**Write-Up CTF RedTeam VSC**

**Ngày 26 tháng 12, 2024**

**Sdsdsdsdsdsdsdsdádasdasda----**

Mục lục

[**1.** **Tóm tắt báo cáo tiến độ** 1](#_Toc186823110)

[**Tổng quan:** 1](#_Toc186823111)

[**Chi tiết kết quả:** 1](#_Toc186823112)

[**2.** **web01-flag1** 2](#_Toc186823113)

[**3.** **web01-flag2** 5](#_Toc186823114)

[**4.** **web01-flag3** 8](#_Toc186823115)

1. **Tóm tắt báo cáo tiến độ**

Trong quá trình thực hiện, em đã hoàn thành một phần các thử thách trong phân loại Web và chưa thực hiện các thử thách thuộc phân loại PWN. Dưới đây là chi tiết kết quả đạt được:

## **Tổng quan:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **STT** | **Phân loại** | **Tổng** | **Hoàn thành** | **Chưa hoàn thành** |
| 1 | Web | 6 | 3 | 3 |
| 2 | PWN | 3 | 0 | 3 |

## **Chi tiết kết quả:**

|  |  |  |
| --- | --- | --- |
| **STT** | **Challenge** | **Flag** |
| 1 | web01-flag1 | VCS{Web01-Flag2-1ccb9abe-32ff-4d58-bbdb-7f8d027fdf85} |
| 2 | web01-flag2 | VCS{Web01-Flag2-1ccb9abe-32ff-4d58-bbdb-7f8d027fdf85} |
| 3 | web01-flag3 | VCS{Web01-Flag3-b7c9d888-8612-4ed8-a161-f3d5aede9180} |

1. **web01-flag1**

**------------**

**Phân tích**

**A screen shot of a computer

Description automatically generated**Sau khi giải nén source code và chạy docker, ta sẽ được trang web như sau:

Trang web có giao diện đơn giản và cung cấp các chức năng như: truy vấn hình ảnh thông qua ID, tải lên hình ảnh mới, và đăng nhập.

A screen shot of a computer

Description automatically generatedVới challenge này, ta cần được lấy được flag1. Vì thế ta sẽ tập trung vào đoạn code có liên quan đến flag1.

Đoạn code trên có logic khá đơn giản, khi truy cập endpoint **freeflag** ta cần truyền tham số **segcret**. Sau đó, server sẽ so sánh tham số này với **app.config["SECRET\_KEY"]**. Nếu bằng sẽ trả về flag1 cho chúng ta.

A screenshot of a computer screen

Description automatically generatedA red arrow on a black background

Description automatically generatedMà **app.config["SECRET\_KEY"]** được tạo ra một cách được tạo ra một cách ngẫu nhiên thông qua **os.urandom()**.

A screenshot of a computer

Description automatically generatedSau khi thử request, ta nhận đoạn thông báo "Incorrect secret key!" kèm theo **session**. Giá trị **session** được ngăn chia thành ba phần và ngăn cách bởi dấu **.**

|  |  |  |
| --- | --- | --- |
| **Phần** | **Dữ liệu mã hóa** | **Mục đích** |
| 1 | eyJmbGFnIjoiVkNTe1dlYjAxLUZ… | Dữ liệu được encode Base64 |
| 2 | Z21INw | Timestamp |
| 3 | uBpUNEWktSiSNoW5iMBTONCTsx4 | HMAC |

**Tại dòng 53**, flag1 đã được gán vào **session**. Vì thế, ta chỉ cần decode base64 phần thứ nhất của giá trị **session** là ta sẽ nhận được flag1.

**Khai thác**

* Bước 1:

A screenshot of a computer screen

Description automatically generatedThực hiện request đến **freeflag** với giá trị **segcret** tùy ý.

* Bước 2:

A screenshot of a computer

Description automatically generatedDecode base64 phần thứ nhất của giá trị **session**.

**Khắc phục**

A screen shot of a computer

Description automatically generated Để khắc phục, ta cần thiết lập flag sau khi đã kiểm tra giá trị **segcret** như sau:

1. **web01-flag2**

**------------**

A black screen with white text

Description automatically generated**Phân tích**

A screen shot of a computer code

Description automatically generatedVới challenge này, ta chỉ cần truy cập đến endpoint **admin** với tư cách admin thì ta sẽ có được flag2.

A black background with colorful text

Description automatically generatedLogic của chức năng đăng nhập khá đơn giản, server sẽ truy vấn cơ sở dữ liệu thông qua **username** do người dùng nhập. Sau đó, lấy password trong cơ sở dữ liệu so sánh với **password** mà người dùng nhập.

Tại hàm **get\_user()** chứa lỗ hổng SQL Injection và lỗ hổng này có thể bị khai thác để thực hiện tấn công theo kiểu Union-Based SQL Injection. Bằng cách lợi dụng lỗ hổng, ta có thể thao túng kết quả câu truy vấn quả trả về và từ đó đăng nhập với quyền admin

SELECT \* FROM users WHERE username=("")

UNION

SELECT "admin", " hash\_md5 "

Câu truy vấn trên sẽ trả về kết quả là username là admin và password là một chuỗi hash md5.

**Khai thác**

* Bước 1:

A screenshot of a computer

Description automatically generatedThực hiện request endpoint **login** với username và password như sau:

Truy cập mẫu đường dẫn sau:

http://IP:PORT/login?username=")+UNION+SELECT+"admin","c4ca4238a0b923820dcc509a6f75849b"--&password=1

**Lưu ý:**

Các giá trị màu đỏ cần được thay thế

* Bước 2:

Sau khi đăng nhập thành công, ta sẽ nhận được một giá trị session. Ta cần truy cập đến endpoint **admin** cùng với giá trị session vừa nhận được.

A screenshot of a computer

Description automatically generated

**Khắc phục**

A screen shot of a computer code

Description automatically generatedGiới hạn độ dài chuỗi **username**.

A computer screen with red and yellow text

Description automatically generatedSử dụng Prepared Statements.

1. **web01-flag3**

**------------**

**Phân tích**

Với challenge này, ta cần đọc được biến môi trường (env) để lấy flag3.

A screenshot of a computer program

Description automatically generatedĐể đọc được biến env, ta có thể lợi dụng biến **filename**. Nếu **filename** có giá là **../../../proc/self/environ** thì sau khi nối chuỗi thì chương trình sẽ mở tệp environ (tệp này chứa danh sách các biến môi trường của tiến trình hiện tại).

A screen shot of a computer

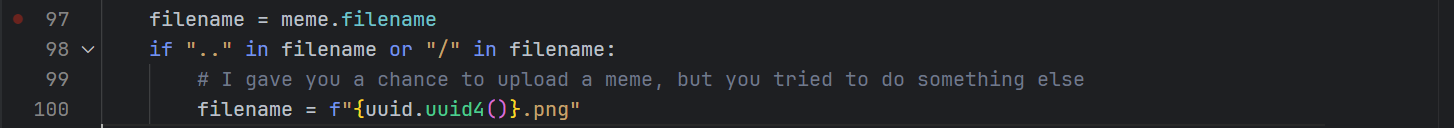
Description automatically generatedMà giá trị của **filename** được lấy từ cơ sở dữ liệu thông qua **db.get\_meme()**.

Tại câu truy vấn, không thể xảy ra lỗ hổng SQL Injection vì giá trị của biến **id** được xác thực tại hàm **meme()** trước khi đưa vào câu truy vấn.

A screen shot of a computer program

Description automatically generatedNhưng may mắn thay tại hàm **upload\_meme()** giải quyết được vấn đề này.

Tại hàm này gọi đến **db.get\_meme()**.

A screen shot of a computer

Description automatically generatedĐoạn code trên xảy ra lỗ hổng SQL Injection, ta có thể injection vào **title** hoặc **filename**.

Nhưng trong trường hợp này, ta sẽ injection tại biến **title** vì **filename** không thể chứa ký tự .. và /

Tóm lại, luồng tấn công như sau:

* Đầu tiên ta sẽ upload tệp và lợi dụng SQL Injection để thiết lập giá trị **filename=../../../proc/self/environ.** Sau đó, server sẽ trả về giá trị **id**.
* Tiếp theo, ta sử dụng giá trị **id** để đọc được danh sách các biến môi trường của tiến trình hiện tại.

**Khai thác**

* Bước 1:

A screenshot of a computer

Description automatically generatedThực hiện upload và injection tại biến **title**

HTTP Request

POST /upload-meme HTTP/1.1

Host: localhost:1337

Content-Length: 318

Content-Type: multipart/form-data; boundary=----WebKitFormBoundaryYrO9SATIv0VUPEDK

------WebKitFormBoundaryYrO9SATIv0VUPEDK

Content-Disposition: form-data; name="title"

","../../../proc/self/environ")--

------WebKitFormBoundaryYrO9SATIv0VUPEDK

Content-Disposition: form-data; name="meme"; filename="hello"

Content-Type: application/octet-stream

------WebKitFormBoundaryYrO9SATIv0VUPEDK--

* Bước 2:

Sau khi upload thành công, request đến endpoint **meme** với **id** mà server vừa trả về.

A screenshot of a computer screen

Description automatically generated

Truy cập mẫu đường dẫn sau:

http://IP:PORT/meme?id=id-ret

**Lưu ý:**

Các giá trị màu đỏ cần được thay thế

**Khắc phục**

A screen shot of a computer code

Description automatically generatedSử dụng Prepared Statements.

Làm sạch đường dẫn tại hàm **meme()**

A screen shot of a computer code

Description automatically generated

**HẾT**