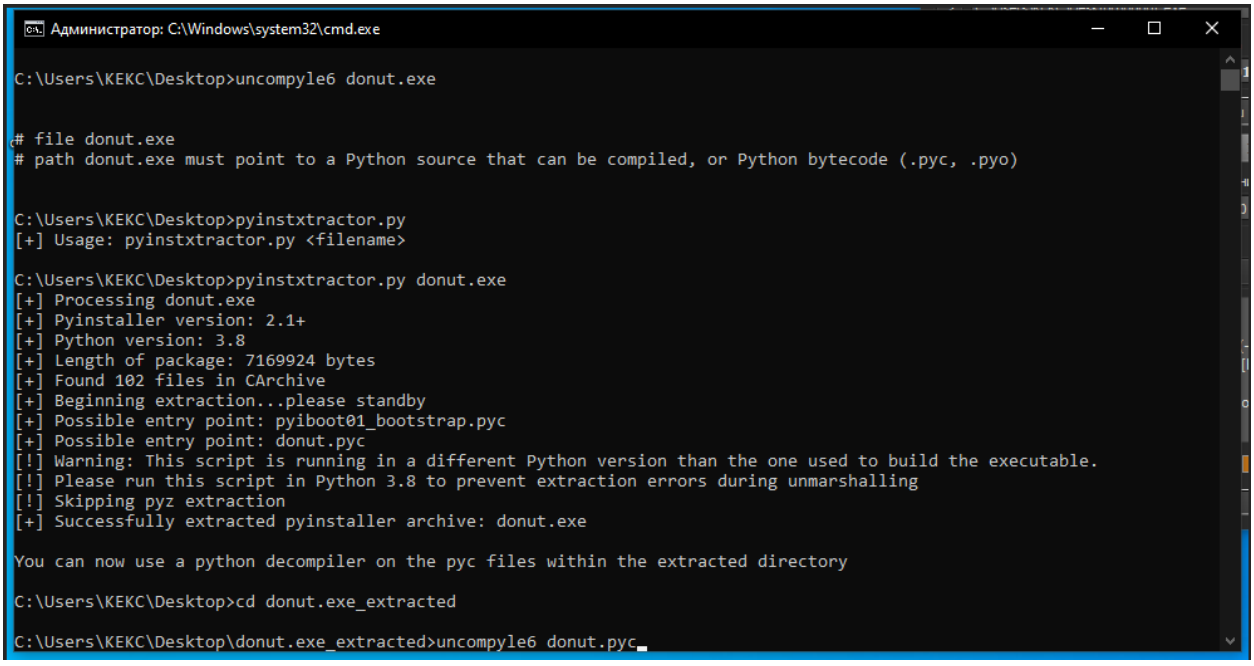
Теги:	`Python, PyInstaller`
Автор:	`ROP`

Прохождение:

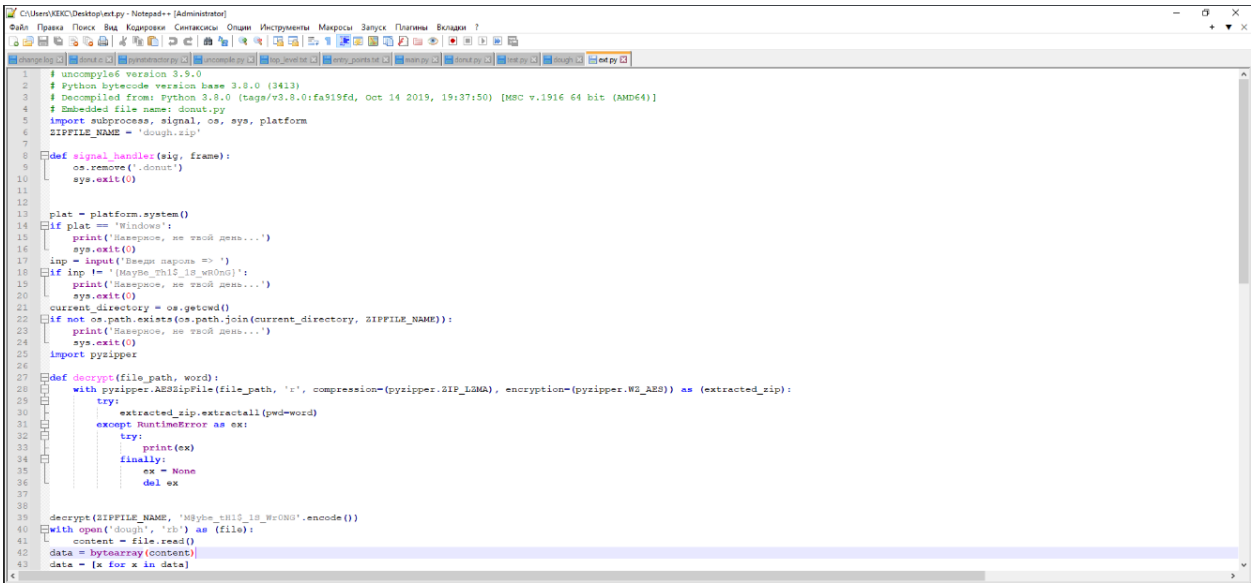
Изучаем файл donut.exe в DIE. Видим, что это pyinstaller.



Используем pyinstxtractor (https://github.com/extremecoders-re/pyinstxtractor) uncompyle6 (установка через pip) и получаем исходник.



Получили декомпилированный код.



```
# uncompyle6 version 3.9.0
# Python bytecode version base 3.8.0 (3413)
# Decompiled from: Python 3.8.0 (tags/v3.8.0:fa919fd, Oct 14 2019, 19:37:50) [MSC v.1916 64 bit (AMD64)]
# Embedded file name: donut.py
import subprocess, signal, os, sys, platform
ZIPFILE_NAME = 'dough.zip'

def signal_handler(sig, frame):
    os.remove('.donut')
    sys.exit(0)


plat = platform.system()
if plat == 'Windows':
    print('Наверное, не твой день...')
    sys.exit(0)
inp = input('Введи пароль => ')
if inp != 'M@ybe_tH1$_1S_Wr0NG':
    print('Наверное, не твой день...')
    sys.exit(0)
current_directory = os.getcwd()
if not os.path.exists(os.path.join(current_directory, ZIPFILE_NAME)):
    print('Наверное, не твой день...')
    sys.exit(0)
import pyzipper

def decrypt(file_path, word):
    with pyzipper.AESZipFile(file_path, 'r', compression=(pyzipper.ZIP_LZMA), encryption=(pyzipper.WZ_AES)) as (extracted_zip):
        try:
            extracted_zip.extractall(pwd=word)
        except RuntimeError as ex:
            try:
                print(ex)
            finally:
                ex = None
                del ex


decrypt(ZIPFILE_NAME, 'M@ybe_tH1$_1S_Wr0NG'.encode())
with open('dough', 'rb') as (file):
    content = file.read()
data = bytearray(content)
data = [x for x in data]
```

Изучаем его и понимаем, какой пароль у архива, что идёт вместе с EXE-файлом. Так же понимаем, что файл в архиве запускается в Linux.

Пароль к архиву: `M@ybe_tH1$_1S_Wr0NG`

Так же этот скрипт декодирует файл. Мы можем его вручную восстановить и убрать ошибки uncompyle6:

```
import subprocess
import signal
import os
import sys
import platform

ZIPFILE_NAME = 'dough.zip'

def signal_handler(sig, frame):
    os.remove('.donut')
    sys.exit(0)

plat = platform.system()

if plat == "Windows":
```

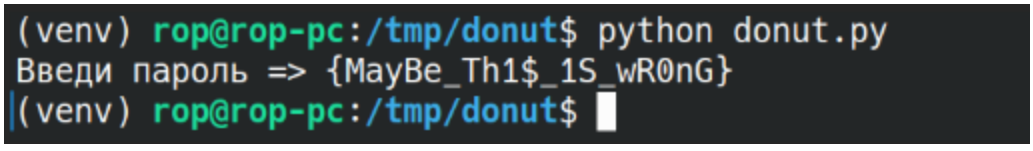
```
    print("Наверное, не твой день...")
    sys.exit(0)

inp = input("Введи пароль => ")
if inp != "{MayBe_Th1$_1S_wR0nG}":
    print("Наверное, не твой день...")
    sys.exit(0)

current_directory = os.getcwd()

if not os.path.exists(os.path.join(current_directory, ZIPFILE
_NAME)):
   print("Наверное, не твой день...")
   sys.exit(0)

import pyzipper


def decrypt(file_path, word):
    with pyzipper.AESZipFile(file_path, 'r', compression=pyzi
pper.ZIP_LZMA, encryption=pyzipper.WZ_AES) \\
            as extracted_zip:
        try:
            extracted_zip.extractall(pwd=word)
        except RuntimeError as ex:
            print(ex)

decrypt(ZIPFILE_NAME, "M@ybe_tH1$_1S_Wr0NG".encode())



# Открываем файл в бинарном режиме
with open('dough', 'rb') as file:
    content = file.read()
data = bytearray(content)
data = [x for x in data]
```

```
# Добавляем к каждому элементу массива 0x50
for i in range(len(data)):
    data[i] = (data[i] + 0x50)%256

# Выполняем XOR с 0x50
for i in range(len(data)):
    data[i] = (data[i] ^ 0x50)%256

# Записываем результат в файл .donut
with open('.donut', 'wb') as f:
    for b in data:
        f.write(bytes([b]))

subprocess.run(['chmod', '+x', '.donut'])
os.remove('dough')

# Запускаем файл .donut
signal.signal(signal.SIGINT, signal_handler)
subprocess.run(['./.donut'])
```

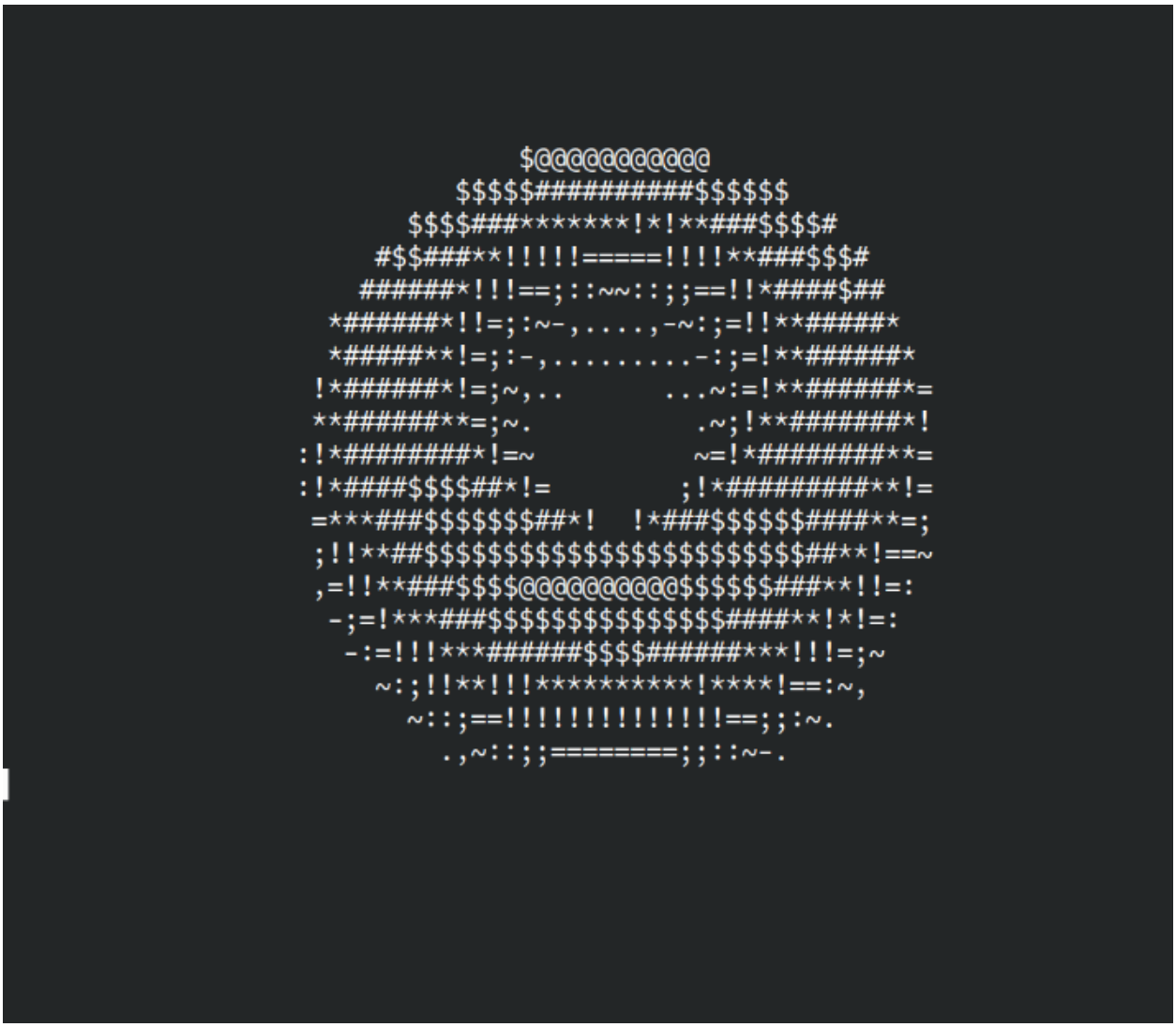
Уберём строки ниже и декодер готов.

```
signal.signal(signal.SIGINT, signal_handler)
subprocess.run(['./.donut'])
```

Пароль для декодирования: `{MayBe_Th1$_1S_wR0nG}`



Попробуем запустить файл `.donut`.



Видим пончик. Если заглянем к нему в строки, то увидим исходник и части флага.

```
(venv) rop@rop-pc:/tmp/donut$ strings .donut
/lib64/ld-linux-x86-64.so.2
_ITM_deregisterTMCloneTable
__gmon_start__
_ITM_registerTMCloneTable
putchar
__libc_start_main
__cxa_finalize
memset
printf
libm.so.6
libc.so.6
GLIBC_2.2.5
GLIBC_2.34
PTE1
u+UH
gfffH
.,-~:;=!*#$@
9*3$"
           int k;double sin()
         ,cos();main(){float A=
       0,B=0,i,j,z[1760];char b[
     1760];printf("\x1b[2J");for(;;
  ){memset(b,32,1760);memset(z,0,7040)
  ;for(j=0;6.28>j;j+=0.07)for(i=0;6.28
 >i;i+=0.02){float c=sin(i),d=cos(j),e=
 sin(A),f=sin(j),g=cos(A),h=d+2,D=1/(c*
 h*e+f*g+5),l=cos      (i),m=cos(B),n=s\
in(B),t=c*h*g-f*        e;int x=40+30*D*
(l*h*m-t*n),y=            12+15*D*(l*h*n
+t*m),o=x+80*y,          N=8*((f*e-c*d*g
 )*m-c*d*e-f*g-l        *d*n);if(22>y&&
 y>0&&x>0&&80>x&&D>z[o]){z[o]=D;;;b[o]=
 ".,-~:;=!*#$@"[N>0?N:0];}}/*****!!-*/
  printf("\x1b[H");for(k=0;1761>k;k++)
   putchar(k%80?b[k]:10);A+=0.04;B+=
     0.02;}}/*********#########!!=;:~
       ~::==!!!CODEBY{****!!!==::-
         .,~~=I;_;=l0vE;;_;:~-.
             ..,--d0nUt--}-,*/
```

Собираем части флага.