PARSONS

Parsons & Friends CTF – 2020.03.20

delivering a better world

Networking

Network Thief

Challenge

A known flag thief has hacked his way onto our network and is stealing flags. We were able to capture his network activity in the attached pcap file and when he was captured we recovered a portion of his home directory. Can you recover the flag he took to use as proof of his nefarious activity?

Network Thief

Examining the pcap file will show that the interesting activity is all TLS traffic with no obvious way to decript it.

There are some dns queries pointing to ctf.zendia.net which are a clue to where the flag might be.

Examining the home directory tar file shows multiple pointers to an ssl.log file and environment variable SSLKEYLOGFILE.

Googling for these leads to several tutorials on how to add the logs to wireshark to decrypt the tls traffic.

Once this is done it is an easy task to use the File -> Export Object option to find the NCX_Flag.png file and read the flag (NCX20{Some_Stupid_Logs}) in it.

Basic Networking 1

Challenge

What is the first domain the user goes to on port 80?

Basic Networking 1

Open up wireshark and filter by tcp.port == 80.

Scroll down to the GET / HTTP 1.1 line, and open up the HTTP protocol parsing to see Host: www.centos.org\r\\n

Basic Networking 2

Challenge

What was the name of the installation file downloaded from centos.org?

Basic Networking 2

Tshark is a good way to look at http (port 80) traffic and see what was downloaded

tshark -r pcap.pcap -nnn "http" | grep GET

283 4.647903 10.0.2.15 -> 81.171.33.201 HTTP 429 GET / HTTP/1.1

302 5.038907 10.0.2.15 -> 72.21.91.29 HTTP 296 GET /MFEwTzBNMEswSTAJBgUrDgMCGgUABBTPJvUY%2Bsl%2Bj4yzQuAcL2oQno5fCgQUUWj%2FkK8CB3U8zNllZGKiErhZcjsCEA%2BBvJL%2Fr0OGTczCptE6c74%3D HTTP/1.1

2740 8.452941 10.0.2.15 -> 72.21.91.29 HTTP 286 GET /MFEwTzBNMEswSTAJBgUrDgMCGgUABBTfqhLjKLEJQZPin0KCzkdAQpVYowQUsT7DaQP4v0cB1JgmGggC72NkK8MCEATh56TcXPLzbcArQrhdFZ8%3D HTTP/1.1

2743 8.499281 10.0.2.15 -> 72.21.91.29 HTTP 296 GET /MFEwTzBNMEswSTAJBgUrDgMCGgUABBTPJvUY%2Bsl%2Bj4yzQuAcL2oQno5fCgQUUWj%2FkK8CB3U8zNllZGKiErhZcjsCEA%2BBvJL%2Fr0OGTczCptE6c74%3D HTTP/1.1

2816 14.519257 10.0.2.15 -> 8.43.84.214 HTTP 520 GET /Download HTTP/1.1

3595 22.885825 10.0.2.15 -> 204.10.37.194 HTTP 523 GET /centos/6/ HTTP/1.1

3611 23.049407 10.0.2.15 -> 204.10.37.194 HTTP 512 GET /centos-design/css/centos.css HTTP/1.1

3613 23.049908 10.0.2.15 -> 204.10.37.194 HTTP 522 GET /centos-design/css/centos-listindex.css HTTP/1.1

3719 23.124346 10.0.2.15 -> 204.10.37.194 HTTP 528 GET /icons/blank.gif HTTP/1.1

3720 23.124454 10.0.2.15 -> 204.10.37.194 HTTP 527 GET /icons/back.gif HTTP/1.1

3723 23.124597 10.0.2.15 -> 204.10.37.194 HTTP 555 GET /centos-design/images/centos-logo-white.png HTTP/1.1

3724 23.124645 10.0.2.15 -> 204.10.37.194 HTTP 529 GET /icons/folder.gif HTTP/1.1

3830 23.211198 10.0.2.15 -> 151.139.128.14 HTTP 288 GET /MFEwTzBNMEswSTAJBgUrDgMCGgUABBReAhtobFzTvhaRmVeJ38QUchY9AwQUu69%2BAj36pvE8hI6t7jiY7NkyMtQCECsuburZdTZsFIpu26N8jAc%3D HTTP/1.1

3885 23.439485 10.0.2.15 -> 204.10.37.194 HTTP 349 GET /centos-design/images/favicon.ico HTTP/1.1

3892 27.875298 10.0.2.15 -> 204.10.37.194 HTTP 576 GET /centos/6/updates/ HTTP/1.1

3956 29.369799 10.0.2.15 -> 204.10.37.194 HTTP 591 GET /centos/6/updates/x86_64/ HTTP/1.1

3998 30.823680 10.0.2.15 -> 204.10.37.194 HTTP 604 GET /centos/6/updates/x86_64/drpms/ HTTP/1.1

4050 31.189910 10.0.2.15 -> 204.10.37.194 HTTP 551 GET /icons/unknown.gif HTTP/1.1

5721 38.010240 10.0.2.15 -> 204.10.37.194 HTTP 654 GET /centos/6/updates/x86_64/drpms/adcli-0.8.1-2.el6_0.8.1-3.el6_10.x86_64.drpm HTTP/1.1

Basic Networking 3

Challenge

What is the sha256 hash of the installation file downloaded from centos.org?

Basic Networking 3

Open up wireshark and and click “File”, “Export Objects”, “HTTP”

The click that file and "save as".

Then run sha256sum <file>:

51e4380f58b3023af4174fd6985525d9f69969993838a7af5ae2f1cf1752ebe5 adcli-0.8.1-2.el6_0.8.1-3.el6_10.x86_64.drpm

Basic Networking 4

Challenge

What is the domain name of the site the user tried to FTP to in this pcap file?

Basic Networking 4

First, we need to figure out the IP the user is FTPing to.

tcpdump -r pcap.pcapng "port 21" -nnn -tttt

reading from file pcap.pcapng, link-type EN10MB (Ethernet)

2020-03-19 15:24:46.658253 IP 10.0.2.15.55439 > 63.73.199.22.21: Flags [S], seq 4014318924, win 8192, options [mss 1460,nop,wscale 0,nop,nop,sackOK], length 0

2020-03-19 15:24:47.666790 IP 10.0.2.15.55439 > 63.73.199.22.21: Flags [S], seq 4014318924, win 8192, options [mss 1460,nop,wscale 0,nop,nop,sackOK], length 0

2020-03-19 15:24:49.682379 IP 10.0.2.15.55439 > 63.73.199.22.21: Flags [S], seq 4014318924, win 8192, options [mss 1460,nop,wscale 0,nop,nop,sackOK], length 0

2020-03-19 15:24:53.683040 IP 10.0.2.15.55439 > 63.73.199.22.21: Flags [S], seq 4014318924, win 8192, options [mss 1460,nop,wscale 0,nop,nop,sackOK], length 0

2020-03-19 15:25:01.698073 IP 10.0.2.15.55439 > 63.73.199.22.21: Flags [S], seq 4014318924, win 8192, options [mss 1460,nop,wscale 0,nop,nop,sackOK], length 0

Okay, now that we know it is 63.73.199.22, we need to know what domain name that resolves too.

tshark -r pcap.pcapng -nnn -t u "dns" | grep -B 1 63.73.199.22

7671 19:24:46.629462 10.0.2.15 -> 192.168.100.201 DNS 69 Standard query 0xc6bc A happy.com

7672 19:24:46.652662 192.168.100.201 -> 10.0.2.15 DNS 85 Standard query response 0xc6bc A 63.73.199.22

7673 19:24:46.654651 10.0.2.15 -> 8.8.8.8 DNS 69 Standard query 0xc6bc A happy.com

7675 19:24:46.673268 8.8.8.8 -> 10.0.2.15 DNS 85 Standard query response 0xc6bc A 63.73.199.22

Networked Pokemon 2

Challenge

One of our friends, Sammy, was sending himself a picture of his favorite Pokemon covertly over the network. Who is Sammy's favorite Pokemon?

Networked Pokemon 2

- Use wireshark to look at the traffic
- Right click and choose "Follow UDP Stream" and save the file to disk
- Then open up the image you just saved to disk to find the flag: SQWIRTTLE!!!

Networked Pokemon 3

Challenge

One of our friends, Sammy, learned from his previous attempt to "hide" his favorite Pokemon, and is at it again with his second favorite Pokemon, sending an image of it "covertly" over the network. Who is Sammy's second favorite Pokemon?

Networked Pokemon 3

- If you open up the pcan, you see a bunch of packets going from one IP:port to another IP:port via UDP.
- The port on the second IP goes up by 2 each time
- And you see traffic in the “data” field of each UDP packet..maybe we need to simple combine the data content of each packet? How do we do that?
- You can run *tshark -r pokemon3.pcap -T fields -e data | xxd -r -p > image.gif* - that extracts the data content of each packet (in ASCII) and then converts the ASCII back to binary so you can view the image.
- Then open up the image and you see the flag: FIREBOI!!!

Networked Pokemon 4

Challenge

One of our friends, Sammy, learned from his previous attempts to "hide" his favorite Pokemon, and is at it again with his third favorite Pokemon, sending an image of it even more "covertly" over the network. Who is Sammy's third favorite Pokemon?

Networked Pokemon 4

- First, open up the pcap in wireshark
- Look around at the packet and you'll see a bunch of TXT responses, with what appears to be hex data in the TXT response...we probably need to extract all of the content in the text responses (only the responses – no queries)
- tcpdump -r pokemon4.pcap -nnn -tttt | grep -F "127.0.0.1.53 > " | sed 's/.* TXT "//g' | sed 's/".*//g' | sed "s/\^J//g" | base64 -d > output.gif
- Open up the image and you see the answer: FARTINGFIRE

TicTacWhat?

TicTacToe 1

Challenge

Dena made a tic-tac-toe game at http://mar2020web.parsonscyber.com:3001/tictactoe1.php. Win the game to find the flag!

TicTacToe 1

If you had trouble figuring it out, it was nothing fancy.

One way to win is to play in the center first, then the bottom right, then the top right, then the bottom left to get the flag - violentils

TicTacToe 2

Challenge

Dena made a tic-tac-toe game at http://mar2020web.parsonscyber.com:3001/tictactoe2.php. Win the game to find the flag!

TicTacToe 2

Playing around a while you can eventually find a flaw in the AI logic.

Play in the center first, then the bottom left, then the top right to win - Munkleotard

TicTacToe 3

Challenge

Dena made a tic-tac-toe game at http://mar2020web.parsonscyber.com:3001/tictactoe3.php. Win the game to find the flag!

TicTacToe 3

This time, you can pass a play value in with the reset.

Go to http://mar2020ctf.parsonscyber.com/tictactoe3.php?reset=1&play=4, then play in the bottom right, then the top right, then the bottom left to get the flag - Phantasmash

TicTacToe 4

Challenge

Dena made a tic-tac-toe game at http://mar2020web.parsonscyber.com:3001/tictactoe4.php. Win the game to find the flag!

TicTacToe 4

This time, you can pass in 2 play values at once (you can pass in 1 and 2 at the same time).

To do that though, it’s easiest to install something to let you modify the post values you submit.

I found it easy to do with Firefox dev tools up.

First click the middle.

Then in Firefox dev tools, right click that post request and click "Edit and Resend". Then edit the post parameters to be play1.x=1&play2.x=5 and hit "Send". It won't update the screen for some reason, but then click the bottom left square and you'll win, and discover the flag is Olivermit

TicTacToe 5

Challenge

Dena made a tic-tac-toe game at http://mar2020web.parsonscyber.com:3001/tictactoe5.php. Win the game to find the flag!

TicTacToe 5

This time you can click on any of the squares that have already been played on and it will switch them. Click the middle, bottom left, then upper right to win and get the flag – Paulaunch.

TicTacToe 6

Challenge

Dena made a tic-tac-toe game at http://mar2020web.parsonscyber.com:3001/tictactoe6.php. Win the game to find the flag!

TicTacToe 6

This one you can’t solve by using only this challenge.

BUT, if you solve tictactoe1.php, and then go to tictactoe6.php in your browser, the completion/win will save and you’ll instantly get the flag – Kainoelle.

Reversing?

Find The Arguments

Challenge

Tony creating a binary where only when passing in the correct parameters do you get full output for the program. Figure out what the parameters need to be passed in and then pass them in to get the final output of the program, which is the flag!

Find The Arguments

If you look at the assembly in your reverse engineering tool of choice, you'll discover the logic looks like this:

```
if (argc != 5) {
    return 1;
}
int second = atoi(argv[3]);
if (second != 0x881) {
    return 2;
}
int first = atoi(argv[4]);
if (floor(((((first * 98)/3)+6)/7)-2345) != 2373) {
    return 3;
}
char* fourth = argv[2];
if (fourth[2] != '5' || fourth[1] != (char)0150 || fourth[5] != (char)0x44 || fourth[3] != '!' || fourth[4] != 'a' || strlen(fourth) != 6 || fourth[0] != (char)45) {
    return 4;
}
int third = atoi(argv[1]);
if ((third|61) != 957) {
    return 5;
}
```

And then the arguments you passed in are md5 hashes.

Find The Arguments

Working it out, you get the 4 parameters

./re1 896 '-h5!aD' 2177 1011

55f24227b25062725dc25105a41b4f9b

Guess a String

Challenge

Test your string guess against the given executable to find the flag!

Guess a String

- Open it up in your favorite reverse engineering tool and start following the code
- Effectively, each character of the input string gets checked one by one, and you can check the comparisons to figure out each character 1 by 1 until you get the answer. You can test the answer by simply running the binary with your guess and seeing if you get the right output!
- The answer is: iDlDrr7

Guess A Number

Challenge

Guess a number to get the flag against the given executable!

Guess A Number

- Open it up in your favorite reverse engineering tool and start following the code
- You'll see it runs atoi on the given argument (hence you know it will need to be a number).
- It then multiples your argument by 5, subtracts 16, and adds 26, and that has to match count run against "11225".
- You can then trace the count function as it effectively does:

```
int d = atoi("11225")*4;
 int e = atoi("11225")+42-12;
 int f = atoi("11225")+d+(e*2);
return f
```

- The answer, when worked out, is: 15725

Lucy In The Sky

Challenge

Lucy is snooping around Clifford's computer. Help Lucy find the flag in plain sight.

Lucy In The Sky

- Always try “strings” against a binary / executable first!
- *strings challenge.bin | grep ctf*
  ctf{boomshakkaboom}

Lucy In The Sky 2

Challenge

Lucy got the flag in plain sight, but there still looks like there's something suspicious going on. See if you can help her enter the right input and decode the secret message!

Lucy In The Sky 2

- Open it up in your favorite reverse engineering tool!
- You'll soon see that it xors 2 words together with some slight manipulation
- Follow through that xor and you'll get the needed argument to pass to it: VWDT[RP
- Then you can pass it in to get the code!
- *./challenge.bin VWDT[RP*
  What is the keyword?
  VWDT[RP
  WELCOME aSd2ZSBiZWVuIHNvIGxvbmVseQo=
- *echo "aSd2ZSBiZWVuIHNvIGxvbmVseQo=" | base64 -d*
- i've been so lonely

Cryptography

Secret Decoder Ring

Challenge

Here is a bit of cypher found written on a piece of graph paper. The plaintext is suspected to begin with the phrase "Thanks to Mauborgne"

WEECG NHYLT VRRIK SAHAE EXATO NIXHM NCHYW EWBXC FMHLD OSFEA IOABN EAKSR ABMHK NRYBH HSDIP AHDNC HKOEN CRIYR MHOYB HTWRI EABHG OSNHI DMWLH EATWY HYBNH TOPNR KNLAX

Can you decrypt it and recover the flag?

Secret Decoder Ring

Playfair is relatively easy to solve with a crib, as in this case, and for those familiar with solving techniques. Otherwise, especially in a timed environment, and without graph paper, it can be very challenging and complex.

Plaintext: Thanks to Mauborgne we have a solution to the playfair cypher. He developed it before world war one. In honor of its creator here is your flag wheatstone was robbed

FLAG: wheatstonewasrobbed

KEY: WHEATSTONE CYPHER

KEYWORD DECIMATED ALPHABET: WHEATSONCYPRBDFGIJKLMQUVXZ

PLAYFAIR SQUARE:

W H E A T

S O N C Y

P R B D F

G I K L M

Q U V X Z

Googling 'Mauborgne' and crypto gets to the WIKI page for Joseph Mauborgne who first described a solution for the Playfair cypher. It should also be obvious, given the crib, that this is not a simple substitution problem. Mapping the known plain text "THANKS TO MAUBORGNE" to the beginning of the cypher text gives us enough information to fully decrypt the message, recover the key, and ultimately recover the keyword used to generate the playfair matrix.

Secret Decoder Ring

First align the plain and cypher text:

WE EC GN HY LT VR RI KS AH

TH AN KS TO MA UB OR GN E_

For each plaintext/cyphertext pair, with four distinct letters, there are three possibilities. Either they are all in the same row, the same column, or they are at opposite corners of a rectangle. Plain/cypher pairs containing only three letters are all adjacent and in a single row or column. Use graph paper to start building a matrix with the known pairs...

Start with WE and TH on the same row (doing otherwise results in an inconsistency after mapping the initial five pairs). Further, TW and HE must be adjacent.

TW HEA

N C S

Y O

K G

M L

V U

B R

RI and OR are obviously on the same row or column. They must map together as either IRO or I. The column solution will be shown to be correct.

R

O

With a little speculation we can try L next to M and some alphabetizing:

WHEAT WHEAT WHEAT

S NC SONCY Adding in some letters SONCY

GIKLM Which leads to: GIKLM in their obvious places: GIKLM

RB RB RB

O Y UV QUVXZ

UV

Secret Decoder Ring

If at any time there is a question about a character placement just go back to the message and decrypt a couple of digraphs using the test relationships. Finally, just by manipulating the matrix...

WHEAT

SONCY

PRBDF Which leads to "WHEATSTONECYPHER" as the alphabet keyword.

GIKLM

QUVXZ

You can then use any playfair cipher solver (like https://www.dcode.fr/playfair-cipher) to convert the original to plaintext: Thanks to Mauborgne we have a solution to the playfair cypher. He developed it before world war one. In honor of its creator here is your flag wheatstone was robbed

Crypto Fun?

Challenge

Can you decrypt this to find the flag?

mo ey em ke o

nk se on yd

Crypto Fun?

- Read the first the groups of characters from top to bottom, right to left, to get "monkey see monkey do"
-

Crypto Funner?

Challenge

Can you decrypt this to find the flag?

44 54 51 1c 52 58 4h 53 1c 55 5f 1c 44 4h 4j 5b 3f 55 5g 5g 61

Crypto Funner?

- When you notice what characters are and aren't there, you may realize it is a base20 encoding. Base20 to ASCII Decoder yields "The flag is TacoKitty".
-

Crypto Funnest?

Challenge

Can you decrypt this to find the flag?

84104101329711011511910111432116104971163212111111732115101101107321051153270108971099711010311 1

Crypto Funnest?

- If you look, it appears to be decimal numbers, but with missing spaces. If you add in spaces to separate numbers that map to ASCII characters, you get: 84 104 101 32 97 110 115 119 101 114 32 116 104 97 116 32 121 111 117 32 115 101 101 107 32 105 115 32 70 108 97 109 97 110 103 111.
- If you convert that to ASCII, you get: The answer that you seek is Flamango

Et tu, Brute?

Challenge

Can you decrypt this to find the flag? If a password is required, use "flat"...

6153 746c 6465 5f5f 7447 4126 3db4 b7f9 a64b 1675 2ae8 624d fbd5 f170 f6d0 1c2d 28e2 3f21 8590 feaf

Et tu, Brute?

- Dump the hex to a file in binary and run various openssl decryption types and eventually openssl enc -d -des-cfg <inputfile> yields the phrase: The flag is Monkeystone

A Story About a Proposed Marshland?

Challenge

Can you decrypt this to find the name of the proposed marshland? I hear it's something about a story related to it

fwi [w iwff cvh y liv4c i.yi 6l y*vhi y +4v+vlwu [y4l.fyju jy[wu +6[+f6[+c

A Story About a Proposed Marshland?

- Using hints from the clue (like story, proposed, and marshland), you should be able to hand solve enough of the cryptogram, which originally was "let me tell you a story that is about a proposed marshland named pimplimpy", to get the name of the proposed marshland

Coding

Range Coding

Challenge

Jake is a beginning computer science student and he is stumped by what he believes to be an easy question:

What is the sum of all the numbers between 101 (inclusive) and 101539 (exclusive) that are either multiples of 3, multiples of 7, or multiples of 5 (that aren't multiples of 4)?

Range Coding

- The code below will solve it (to get you the answer of 2651228580

```
#!/usr/bin/python3
  sum = 0
  for i in range(101,101539):
    if i%7 == 0 or i%3 == 0:
       sum += i
    elif i%5 == 0 and i%4 != 0:
       sum += i
       print(sum)
```

User Agents

Challenge

In the provided http.log (a tab separated file where the top of the file shows what the name of each field is), what is the full user agent string that is seen exactly 6 times in the file?

An example user agent is - Chrome: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/77.0.3865.90 Safari/537.36

User Agents

- The below Linux shell command will get you the answer! (It prints out the 12th field (using a tab separator), then sort them and counts them and greps for one that's count is 6)
- *awk -F "\t" '{print $12}' http.log | sort | uniq -c | grep "^ *6 "*
- 6 Mozilla/5.0 (iPhone; CPU iPhone OS 7_1 like Mac OS X) AppleWebKit/537.51.2 (KHTML, like Gecko) Mobile/11D167

Log Correlation Challenge

Challenge

You have 3 Bro logs, a file.log (which logs files seen in traffic), a http.log (which logs http connections seen in traffic, and which has a "fuid" that correlates entries from the file.log with the http.log), and a conn.log (which lots all connections seen in traffic, and which has a "uid" field that correlates entries from the conn.log with the http.log).

Each field in the lab is separated by a tab. Each file also has a couple lines at the start of the file that let you know what each field in the file is.

Using the information about to correlate entries from the file.log, http.log, and conn.log (for entires that have that correlation), starting with the file.log as the base, correlate the entries and output data in this format (only if the service duration is greater than 2), then sort the entires by the UID field (ascending (so a to z)) and give me the full line for the 196th one.

Timestamp (from files log)|fuid (from files log)|tx host (files log)|rx host (files log)|md5sum|sha1sum|host (from http)|uri (from http)|user agent (from http)|uid (from http)|service duration (from conn log)

An example line is: 1408502841.520429|FBcZyx1t52FplnMj5k|23.66.231.25|192.168.1.2|81460bf718b14bfb1e1cf1f8bff92e07|c302e4a2c8d270b78bb2378571a9166f23ba8f2d|api.bing.com|/asjson.aspx?query=caramello+c&form=APIPH2|Mozilla/5.0 (iPhone; CPU iPhone OS 7_1 like Mac OS X) AppleWebKit/537.51.2 (KHTML, like Gecko) Version/7.0 Mobile/11D167 Safari/9537.53|CztPsB1zK5ASSREb26|36.134615

Log Correlation Challenge

- First I removed the header and footer from each file. Then I ran this code to get the answer - 1408502463.345337|FTtzal27fwVbrLDOx5|72.21.202.185|192.168.1.2|d5d1dc9ee159bf2801c1e38db528ee4f|97164d8af71b73173169b8cb0049f694b258d9fa|aax.amazon-adsystem.com|/e/dtb/bid?src=3055&u=http://www.cafemom.com/answers/404961/Do_you_hate_being_a_stay_at_home_mom?cat_id=&order=&next=11&cb=6283577|Mozilla/5.0 (iPhone; CPU iPhone OS 7_1 like Mac OS X) AppleWebKit/537.51.2 (KHTML, like Gecko) Version/7.0 Mobile/11D167 Safari/9537.53|CVXiwR2CBuFbgcLn9c|31.703925

```
#!/usr/bin/python3

separator = "|"

fileDict = {}
with open('file.log') as f:
  for unsplitLine in f:
      line = unsplitLine.split("\t")
      ts = line[0]
      fuid = line[1]
      txhost = line[2]
      rxhost = line[3]
      md5sum = line[20]
      sha1sum = line[21]
      fileDict[fuid] = line

httpDict = {}
with open('http.log') as f:
  for unsplitLine in f:
      line = unsplitLine.split("\t")
      #print "FUID - " + fuid
      fuid = line[25]
      httpDict[fuid] = line

connDict = {}
with open('conn.log') as f:
  for unsplitLine in f:
      line = unsplitLine.split("\t")
      uid = line[1]
      connDict[uid] = line
```

Log Correlation Challenge

```
finalDict = {}
  for fuid in fileDict.keys():
    fileLine = fileDict[fuid]
    if fuid in httpDict.keys():
        httpLine = httpDict[fuid]
    else:
        continue
    if httpLine[1] in connDict.keys():
        connLine = connDict[httpLine[1]]
    else:
        continue
    ts = fileLine[0]
    fuid = fileLine[1]
    txhost = fileLine[2]
    rxhost = fileLine[3]
    md5sum = fileLine[19]
    sha1sum = fileLine[20]
    host = httpLine[8]
    uri = httpLine[9]
    userAgent = httpLine[11]
    uid = httpLine[1]
    serviceDuration = connLine[8]
    if float(serviceDuration) > 2:
        finalDict[uid] = ts + separator + fuid + separator + txhost + separator + rxhost + separator + md5sum + separator + sha1sum + separator + host + separator + uri + separator +
userAgent + separator + uid + separator + serviceDuration

  counter = 0
  for uid in sorted(finalDict.keys()):
    counter += 1
    if counter == 196:
        print(finalDict[uid])
```

Tigers

Challenge

Something is wrong with the tiger.jpg. Figure out what's wrong and fix it to find the flag!

Tigers

- It's shifted 10 bytes up (and rotates around if it hits the max). You can tell by the normal JFIF header is now TPSP. The code below will reverse it for you (and output an image you can look at and see the flag of CuddleKitties.

```
#!/usr/bin/python3

import sys

if len(sys.argv) < 3:
  print("Syntax: %s <filename1> <filename2>" % sys.argv[0])
  sys.exit()

file1 = open(sys.argv[1], 'rb')
file2 = open(sys.argv[2], 'wb')
byte1 = file1.read(1)
counter = 3
while byte1:
  byte2ord = ord(byte1) - 10
  if byte2ord < 0:
     byte2ord += 256
  byte2 = byte2ord.to_bytes(1,"little")
  file2.write(byte2)
  byte1 = file1.read(1)
file1.close()
file2.close()
```

Tigers

The Eye of the Tiger

Challenge

A known flag thief was caught leaving the scene of a crime from which a valuable flag was stolen. We believe the thief hid it in this file.

Can you recover the flag and write case closed to the exploits of this villian?

The Eye of the Tiger

- The Flag is contained as a subtitle track within the mp4 file. The subtitle itself plays for .001 seconds at 20 seconds into the movie. It is intentionally small and of very short duration so to not be easily visible. The flag is also base64 encoded and broken up into 4-5 character lines to make it very difficult to use strings or other methods of examining the file. A lion roar also starts at 20 seconds to distract from the real flag.
-
- To solve this use ffmpeg to extract the subtitle track into an easily readable text file thusly
- ffmpeg -i tiger.mp4 junk.srt
-
- junk.srt then contains...
-
- 1
- 00:00:20,000 --> 00:00:20,001
- TkNYM
- jB7c3
- VidGx
- lIFRp
- Z2Vy
- fQo=
-
- The flag base64 decoded is:
-
- NCX20{subtle Tiger}

Roar

Challenge

Tigers are in. Can you find the flag in this tiger video? Don't let the roar scare you!

Roar

- The QRCode changes for a single frame about 6.7s into the video. I had to step through the video frame by frame starting about 6.4s (using the . in mplayer), and when I noticed the qrcode change, I then had to screenshot my desktop and run it through zbarimg (as my phone could process the normal qrcode fine but not the changed one).

Leo the Tiger

Challenge

Leo the Tiger has eaten some letters and thrown him around his video. Can you read them to find the flag?

Leo the Tiger

- The only way I know is to track is frame by frame and write each letter down. Eventually you'll get: The flag is the answer to this - what is the past tense of the word shower, prepended with the number the comes after five? (2showered)

Albino

Challenge

This poor tiger is white. Can you find the red flag he is hiding?

Albino

- Try opening the file, which is a postscript file (.ps) in an editor made to view those, like evince, to see the flag (Whitiger)

Tiger USBDrive

Challenge

We found an image of a talking tiger telling you the flag. Unfortunately, the tiger then tried to bite the keyboard and accidentally deleted the file off the usbdrive. Can you recover the image and find the flag?

Tiger USBDrive

- Run foremost on the usbdrive (after decompressing it), and in the output/jpgs directory will be an image with a tiger and a cartoon talking bubble that says "The flag is SCARYTOOTHS"

Tiger USBDrive 2

Challenge

We found a file telling you about a flag. Unfortunately, the tiger then tried to bite the keyboard and accidentally deleted the file off the usbdrive. He then whacked the keyboard and did something to mess up the recovery. Can you recover the file and find the flag?

Tiger USBDrive 2

- Run foremost on the file to extract a .pdf file. It won't open. View the file in hex and you'll see repeated 4 times, at the start of the file, DELETEME. Delete all 4 of those and the file opens fine to reveal the flag - Piratiger