

IP发现:

```
Nmap scan report for 192.168.128.2
Host is up (0.00047s latency).
Not shown: 999 closed ports
PORT      STATE SERVICE
53/tcp    open  domain
MAC Address: 00:50:56:F3:E0:19 (VMware)

Nmap scan report for 192.168.128.138
Host is up (0.00042s latency).
Not shown: 997 filtered ports
PORT      STATE SERVICE
22/tcp    closed ssh
80/tcp    open  http
443/tcp   open  https
MAC Address: 00:0C:29:DC:2B:29 (VMware)

Nmap scan report for 192.168.128.254
Host is up (0.00015s latency).
All 1000 scanned ports on 192.168.128.254 are filtered
MAC Address: 00:50:56:E9:EE:45 (VMware)

Nmap scan report for 192.168.128.106
Host is up (0.000017s latency).
All 1000 scanned ports on 192.168.128.106 are closed

Nmap done: 256 IP addresses (5 hosts up) scanned in 10.71 seconds
root@kali:~#
```

端口扫描:

```
Nmap done: 256 IP addresses (5 hosts up) scanned in 10.71 seconds
root@kali:~# nmap -sV -O 192.168.128.138

Starting Nmap 7.60 ( https://nmap.org ) at 2018-06-10 03:06 CST
Nmap scan report for 192.168.128.138
Host is up (0.00085s latency).
Not shown: 997 filtered ports
PORT      STATE SERVICE VERSION
22/tcp    closed ssh
80/tcp    open  http    Apache httpd 2.4.18 ((Ubuntu))
443/tcp   open  ssl/http Apache httpd 2.4.18 ((Ubuntu))
MAC Address: 00:0C:29:DC:2B:29 (VMware)
Device type: general purpose
Running: Linux 3.X|4.X
OS CPE: cpe:/o:linux:linux_kernel:3 cpe:/o:linux:linux_kernel:4
OS details: Linux 3.10 - 4.8
Network Distance: 1 hop

OS and Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 23.51 seconds
root@kali:~#
```

访问web页面:



查看网站源码，发现一个目录：

```
view-source: http://192.168.128.138/index.html
DNT
Load URL
Split URL
Execute
Enable Post data
Enable Referrer
<body>
  <div id="social" class="visible-lg">
    <ul class="social-icons pull-right hidden-xs">
      <li class="social-rss">
        <a href="#" target="_blank" title="RSS"></a>
      </li>
      <li class="social-twitter">
        <a href="https://twitter.com/jamesbower" target="_blank" title="Twitter"></a>
      </li>
    </ul>
  </div>
  <!--[if IE 8]> <html lang="en" class="ie8"> <![endif]-->
  <!--[if IE 8]> <html lang="en" class="ie8"> <![endif]-->
  <!--[if IE 4]><script src="/oldIE/html5.js"></script><![Make sure to remove this before going to PROD]-->
  <!--[if !IE]><!--
  <!-- Header -->
  <div id="header" style="background-position: 50% 0%; height:100%; data-stellar-background-ratio="0.5">
    <div class="container">
      <div class="row">
        <!-- Logo -->
        <div class="logo">
          <a href="index.html" title="">
            
          </a>
        </div>
      </div>
    </div>
  </div>
  <!--[if !IE]><!--
```

访问文件，发现第一个flag，是十六进制编码：

```
1 666c61677b376330313332303730613065663731643534323636336539646331663564
65657d
```




所给的线索是nmap，用nmap 对所有65535端口进行更完整的扫描，确定服务器正在端口22222上运行SSH服务器。这一定是我进入服务器的方式

```
root@kali:~# echo 666c61677b37633031333230373061306566373164353432363633653964633166356465657d | xxd -r -p
flag{7c0132070a0ef71d542663e9dc1f5dee}root@kali:~#
root@kali:~# nmap -p- 192.168.128.138

Starting Nmap 7.60 ( https://nmap.org ) at 2018-06-10 03:57 CST
Stats: 0:02:03 elapsed; 0 hosts completed (1 up), 1 undergoing SYN Stealth Scan
SYN Stealth Scan Timing: About 67.22% done; ETC: 04:00 (0:01:00 remaining)
Nmap scan report for 192.168.128.138
Host is up (0.0018s latency).
Not shown: 65531 filtered ports
PORT      STATE SERVICE
22/tcp    closed ssh
80/tcp    open  http
443/tcp   open  https
22222/tcp open  easyengine
MAC Address: 00:0C:29:DC:2B:29 (VMware)
root@kali:~#
```

通过 ssh 连接，拿到了第二个 flag：
Flag{53c82eba31f6d416f331de9162ebe997}

```
root@kali:~# ssh 192.168.128.138 -p 22222
The authenticity of host '[192.168.128.138]:22222 ([192.168.128.138]:22222)' can't be established.
ECDSA key fingerprint is SHA256:DeCMZ74o5wesBHFLyaVY7UTCA7mW+bx6WroHm6AgMqU.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '[192.168.128.138]:22222' (ECDSA) to the list of known hosts.
#####
# WARNING #
# FBI - Authorized access only! #
# Disconnect IMMEDIATELY if you are not an authorized user!!! #
# All actions Will be monitored and recorded #
# Flag{53c82eba31f6d416f331de9162ebe997} #
#####
root@192.168.128.138's password:
Connection closed by 192.168.128.138 port 22222
```

md5解密flag，解密后的线索是：encrypt

密文: 53c82eba31f6d416f331de9162ebe997

类型: 自动 [帮助]

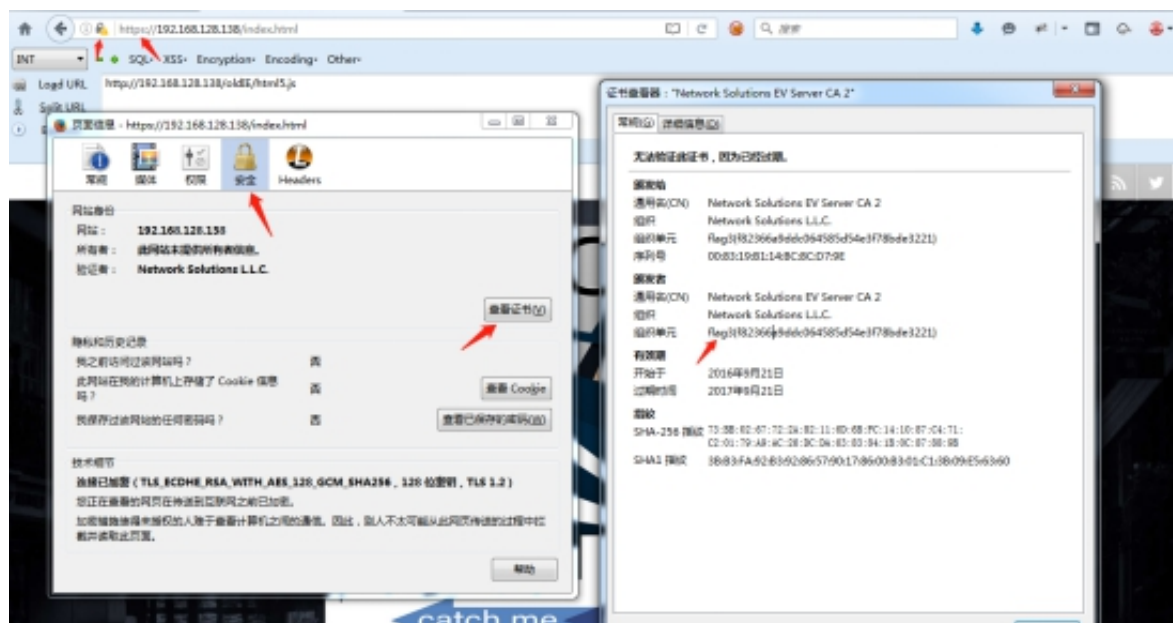
[查询](#) [加密](#)

查询结果:
encrypt

[添加备注](#)

本站对于md5、sha1、mysql、ntlm等的实时解密成功率在全球遥遥领先。成立13年，一直被抄袭，从未被超越。

- 1 线索是“加密”。到目前为止，截获流量和加密的唯一方法就是用于默认站点的SSL。所以我仔细看看SSL证书和BOOM。



拿到第三个flag: flag3{f82366a9ddc064585d54e3f78bde3221}

解密后的线索: personnel



发现personnel是一个目录, 结合flag信息, 又看了看之前找到的

http://192.168.128.138/oldIE/html5.js

发现如下信息

```
(a.compareDocumentPosition(b)&16);for(;b&&a!=b;)b=b.parentNode;return b==a},de=function(a){return 9==a.nodeType?a.ownerDocument||a.document},de=function(a){try{return a.contentWindow||a.contentDocument?ne(a.contentDocument):null}catch(b){return null}},Be=function(a,b){if("textContent" in a)a.textContent=b;else if(3==a.nodeType)a.data=b;else if(a.firstChild&&3==a.firstChild.nodeType){for(;a.lastChild!=a.firstChild;)a.removeChild(a.lastChild);a.firstChild.data=b}else te(a),a.appendChild(de(a).createTextNode(String(b))),De=function(a){var b=[];Ce(a,Hc,b,!1);return b},Ce=function(a,b,c,d){if(null!=a)for(a=a.firstChild;a;(if(b(a)&&(c.push(a),d)||Ce(a,b,c,d))return!0;a=a.nextSibling)return!1),Ee={SCRIPT:1,STYLE:1,HEAD:1,IFRAME:1,OBJECT:1},Fe=[IMG:" ",BR:"\n"],Ge=function(a,b){b?a.tabIndex=0:(a.tabIndex=-1,a.removeAttribute("tabIndex"))},Je=function(a){var b; /* maindev - 6/7/02 Adding temporary support for IE4 FBI Workstations */ /* newmaindev - 5/22/16 Last maindev was and idoit and IE4 is still Gold image -@Support doug.perterson@fbi.gov */ (b="A"==a.tagName||"INPUT"==a.tagName||"TEXTAREA"==a.tagName||"SELECT"==a.tagName||"BUTTON"==a.tagName?!a.disabled&&(!He(a)||Ie(a):He(a)&&Ie(a))&&B?(a=!pa(a.getBoundingClientRect)||B&&null==a.parentElement?(height:a.offsetHeight,width:a.offsetWidth):a.getBoundingClientRect(),a=null!=a&&0<a.height&&0<a.width):a=b;return a},He=function(a){a=a.getAttributeNode("tabindex");return null!=a&&a.specified},Ie=function(a){a=a.tabIndex;return oa(a)&&0<=a&&32768>a},Le=function(a){if(a&&null!=a&&"innerHTML" in a)a=a.innerHTML.replace(/(\r\n|\r|\n)/g,"\\n");else var b=[];Ke(a,b,!0);a=b.join("");a=a.replace(/\\xAD/g," ").replace(/\\xAD/g,"");a=a.replace(/\\u200B/g,"");ae||(a=a.replace(/ +/g," "));" "!=a&&(a=a.replace(/\\s*/," "));return a},Me=function(a){var b=[];Ke(a,b,!1);return b.join("")},Ke=function(a,b,c){if(!(a.nodeName in Ee))if(3==a.nodeType)c?b.push(String(a.nodeValue).replace(/(\r\n|\r|\n)/g," ")):b.push(a.nodeValue);else if(a.nodeName in Fe)b.push(Fe[a.nodeName]);else for(a=a.firstChild;a;)Ke(a,b,
```

FLAGS

Flag#1 - "Don't go Home Frank! There's a Hex on Your House"

Flag#2 - "Obscurity or Security? That is the Question"

Flag#3 - "During his Travels Frank has Been Known to Intercept Traffic"

Flag#4 - "A Good Agent is Hard to Find"

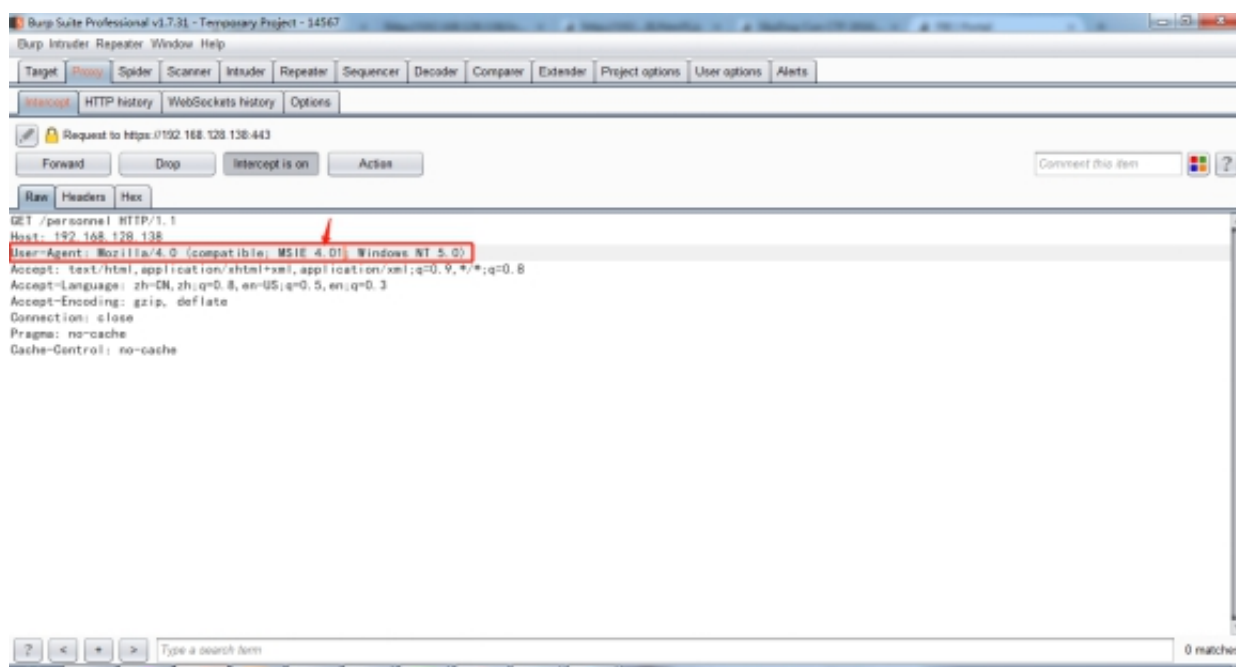
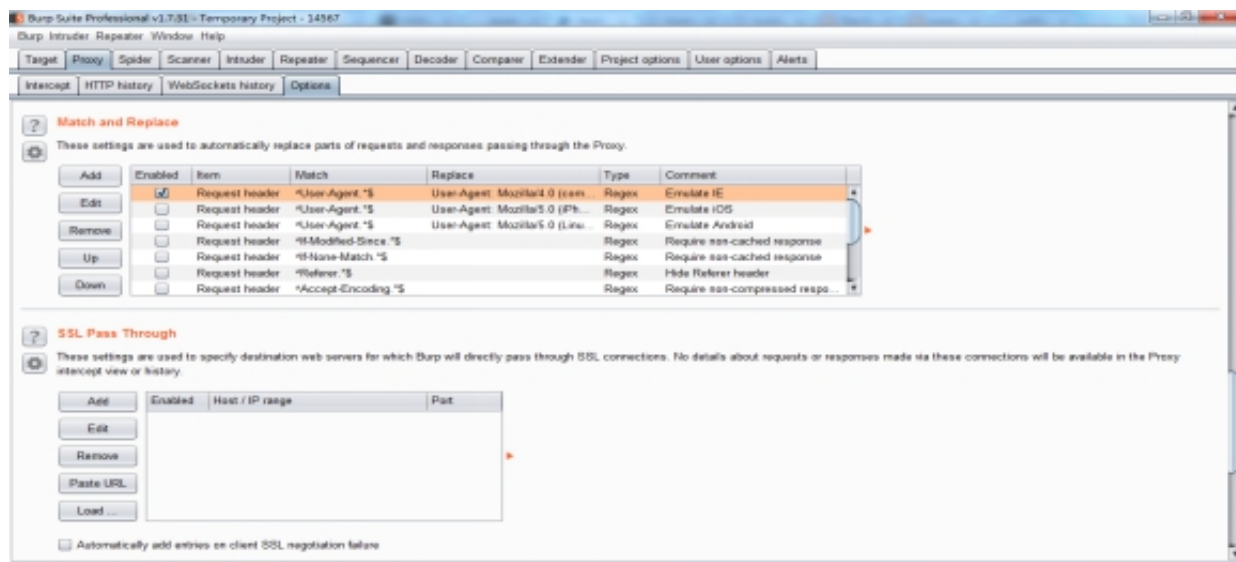
Flag#5 - "The Devil is in the Details - Or is it Dialogue? Either Way, if it's Simple, Guessable, or Personal it Goes Against Best Practices"

Flag#6 - "Where in the World is Frank?"

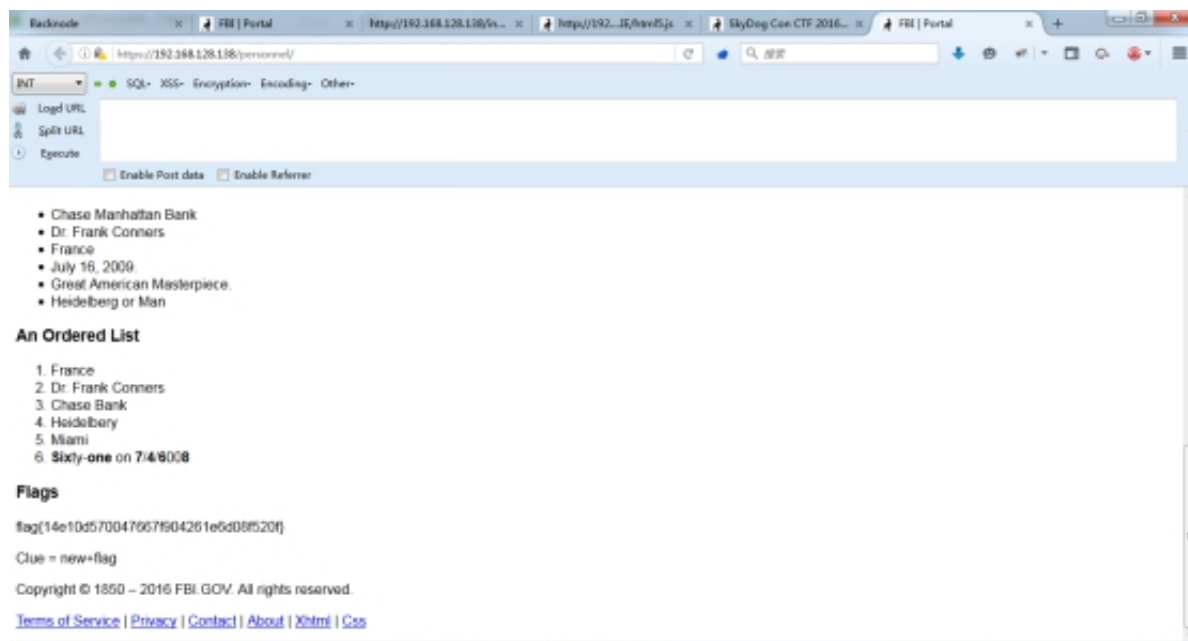
Flag#7 - "Frank Was Caught on Camera Cashing Checks and Yelling - I'm The Fastest Man Alive!"

Flag#8 - "Franks Lost His Mind or Maybe it's His Memory. He's Locked Himself Inside the Building. Find the Code to Unlock the Door Before He Gets Himself Killed!"

用burp挂上代理，修改如下地方， 修改User-Agent，使用IE4访问：



现在我用IE4用户代理刷新页面，我们迎接着欢迎Agent Hanratty的
 FBI门户网站
 浏览器css做了过滤，但是不影响找到flag, 第四个flag:
 flag{14e10d570047667f904261e6d08f520f}

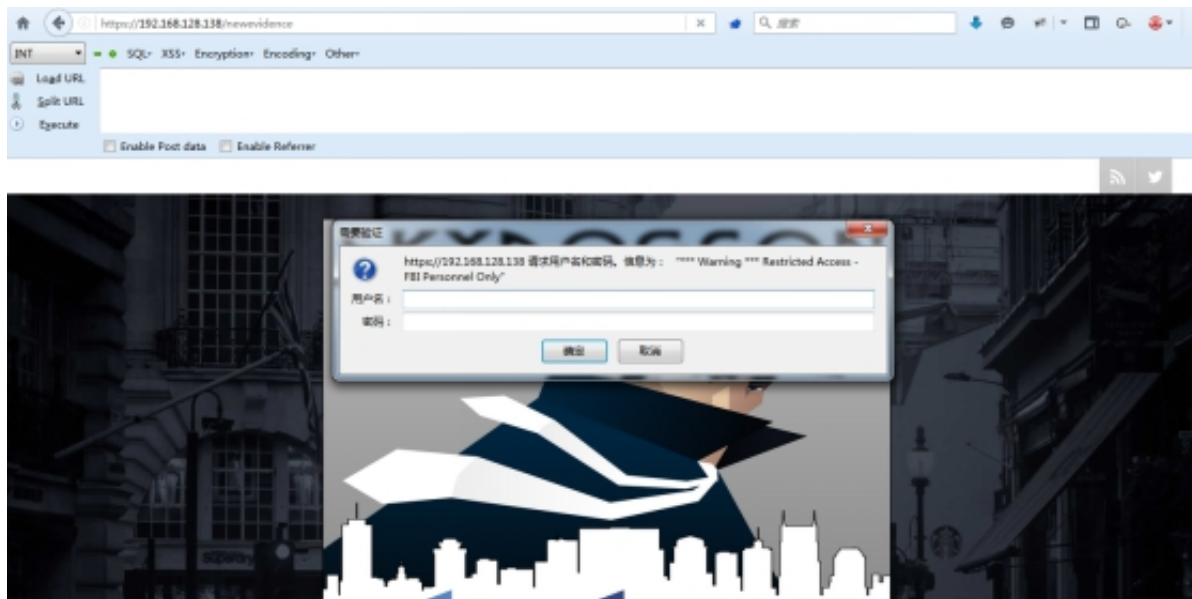


在门户网站的底部，我们发现我们的第四个标志 {14e10d570047667f904261e6d08f520f} 和一个新线索 “Clue = new+flag”。

解密后线索是：evidence



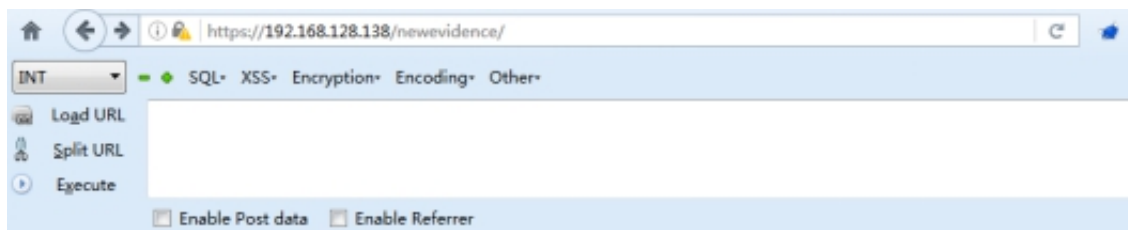
通过我们刚刚从Flag4得到的信息，猜测/newevidence是否是一个目录。确实是，但需要用户名/密码，而且要使用IE4用户代理登录。



用户名，密码怎么办，我在上一个页面用cewl生成了一个字典，用hydra果然跑出了口令没跑出来

然后又去搜了这个电影相关的信息，反正是找了好多信息终于找到口令：carl.hanratty/ Grace

登录任然需要代理采用IE4浏览器



New Evidence in Case# 982318212-A8732

This is the home page for all new evidence logged in for Case# 982318212-A8732.

Direct access for your agent level clearance can be executed below:

- [Evidence Summary File](#)
- [Possible Location](#)
- [Case Invoices](#)

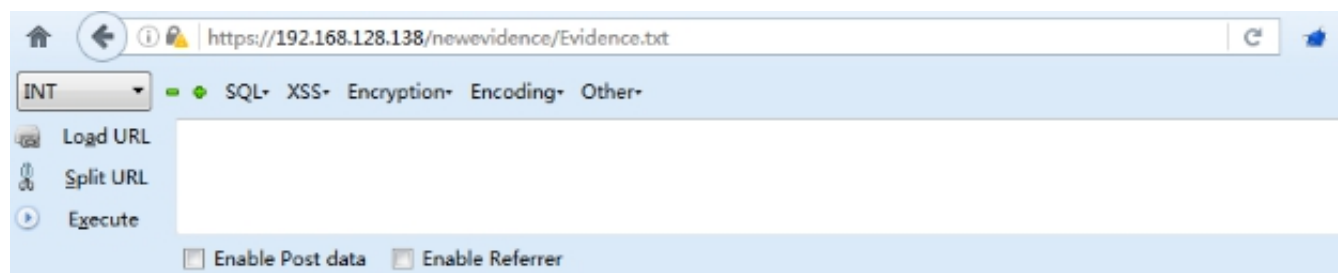


New Evidence in Case# 982318212-A8732

This is the home page for all new evidence logged in for Case# 982318212-A8732.

Direct access for your agent level clearance can be executed below:

- [Evidence Summary File](#)
- [Possible Location](#)
- [Case Invoices](#)



flag {117c240d49f54096413dd64280399ea9}

找到了第五个flag:

flag {117c240d49f54096413dd64280399ea9}

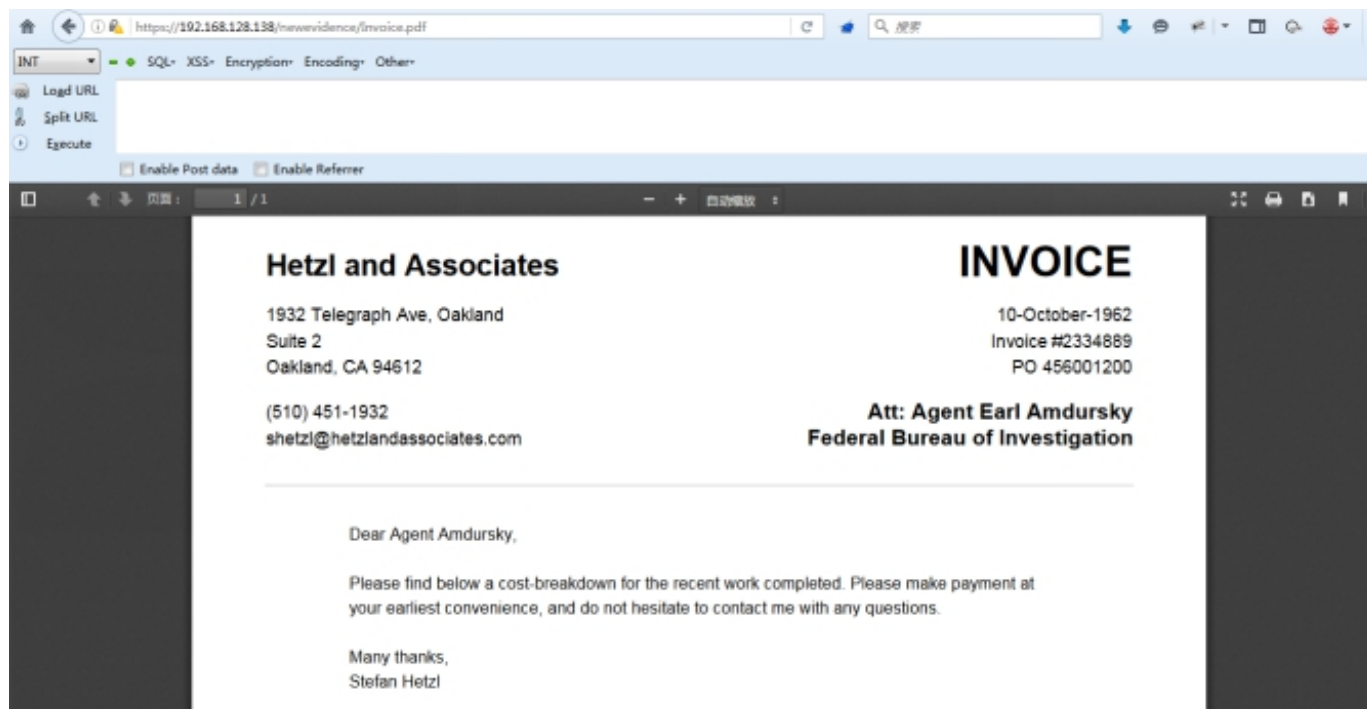
解密后的线索是: panam

密文: 117c240d49f54096413dd64280399ea9
类型: 自动 [帮助]
[查询] 加密

查询结果:
panam
[添加备注]

本站对于md5、sha1、mysql、ntlm等的实时解密成功率在全球遥遥领先。成立13年，一直被抄袭，从未被超越。

访问 两个文件image.jpg和Invoice.pdf



把图片下载下来，打开图片发现第六个flag
flag{d1e5146b171928731385eb7ea38c37b8}

解密后的线索是： ILoveFrance

```
root@Intrusion-Exploitation:~/Desktop/SKycon# steghide --info image.jpg
"image.jpg":
  format: jpeg
  capacity: 230.1 KB
Try to get information about embedded data ? (y/n) y
Enter passphrase:
  embedded file "flag.txt":
    size: 71.0 Byte
    encrypted: rijndael-128, cbc
    compressed: yes
root@Intrusion-Exploitation:~/Desktop/SKycon# steghide extract -sf flag.txt
Enter passphrase:
steghide: could not open the file "flag.txt".
root@Intrusion-Exploitation:~/Desktop/SKycon# steghide extract -sf image.jpg
Enter passphrase:
wrote extracted data to "flag.txt".
root@Intrusion-Exploitation:~/Desktop/SKycon# ls
flag.txt  image.jpg  Invoice.pdf
root@Intrusion-Exploitation:~/Desktop/SKycon# cat flag.txt
flag{d1e5146b171928731385eb7ea38c37b8}
=ILoveFrance

clue=iheartbrenda
root@Intrusion-Exploitation:~/Desktop/SKycon#
```

这是一个奇怪的线索。为什么弗兰克大喊“我是最快的人活着！”？这听起来很奇怪，我谷歌这句话，果然是超级英雄barry.allen；又名Flash。

现在我做了一堆“barry.allen”和“flash”的不同组合，现在使用我发现的任何凭据的唯一地方就是SSH。所以我尝试使用密码“iheartbrenda”的“barry.allen”，但这不起作用。接下来，我尝试使用“barryallen”和“iheartbrenda”作为密码，登陆成功。


```
root@kali:~# ssh barryallen@192.168.128.138 -p 22222
#####
#                               WARNING                               #
# the wall    FBI - Authorized access only!                          #
# Disconnect IMMEDIATELY if you are not an authorized user!!! #
# All actions Will be monitored and recorded                       #
# Flag{53c82eba31f6d416f331de9162ebe997}                          #
#####
barryallen@192.168.128.138's password:
Welcome to Ubuntu 16.04.1 LTS (GNU/Linux 4.4.0-38-generic x86_64)

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

235 packages can be updated.
136 updates are security updates.

barryallen@skydogconctf2016:~$ ls
flag.txt security-system.data
barryallen@skydogconctf2016:~$ cat flag.txt
flag{bd2f6a1d5242c962a05619c56fa47ba6}
barryallen@skydogconctf2016:~$
```

成功拿到flag，flag{bd2f6a1d5242c962a05619c56fa47ba6}
并且有一个名为“security-system.data”的非常大的文件。
解密flag的线索： theflash



密文: bd2f6a1d5242c962a05619c56fa47ba6
类型: 自动 [帮助]
[查询] 加密

查询结果:
theflash
[添加备注]

本站对于md5、sha1、mysql、ntlm等的实时解密成功率在全球遥遥领先。成立13年，一直被抄袭，从未被超越。

现在可以通过SSH访问Barry Allen帐户，我开始仔细查看主目录中的security-system.data文件。

在kali执行如下命令，将文件下载下来

```
scp -P 22222 barryallen@192.168.128.138:/home/barryallen/security-system.data ~/
```

```
root@kali:~# scp -P 22222 barryallen@192.168.128.138:/home/barryallen/security-system.data ~/
#####
# WARNING
# password: FBI - Authorized access only!
# Disconnect IMMEDIATELY if you are not an authorized user!!!
# All actions Will be monitored and recorded
# Flag{53c82eba31f6d416f331de9162ebe997}
#####
barryallen@192.168.128.138's password:
security-system.data 100% 1024MB 27.9MB/s###00:36#####
root@kali:~#
```

我将文件下载到Kali，查看它是什么类型的文件。该文件命令显示它是一个zip文件，所以我运行命令

```
root@kali:~# volatility -f '/root/security-system.data' pslist
Volatility Foundation Volatility Framework 2.6
Offset(V) Name PID PPID Thds Hnds Sess Wow64 Start Time(UTC) Exit
-----
0x867c6838 System 4 0 57 171 ----- 0
0x86262900 smss.exe 332 4 3 19 ----- 2016-10-10 21:59:14 UTC+0800
0x8623b978 csrss.exe 560 332 10 423 barrya0 2016-10-10 21:59:14 UTC+0800
0x865ed020 winlogon.exe 588 332 24 512 barrya0 2016-10-10 21:59:14 UTC+0800
0x8662d808 services.exe 664 588 15 263 alia0 2016-10-10 21:59:14 UTC+0800
0x866a5670 lsass.exe 676 588 25 356 rwxr0 2016-10-10 21:59:14 UTC+0800
0x86358a70 vmacthlp.exe 848 664 1 25 rwxr0 2016-10-10 21:59:14 UTC+0800
0x86651da0 svchost.exe 860 664 21 202 rwxr0 2016-10-10 21:59:14 UTC+0800
0x865c2790 svchost.exe 944 664 11 258 rwxr0 2016-10-10 21:59:14 UTC+0800
0x86554820 svchost.exe 1040 664 82 1287 rwxr0 2016-10-10 21:59:14 UTC+0800
0x866196b8 svchost.exe 1092 664 5 59 rwxr0 2016-10-10 21:59:14 UTC+0800
0x8643ca18 svchost.exe 1144 664 17 213 rwxr0 2016-10-10 21:59:15 UTC+0800
0x866fca88 explorer.exe 1540 1520 14 417 rwxr0 2016-10-10 21:59:16 UTC+0800
0x8656b4d0 spoolsv.exe 1636 664 15 125 rwxr0 2016-10-10 21:59:16 UTC+0800
0x86338640 VGAuthService.e 1900 664 2 60 rwxr0 2016-10-10 21:59:25 UTC+0800
0x8667bda0 vmtoolsd.exe 2012 664 9 271 rwxr0 2016-10-10 21:59:28 UTC+0800
0x864f6440 wmiprvse.exe 488 860 14 251 barrya0 2016-10-10 21:59:28 UTC+0800
0x864fbad0 wscntfy.exe 536 1040 1 311 rwxr0 2016-10-10 21:59:28 UTC+0800
0x85e5dd48 alg.exe 624 664 8 110 rwxr0 2016-10-10 21:59:28 UTC+0800
0x866f98b0 vmtoolsd.exe 1352 1540 7 242 barrya0 2016-10-10 21:59:29 UTC+0800
0x86674410 ctfmon.exe 1356 1540 1 79 rwxr0 2016-10-10 21:59:29 UTC+0800
0x865bea48 CCleaner.exe 1388 1540 5 108 rwxr0 2016-10-10 21:59:29 UTC+0800
0x865c3d78 cmd.exe 1336 1540 1 30 barrya0 2016-10-10 22:00:05 UTC+0800
0x8634fbb0 wuaucit.exe 1884 1040 9 198 rwxr0 2016-10-10 22:00:13 UTC+0800
0x86260a70 wuaucit.exe 1024 1040 6 172 rwxr0 2016-10-10 22:00:29 UTC+0800
```

现在security-system.data显示为简单的数据。在文件上运行字符串我看到很多提及内存的内容，所以我在想它是一台机器的内存映像。下一步是使用volatility来查看文件

```

root@kali:~# volatility -f '/root/security-system.data' imageinfo
Volatility Foundation Volatility Framework 2.6
INFO      : volatility.debug      : Determining profile based on KDBG search:
Suggested Profile(s) : WinXPSP2x86, WinXPSP3x86 (Instantiated with WinXPSP2x86)
AS Layer1 : IA32PagedMemoryPae (Kernel AS)
AS Layer2 : FileAddressSpace (/root/security-system.data)
PAE type  : PAE
DTB       : 0x33e000L
KDBG      : 0x80545b60L
Number of Processors : 1
Image Type (Service Pack) : 3
KPCR for CPU 0 : 0xffdff000L
KUSER_SHARED_DATA : 0xffdf0000L
Image date and time : 2016-10-10 22:00:50 UTC+0000
Image local date and time : 2016-10-10 18:00:50 -0400

```

确实能够显示一些有趣的信息。我继续使用volatility进一步挖掘。

```

root@kali:~# volatility -f '/root/security-system.data' --profile=WinXPSP2x86 iehistory
Volatility Foundation Volatility Framework 2.6
*****
Process: 1540 explorer.exe
Cache type "DEST" at 0x159d0f
Last modified: 2016-10-10 18:00:41 UTC+0000
Last accessed: 2016-10-10 22:00:42 UTC+0000
URL: test@file:///C:/Documents%20and%20Settings/test/Desktop/code.txt

```

我可以在桌面上看到一个名为code.txt的文件的引用，它是对我们线索的直接引用。

最后我能够抓住一些图像，但只有一个有可见的东西显示空的code.txt，但没有别的。我的下一步是查看是否有任何内容输入到控制台中。

```
root@kali:~# volatility -f '/root/security-system.data' --profile=WinXPSP2x86 consoles
Volatility Foundation Volatility Framework 2.6
*****
ConsoleProcess: csrss.exe Pid: 560
Console: 0x4f23b0 CommandHistorySize: 50
HistoryBufferCount: 1 HistoryBufferMax: 4
OriginalTitle: %SystemRoot%\system32\cmd.exe
Title: C:\WINDOWS\system32\cmd.exe
AttachedProcess: cmd.exe Pid: 1336 Handle: 0x2d4
----
CommandHistory: 0x10186f8 Application: cmd.exe Flags: Allocated, Reset 3 barryallen barryallen 4096 Jun 9 12:30 .
CommandCount: 2 LastAdded: 1 LastDisplayed: 1
FirstCommand: 0 CommandCountMax: 50
ProcessHandle: 0x2d4
Cmd #0 at 0x1024400: cd Desktop
Cmd #1 at 0x4f2660: echo 66 6c 61 67 7b 38 34 31 64 64 33 64 62 32 39 62 30 66 62 62 62 64 38 39 63 37 62 35 62 65 37 36 38 63 64 63 38 31 7d > code.txt
----
Screen 0x4f2ab0 X:80 Y:300
Dump:
Microsoft Windows XP [Version 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.

C:\Documents and Settings\test>cd Desktop
C:\Documents and Settings\test\Desktop>echo 66 6c 61 67 7b 38 34 31 64 64 33 64 62 32 39 62 30 66 62 62 62 62 64 38 39 63 37 62 35 62 65 37 36 38 63 64 63 38 31 7d > code.txt
code.txt
```

真棒！我可以看到code.txt是通过在文件中回显hex来在桌面上创建的。有时间看看十六进制表示什么，所以我再次运行xxd命令。

```
echo 66 6c 61 67 7b 38 34 31 64 64 33 64 62 32 39 62 30 66
62 62 64 38 39 63 37 62 35 62 65 37 36 38 63 64 63 38 31 7d
| xxd -r -p
```

```
root@kali:~# echo 66 6c 61 67 7b 38 34 31 64 64 33 64 62 32 39 62 30 66 62 62 62 64 38 39 63 37 62 35 62 65 37 36 38 63 64 63 38 31 7d | xxd -r -p
flag{841dd3db29b0fbbd89c7b5be768cdc81}root@kali:~#
```

最后的flag: flag{841dd3db29b0fbbd89c7b5be768cdc81}

解密: Twolittlemice

