SCENARIO

The application contains a simple DOM based cross site scripting vulnerability in the search query tracking functionality which uses **document.write** function to write data out to the page and that function is called with data from **location.search** which we can control using the website URL, so we will try to exploit it by injecting some payload containing malicious script in the comment box.

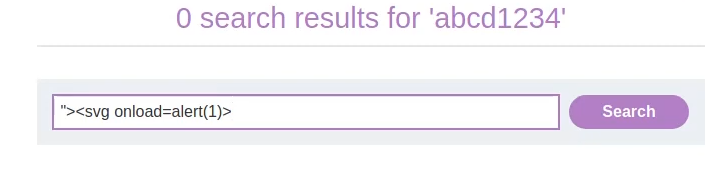
**PROCEDURE**

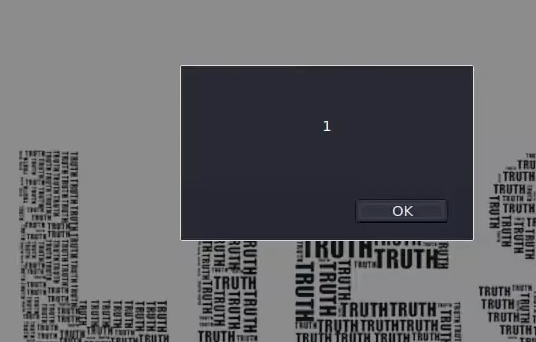
1. Go to the vulnerable web application.
2. Try to search for something in the search box and hit enter.
3. Now inspect the search box element and below it we can see that our searched keyword is injected into HTML image tag.
4. So, we will modify the search key word and change it with the payload given below which will force the application to execute the payload everytime that image element is loaded
5. Now we get an alert, it is clear that we can exploit the application more by injecting DOM based XSS.

**PAYOAD**

"><svg onload=alert(1)>

**PROOF OF CONCEPT**

****



**REMEDIATION**

1. **Avoid DOM-Based Operations:** Instead of using functions like document.write or innerHTML, utilize safer methods like textContent or use frameworks that automatically escape data for you.
2. **Validate and Sanitize Input:** User-controlled inputs that are used in DOM manipulation should be both validated (ensuring the data matches expected formats) and sanitized (ensuring any malicious content is removed or made safe).
3. **Content Security Policy (CSP):** Implement a strong Content Security Policy that disallows the execution of inline scripts. While this may not completely prevent all DOM-based XSS attacks, it can make exploitation more difficult.