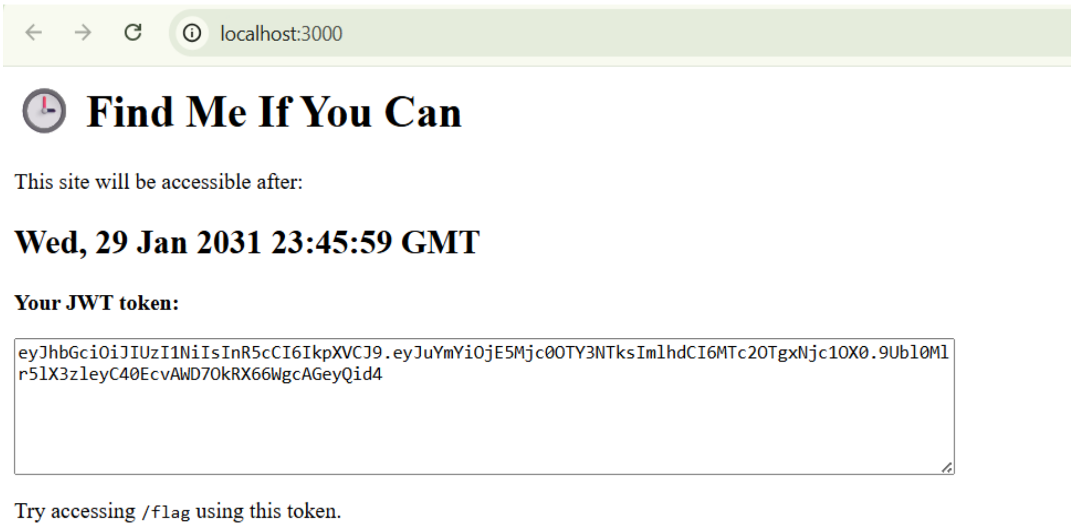
Find Me If You Can – JWT nbf Challenge

This challenge focuses on understanding and exploiting JSON Web Tokens (JWT), specifically the *nbf* (Not Before) claim. The application restricts access to protected resources until a future timestamp. By decoding, modifying, and re-encoding the JWT payload, this restriction can be bypassed.


localhost:3000

Find Me If You Can

This site will be accessible after:

Wed, 29 Jan 2031 23:45:59 GMT

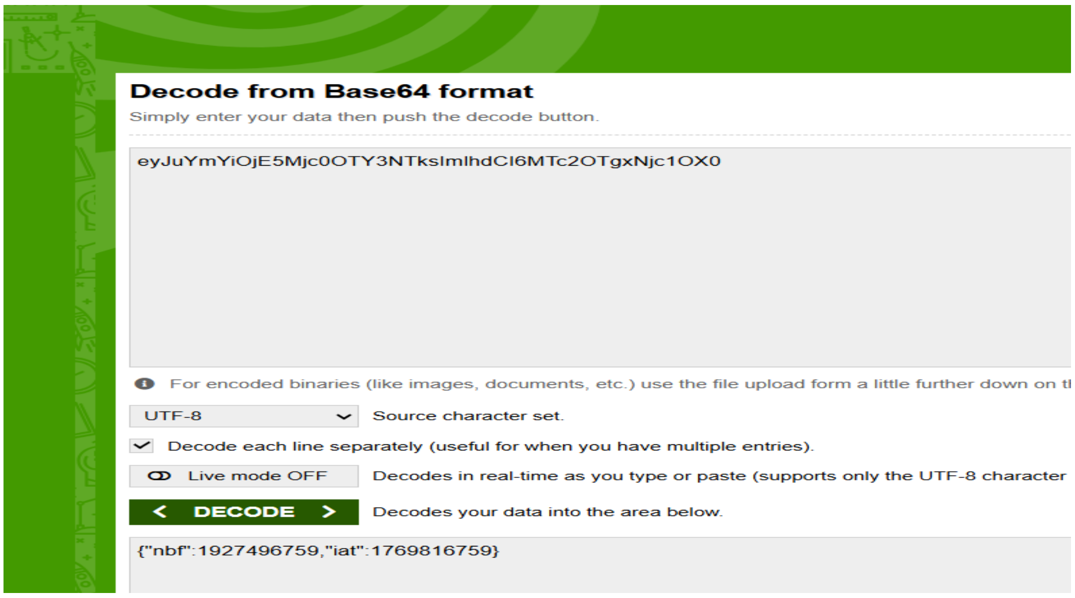
Your JWT token:

```
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJuYmYiOjE5Mjc0OTY3NTksImlhdCI6MTc2OTgxNjc1OX0.9Ubl0Ml
r5lX3zleyC40EcvAWD7OkRX66WgcAGeyQid4
```

Try accessing `/flag` using this token.

Step 1: Extract the JWT Payload

Copy the middle portion of the JWT token. This section is located between the first and second full stops (dots) and represents the Base64-encoded payload.


Decode from Base64 format

Simply enter your data then push the decode button.

```
eyJuYmYiOjE5Mjc0OTY3NTksImlhdCI6MTc2OTgxNjc1OX0
```

For encoded binaries (like images, documents, etc.) use the file upload form a little further down on tl

UTF-8 — Source character set.

✓ Decode each line separately (useful for when you have multiple entries).

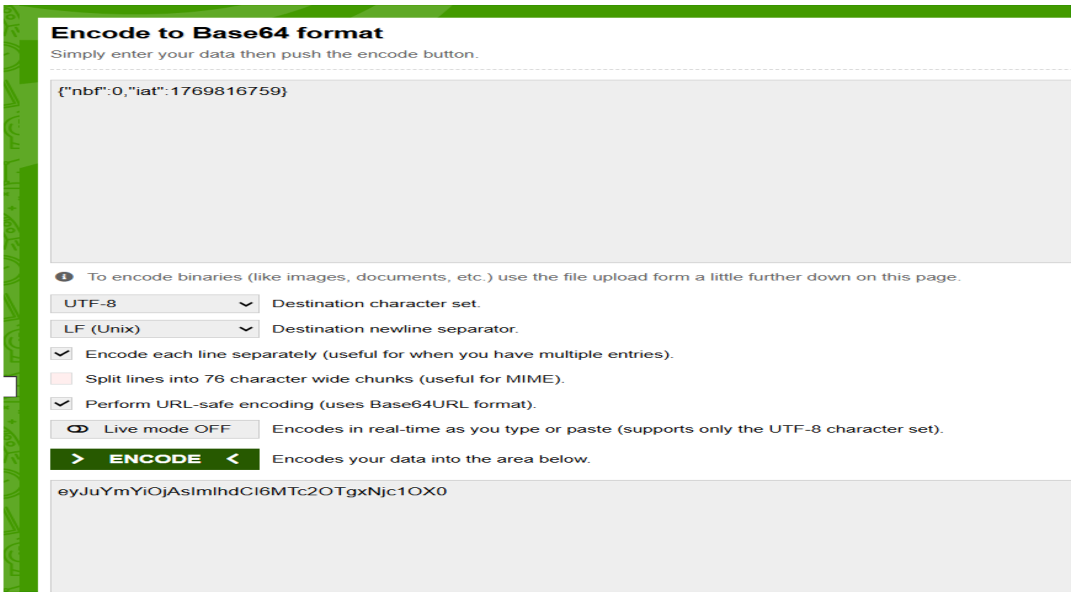
Live mode OFF — Decodes in real-time as you type or paste (supports only the UTF-8 character

DECODE — Decodes your data into the area below.

```
{"nbf":1927496759,"iat":1769816759}
```

Step 2: Decode and Modify the nbf Claim

Paste the extracted payload into a Base64 decoder. After decoding, modify the *nbf* value to 0. This removes the time-based restriction and allows immediate access.



Encode to Base64 format

Simply enter your data then push the encode button.

```
{"nbf":0,"iat":1769816759}
```

To encode binaries (like images, documents, etc.) use the file upload form a little further down on this page.

UTF-8 — Destination character set.

LF (Unix) — Destination newline separator.

✓ Encode each line separately (useful for when you have multiple entries).

Split lines into 76 character wide chunks (useful for MIME).

✓ Perform URL-safe encoding (uses Base64URL format).

Live mode OFF — Encodes in real-time as you type or paste (supports only the UTF-8 character set).

> ENCODE < — Encodes your data into the area below.

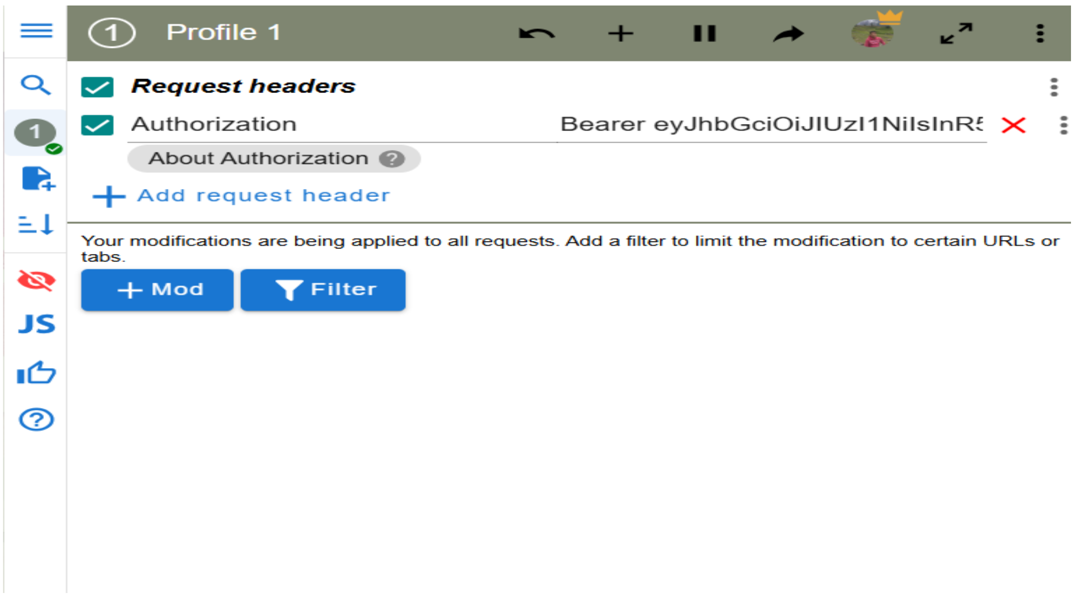
```
eyJuYmYiOjAsImlhdCI6MTc2OTgxNjc1OX0
```

Step 3: Re-encode the Modified Payload

After editing the payload, encode it back into Base64 format. Ensure URL-safe encoding is enabled to preserve JWT compatibility.

Step 4: Reconstruct the JWT

Replace the original payload section of the JWT with the newly encoded payload, while keeping the header and signature unchanged.



✓ **Request headers**

✓ Authorization — Bearer eyJhbGciOiJIUzI1NiIsInR5

About Authorization

Add request header

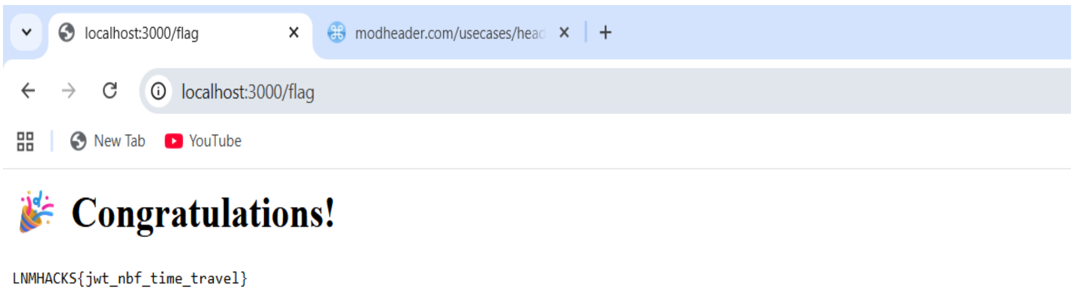
Your modifications are being applied to all requests. Add a filter to limit the modification to certain URLs or tabs.

Step 5: Inject the Modified Token

Install the ModHeader browser extension and add an Authorization header with the value *Bearer <modified_jwt>*. Ensure the extension is enabled.

Step 6: Access the Flag Endpoint

With the modified JWT injected, navigate to the */flag* endpoint. Since the *nbf* restriction has been bypassed, the server validates the token and returns the flag.



🎉 Congratulations!

```
LNMHACKS{jwt_nbf_time_travel}
```

Flag

LNMHACKS{jwt_nbf_time_travel}