SCENARIO

The application contains a stored cross site scripting vulnerability in the comment box functionality, website input field to be specific but with angle brackets and double quotes HTML-encoded and single quotes and backslash escaped. We will try to trigger an alert message by injecting a payload into the application.

**PROCEDURE**

1. Go to the vulnerable application and comment anything in any blog with all fields filled and we’ll notice that the website we provided is sent in a href attribute.
2. As we know that some characters are escaped so based on that we will create a payload to inject in the application’s blog’s comment box.
3. It will force the application to trigger an alert whenever clicked on the commentor’s name.

**PAYLOAD**

http://hack?&apos;-alert("HACKED!")-&apos;

**PROOF OF CONCEPT**

**REMEDIATION**

1. **Input Validation and Sanitization:** Always validate and sanitize user input. For the website field, use regular expressions to ensure the input is a valid URL format without any extraneous characters or script payloads.
2. **Escape All Content:** Rather than just escaping a few characters like <, >, ', " and \, ensure that all content displayed on the site which originated from user input is properly escaped or sanitized.
3. **Use Safe Functions:** Instead of inserting user content directly into the web page, use functions that inherently make content safe. For example, when using JavaScript frameworks, utilize built-in functions that automatically escape user input.