

## CMPE - 279 Assignment – 4

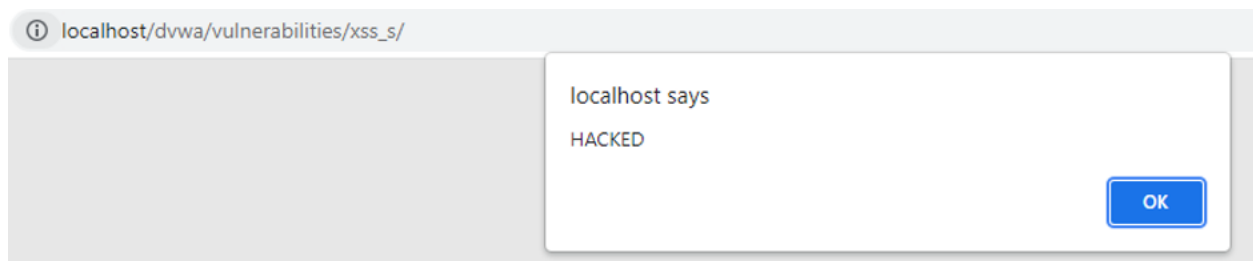
**Name: Saurabh Laxman Dake**  
**SJSU ID: 015970058**

**Name: Adarsh Patil**  
**SJSU ID: 014749228**

**Q1.** Describe the attack you used. How did it work?

**Solution :**

For this, we utilized a stored-XSS attack. A cross-site scripting (XSS) attack is a sort of XSS injection attack that involves injecting malicious scripts into trusted websites. An XSS attack takes place when an attacker uses an online application to send malicious code to other end users often in the form of browser-side scripting. These attacks are enabled by a slew of flaws that can be found anywhere within a web - based application. We know that the stored XSS is kept within the database until it is reset or the payload is deliberately eliminated. Everyone will be redirected to the URL you select.



At the low level, the requested input will not be validated before being utilized in the output text. At the low level, input requests are not validated until and unless the requested input has been included in the output text.

Since the input was not sanitized, we were allowed to insert script tags. Also because alerts are stored in the database, it will be displayed whenever someone accesses this page.



- Home
- Instructions
- Setup / Reset DB
- Brute Force
- Command Injection
- CSRF
- File Inclusion
- File Upload
- Insecure CAPTCHA
- SQL Injection
- SQL Injection (Blind)
- Weak Session IDs
- XSS (DOM)
- XSS (Reflected)
- XSS (Stored)**
- CSP Bypass

## Vulnerability: Stored Cross Site Scripting (XSS)

Name \*

Message \*

### More Information

- <https://owasp.org/www-community/attacks/xss>
- <https://owasp.org/www-community/xss-filter-evasion-cheatsheet>
- [https://en.wikipedia.org/wiki/Cross-site\\_scripting](https://en.wikipedia.org/wiki/Cross-site_scripting)
- <http://www.cgisecurity.com/xss-faq.html>
- <http://www.scriptalert1.com/>



- Home
- Instructions
- Setup / Reset DB
- Brute Force
- Command Injection
- CSRF
- File Inclusion
- File Upload
- Insecure CAPTCHA
- SQL Injection
- SQL Injection (Blind)
- Weak Session IDs
- XSS (DOM)
- XSS (Reflected)
- XSS (Stored)**
- CSP Bypass

## Vulnerability: Stored Cross Site Scripting (XSS)

Name \*

Message \*

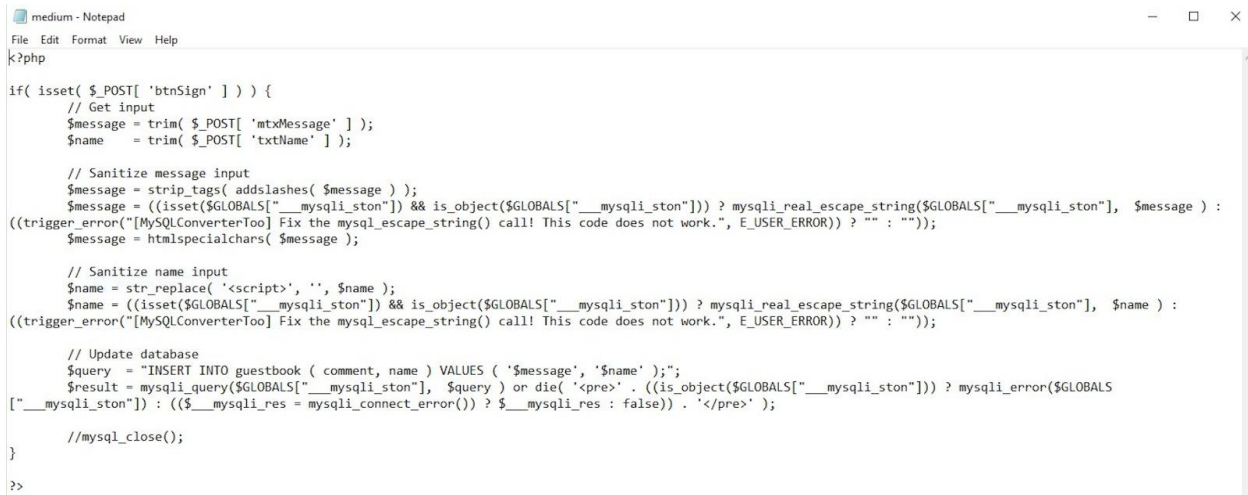
Name: Saurabh  
Message: alert('YOU ARE HACKED')

### More Information

- <https://owasp.org/www-community/attacks/xss>
- <https://owasp.org/www-community/xss-filter-evasion-cheatsheet>
- [https://en.wikipedia.org/wiki/Cross-site\\_scripting](https://en.wikipedia.org/wiki/Cross-site_scripting)
- <http://www.cgisecurity.com/xss-faq.html>
- <http://www.scriptalert1.com/>

**Q2.** Does your attack work in “Medium” security level?

**Solution :**



```
medium - Notepad
File Edit Format View Help
k?php

if( isset( $_POST[ 'btnSign' ] ) ) {
    // Get input
    $message = trim( $_POST[ 'mtxMessage' ] );
    $name     = trim( $_POST[ 'txtName' ] );

    // Sanitize message input
    $message = strip_tags( addslashes( $message ) );
    $message = ((isset($GLOBALS["__mysqli_ston"]) && is_object($GLOBALS["__mysqli_ston"]))) ? mysqli_real_escape_string($GLOBALS["__mysqli_ston"], $message ) :
    ((trigger_error("MySQLConverterToo Fix the mysql_escape_string() call! This code does not work.", E_USER_ERROR)) ? "" : "");
    $message = htmlspecialchars( $message );

    // Sanitize name input
    $name = str_replace( '<script>', '', $name );
    $name = ((isset($GLOBALS["__mysqli_ston"]) && is_object($GLOBALS["__mysqli_ston"]))) ? mysqli_real_escape_string($GLOBALS["__mysqli_ston"], $name ) :
    ((trigger_error("MySQLConverterToo Fix the mysql_escape_string() call! This code does not work.", E_USER_ERROR)) ? "" : "");

    // Update database
    $query = "INSERT INTO guestbook ( comment, name ) VALUES ( '$message', '$name' );";
    $result = mysqli_query($GLOBALS["__mysqli_ston"], $query ) or die( '<pre>' . ((is_object($GLOBALS["__mysqli_ston"]))) ? mysqli_error($GLOBALS
    ["__mysqli_ston"]): (($__mysqli_res = mysqli_connect_error()) ? $__mysqli_res : false)) . '</pre>' );

    //mysqli_close();
}

?>
```

No, it didn't work at the 'Medium' security level.

Strip labels () eliminates all HTML labels from the message field and guarantees that any content counting citations or bad characters that pass through this approach are encoded into comparable HTML characters. Our XSS payload is rendered useless as a result. Hence , reducing the script tag in order to sanitize input does not make it more susceptible or vulnerable.

**Q3.** Set the security mode to “Low” and examine the code that is vulnerable, and then set the security mode to “High” and reexamine the same code. What changed? How do the changes prevent the attack from succeeding?

**Solution :**

The security mode is set to 'Low' as it is explained in question 1. The same cannot be said for 'High' mode. We have sanitized name and message field inputs , and the script tag is trimmed with each of the script characters, so that if it gets a script word, it will simply deny it and the attack won't occur.

```
low - Notepad
File Edit Format View Help
k?php

if( isset( $_POST[ 'btnSign' ] ) ) {
    // Get input
    $message = trim( $_POST[ 'mtxMessage' ] );
    $name     = trim( $_POST[ 'txtName' ] );

    // Sanitize message input
    $message = stripslashes( $message );
    $message = ((isset($GLOBALS["__mysqli_ston"]) && is_object($GLOBALS["__mysqli_ston"])) ? mysqli_real_escape_string($GLOBALS["__mysqli_ston"], $message ) :
    ((trigger_error("MySQLConverterToo] Fix the mysql_escape_string() call! This code does not work.", E_USER_ERROR)) ? "" : ""));

    // Sanitize name input
    $name = ((isset($GLOBALS["__mysqli_ston"]) && is_object($GLOBALS["__mysqli_ston"])) ? mysqli_real_escape_string($GLOBALS["__mysqli_ston"], $name ) :
    ((trigger_error("MySQLConverterToo] Fix the mysql_escape_string() call! This code does not work.", E_USER_ERROR)) ? "" : ""));

    // Update database
    $query = "INSERT INTO guestbook ( comment, name ) VALUES ( '$message', '$name' );";
    $result = mysqli_query($GLOBALS["__mysqli_ston"], $query ) or die( '<pre>' . ((is_object($GLOBALS["__mysqli_ston"])) ? mysqli_error($GLOBALS
["__mysqli_ston"]) : (($__mysqli_res = mysqli_connect_error()) ? $__mysqli_res : false)) . '</pre>' );

    //mysqli_close();
}

?>
```

```
high - Notepad
File Edit Format View Help
k?php

if( isset( $_POST[ 'btnSign' ] ) ) {
    // Get input
    $message = trim( $_POST[ 'mtxMessage' ] );
    $name     = trim( $_POST[ 'txtName' ] );

    // Sanitize message input
    $message = strip_tags( addslashes( $message ) );
    $message = ((isset($GLOBALS["__mysqli_ston"]) && is_object($GLOBALS["__mysqli_ston"])) ? mysqli_real_escape_string($GLOBALS["__mysqli_ston"], $message ) :
    ((trigger_error("MySQLConverterToo] Fix the mysql_escape_string() call! This code does not work.", E_USER_ERROR)) ? "" : ""));
    $message = htmlspecialchars( $message );

    // Sanitize name input
    $name = preg_replace( '/<(.*)s(.*)c(.*)r(.*)i(.*)p(.*)t/i', '', $name );
    $name = ((isset($GLOBALS["__mysqli_ston"]) && is_object($GLOBALS["__mysqli_ston"])) ? mysqli_real_escape_string($GLOBALS["__mysqli_ston"], $name ) :
    ((trigger_error("MySQLConverterToo] Fix the mysql_escape_string() call! This code does not work.", E_USER_ERROR)) ? "" : ""));

    // Update database
    $query = "INSERT INTO guestbook ( comment, name ) VALUES ( '$message', '$name' );";
    $result = mysqli_query($GLOBALS["__mysqli_ston"], $query ) or die( '<pre>' . ((is_object($GLOBALS["__mysqli_ston"])) ? mysqli_error($GLOBALS
["__mysqli_ston"]) : (($__mysqli_res = mysqli_connect_error()) ? $__mysqli_res : false)) . '</pre>' );

    //mysqli_close();
}

?>
```