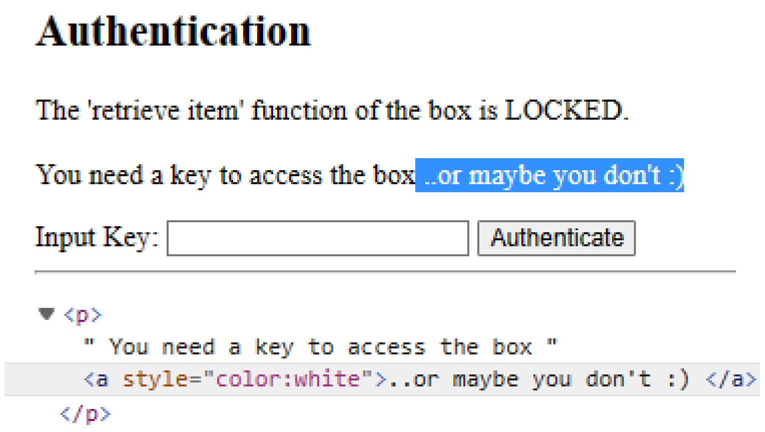
CS561 CTF: self service box

Description

This is a simple web CTF challenge which requires basic web development skills(like HTML, JavaScript, and browser tools).

Solution

1. (optional) Find the hint which has the same color as background, by checking the HTML element or selecting the text.



Authentication

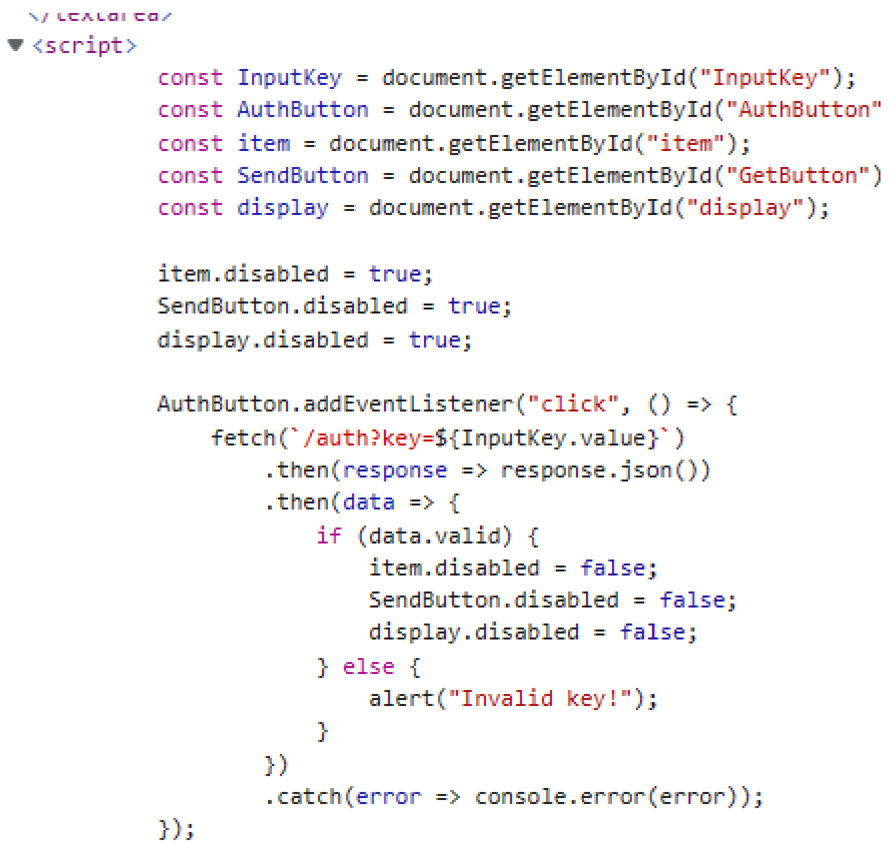
The 'retrieve item' function of the box is LOCKED.

You need a key to access the box ..or maybe you don't :)

Input Key: Authenticate

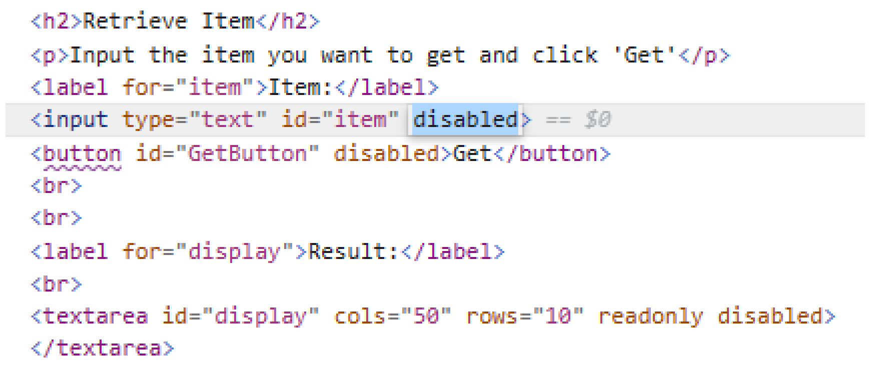
```
▼ <p>
    " You need a key to access the box "
    <a style="color:white">..or maybe you don't :) </a>
  </p>
```

2. Check the JavaScript code of the page, we can see the auth button is only used for setting several elements attribute 'disable' to false, so we can simply ignore the authentication



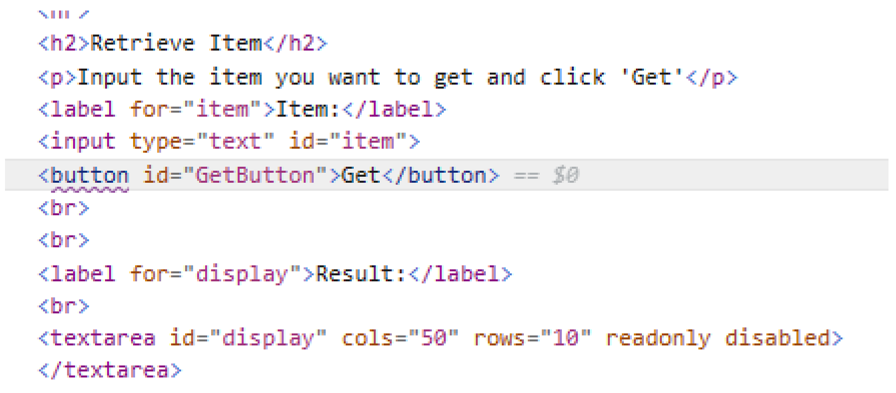
```
</textarea>
▼ <script>
            const InputKey = document.getElementById("InputKey");
            const AuthButton = document.getElementById("AuthButton")
            const item = document.getElementById("item");
            const SendButton = document.getElementById("GetButton")
            const display = document.getElementById("display");

            item.disabled = true;
            SendButton.disabled = true;
            display.disabled = true;

            AuthButton.addEventListener("click", () => {
                fetch(`/auth?key=${InputKey.value}`)
                    .then(response => response.json())
                    .then(data => {
                        if (data.valid) {
                            item.disabled = false;
                            SendButton.disabled = false;
                            display.disabled = false;
                        } else {
                            alert("Invalid key!");
                        }
                    })
                    .catch(error => console.error(error));
            });
```

3. Edit the HTML elements directly -- remove the 'disabled' from the 'item' field, the 'get' button.

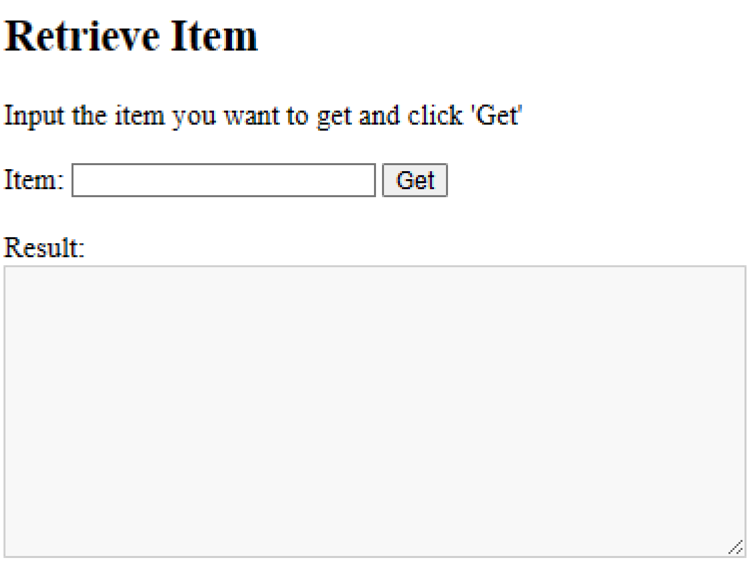


```
<h2>Retrieve Item</h2>
<p>Input the item you want to get and click 'Get'</p>
<label for="item">Item:</label>
<input type="text" id="item" disabled> == $0
<button id="GetButton" disabled>Get</button>
<br>
<br>
<label for="display">Result:</label>
<br>
<textarea id="display" cols="50" rows="10" readonly disabled>
</textarea>
```

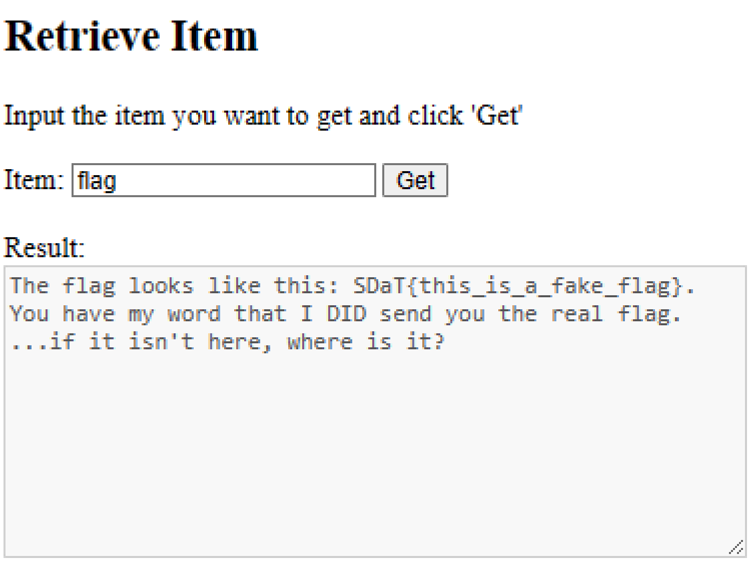
Then we can see these elements are enabled



```
<h2>Retrieve Item</h2>
<p>Input the item you want to get and click 'Get'</p>
<label for="item">Item:</label>
<input type="text" id="item">
<button id="GetButton">Get</button> == $0
<br>
<br>
<label for="display">Result:</label>
<br>
<textarea id="display" cols="50" rows="10" readonly disabled>
</textarea>
```

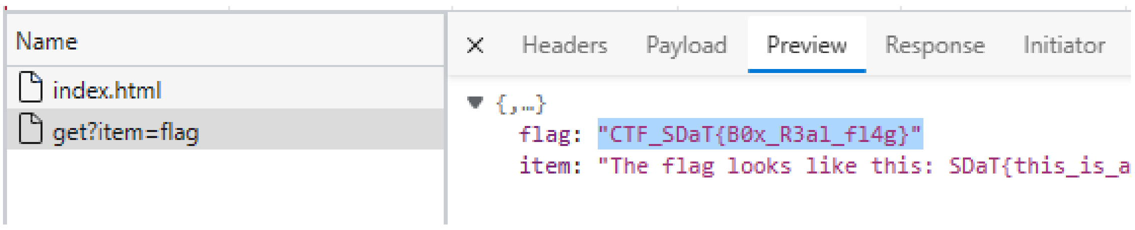


4. After typing 'flag' on the Item field and clicking 'Get' button, we can see the following messages



```
The flag looks like this: SDaT{this_is_a_fake_flag}.
You have my word that I DID send you the real flag.
...if it isn't here, where is it?
```

5. Use the Network tool of the browser to check the response from the backend, we can see the real flag is added to the JSON



```
▼ {,…}
    flag: "CTF_SDaT{B0x_R3al_f14g}"
    item: "The flag looks like this: SDaT{this_is_a
```