

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.



Find Me If You Can

This site will be accessible after:

Wed, 29 Jan 2031 23:45:59 GMT

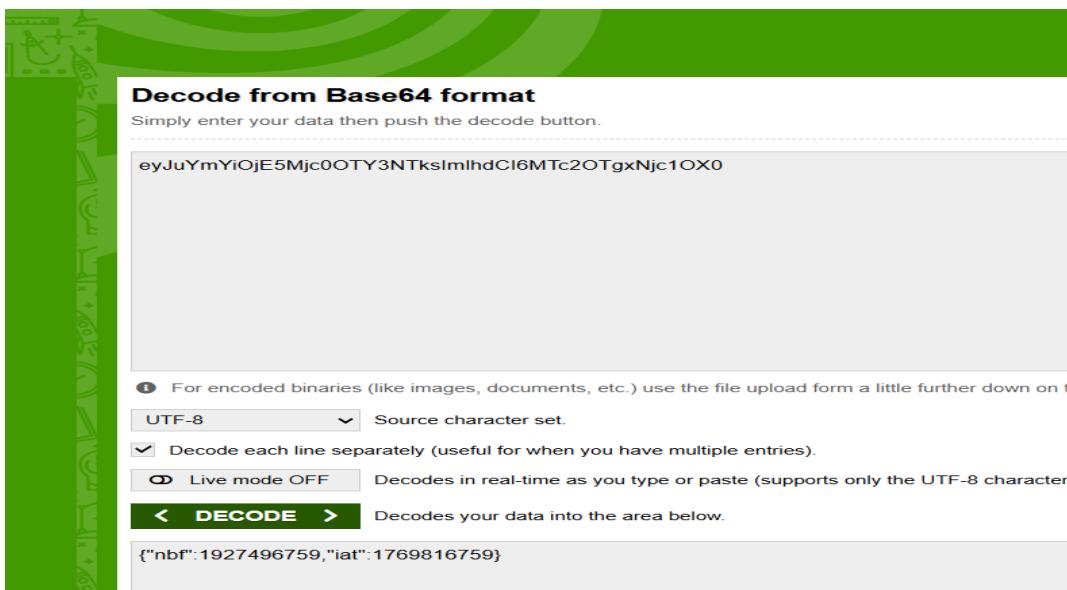
Your JWT token:

```
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJyYmYiOjE5Mjc0OTY3NTksIm1hdCI6MTc2OTgxNjc1OX0.9Ub10M1r51X3zleyC40EcwAWD70kRX66WgcAGeyQid4
```

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.

```
eyJyYmYiOjE5Mjc0OTY3NTksIm1hdCI6MTc2OTgxNjc1OX0
```

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

Source character set: **UTF-8**

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 set).

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.

The screenshot shows a web-based Base64 encoder tool. At the top, it says "Encode to Base64 format" and "Simply enter your data then push the encode button.". Below this is a text input field containing the JSON payload: `{"nbf":0,"iat":1769816759}`. Underneath the input field are several configuration options:

- To encode binaries (like images, documents, etc.) use the file upload form a little further down on this page.
- Character set: UTF-8 (selected) → Destination character set.
- Newline separator: 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).

A large green "ENCODE" button is centered below these settings. To the right of the button, it says "Encodes your data into the area below." Below the button, the encoded output is displayed in a text area:
`eyJJuYmYiOjAsImlihdCi6MTc2OTgxNjc1OXo`

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.

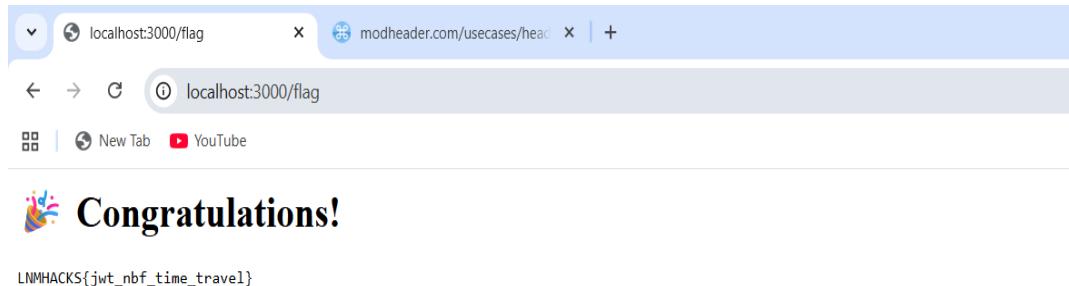
The screenshot shows a browser extension interface for modifying requests. The main header bar displays "Profile 1". On the left, there's a sidebar with icons for search, refresh, add, and other tools. The main panel shows two checked items under "Request headers": "Request headers" and "Authorization". The "Authorization" item has a dropdown menu showing "About Authorization" and a copy icon. To the right of the dropdown is the token: "Bearer eyJhbGciOiJIUzI1NilsInR...". There are also "Mod" and "Filter" buttons at the bottom of the panel. A note at the bottom says "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.



Flag

LNMHACKS{jwt_nbf_time_travel}