DEPARTMENT OF
COMPUTER SCIENCE & ENGINEERING

Discover. Learn. Empower.

CHANDIGARH UNIVERSITY
UNIVERSITY INSTITUTE OF ENGINEERING
DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Submitted By: Pranjal Kumar	**Submitted To:** Renuka Ratten
Subject Name	WEB AND MOBILE SECURITY LAB
Subject Code	20CSP-338
Branch	Computer Science
Semester	5th

DEPARTMENT OF
COMPUTER SCIENCE & ENGINEERING

Discover. Learn. Empower.

UNIVERSITY INSTITUTE OF ENGINEERING

Department of Computer Science & Engineering

Subject Name: WEB AND MOBILE SECURITY LAB

Subject Code: 20CSP-338

Submitted to: Renuka Ratten

Faculty name: Renuka Ratten

Submitted by: Pranjal Kumar

Name: Pranjal Kumar

UID: 20BCS3504

Section: 607

Group: B

Ex. No	List of Experiments	Date	Conduct (MM: 12)	Viva (MM: 10)	Record (MM: 8)	Total (MM: 30)	Remarks/Signature
1.1	Open any website on computer system and identify http packet on monitoring tool like Wireshark.	19/08/22					
1.2	Design a method to simulate the html injection and cross sites cripting to exploit the attackers	28/08/22					
1.3	Working of CSRF (cross site request forgery) attack/ Vulnerability.	16/09/22					
2.1	Design methods to break authentication schemes (SQL Injection Attack).	04/09/22					
2.2							
2.3							
2.4							
3.1							
3.2							
3.3							

Experiment No 2

Aim:
Design a method to simulate the html injection and cross site scripting to exploit the attackers

Objective:
To test HTML and xss injection

Software/Hardware Requirements:
Window 7 and above version

Tools to be used:
1. OSASP Mutillidae ll: Web Pwn Mass Production
2. XSS game site

Introduction:
Acunetix is a web application security scanner that gives you a 360-degree view of the organization's security. This end-to-end web security scanner can identify over 7000 vulnerabilities like XSS and misconfigurations. It has capabilities for scanning all pages, web apps, complex web applications, etc. Acunetix offers specialized technologies that let you detect more and fix faster

Html Injection
1. The attacker finds the vulnerable web application.
2. The attacker sends the modified URL to the user by any means, usually via email. This URL has text injected.
3. By clicking on the URL user is navigated to the attackers webpage, looks like legitimate one.
4. User asked the information like username, password, card pins etc.
5. This information gets transferred to the attackers server.

for example

www.testing.com/siteAdcontent?divMessage=<h1>Click Here!!</h1> It is possible to modify it as −
www.testing.com/siteAdcontent?divMessage=<hack><h1>Do not Click!!</h1><hack>

Cross Site Scripting(XSS)

It happens whenever an application takes untrusted data and sends it to the client browser without validation. This allows attackers to execute malacious scripts in the victim's browser which can result in user sessions hijack, defacing web sites or redirect the user to malicious sites.

Steps/Method/Coding:

HTML Injection

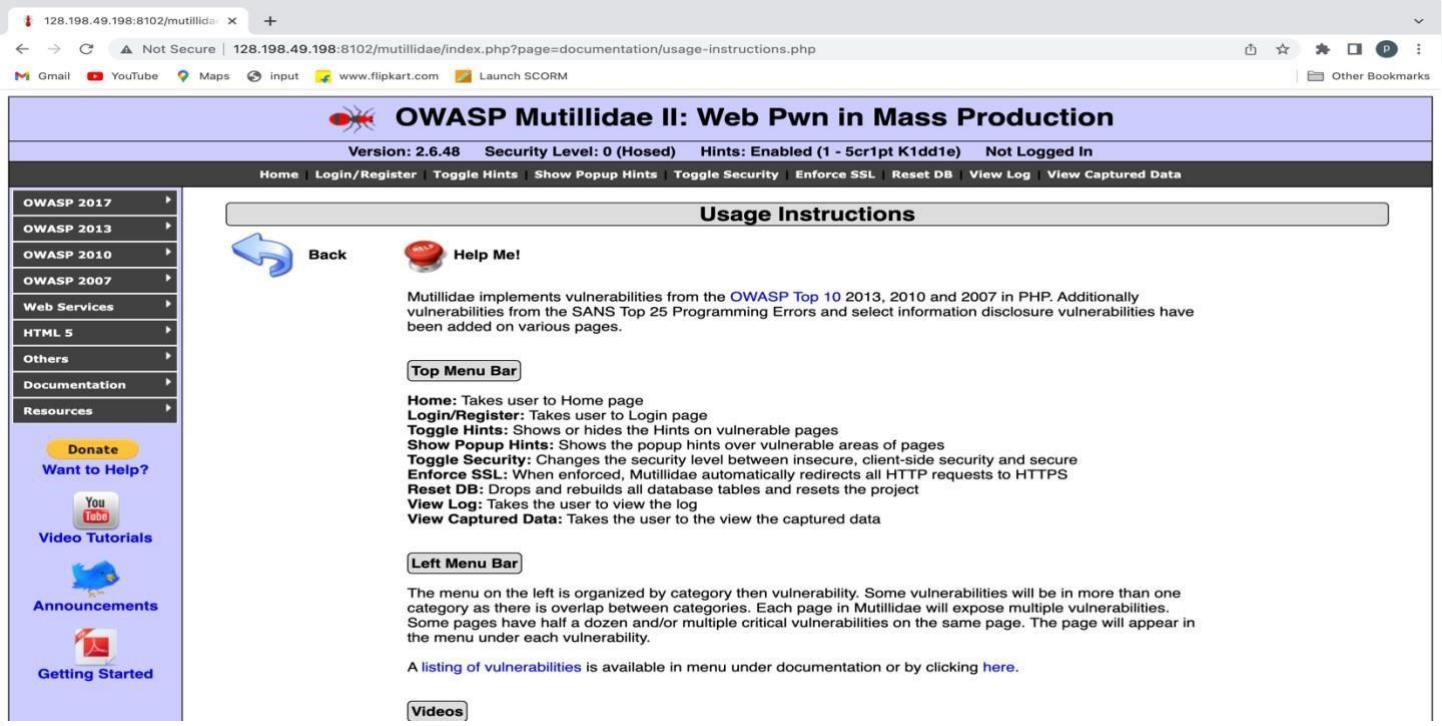
1.Open website : OWASP Mutillidae II: Web Pwn in Mass Production (URL: http://128.198.49.198:8102/mutillidae/index.php?page=documentation/usage-instructions.php)

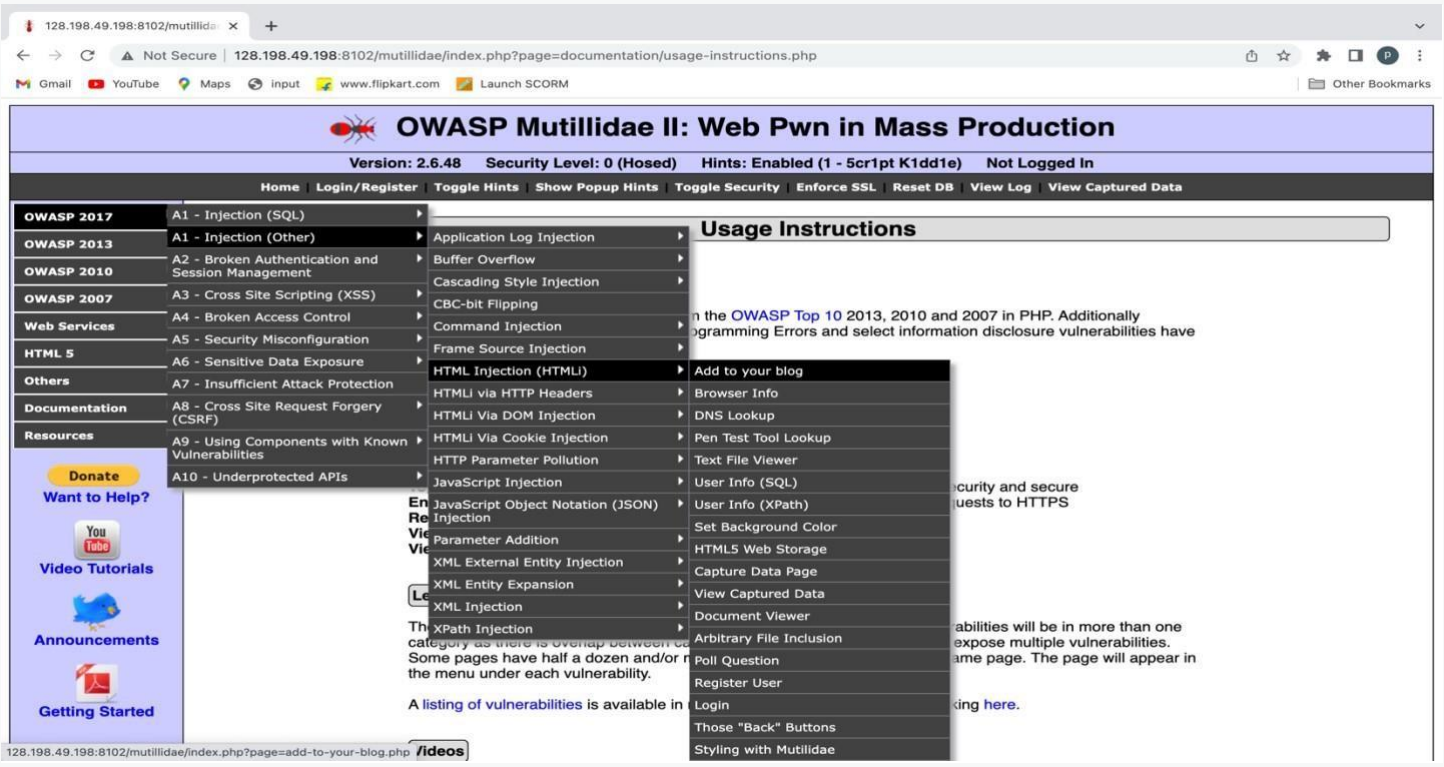
2.Now, we'll be redirected to the web page which is suffering from an **HTML Injection vulnerability** which allows the user to submit his entry in the blog.
3.On the left hand side, click on OWASP 2017❼A1-injection(others)❼HTML injection❼Add to your blog
(check screenshot)

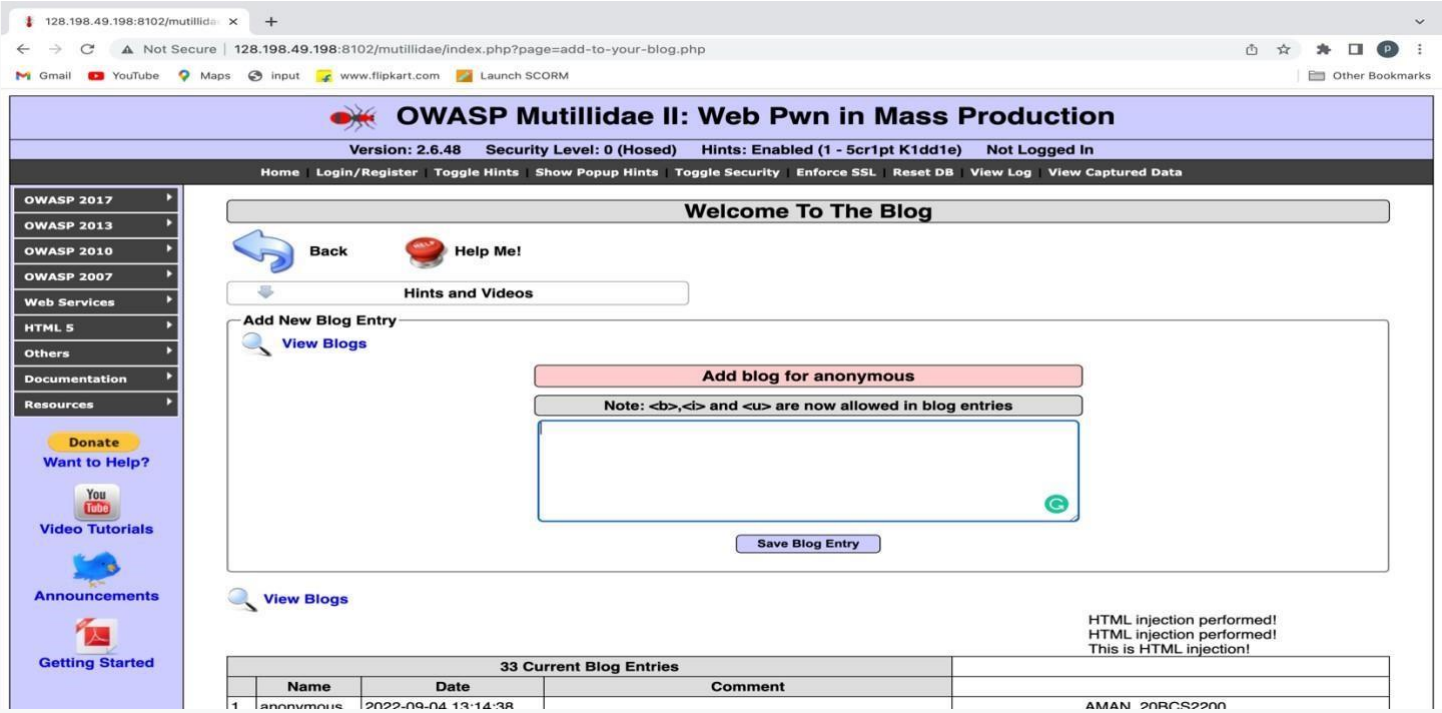
4.Welcome to blog window will appear on the screen. Now, let's try to inject malicious code. Enter the HTML code inside the given text area in order to set up the HTML attack.

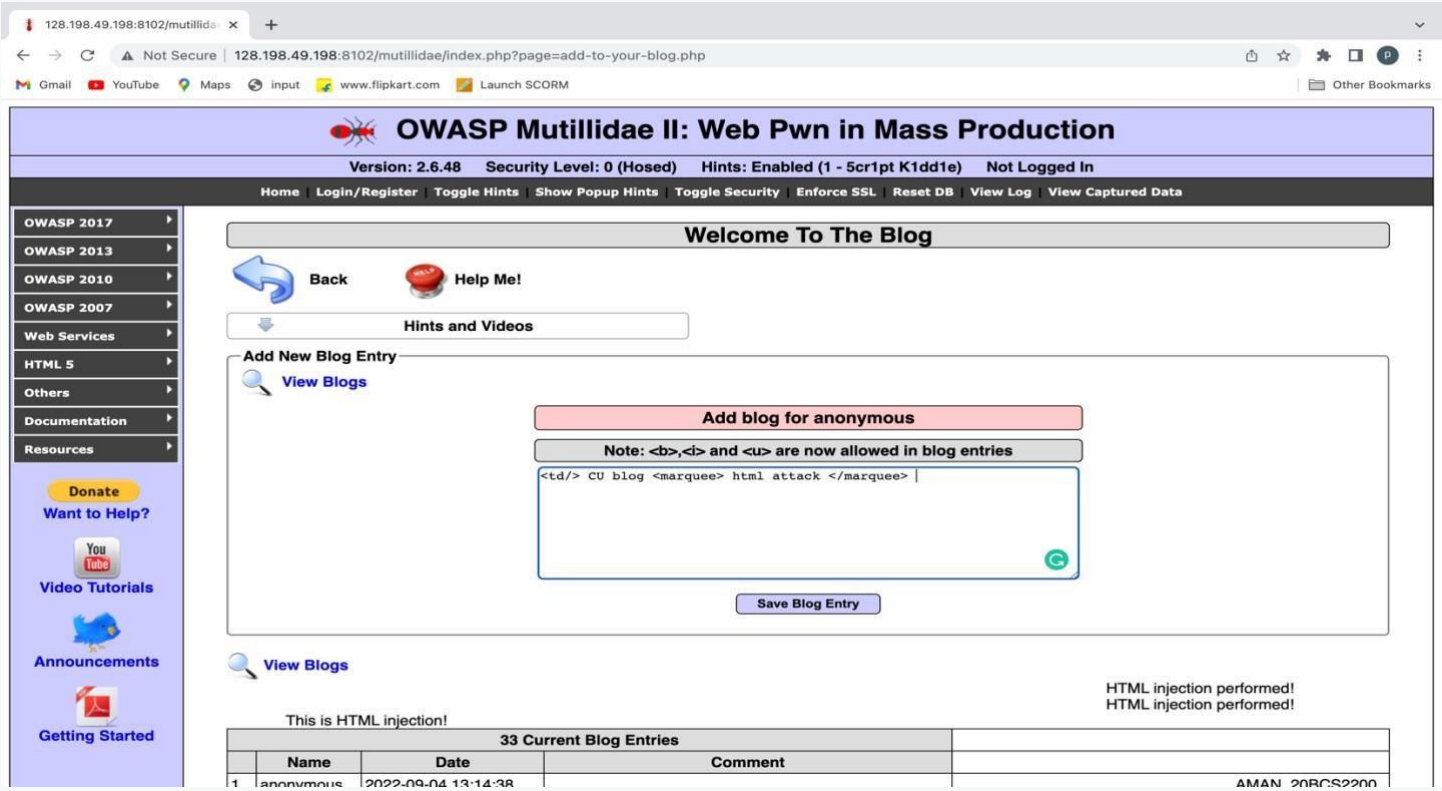
5.For example injected code is : <td/> CU blog <marquee> html attack </marquee> then save blog entry

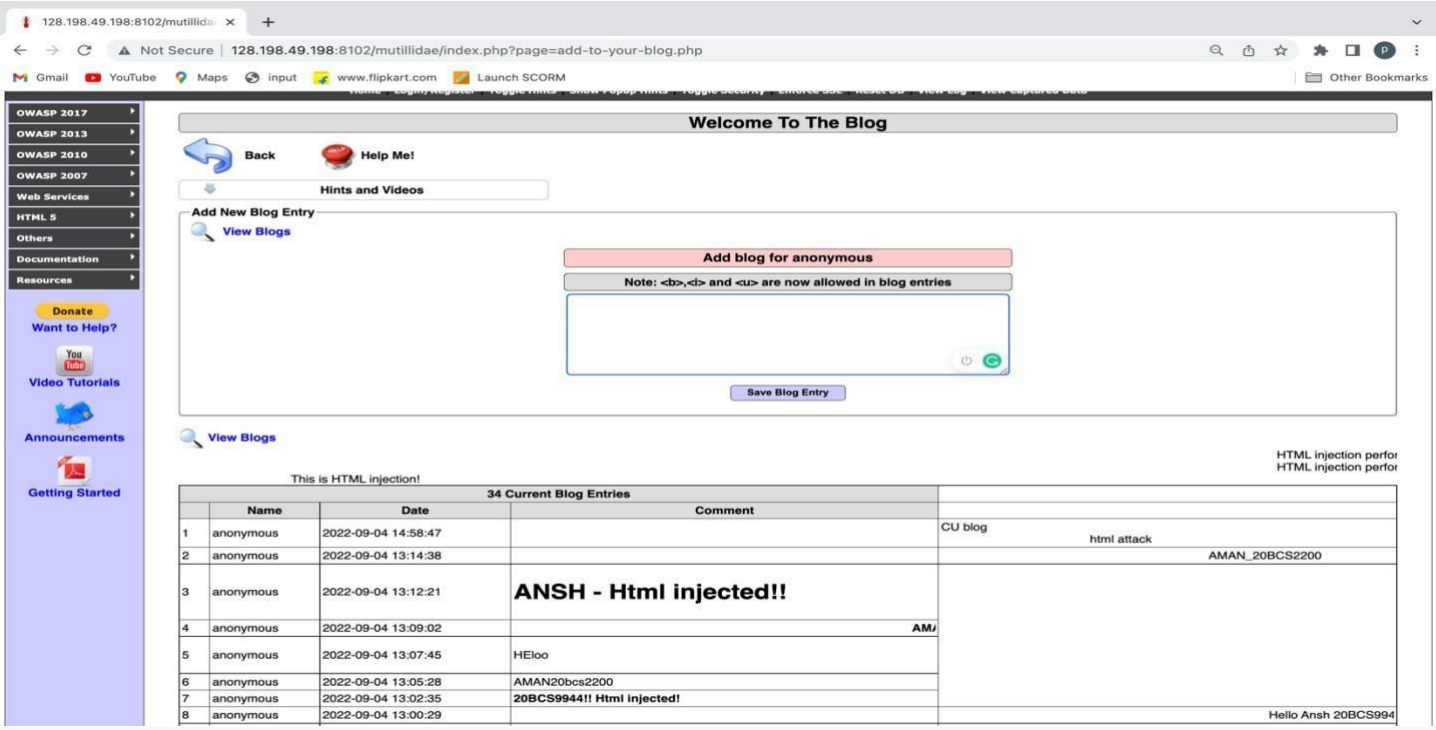
6.That html code is thus now into the application's web server, which gets rendered every time whenever the victim visits this malicious page, he'll always have this code which looks official to him.

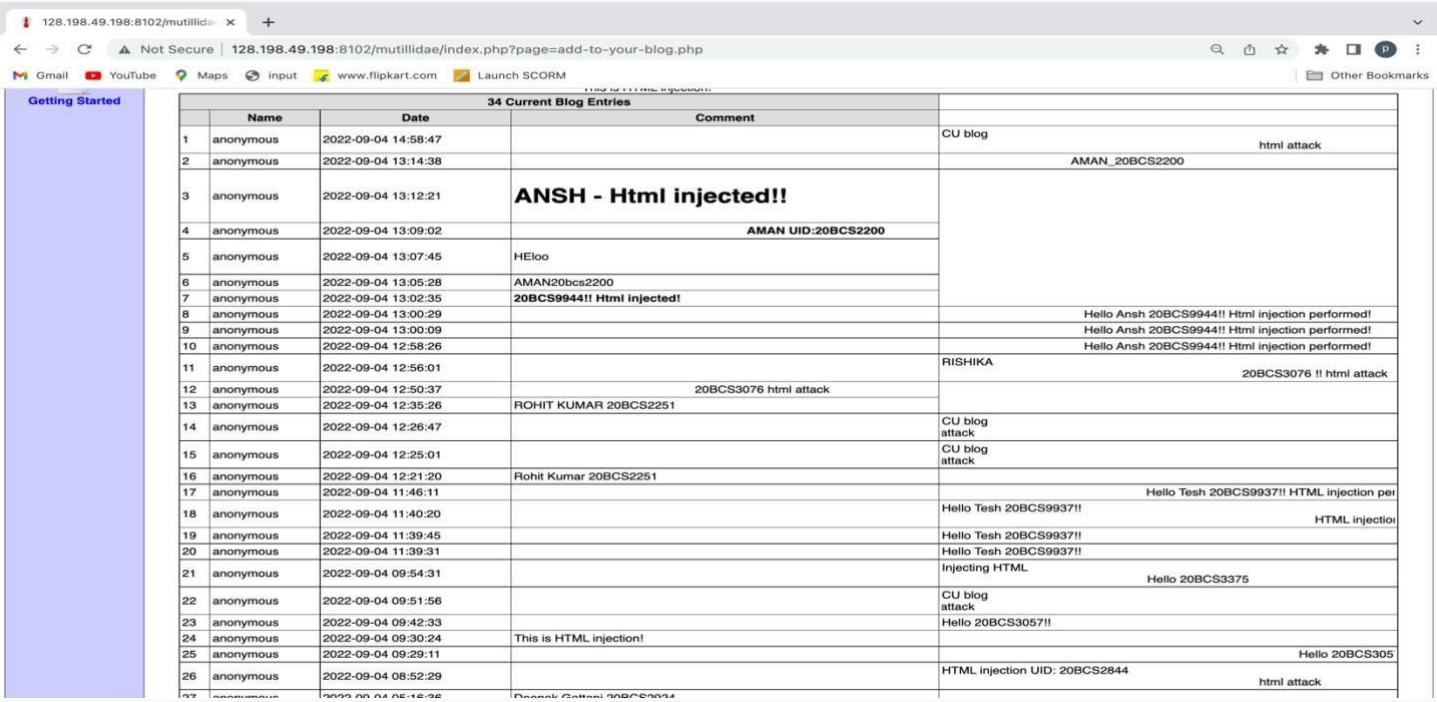

128.198.49.198:8102/mutillidae/index.php?page=documentation/usage-instructions.php
OWASP Mutillidae II: Web Pwn in Mass Production
Version: 2.6.48 Security Level: 0 (Hosed) Hints: Enabled (1 - 5cr1pt K1dd1e) Not Logged In
Home | Login/Register | Toggle Hints | Show Popup Hints | Toggle Security | Enforce SSL | Reset DB | View Log | View Captured Data
OWASP 2017
OWASP 2013
OWASP 2010
OWASP 2007
Web Services
HTML 5
Others
Documentation
Resources
Donate
Want to Help?
Video Tutorials
Announcements
Getting Started
Usage Instructions
Back
Help Me!
Mutillidae implements vulnerabilities from the OWASP Top 10 2013, 2010 and 2007 in PHP. Additionally vulnerabilities from the SANS Top 25 Programming Errors and select information disclosure vulnerabilities have been added on various pages.
Top Menu Bar
Home: Takes user to Home page
Login/Register: Takes user to Login page
Toggle Hints: Shows or hides the Hints on vulnerable pages
Show Popup Hints: Shows the popup hints over vulnerable areas of pages
Toggle Security: Changes the security level between insecure, client-side security and secure
Enforce SSL: When enforced, Mutillidae automatically redirects all HTTP requests to HTTPS
Reset DB: Drops and rebuilds all database tables and resets the project
View Log: Takes the user to view the log
View Captured Data: Takes the user to the view the captured data
Left Menu Bar
The menu on the left is organized by category then vulnerability. Some vulnerabilities will be in more than one category as there is overlap between categories. Each page in Mutillidae will expose multiple vulnerabilities. Some pages have half a dozen and/or multiple critical vulnerabilities on the same page. The page will appear in the menu under each vulnerability.
A listing of vulnerabilities is available in menu under documentation or by clicking here.
Videos

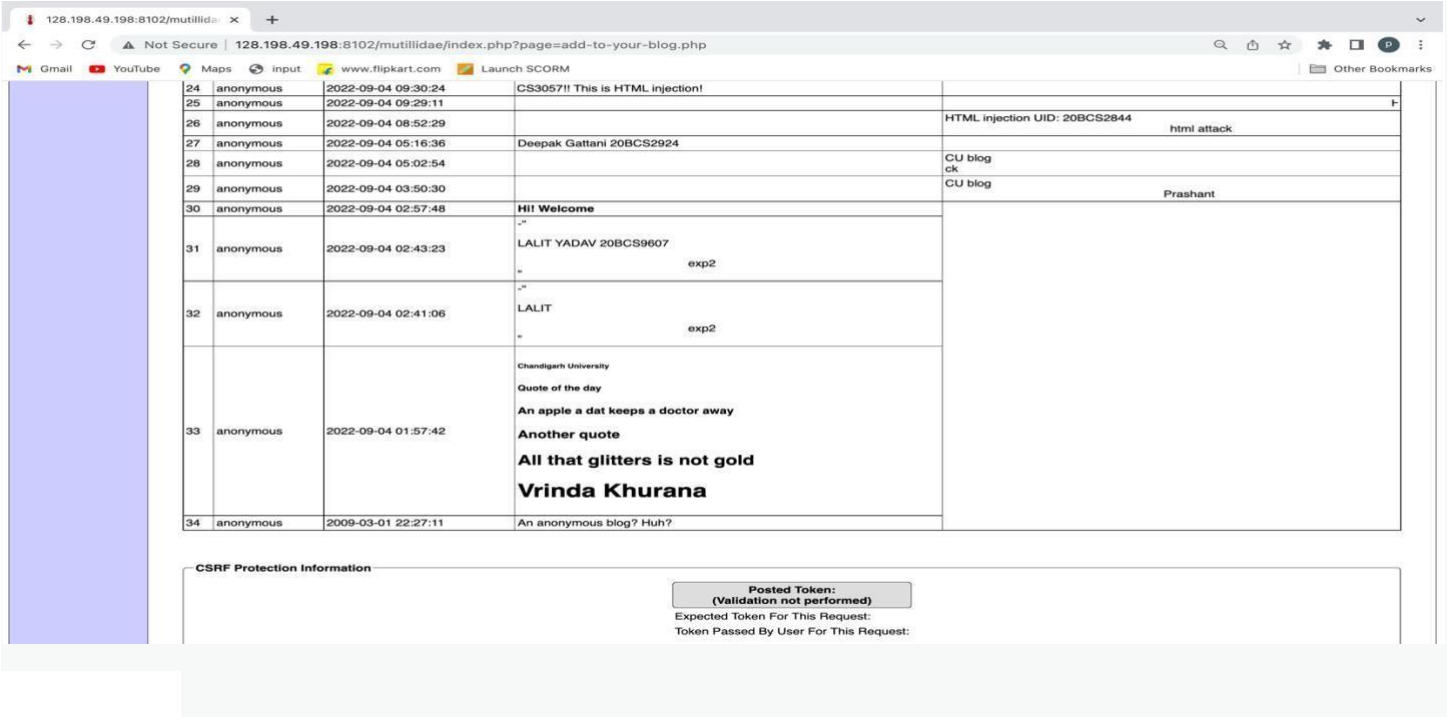

128.198.49.198:8102/mutillidae/index.php?page=documentation/usage-instructions.php
OWASP Mutillidae II: Web Pwn in Mass Production
Version: 2.6.48 Security Level: 0 (Hosed) Hints: Enabled (1 - 5cr1pt K1dd1e) Not Logged In
Home | Login/Register | Toggle Hints | Show Popup Hints | Toggle Security | Enforce SSL | Reset DB | View Log | View Captured Data
OWASP 2017
OWASP 2013
OWASP 2010
OWASP 2007
Web Services
HTML 5
Others
Documentation
Resources
A1 - Injection (SQL)
A1 - Injection (Other)
A2 - Broken Authentication and Session Management
A3 - Cross Site Scripting (XSS)
A4 - Broken Access Control
A5 - Security Misconfiguration
A6 - Sensitive Data Exposure
A7 - Insufficient Attack Protection
A8 - Cross Site Request Forgery (CSRF)
A9 - Using Components with Known Vulnerabilities
A10 - Underprotected APIs
Application Log Injection
Buffer Overflow
Cascading Style Injection
CBC-bit Flipping
Command Injection
Frame Source Injection
HTML Injection (HTMLi)
HTMLi via HTTP Headers
HTMLi Via DOM Injection
HTMLi Via Cookie Injection
HTTP Parameter Pollution
JavaScript Injection
JavaScript Object Notation (JSON) Injection
Parameter Addition
XML External Entity Injection
XML Entity Expansion
XML Injection
XPath Injection
Add to your blog
Browser Info
DNS Lookup
Pen Test Tool Lookup
Text File Viewer
User Info (SQL)
User Info (XPath)
Set Background Color
HTML5 Web Storage
Capture Data Page
View Captured Data
Document Viewer
Arbitrary File Inclusion
Poll Question
Register User
Login
Those "Back" Buttons
Styling with Mutilidae
Donate
Want to Help?
Video Tutorials
Announcements
Getting Started
Usage Instructions
128.198.49.198:8102/mutillidae/index.php?page=add-to-your-blog.php


128.198.49.198:8102/mutillida
Not Secure | 128.198.49.198:8102/mutillidae/index.php?page=add-to-your-blog.php
Gmail YouTube Maps input www.flipkart.com Launch SCORM
Other Bookmarks
OWASP Mutillidae II: Web Pwn in Mass Production
Version: 2.6.48 Security Level: 0 (Hosed) Hints: Enabled (1 - 5cr1pt K1dd1e) Not Logged In
Home | Login/Register | Toggle Hints | Show Popup Hints | Toggle Security | Enforce SSL | Reset DB | View Log | View Captured Data
OWASP 2017
OWASP 2013
OWASP 2010
OWASP 2007
Web Services
HTML 5
Others
Documentation
Resources
Donate
Want to Help?
Video Tutorials
Announcements
Getting Started
Welcome To The Blog
Back
Help Me!
Hints and Videos
Add New Blog Entry
View Blogs
Add blog for anonymous
Note: <b>,<i> and <u> are now allowed in blog entries
Save Blog Entry
View Blogs
HTML injection performed!
HTML injection performed!
This is HTML injection!
33 Current Blog Entries
Name
Date
Comment


128.198.49.198:8102/mutillida
Not Secure | 128.198.49.198:8102/mutillidae/index.php?page=add-to-your-blog.php
Gmail YouTube Maps input www.flipkart.com Launch SCORM
Other Bookmarks
OWASP Mutillidae II: Web Pwn in Mass Production
Version: 2.6.48 Security Level: 0 (Hosed) Hints: Enabled (1 - 5cr1pt K1dd1e) Not Logged In
Home | Login/Register | Toggle Hints | Show Popup Hints | Toggle Security | Enforce SSL | Reset DB | View Log | View Captured Data
OWASP 2017
OWASP 2013
OWASP 2010
OWASP 2007
Web Services
HTML 5
Others
Documentation
Resources
Donate
Want to Help?
Video Tutorials
Announcements
Getting Started
Welcome To The Blog
Back
Help Me!
Hints and Videos
Add New Blog Entry
View Blogs
Add blog for anonymous
Note: <b>,<i> and <u> are now allowed in blog entries
<td/> CU blog <marquee> html attack </marquee>
Save Blog Entry
View Blogs
HTML injection performed!
HTML injection performed!
This is HTML injection!
33 Current Blog Entries
Name
Date
Comment


128.198.49.198:8102/mutillida
Not Secure | 128.198.49.198:8102/mutillidae/index.php?page=add-to-your-blog.php
Gmail YouTube Maps input www.flipkart.com Launch SCORM
Other Bookmarks
OWASP 2017
OWASP 2013
OWASP 2010
OWASP 2007
Web Services
HTML 5
Others
Documentation
Resources
Donate
Want to Help?
Video Tutorials
Announcements
Getting Started
Welcome To The Blog
Back
Help Me!
Hints and Videos
Add New Blog Entry
View Blogs
Add blog for anonymous
Note: <b>,<i> and <u> are now allowed in blog entries
Save Blog Entry
View Blogs
This is HTML injection!
34 Current Blog Entries
Name
Date
Comment
1 anonymous 2022-09-04 14:58:47 CU blog html attack
2 anonymous 2022-09-04 13:14:38 AMAN_20BCS2200
3 anonymous 2022-09-04 13:12:21 ANSH - Html injected!!
4 anonymous 2022-09-04 13:09:02
5 anonymous 2022-09-04 13:07:45 HEloo
6 anonymous 2022-09-04 13:05:28 AMAN20bcs2200
7 anonymous 2022-09-04 13:02:35 20BCS9944!! Html injected!
8 anonymous 2022-09-04 13:00:29 Hello Ansh 20BCS994

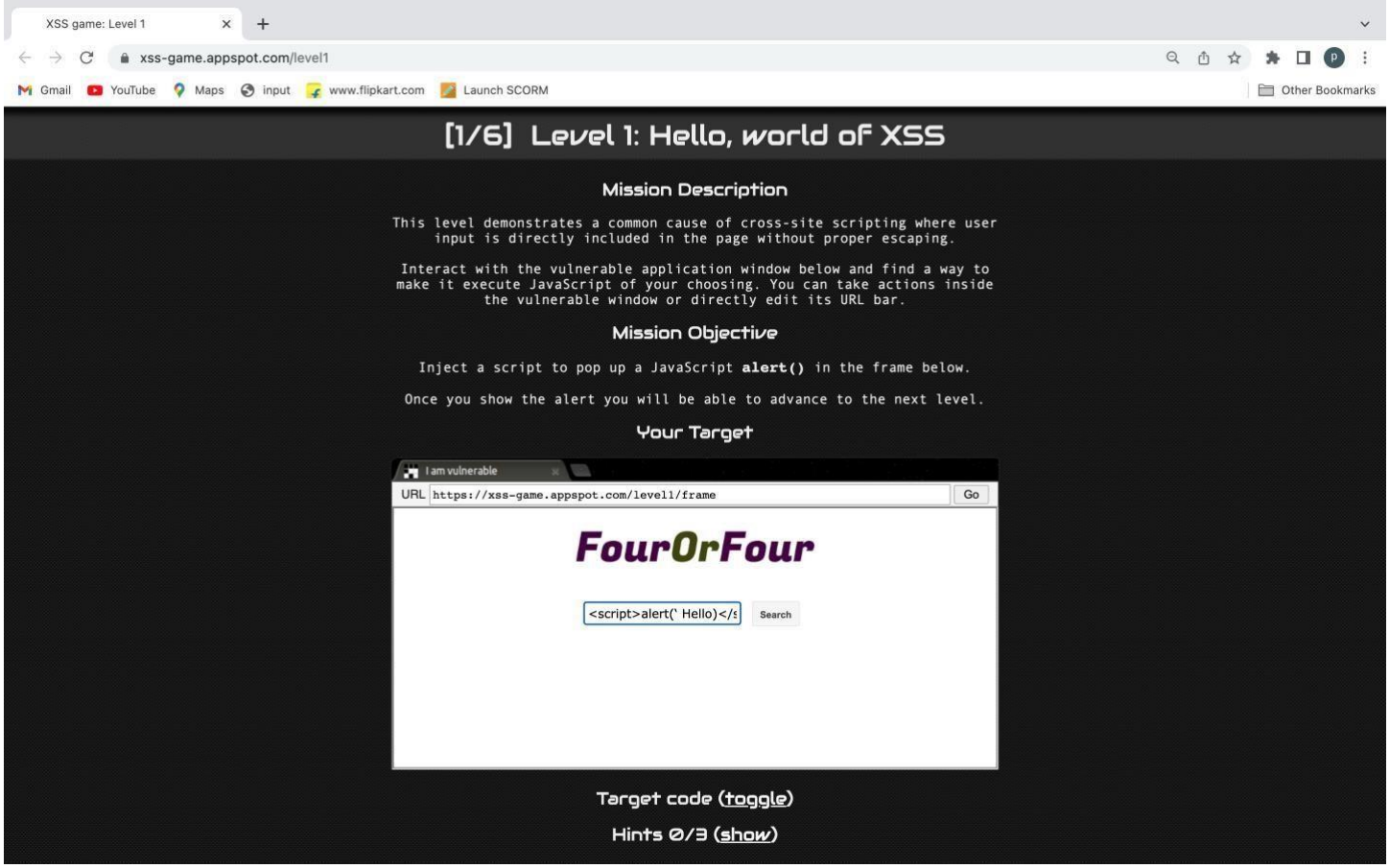

128.198.49.198:8102/mutillida
Not Secure | 128.198.49.198:8102/mutillidae/index.php?page=add-to-your-blog.php
Gmail YouTube Maps input www.flipkart.com Launch SCORM
Other Bookmarks
Getting Started
34 Current Blog Entries
Name
Date
Comment
1 anonymous 2022-09-04 14:58:47 CU blog html attack
2 anonymous 2022-09-04 13:14:38 AMAN_20BCS2200
3 anonymous 2022-09-04 13:12:21 ANSH - Html injected!!
4 anonymous 2022-09-04 13:09:02 AMAN UID:20BCS2200
5 anonymous 2022-09-04 13:07:45 HEloo
6 anonymous 2022-09-04 13:05:28 AMAN20bcs2200
7 anonymous 2022-09-04 13:02:35 20BCS9944!! Html injected!
8 anonymous 2022-09-04 13:00:29 Hello Ansh 20BCS9944!! Html injection performed!
9 anonymous 2022-09-04 13:00:09 Hello Ansh 20BCS9944!! Html injection performed!
10 anonymous 2022-09-04 12:58:26 Hello Ansh 20BCS9944!! Html injection performed!
11 anonymous 2022-09-04 12:56:01 RISHIKA 20BCS3076 !! html attack
12 anonymous 2022-09-04 12:50:37 20BCS3076 html attack
13 anonymous 2022-09-04 12:35:26 ROHIT KUMAR 20BCS2251
14 anonymous 2022-09-04 12:26:47 CU blog attack
15 anonymous 2022-09-04 12:25:01 CU blog attack
16 anonymous 2022-09-04 12:21:20 Rohit Kumar 20BCS2251
17 anonymous 2022-09-04 11:46:11 Hello Tesh 20BCS9937!! HTML injection per
18 anonymous 2022-09-04 11:40:20 Hello Tesh 20BCS9937!! HTML injectio
19 anonymous 2022-09-04 11:39:45 Hello Tesh 20BCS9937!!
20 anonymous 2022-09-04 11:39:31 Hello Tesh 20BCS9937!!
21 anonymous 2022-09-04 09:54:31 Injecting HTML Hello 20BCS3375
22 anonymous 2022-09-04 09:51:56 CU blog attack
23 anonymous 2022-09-04 09:42:33 Hello 20BCS3057!!
24 anonymous 2022-09-04 09:30:24 This is HTML injection!
25 anonymous 2022-09-04 09:29:11 Hello 20BCS305
26 anonymous 2022-09-04 08:52:29 HTML injection UID: 20BCS2844 html attack



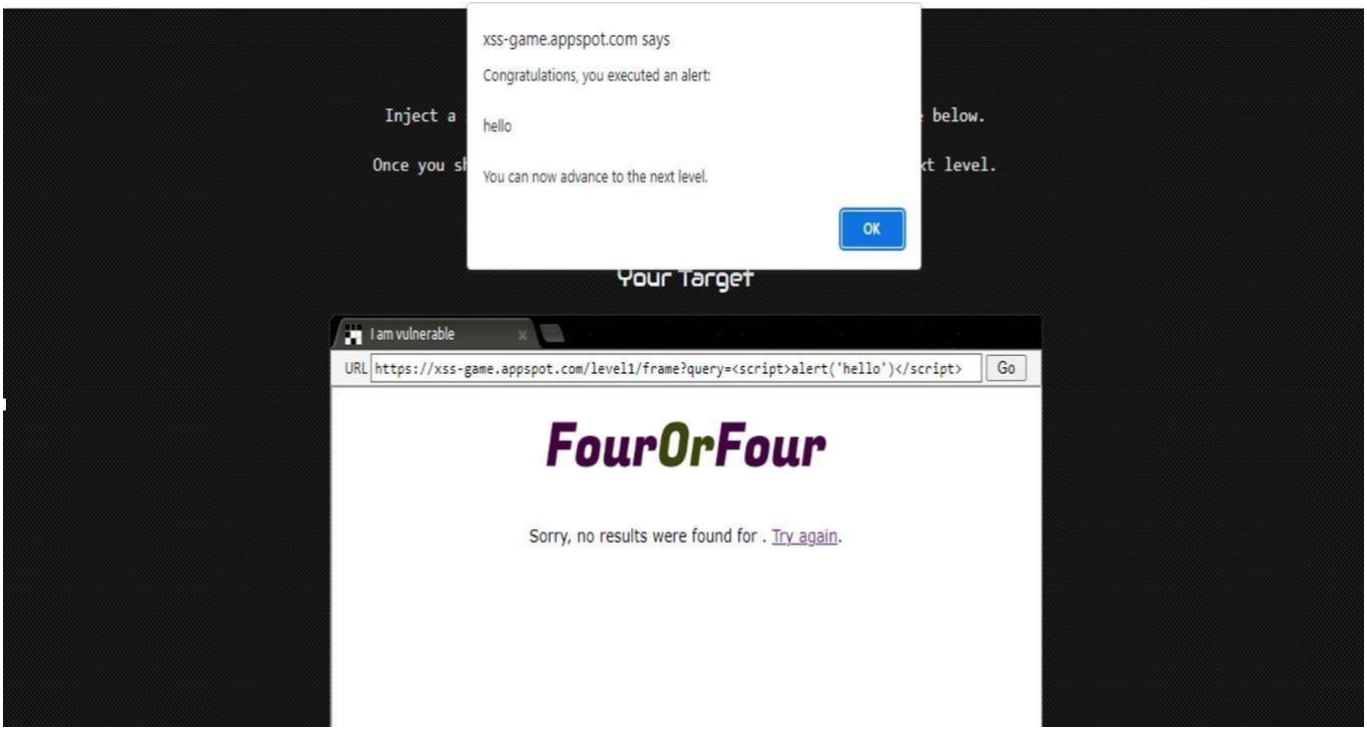
1. Open the link https://xss-game.appspot.com/level1 (or Google XSS game website)

2. If the search field is vulnerable, when the user enters any script, then it will be executed. Consider, a user enters a very simple script as shown below:

<script>alert(‘ Hello)</script>



3. Then after clicking on the **“Search”** button, the entered script will be executed. The script typed into the search field gets executed. This just shows the vulnerability of the XSS attack.



Learning outcome:

We have learned what HTML injection is and XSS injection .An overview of how these attacks are constructed and applied to real system. If the app or website lacks proper data sanitization, the malicious link executes the attacker's chosen code on the user's system. As a result, the attacker can steal the user's active session cookie and can be the harmful for the website.

Evaluation Grid (To be created as per the SOP and Assessment guidelines by the faculty):

Sr. No.	Parameters	Marks Obtained	Maximum Marks
1			
2			
3			
4			