Challenge 1

Alice has sent Bob an encrypted file. Find it, decrypt it, and find the secret inside.

Look in the alice.pcap file to answer this question.

Hint: Alice is often quite chatty with Bob, and phrases she references could be useful to use as passwords (or passphrases). You won't need to use wordlists, mutation, or brute-force of any kind to decrypt the encrypted file.

Whenever I come across a pcap, and I don't know exactly what I'm looing for, I like open it with three different tools almost immediately – Wireshark, NetworkMiner, and Cain&Abel. For this challenge, most of my work was done in Wireshark, but it's worth mentioning the other two tools because they have their own unique uses (and we will use Cain a little later on to get our hands on some NTLMv2 Hahses).

Start by opening alice.pcap in Wireshark. We know that Alice and Bob like to chat, and we can see at Frame 125 that the machine with IP Address 172.16.14.138 sent a DNS query for webchat.freenode.net. After that, we can see communication over HTTP that includes the IRC chat session. After a little inspection, I built this Wireshark Filter to show me the most interesting parts of that HTTP communication:

http and !(frame.len == 568) and !(frame.len == 212) and !(frame.len == 221) and !(frame.len == 567)

Removing the frames with lengths 568, 212, 221, and 567 result in most of the irrelevant Protocol overhead being stripped from view

In frame 133 we see the IRC nickname "AL1C3" sent to the IRC server, so we assume that Alice's computer is 172.16.14.138. AL1CE joins the #shmoocon channel and proceeds to have a series of Private Messages with "I_am_Bob". If you parse out the conversation from the HTTP data, this is what you find:

PRIVMSG #shmoocon :I_am_Bob: Hi there, Bob! You heading off to Shmoo next weekend?

[["c","PRIVMSG","I_am_Bob!

PRIVMSG #shmoocon :I_am_Bob: That's a shame! There's lots of excitement going on. Talks, events, labs... I even hear there's some kind of challenge involving placeholder names used in crypto.

PRIVMSG #shmoocon :I_am_Bob: Oh, and my favorite, there's a game going on that blends game hacking, first-person shooting, and role-playing mechanics!

[["c","PRIVMSG","I_am_Bob!

PRIVMSG #shmoocon :I_am_Bob: You'll have to check the website yourself ;)

PRIVMSG #shmoocon :I_am_Bob: By the way, I'll send you my latest message via SMB and an

encrypted zip file, per our normal protocol. Silly eavesdroppers...

PRIVMSG #shmoocon :I_am_Bob: See you soon!

[["c","PRIVMSG","I_am_Bob!

PRIVMSG #shmoocon :I_am_Bob: My pleasure

Now we know that a file has been transferred using SMB. In Wireshark, click File → Export Objects → SMB, and we see the "another_message.7z" that Alice referenced in her IRC message to Bob. We also see some other very suspicious files, "not_exactly_inconspicious.

Now that we have Alice's encrypted zip file, we need to open it. The hint said that phrases she references might be useful as passwords or passphrases. At this point, I began trying words and phrases copied directly from Alice's chat session. Eventually, after many failed attempts, I went to www.shmoocon.org to find the game that Alice referenced as being her favorite. That event was called "Ghost in the Shellcode". When that is used as a passphrase, it will decrypt the zip.

The secret is: Build It, Belay It, and Bring It On/

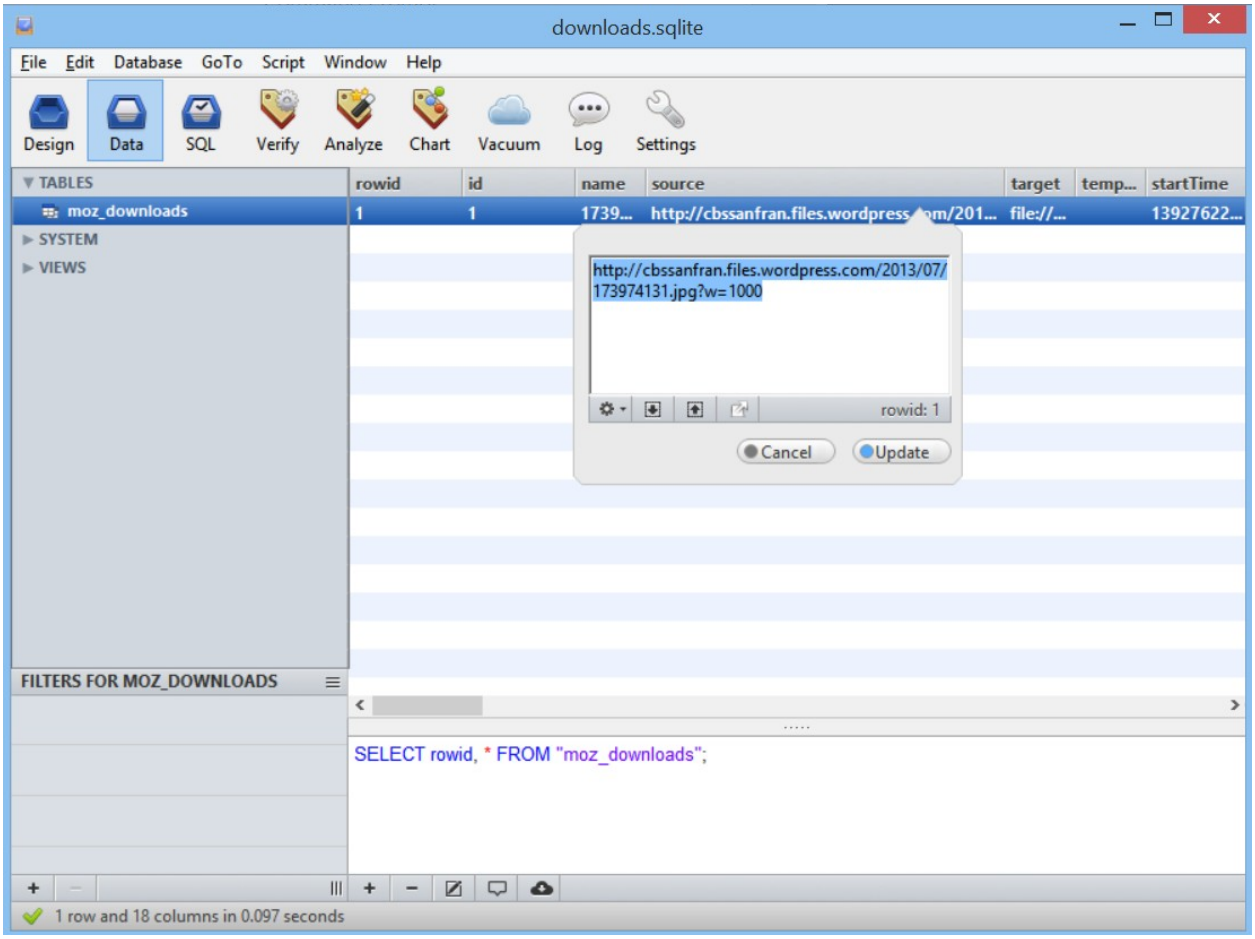
Challenge 2

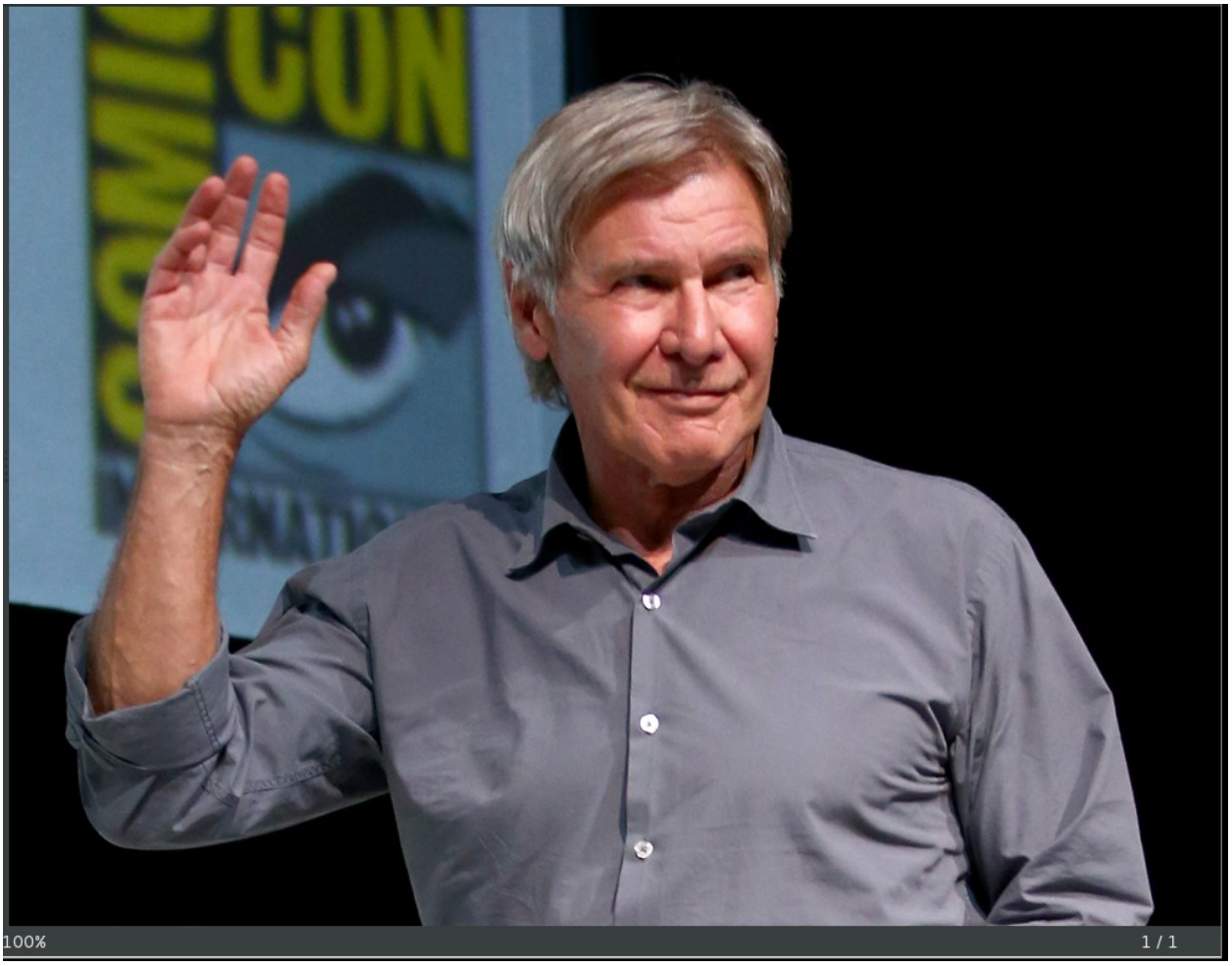
Carol has used Firefox for Android to search for, browse, and save a particular image. A compressed copy of her /data/data/org.mozilla.firefox folder is in the question_assets folder, named "org.mozilla.firefox.tgz". Find the serial number of the lens used to take the download picture, which is the secret for this question.

Hint: You may have to use resources outside the org.mozilla.firefox folder to fully answer this question.

7zip can open the "org.mozilla.firefox.tgz" file, as well as the "org.mozilla.firefox.tar" that is found inside. Once we have the uncompressed "org.mozilla.firefox" directory, we need to look for the downloads.sqlite file to look for the file she downloaded. That file is located in \files\mozilla\9tnld04f.

When you open downloads.sqlite with SQLite Manager, and view the moz_downloads table, you can see that Carol (a fan of Star Wars, and Han Solo in particular) downloaded a photo of Harrison Ford at the 2013 Comic Con from the CBS San Francisco Wordpress site:


downloads.sqlite
File Edit Database GoTo Script Window Help
Design Data SQL Verify Analyze Chart Vacuum Log Settings
TABLES
moz_downloads
SYSTEM
VIEWS
rowid id name source target temp... startTime
1 1 1739... http://cbssanfran.files.wordpress.com/201... file://... 13927622...
http://cbssanfran.files.wordpress.com/2013/07/173974131.jpg?w=1000
rowid: 1
Cancel
Update
FILTERS FOR MOZ_DOWNLOADS
SELECT rowid, * FROM "moz_downloads";
1 row and 18 columns in 0.097 seconds



The challenge asks for the Serial Number of the Lens used to take the picture. That information can be gathered by running from the exif data stored inside the 173974131.jpg file. Download a copy of the file, and run the command below in a Terminal to display exif data. the 64th line of the output is the Lense Serial Number.

```
exiftool /root/173974131.jpg
```

```
ExifTool Version Number         : 8.60
File Name                       : 173974131.jpg
Directory                       : /root
File Size                       : 362 kB
File Modification Date/Time     : 2015:02:11 20:11:38-05:00
File Permissions                : rw-r--r--
File Type                       : JPEG
MIME Type                       : image/jpeg
JFIF Version                    : 1.01
Exif Byte Order                 : Little-endian (Intel, II)
Photometric Interpretation      : RGB
Image Description               : SAN DIEGO, CA - JULY 18:  Actor Harrison Ford onstage at the
```

Owner Name :
Serial Number : 088015001238
Lens Info : 70-200mm f/0
Lens Model : EF70-200mm f/2.8L IS II USM
Lens Serial Number : 0000c15998
GPS Version ID : 2.3.0.0
Compression : JPEG (old-style)
Thumbnail Offset : 1752
Thumbnail Length : 5243
Current IPTC Digest : 4070c4df48c719664a9df0314ac3ea
Coded Character Set : UTF8
Application Record Version : 4
Caption-Abstract : SAN DIEGO, CA - JULY 18: Actor Harrison Ford onstage at the "Ender's Game" press conference during Comic-Con International 2013 at San Diego Convention Center on July 18, 2013 in San Diego, California. (Photo by Joe Scarnici/Getty Images for Summit Entertainment)
Writer-Editor : hg
Headline : "Ender's Game" Press Conference
By-line : Joe Scarnici
By-line Title : Stringer
Credit : Getty Images for Summit Entertai
Source : Getty Images North America
Object Name : 174014009HG00008_Ender_s_Ga
Date Created : 2013:07:18
Time Created : 00:00:00+00:00
City : San Diego
Sub-location : San Diego Convention Center
Province-State : CA
Country-Primary Location Name : United States
Country-Primary Location Code : USA
Original Transmission Reference : 174014009
Category : E
Supplemental Categories : ACE, CEL, ENT
Urgency : 2
Keywords : Celebrities
Copyright Notice : 2013 Getty Images
IPTC Digest : 4070c4df48c719664a9df0314ac3ea
Displayed Units X : inches
Displayed Units Y : inches
Global Angle : 30
Global Altitude : 30
Photoshop Thumbnail : (Binary data 5243 bytes, use -b option to extract)
Photoshop Quality : 12
Photoshop Format : Standard
Progressive Scans : 3 Scans
Profile CMM Type : Lino
Profile Version : 2.1.0
Profile Class : Display Device Profile
Color Space Data : RGB
Profile Connection Space : XYZ

Depth Of Field : 0.17 m (6.98 - 7.14)
Field Of View : 4.2 deg
Focal Length : 102.0 mm (35 mm equivalent: 490.0 mm)
Hyperfocal Distance : 594.07 m
Light Value : 5.3

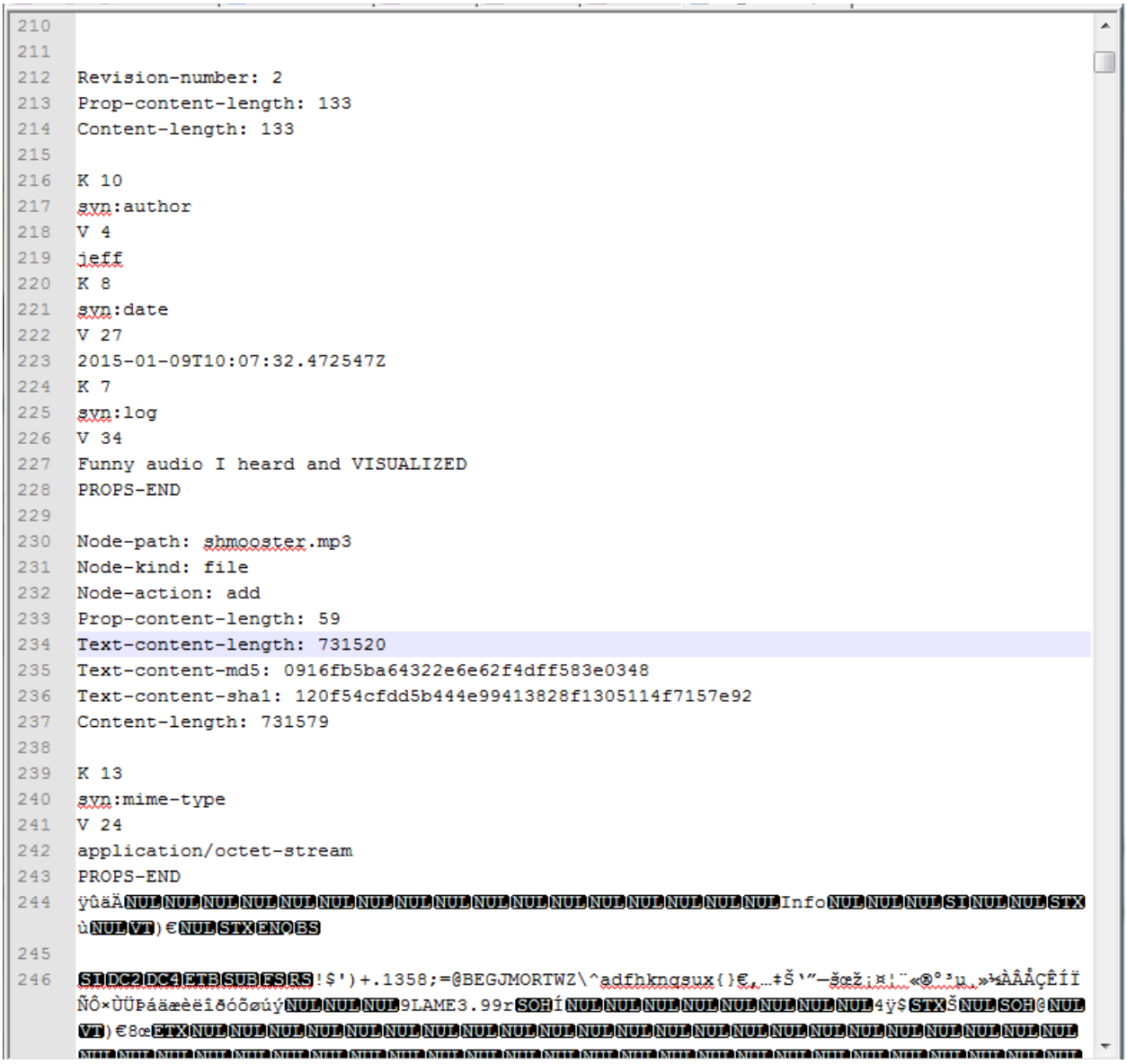
The Lens Serial Number is 0000c15998

Challenge 3

Dave messed up and deleted his only copy of an MP3 file. He'd really appreciate it if you could retrieve it for him - look inside svn_2015.dump.gz to get started.

Once you've recovered the audio file, look at it carefully to find the secret.

This file is a dump of an Apache Subversion Repository. One way to recover data from this file is to create a new Subversion Repository and load this dump into it. Since I don't really need the full repo I'm going to just carve it up with a text editor. For example, if we open it in Notepad++ and scroll down to line 212, we can see that Revision 2 included an audio file named shmooster.mp3.



Just delete everything from the start of the file until line 243 (the “PROPS-END” line) and from until line 7326 (just before the “Revision-Number 3” line) until the end of the file, and save it as shmooster.mp3. After you create the file, you can confirm its content by running a SHA1 or MD5 hash against it and comparing it to the results on lines 235 or 236 in the above screenshot.

When you listen to the mp3, it says:

Which of the following would you most prefer?

1. *A puppy*
2. *A pretty flower from your sweetie*

or

3. *A large properly formatted data file*

…..

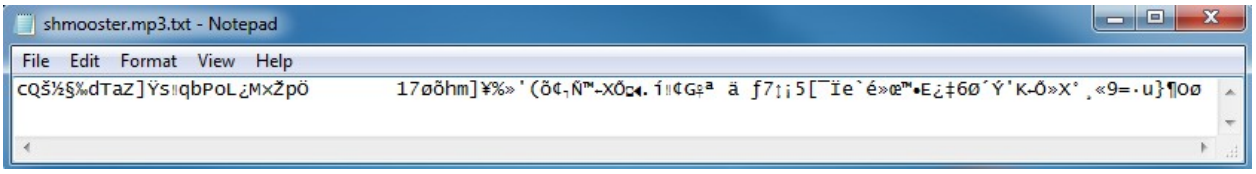
You have failed this Reverse Turning test. Now suffer the consequences.

The next few paragraphs on MP3Stego doesn't actually help solve the challenge - it was a dead end that may have intentionally included some mis-information.
The challenge said to look at the MP3 file carefully to find the secret. There were no ID3 tags included in the file, and no exif data of any use. Text files can be hidden in MP3s using the MP3Stego program, and the audio portion of the file is a hint to the password. When you use the password is "c", a text file is successfully extracted. Using MP3Stego we need to execute:
Decode.exe –X –P c \path\to\shmooster.mp3

The result is:

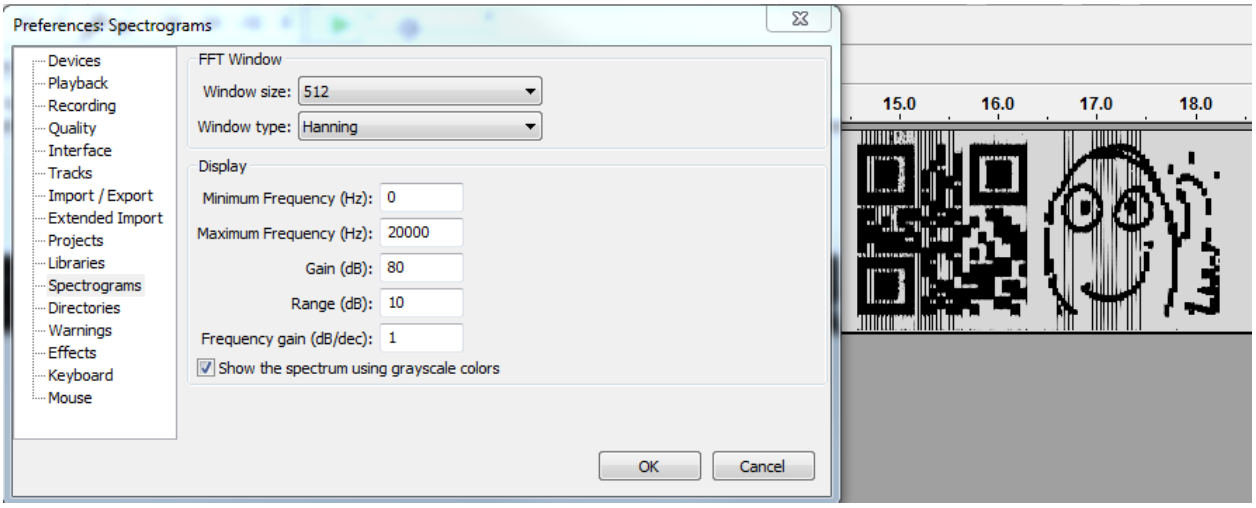
Input file = 'C:\path\to\shmooster.mp3' output file = 'mp3'
Will attempt to extract hidden information. Output: C:\path\to\shmooster.mp3.txt
the bit stream file C:\path\to\shmooster.mp3 is a BINARY file
HDR: s=FFF, id=1, l=3, ep=off, br=E, sf=1, pd=0, pr=0, m=3, js=0, c=0, o=1, e=0
alg.=MPEG-1, layer=III, tot bitrate=320, sfrq=48.0
mode=single-ch, sblim=32, jsbd=32, ch=1
Frame cannot be located
Input stream may be empty
Avg slots/frame = 960.002; b/smp = 6.67; br = 320.001 kbps
Decoding of "C:\path\to\shmooster.mp3" is finished
The decoded PCM output file name is "mp3"

The shmooster.mp3.txt file that is extracted contains the string of ASCII characters shown below. I cannot get that string to work, in combination with the other password, to open the open_this_to_win.7z file. I have tried almost countless manipulations by converting to Hex, Binary, Base64 encode/decode, URL encoding, etc, and can't get anything to work.



Is it an odd coincidence that text is successfully extracted using the password "c" with MP3Stego or did Dave intentionally embed bad information to keep his adversaries occupied with a red herring?

The real solution to Challenge 3 is to open the mp3 in Audacity and use the Spectrogram view to reveal a hidden QR code. The settings that I used were: Windows Size: 512, Window Type: Hannning, Min Freq -, Max Freq 20000, Gain 80, Range 10, Freq Gain 1, a Grayscale Colors. Below is a screenshot:



When you scan that QR code, the text “3e9cd9ea80d80606” is displayed.

The Secret in Challenge 3 is 3e9cd9ea80d80606

Challenge 4

> *Eve suspects that one of Alice, Bob, or Carol might not be as innocent as they seem. She'll need your help to prove it, however. Examine the other three questions and their included files. Which user, based off their malicious behavior, might be a Cylon?*
>
> *Once you know who it is, find that user's password, which is the secret for this question.*

Based on the additional files that Alice dropped on Bob’s PC, it’s fairly obvious that Alice isn’t very innocent. At frame 1016 of the pcap, we can see that Alice started flooding Bob’s PC with TCP Resets. We can also see in Frame 712’s DHCP request and the various SMB NTLMSSP_NEGOTIATE and NTLMSSP_AUTH frames (i.e. Frames 801, 803, 3336, 3338, etc) that Alice’s Host Name is “KALI”, which is a well-known and powerful Linux Security Distro.

If we open alice.pcap in Cain & Abel, and go to the Sniffer → Passwords Tab, we can see that Cain successfully extracted a bunch of hashes from Alice’s password from the pcap. Unfortunately, they are NTLMv2 hashes, and cracking them (even using a very efficient tool like oclHashcat with power GPUs) is not likely to happen in a timely manner. Out of curiosity, I did upload the hashes to an Amazon Web Services G2.2XLarge instance to see if they could be brute forced, but didn’t have a luck. The maximum length I ran was 6 characters (which takes about 4 hours). Beyond that, 7 characters takes a few days and 8 characters takes years. Had Alice’s password been 6 characters or less, I could have recovered it with oclHashcat. Below are the steps you would take to get oclHashcat running on an Amazon Web Services GPU Instance, and crack with oclHashcat:

First, you need to get an AWS account if you don’t already have one, and launch a GPU Instance (as of Feb 2015, it’s called an G2.2xlarge, and the OS it runs is Amazon Linux AMI).

As of now, it costs about $0.60 per hour to run. Follow Amazon's steps for authenticating to the console using SSH and a private key file (either PEM, or PPK if you're using PuTTy). To get oclHashcat (actually, cudaHashcat since we're using nVidia GPUs) running, I needed to remove the nVidia driver that's pre-installed, and install a driver directly from nVidia. If you don't have a proper driver, you will receive cuModuleLoad()209 errors when you try to execute the program . Run these commands:
First, download 7zip and cudaHashcat:
*wget rpmfind.net/linux/epel/6/x86_*

wget http://hashcat.net/files/

Install 7zip:
sudo rpm -ivh p7zip-9.20.1-2.el6.x86_64.rpm

extract the cudaHashcat compressed 7z file:
7za x cudaHashcat-1.32.7z

delete the driver:
sudo yum erase nvidia cudatoolkit

download the driver from nVidia and run it:
wget http://us.download.nvidia.com/
sudo /home/ec2-user/NVIDIA-Linux-

To extract the NTLMv2 Hashes from Cain and put them in the correct format for oclHashcat, you can take the NTLMv2.LST file from Cain's installation directory and run this AWK command against it:

awk -v OFS=":" -F "\t" '{print($1,"",$2,$5,$4,$6)}' NTLMv2.LST > ntlmv2.hashes

You can also do this manually, but running that command makes it easy (especially when dealing with many hashes). Here is an example of the proper format for the 3 hashes captured from alice.pcap:
alice::WORKGROUP:
alice::WORKGROUP:
Alice::WORKGROUP:

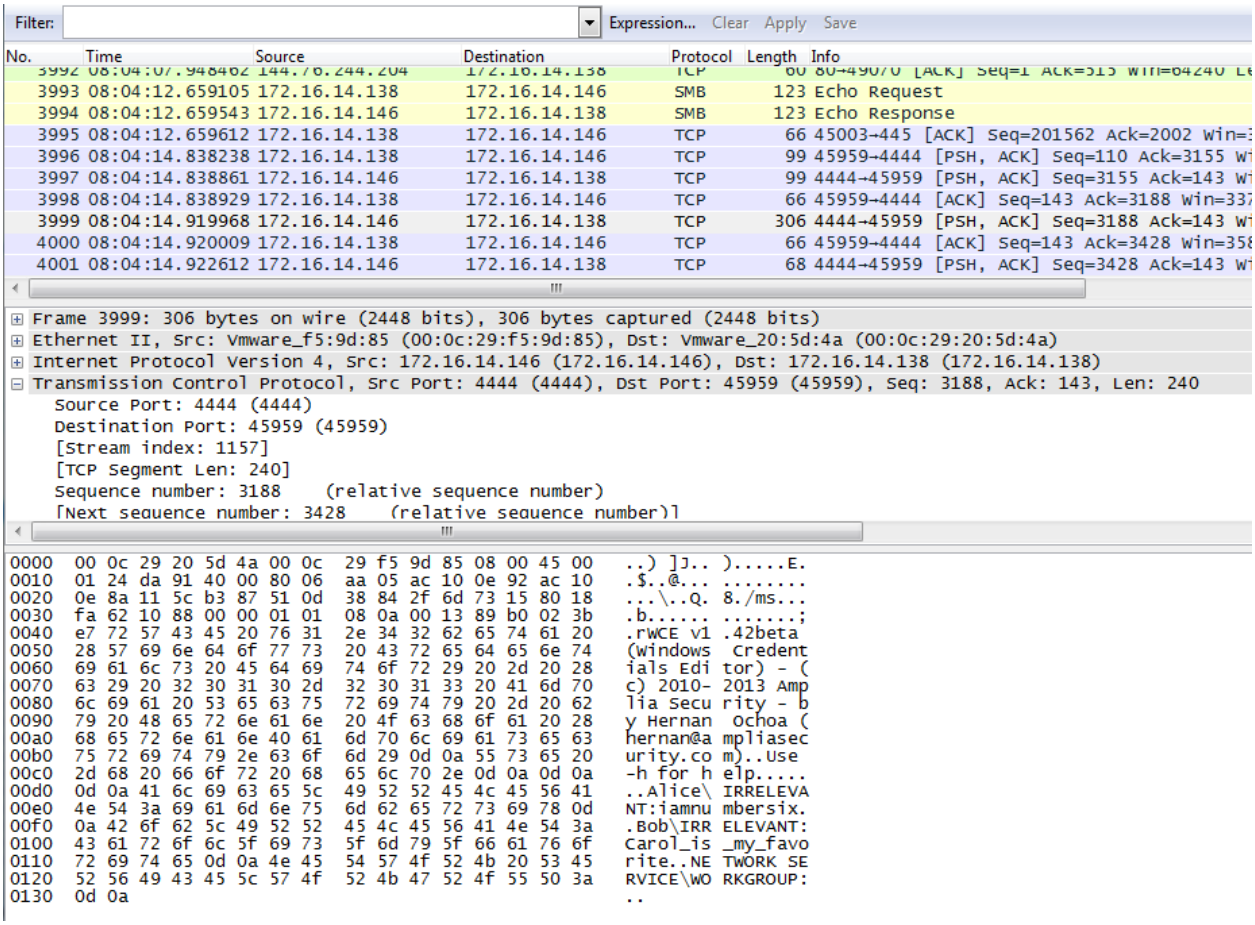
Upload the NTLMv2.hashes file to your Amazon GPU instance. I like to use WinSCP for this.
To brute force the NTLMv2 hashes with oclHashcat (implemented as a Mask Attack), using either a lowercase alpha, uppercase alpha, number, or special character in each position, you would run each of these commands (first command for a 1 character password length, second for a 2 character password length, etc.), and wait for the results:

sudo ./cudaHashcat64.bin -m5600 -a 3 ntlmv2.hashes ?a
sudo ./cudaHashcat64.bin -m5600 -a 3 ntlmv2.hashes ?a?a
sudo ./cudaHashcat64.bin -m5600 -a 3 ntlmv2.hashes ?a?a?a

sudo ./cudaHashcat64.bin -m5600 -a 3 ntlmv2.hashes ?a?a?a?a
sudo ./cudaHashcat64.bin -m5600 -a 3 ntlmv2.hashes ?a?a?a?a?a
sudo ./cudaHashcat64.bin -m5600 -a 3 ntlmv2.hashes ?a?a?a?a?a?a

oclHashcat can also perform dictionary attacks. Since the note from Challenge 1 mentioned that Alice mentions her passwords when she chats with Bob, I built a quick dictionary from their IRC conversations. That also didn't result in a cracked Hash, but a dictionary file based on good reconnaissance or social engineering is always worth a try.

Ultimately, finding Alice's password was accomplished by looking through the pcap file after she compromises Bob's PC. In Frame 3999, we can see a connection from Bob's PC back to Alice's PC over TCP Port 4444. Alice is running the "not_exactly_inconspicious.



If we take Alice's password that we just recovered, iamnumbersix, and add it to a dictionary file, we can run it through oclHashcat and crack the NTLMv2 hashes with it to confirm it is valid.

[ec2-user@ip-172-31-43-9 cudaHashcat-1.32]$ sudo ./cudaHashcat64.bin -m 5600 -a 3 ntlmv2.hashes /home/ec2-user/password.txt
cudaHashcat v1.32 starting...
Device #1: GRID K520, 4095MB, 797Mhz, 8MCU
Hashes: 3 hashes; 3 unique digests, 3 unique salts

Bitmaps: 8 bits, 256 entries, 0x000000ff mask, 1024 bytes
Applicable Optimizers:
** Zero-Byte*
** Not-Iterated*
** Brute-Force*
Watchdog: Temperature abort trigger set to 90c
Watchdog: Temperature retain trigger set to 80c
*Device #1: Kernel ./kernels/4318/m05600_a3.sm_*
Device #1: Kernel ./kernels/4318/markov_le_v1.
INFO: approaching final keyspace, workload adjusted
ALICE::WORKGROUP:
0000000008014958a0c2cd0015528f
c004500560041004e0054000100140
0140069007200720065006c0065007
500760061006e00740000000000:
ALICE::WORKGROUP:
00000000000d36b480c2cd001c0d15
c004500560041004e0054000100140
0140069007200720065006c0065007
500760061006e00740000000000:
ALICE::WORKGROUP:
00000000080986ca20c2cd00159fe4
c004500560041004e0054000100140
0140069007200720065006c0065007
500760061006e00740007000800809
0370032002e00310036002e0031003
rsix
Session.Name...: cudaHashcat
Status.....: Cracked
Input.Mode.....: Mask (iamnumbersix) [12] (0.00%)
Hash.Target....: File (ntlmv2.hashes)
Hash.Type......: NetNTLMv2
Time.Started...: 0 secs
Speed.GPU.#1...: 0 H/s
Recovered......: 3/3 (100.00%) Digests, 3/3 (100.00%) Salts
Progress.......: 3/3 (100.00%)
Skipped........: 0/3 (0.00%)
Rejected.......: 0/3 (0.00%)
HWMon.GPU.#1...: 0% Util, 35c Temp, -1% Fan
Started: Tue Feb 10 20:12:01 2015
Stopped: Tue Feb 10 20:12:03 2015

Alice’s password is: iamnumbersix

Open_this_to_win.zip