**Bài 1: SSFS**

Bài cho upload file và download file

Bug nằm ở đoạn code sau:



Bạn có thể kiểm soát file\_id trong route download 🡪Hàm send\_file('uploads/' + file\_id)

Payload: file\_id=../../flag.txt

**Flag: bctf{4lw4y5\_35c4p3\_ur\_p4th5}**

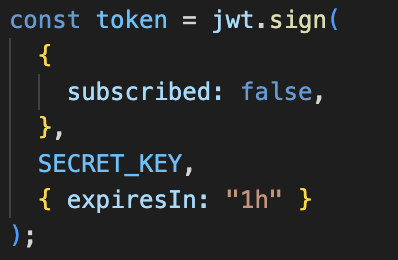
**Bài 2: Quotes**

Route /register

Cấp JWT nội dung: {subscribed: false}

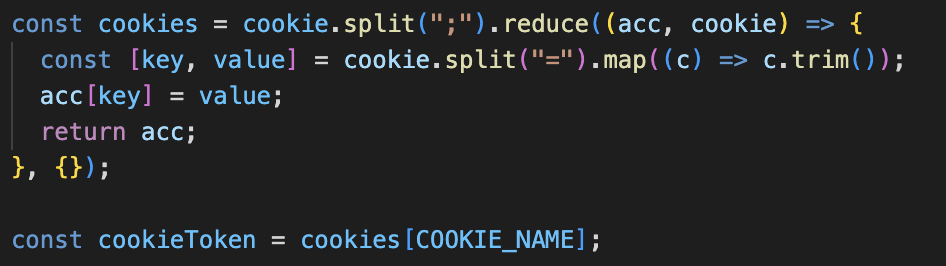
Trả về message: "Signed in!"

Thuật toán tạo JWT là HS256 (mã hóa đối xứng)



Route /quote

Mình không biết code này có lỗ hổng gì không nhưng nó đang xử lí cookie

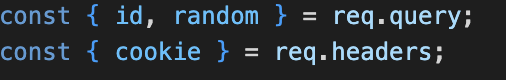


COOKIE\_NAME = "quotes-auth";

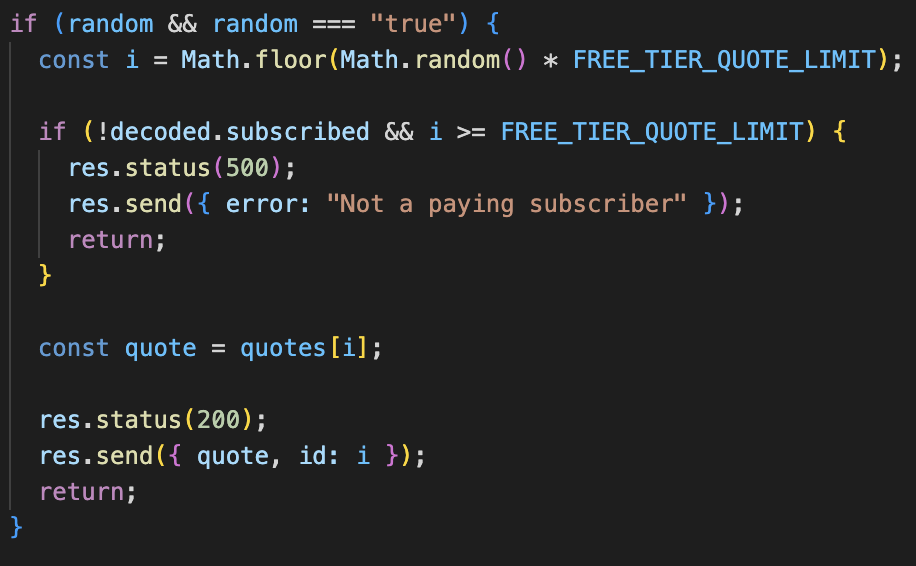
Xác thực JWT:



Nó nhận 2 tham số id và random:

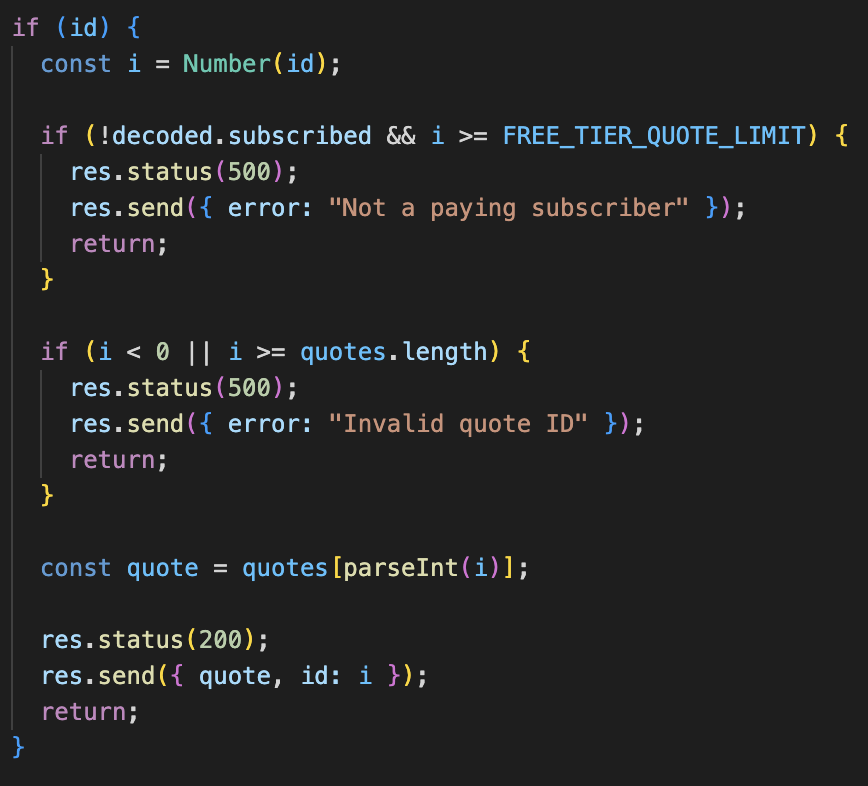


Nếu random= true:

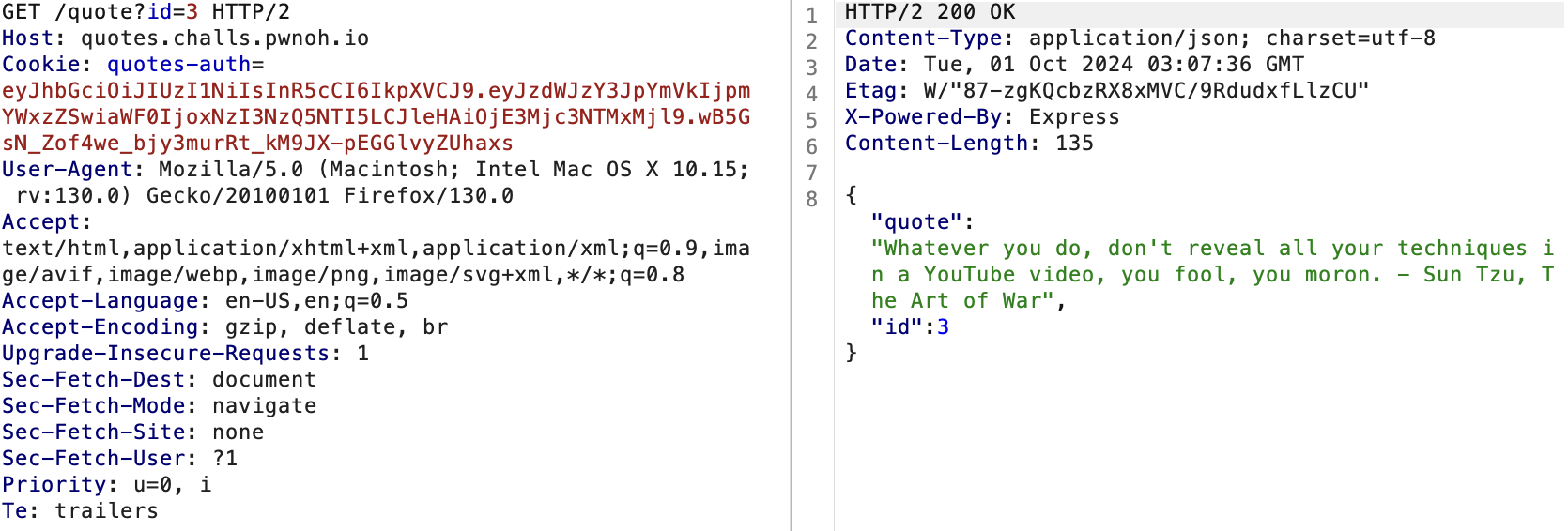


quotes là list các bài viết. Có tất cả 8 bài viết, bài viết thứ 8 là flag

Trang web còn cho ta chỉ định id bài viết:



const FREE\_TIER\_QUOTE\_LIMIT = 5;



Nhưng tham số i của bạn không bao giờ lớn hơn hoặc bằng 5 được

Nó sẽ báo lỗi “Not a paying subscriber”

Để đọc được flag thì id phải bằng 8. Muốn id ==8 thì decoded.subscribed==true

Tức là ta phải can thiệp vào JWT.

Mình thử bruteforce secret key nhưng không thành công. Nó không có dấu hiệu gì về lỗ hổng JWT cả

Đến đây thì mình hết ý tưởng rồi ☺ nên phải xem writeup:

Để xem nào:

1. Chúng ta cung cấp 1 id
2. id được chuyển thành number(i)

* Nếu không chuyển tiếp được, trả về NaN
* Nếu chuyển được, giá trị vẫn giữ nguyên

1. parseInt(i) sẽ xóa các kí tự khác và chỉ giữ lại số. Ví dụ ‘1b’ sẽ chuyển thành ‘1’

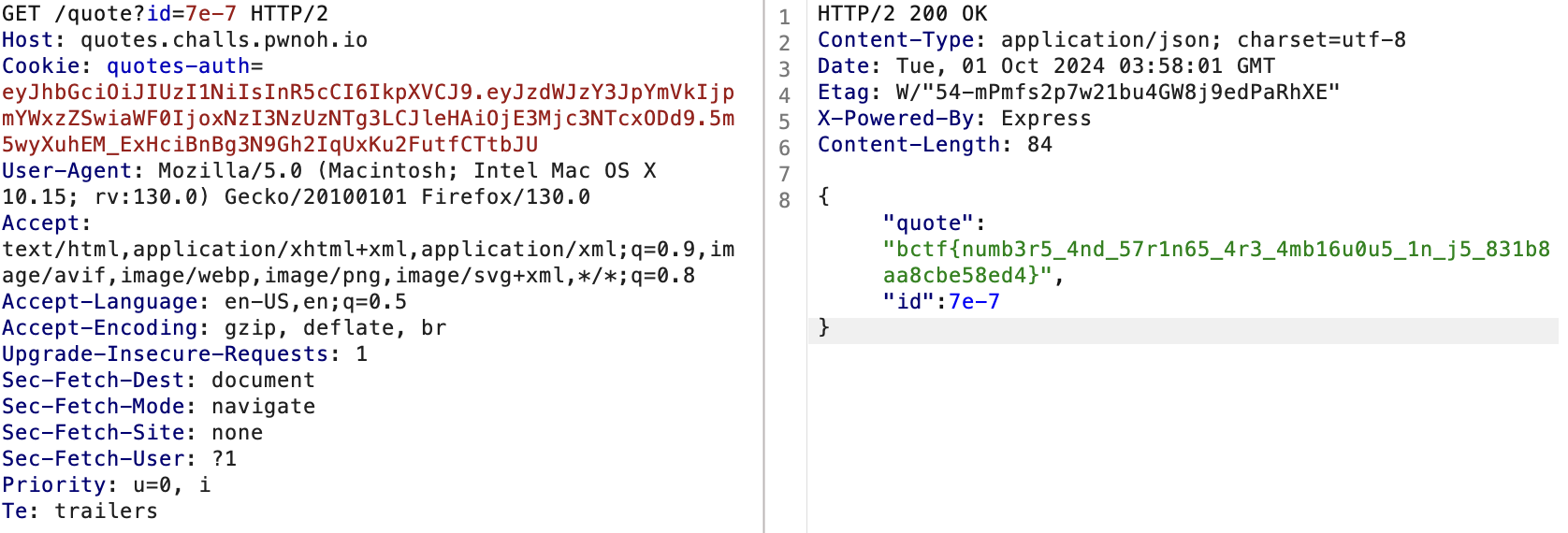
Mình đã thử một vài giá trị và thăm dò 1 chút nhưng nó chỉ trả về id: null hoặc lỗi

Sau đó, mình tìm được con số có thể hoạt động được trong bối cảnh của chúng ta:

Bởi vì các con số lớn dùng kí tự e để biểu diễn trong chuỗi của chúng (ví dụ: 6.022e23 là 6.022 × 1023), dùng hàm parseInt(i) để rút gọn số sẽ sinh ra kết quả không mong đợi khi dùng trên 1 con số quá lớn hoặc quá nhỏ.

console.log(parseInt(7e-3)); 🡪 7\*10-3=0

console.log(parseInt(7e-7)); 🡪 7\*10-7=7



**Flag: bctf{numb3r5\_4nd\_57r1n65\_4r3\_4mb16u0u5\_1n\_j5\_831b8aa8cbe58ed4}**