SCENARIO

The application contains a reflected cross site scripting vulnerability in the search box functionality but it uses a web application firewall to protect against common XSS vectors by blocking all HTML tags except custom ones. We will try to trigger an alert message by injecting a payload into the search box.

**PROCEDURE**

1. Go to the vulnerable application and try searching for anything.
2. As we’re already told that we need to exploit the vulnerability by triggering an alert.
3. We’ll create a payload and then we will send it to the target using our exploit server containing a custom tag with on-focus event handler that fires alert function as soon as the page loads.

**PAYLOAD**

1. <script>

location = 'https://YOUR-LAB-ID.web-security-academy.net/?search=%3Cxss+id%3Dx+onfocus%3Dalert%28document.cookie%29%20tabindex=1%3E#x';

</script>

**PROOF OF CONCEPT**

****

**REMEDIATION**

1. **Input Validation:** Implement strict whitelisting input validation rules. Accept only known good input. Ensure that all user-submitted data is validated and filtered. Invalid data should be rejected and not reflected back to the user.
2. **Escape Output:** Before displaying user content on the web page, escape all user input. Convert characters with special meanings in HTML, such as <, >, and &, into their escaped counterparts, like &lt;, &gt;, and &amp;.
3. **Update and Configure WAF:** Regularly update your web application firewall (WAF) and configure it to block or sanitize custom HTML tags. A properly configured WAF can provide an additional layer of defense against XSS attacks.
4. **Avoid Reflecting User Data:** Avoid reflecting user data directly back onto the page. If necessary, ensure it's securely validated and sanitized.
5. **HTTP-Only and Secure Cookie Flags:** Ensure that session cookies are flagged as HTTP-Only. This ensures that JavaScript cannot access them, reducing the risk if an attacker is able to execute a script.