Cross Site Scripting (XSS)

Presented by **Ali Tare**

CYBERUS Stud

TABLE OF CONTENTS

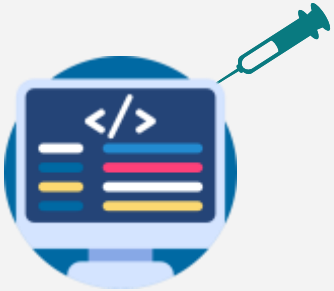
01

What is XSS

DEFINITION OF XSS

What is XSS ?

Cross-site scripting (XSS) is a type of security vulnerability in web applications where attackers inject malicious code into a website, which is then executed by the user's web browser.



DEFINITION OF XSS

Why XSS Happens ?

This occurs when a web application does not properly validate or sanitize user input before displaying it on a webpage, allowing attackers to inject their own code into the application's output.

TABLE OF CONTENTS

02

Impacts of XSS

What's the Impact of XSS ?

- Capture keystrokes
- Perform phishing attacks
- Steal sensitive information
- Perform unauthorized action
- Capture the user's cookies
- Inject trojan functionality into the website

CYBERUS Stud

TABLE OF CONTENTS

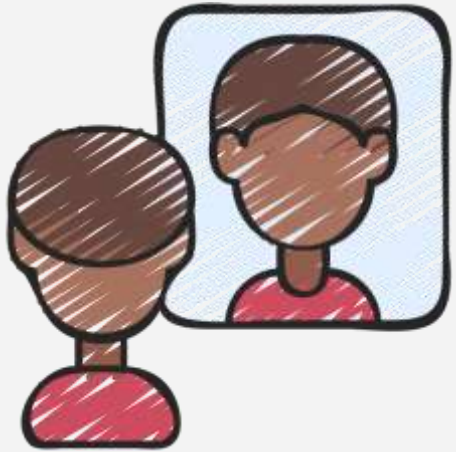
03

Types of XSS

Types of XSS

Reflected XSS

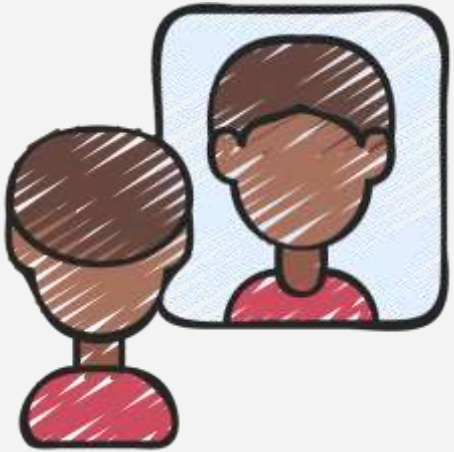
Reflected XSS, also known as non-persistent XSS, occurs when an attacker injects malicious code into a website that is then reflected back to the user. The user's browser will execute the malicious code, potentially leading to data theft or other attacks.



Types of XSS

Reflected XSS

If the website is vulnerable, the script would be reflected back to the user, causing an alert box to appear on the screen. This could be used to steal user credentials, redirect the user to a malicious website, or perform other malicious actions.



Stored XSS

Stored XSS (also known as persistent or second-order XSS) is an attack in which the malicious payload is stored on the server and served to all users who access the vulnerable application.

Stored XSS is considered more dangerous than reflected XSS because it can affect multiple users and persists over time. It also allows an attacker to execute the payload even if the victim does not click on a malicious link or visit a malicious website.

CYBERUS Stud

TABLE OF CONTENTS

04

Exploitation & Mitigation

Payloads

This payload injects a simple JavaScript alert message that will pop up when the victim loads the page. This is a basic example of an XSS payload that demonstrates how an attacker can execute arbitrary code on a victim's browser.

```
<script> alert("XSS"); </script>
```

Payloads

This payload injects an image tag with a source value of "1" and an onerror event that executes a JavaScript alert message. The purpose of this payload is to demonstrate how an attacker can use a benign HTML tag like the image tag to execute arbitrary code.

```
<img src=1 onerror=alert("XSS")>
```

Payloads

This payload displays the cookies of the victim's browser.

```
<script>alert(document.cookie)</script>
```

Mitigation XSS vulnerabilities

- Sanitize User Input: User input should be sanitized to remove any potentially harmful code. This can be done by validating and filtering user input at the server-side, using input validation libraries, or using encoding techniques to render input harmless.

HTML ESCAPING

In HTML, escaping refers to the process of converting special characters to their corresponding character entities to prevent them from being interpreted as part of the HTML markup. This is important in preventing XSS attacks, where attackers can inject malicious scripts by exploiting unescaped special characters in user input fields.

ESCAPING

Symbols	Encoding
&	&amp;
<	&lt;
>	&gt;
"	&quot;
`	&#x60;
'	&#x27;
/	&#x2F;

Mitigation XSS vulnerabilities

- Educate Users: Educating users about the risks of XSS attacks and how to protect themselves can also help prevent these types of attacks. Users should be encouraged to use up-to-date browsers and to be cautious when clicking on links or entering sensitive information

05

Recap and Q&A

RESOURCES

https://owasp.org/www-community/attacks/xss/

https://owasp.org/www-project-top-ten/2017/A7_2017-Cross-Site_Scripting_(XSS).html

“The Web Application Hacker's Handbook”

https://portswigger.net/web-security/cross-site-scripting

"XSS Attacks: Cross Site Scripting Exploits and Defence"

https://github.com/InsiderPhD/hackerone-reports/blob/master/tops_by_bug_type/TOPXSS.md

Thank You

Presented by **Ali Tarek**