

XSS (Cross-Site Scripting)

XSS (Cross-Site Scripting) to jeden z najczęstszych typów ataków na aplikacje webowe, w którym atakujący wstrzykuje **złośliwy kod JavaScript do strony internetowej**, a ten kod wykonuje się w przeglądarce innych użytkowników.

1. Na czym polega XSS?

Jeśli aplikacja **nie filtruje danych od użytkownika**, ktoś może wpisać np.:

```
<script>alert('HACK')</script>
```

A potem strona to wyświetli bez zabezpieczeń:

Witaj: `<script>alert('HACK')</script>`

- przeglądarka użytkownika to wykona
- pojawia się okienko / kradzież danych / inne działania

2. Co może zrobić atak XSS?

Zależnie od typu ataku:

- 1) **Wyświetlanie fałszywych treści** - manipulacja stroną
- 2) **Kradzież cookies**- `document.cookie`- może prowadzić do przejęcia sesji użytkownika
- 3) **Przejęcie konta** Jeśli ukradnie cookie sesji: loguje się jako ofiara
- 4) **Keylogger (w skrajnych przypadkach)**- zapisuje wpisywane hasła

3. Przykład w PHP (problem)

NIEBEZPIECZNE:

```
echo $_POST['komentarz'];
```

Jeśli ktoś wpisze:

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

kod się wykona

4. Jak się przed tym bronić?

`htmlspecialchars()`

```
echo htmlspecialchars($_POST['komentarz']);
```

Co robi? Zamienia:

- `<` → `<`;
- `>` → `>`;

kod NIE działa, tylko się wyświetla

5. Efekt zabezpieczenia

zamiast wykonania:

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

dostajesz:

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

widoczne jako tekst

6. Najprostsza definicja

XSS to sytuacja, gdy:

ktoś wstrzykuje kod JavaScript do strony, a inni użytkownicy go nieświadomie uruchamiają

7. Rodzaje XSS

- **Stored XSS** – zapisany w bazie (najgroźniejszy)
 - **Reflected XSS** – z URL/formularza
 - **DOM-based XSS** – działa w JS po stronie przeglądarki
-

8. Najważniejsze do zapamiętania

- XSS = wstrzyknięcie JS do strony
- działa w przeglądarce użytkownika
- najprostsza obrona: `htmlspecialchars()`

✗ A) WERSJA PODATNA (NIE RÓB TAK)

```
<?php
$komentarz = $_GET['komentarz'];
echo "Komentarz: " . $komentarz;
?>
```

☑ B) WERSJA ZABEZPIECZONA

```
<?php
$komentarz = $_GET['komentarz'];
echo "Komentarz: " .
htmlspecialchars($komentarz);
?>
```

Wersja	Co widzi użytkownik	Czy JS działa
✗ podatna	alert / kradzież cookies	TAK
☑ bezpieczna	tekst <script>...</script>	NIE

Przykład:

Formularz produktu (* pola wymagane)

Nazwa produktu *:

Cena *:

Kategoria *:

```
<?php
```

```
$connect = mysqli_connect("localhost", "root", "", "");
```

```
if (!$connect) {
```

```
    die("Błąd połączenia: " . mysqli_connect_error());
```

```
}
```

```
mysqli_query($connect, "CREATE DATABASE IF NOT EXISTS sklep_db");
```

```

mysqli_select_db($connect, "sklep_db");

$zapytanie = "CREATE TABLE IF NOT EXISTS produkt (
    id_produkt INT AUTO_INCREMENT PRIMARY KEY,
    nazwa VARCHAR(100),
    cena DECIMAL(10,2),
    kategoria VARCHAR(100)
)";

mysqli_query($connect, $zapytanie);

```

/*wyjaśnienie:

```

mysqli_stmt_bind_param(...)
mysqli_stmt_bind_param($stmt, "sds", $nazwa, $cena, $kategoria);

```

Przypisuje wartości do znaków ? w zapytaniu SQL

Litera	Typ danych	Przykład
-----	-----	-----
`s`	string (tekst)	"Laptop"
`d`	double (liczba zmiennoprzecinkowa)	1999.99
`i`	integer	5
`b`	blob	plik

NIE wstawiasz danych do SQL jako tekstu. One są przekazywane osobno do silnika bazy danych, dzięki temu nie mogą zmienić zapytania

mysqli_stmt_execute(\$stmt); - Wykonuje przygotowane zapytanie z przypisanymi danymi, zapytanie trafia do MySQL, dane są faktycznie zapisane w tabeli

```

*/

if (isset($_POST['dodaj'])) {

```

```

$nazwa = $_POST['nazwa'];
$cena = $_POST['cena'];
$katgoria = $_POST['katgoria'];

$stmt = mysqli_prepare($connect,
    "INSERT INTO produkt (nazwa, cena, katgoria) VALUES (?, ?, ?)");

mysqli_stmt_bind_param($stmt, "sds", $nazwa, $cena, $katgoria);
mysqli_stmt_execute($stmt);

echo "<p style='color:green;'>Produkt został dodany!</p>";
}
?>

<!DOCTYPE html>
<html lang="pl">
<head>
    <meta charset="UTF-8">
    <title>Formularz produktu</title>
</head>
<body>

<h2>Formularz produktu (* pola wymagane)</h2>

<form method="post">

    <label>Nazwa produktu *:</label><br>
    <input type="text" name="nazwa" required><br><br>

    <label>Cena *:</label><br>
    <input type="number" name="cena" step="0.01" min="0.01" required><br><br>

```

```
<label>Kategoria *:</label><br>
<select name="kategoria" required>
  <option value="">-- wybierz --</option>
  <option value="elektronika">Elektronika</option>
  <option value="odziez">Odzież</option>
  <option value="spozywcze">Spożywcze</option>
  <option value="ksiazki">Książki</option>
  <option value="inne">Inne</option>
</select><br><br>

<button type="submit" name="dodaj">Dodaj</button>
<button type="reset">Reset</button>

</form>

<hr>

<form method="get" action="pokaz_produkty.php">
  <button type="submit">Pokaż produkty</button>
</form>

</body>
</html>
```

Lista produktów (tabela)

ID	Nazwa	Cena	Kategoria
1	Plecak	10.50 zł	elektronika

Lista produktów (lista punktowana)

- **Plecak** - 10.50 zł (elektronika)

[Powrót](#)

<?php

```
$connect = mysqli_connect("localhost", "root", "", "");
```

```
if (!$connect) {
```

```
    die("Błąd połączenia: " . mysqli_connect_error());
```

```
}
```

```
mysqli_select_db($connect, "sklep_db");
```

```
// Pobranie danych (prepared statement)
```

```
$stmt = mysqli_prepare($connect, "SELECT id_produkt, nazwa, cena, kategoria FROM  
produkt");
```

```
mysqli_stmt_execute($stmt);
```

```
$result = mysqli_stmt_get_result($stmt);
```

```
?>
```

```
<!DOCTYPE html>
```

```
<html lang="pl">
```

```
<head>
```

```
    <meta charset="UTF-8">
```

```
    <title>Produkty</title>
```

```
</head>
```

```
<body>
```

```
<h2>Lista produktów (tabela)</h2>
```

```
<table border="1" cellpadding="5" cellspacing="0">
```

```
<tr>
```

```
<th>ID</th>
```

```
<th>Nazwa</th>
```

```
<th>Cena</th>
```

```
<th>Kategoria</th>
```

```
</tr>
```

```
<?php
```

```
    //htmlspecialchars - zamienia znaki specjalne HTML na bezpieczne, NIE wykona się  
kod JS, tylko wyświetli jako tekst
```

```
    //. - Bo PHP NIE łączy tekstów automatycznie, . w PHP = konkatenacja (łączenie  
tekstu)
```

```
while ($row = mysqli_fetch_assoc($result)) {
```

```
    echo "<tr>";
```

```
    echo "<td>" . $row['id_produkt'] . "</td>";
```

```
    echo "<td>" . htmlspecialchars($row['nazwa']) . "</td>";
```

```
    echo "<td>" . $row['cena'] . " zł</td>";
```

```
    echo "<td>" . htmlspecialchars($row['kategoria']) . "</td>";
```

```
    echo "</tr>";
```

```
}
```

```
?>
```

```
</table>
```

```
<hr>
```

```
<h2>Lista produktów (lista punktowana)</h2>
```

```
<ul>
```

```
<?php
```

```

// ponownie zapytanie (bo wynik został już "zużyty")

$stmt = mysqli_prepare($connect, "SELECT id_produkt, nazwa, cena, kategoria FROM
produkt");

mysqli_stmt_execute($stmt);

$result = mysqli_stmt_get_result($stmt);

while ($row = mysqli_fetch_assoc($result)) {
    echo "<li>";
    echo "<b>" . htmlspecialchars($row['nazwa']) . "</b> - ";
    echo $row['cena'] . " zł (" . htmlspecialchars($row['kategoria']) . ")";
    echo "</li>";
}
?>
</ul>

<hr>

<form action="form.php">
    <button type="submit">Powrót</button>
</form>

</body>
</html>

```