HTML INJECTION

* An HTML code is injected into a web app.
* Rendered by the Browser => Client side attack.
* Exploits flaws in the application's handling of user Input for web pages.
* Aims to modify appearance of a web page.
* Tricking user to interact with fake contents.

* User provides input to the app.
* If the input is not sanitized, then malacious input can get injected into the page
* The malacious input is in the form of HTML code which is rendered by the Browser

Example

```
$ cat htmli.php

<html>

<body>
      <span>
      Welcome
      <?php

         echo $_REQUEST['nickname'];

      ?>
      </span>
</body>

</html>
```

Post — used when submitting form data
 core HTTP method should be specified in form
Get — when user enters a URL or clicks on a link, browser generates a GET request.

or default Get is used.

span keep the data inline.

Goto /var/www/html/
create a dir there htmlinjection
$ cd htmlinjection
$ place the file htmli.php here

Now open a new tab
Get your kali ip.

start apache2 service

```
$ systemctl status apache2.service
$ systemctl start apache2.service
```

Now open firefox browser in kali
Go to URL bar & type

<ip of kali>/htmlinjection/htmli.php

You will see

Welcome !

Now on URL bar pass input

i.e., kali_ip/htmlinjection/htmli.php?nickname=sbee

or Raaf Buri
or <b>sbee</b>

You'll see

Welcome sbee !

So Any thing you type in here, it's reflected back.

* If you look at htmli.php, you'll notice

```
echo $_REQUEST['nickname'];
```

↑ super global array that collects data from multiple sources like
- form submission
- URL parameters
- cookies.

we are echoing user input.

Now because this user input is not being senalized, I can actually add any melacious code

eg ... ?nickname=<b>rbee</b>

Rt. click & look at the source code of the page.
See the echo line is replaced by user input.

The melacious user can also do

~~...?nickname=click here <a href="http://google.com"~~>

...?nickname=<a href="google.com">click Here </a>

https://

Note:- Write above injection an URL parameter. It works.

* If you try to write it on Burp it fails.
* Set interception on Burp.
* Write this on URL
* On Burp look at the intercepted request
  It is encoded
  What you can do is type fccollege.edu.pk in place of google.com

To Encode.

" →	%22
< →	%3C
> →	%3E
space →	%20
& →	%26
~~→~~	~~%3D~~

So
?nickname=<a href="http://google.com">click Here</a>
becomes
?nickname=%3Ca%20href=%22http://google.com%22%3Eclick%20Here%3C/a%3E

Example

/var/www/html/infection/htmltags

Explaini

```
<html>
<body>
    <b> Welcome! Please Login </b>

    <br>

    <form action='/'>
        <input type="hidden"
         name="sessionid"
         value="<?php echo
                $_REQUEST['sid'];
                ?>">

        Username: <input type="text"
                   name="username">
                   <br>

        Password: <input type="password"
                   name="pass"><br>

        <input type="submit"
         value="Login">

    </form>
</body>
</html>
```

Here since method is not explicitly specified ⇒ default method GET is used & form data is appended to the URL in the form of query param

output :-

```
Welcome! Please Login
Username: [          ]
Password: [          ]
```

Explaining the code

<form> tag defines start of a form in HTML. Form is something where you can ~~find~~ add fields like text boxes, labels, radio btns, buttons etc.

action='/' means when the form is submitted, the data will be sent to the root of the server. The server should then have an end point that listens for the form submission at this URL.
If no action is specified, the form will submit to the current page by default.

<input> tag is used to create various types of input fields.

eg <input type="text" name="username">

Here common attributes of input tag are type, name, id, value, ...
type can be text, password, radio, submit, ...

In the code,
we have username text box
password text box
submit button
& a hidden field called sessionid
its value is based on user input

On URL bar type
<kali ip>/htmlinjection/htmltags1.php

you'll see the page

* Now type
-- htmltags1.php?sid=rauf

nothing appears on page
Goto view page source
Here in the code, you'll see "sid" is
replaced by "rauf"

* Next type a " at the end of URL etc.
View page source, you'll see a " at
the end of "rauf"

* Try placing different symbols like ">
or <!-- etc & see what happens

* It / [illegible]

* As an attacker you can change the view ~~the~~ of the page
eg.
go to view source & try to create a third text box with ATM pin label.

Copy the code in a text editor & make changes as follows. in a single line

```
"></form><form action='/'><input type=
"hidden" name="sessionid" value="some">
Username: <input type="text" name=
"username"><br>Password: <input type=
"password" name="pass"><br>ATM pin:
<input type="password" name="atmpin">
<br><input type="submit" value="Login">
</form><!--
```

Let us try it on DVWA

Open

* DVWA is a deliberately vulnerable web app.
* Used to practice web app security testing in a safe environment.
* Open source
* Adjustable security levels
* Burp Suite, OWASP ZAP can be tested & practiced on DVWA.
* Common Vulnerabilities in DVWA
  - HTML Inj
  - SQL Inj
  - X-SS
  - CSRF
  - Command Inj
  - File Inclusion

Open M2 & kali

- Goto firefox on kali
- Type ip addr of M2 on URL bar
- Click on DVWA

  username : admin
  password : password.

- Goto XSS Reflected

  Type on text Box
  Rauf
  o/p : Hello Rauf

Type <h1>Rauf</h1>

- o/p Hello
  Rauf as in Heading => Vuln.

Type
<a href="http://google.com">click Me</a>

o/p Hello clickMe
when you click on it you will be
redirected to google.com

Visit the View Source Btn

For Low security

echo 'Hello' . $_GET ['name']

is used
This is a superglobal (built-in global array) in php
It directly retrieves & processes data without
any validation. If the data is a valid HTML
code, it simply is rendered.

For Medium Security

echo 'Hello' . str_replace('<script>', '',
$_GET['name']);

This line replaces <script> by an empty
space. But still uses $_GET[]
⇒ vulnerable

For high Security

There htmlspecialchars() fn is used.
This fn ensures that chars like <, >, &, "
are treated as plain text instead of
being Interpreted as HTML or Java Script
by the browser.