

2.1: Command and SQL injection	1
Simple	1
2. os-command-injection (2)	3
Blind-time-delays	3
3. os-command-injection (3)	4
4. os-command-injection (4)	5
5. sql-injection (1)	5
6. sql-injection (2)	6
7. sql-injection/union-attacks (1)	7
determine-number-of-columns	7
8. sql-injection/union-attacks (2)	8
9. sql-injection/union-attacks (3)	8
10. sql-injection/examining-the-database (1)	9
11. sql-injection/examining-the-database (2)	10

2.1: Command and SQL injection

[Simple](#)

- Take a screenshot for your lab notebook of the output

```
7 s = requests.Session()
8
9 site = 'https://acb21f131f030067c029524400190032.web-security-academy.net'
10 stock_post_url=f'{site}/product/stock'
11 post_data = {
12     'productId' : '1',
13     'storeId' : '1 | date'
14 }
15 resp = s.post(stock_post_url, data=post_data)
16 print(resp.text)
```

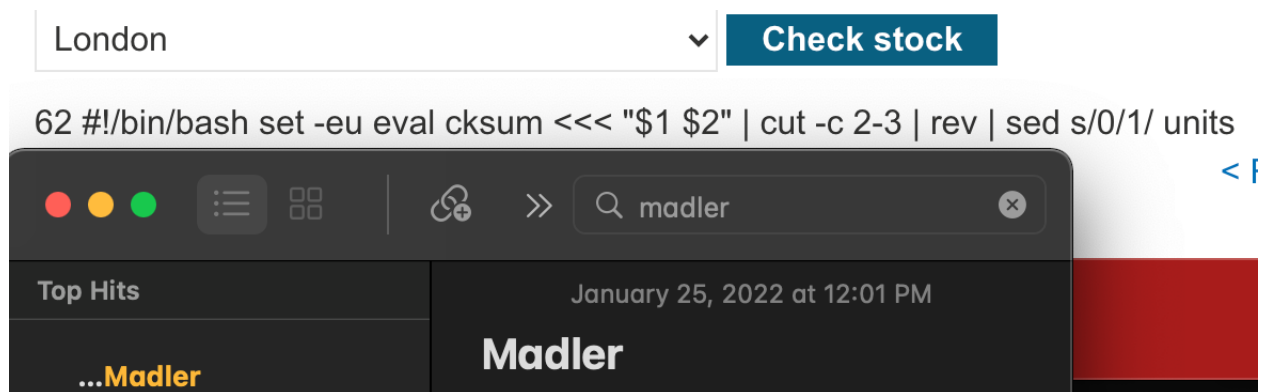
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL 1: b

```
(base) Matthews-Air:labs2 matthewadler$ python 1.py
Thu Jan 27 19:15:52 UTC 2022

(base) Matthews-Air:labs2 matthewadler$
```

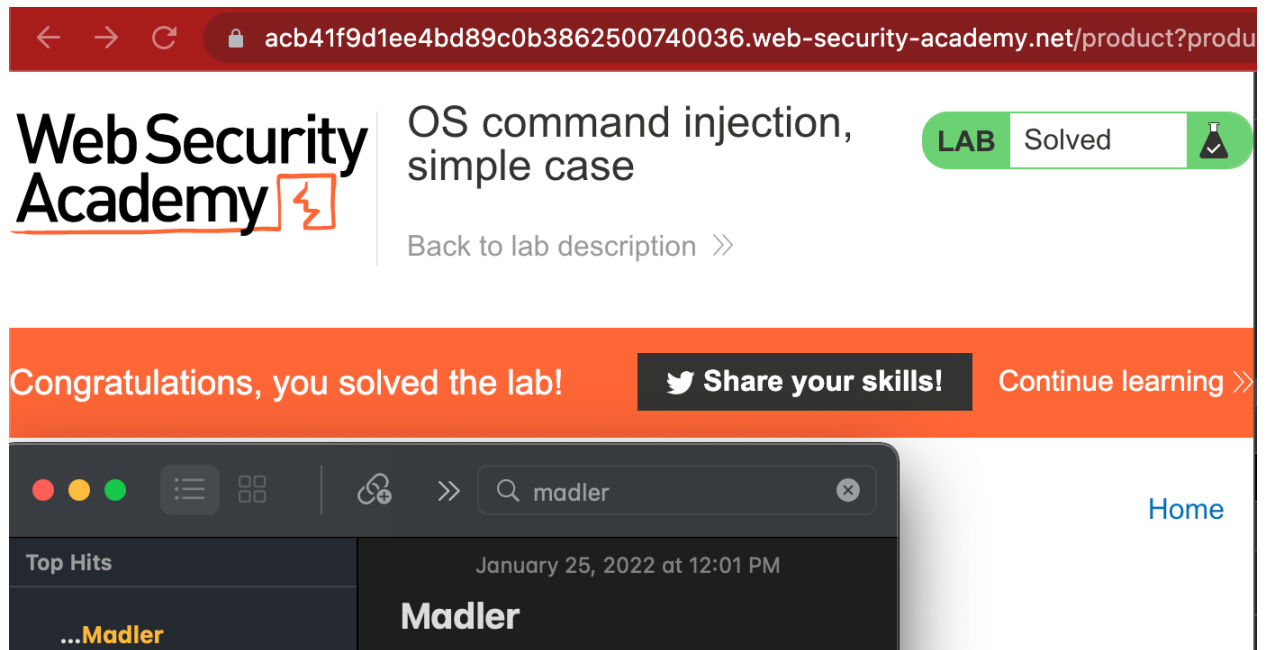
Edit the option value again to instead run the `cat` command in order to return the contents of the shell script on the web page similar to that shown below

- Take a screenshot for your lab notebook of the output



Finally, finish the level by running the command that identifies the [username](#) of the account running the web site.

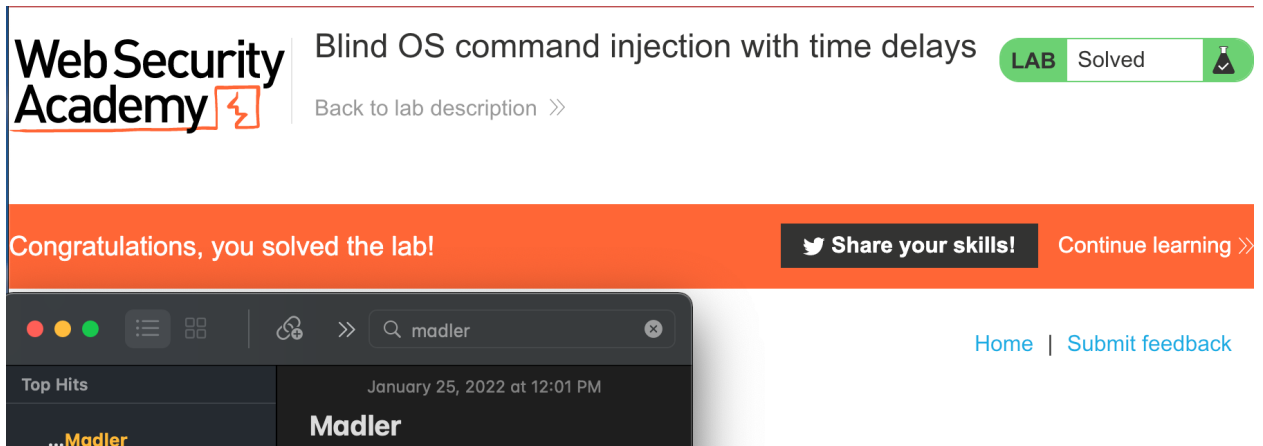
- Take a screenshot showing completion of the level that includes your OdinId



2. os-command-injection (2)

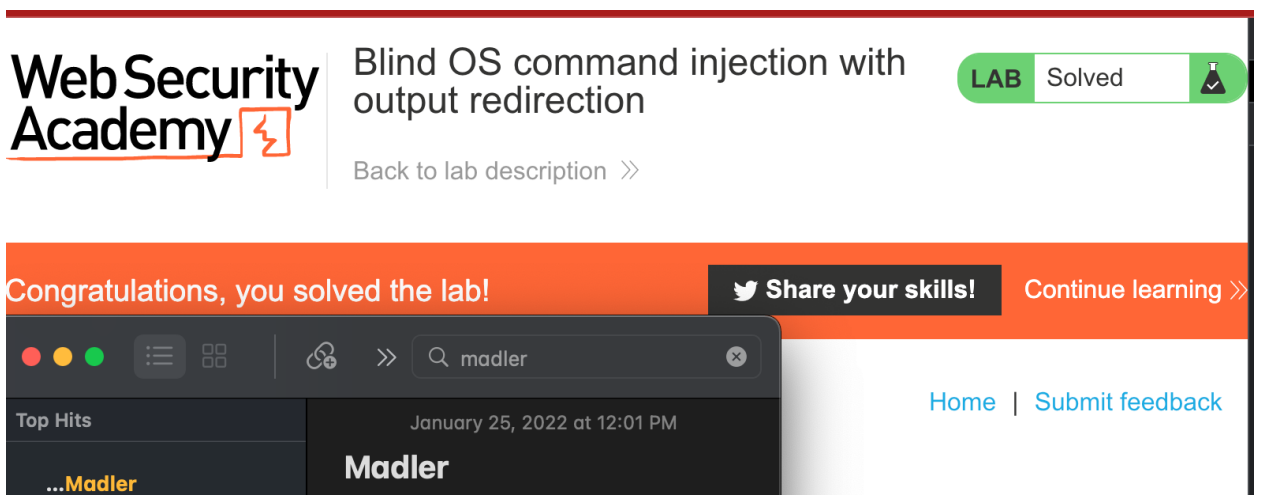
[Blind-time-delays](#)

Web applications may be written in a way that the results of a command are not directly returned. 'Blind' attacks allow an adversary to learn whether a web site contains a command injection vulnerability via other means. One way to do so is to inject a command that results in a delay in the processing of the request. For example, if one is able to successfully inject a 'sleep 5', the web application will pause for 5 seconds before returning a response, thus revealing that the application contains the vulnerability.



3. os-command-injection (3)

- Take a screenshot showing completion of the level that includes your OdinId



- Take an additional screenshot that shows the username that runs the site

```
(env) (base) Matthews-Air:labs2 matthewadler$ python 3.py  
{  
peter-4VCP5b  
(env) (base) Matthews-Air:labs2 matthewadler$
```

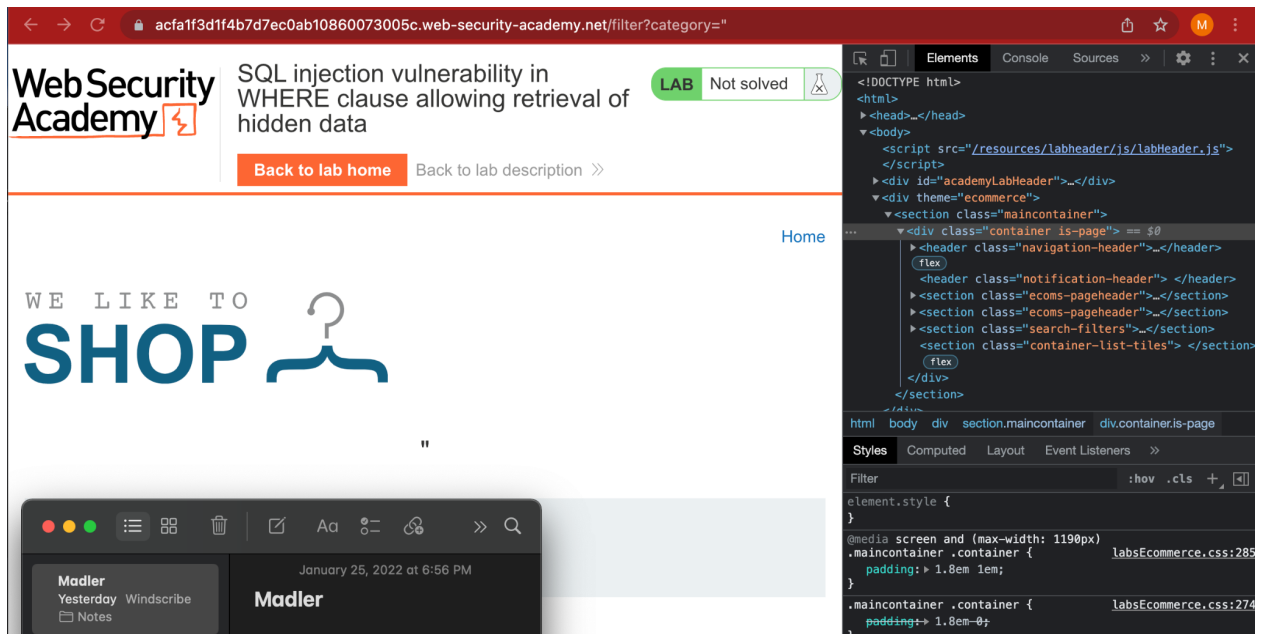
4. os-command-injection (4)

The screenshot shows a web browser window with the address bar displaying the URL `ac1c1f321e92cabbc0cb0f60005400f5.web-security-academy.net/feedback`. The page features the Web Security Academy logo on the left and the title "Blind OS command injection with out-of-band interaction" in the center. To the right of the title is a green "LAB Solved" badge with a checkmark icon. Below the title is a link that says "Back to lab description >>". An orange banner at the top of the page reads "Congratulations, you solved the lab!" and includes a "Share your skills!" button with a Twitter icon and a "Continue learning >>" link. At the bottom, there is a dark-themed sidebar with a "Madler" profile card showing "Yesterday" and "Windscribe" notes, and a main content area with the name "Madler" and the date "January 25, 2022 at 6:56 PM". Navigation links for "Home" and "Submit feedback" are visible in the bottom right corner.

5. sql-injection (1)

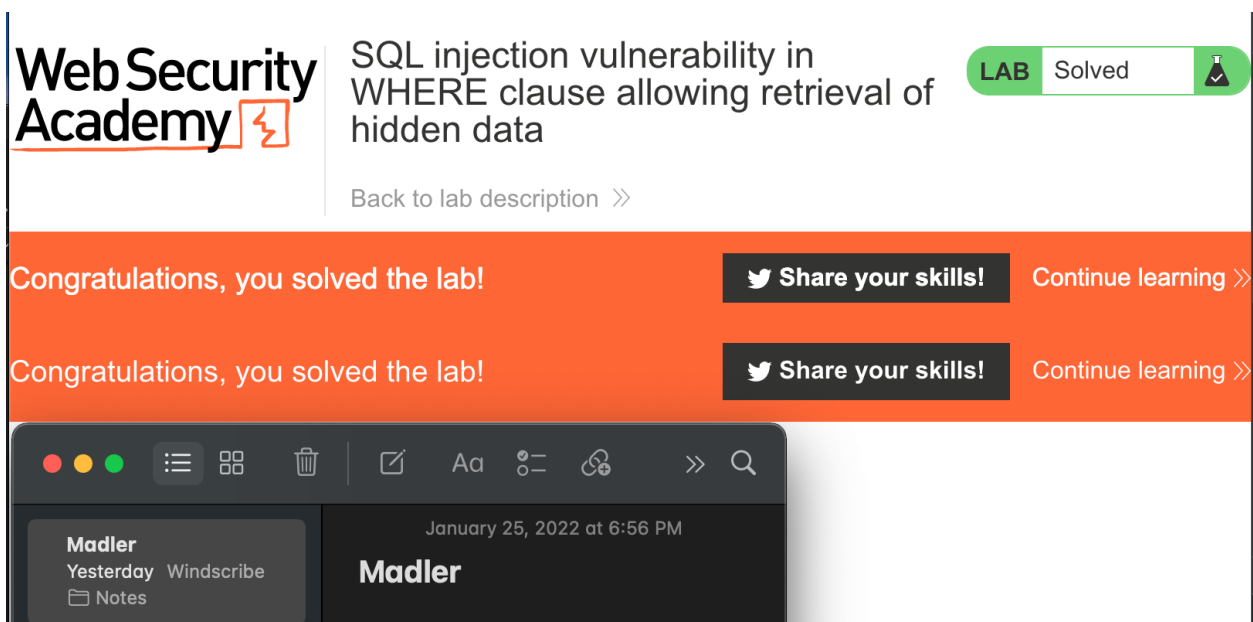
Go back to the web site and select the `Gifts` category. In the URL, replace `Gifts` with a double-quote character and load the resulting page.

- Take a screenshot of the output.



Then, replace the category string with a single-quote character and load the resulting page.

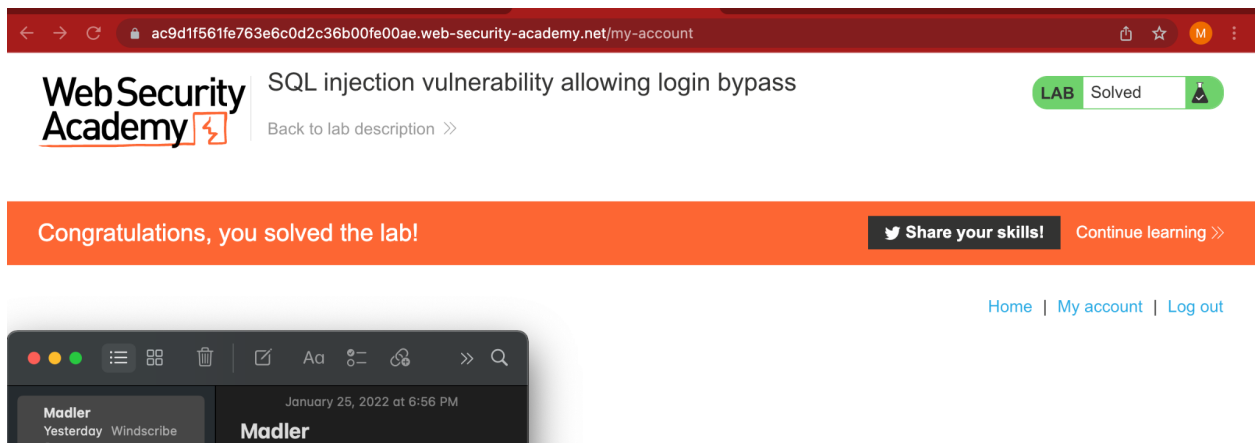
- What output is returned? **Internal Service Error**
- Take a screenshot showing completion of the level that includes your OdinId



6. sql-injection (2)

- Is the username field vulnerable to SQL injection? If so, what character breaks syntax? **Yes, single quote.**
- Is the password field vulnerable to SQL injection? If so, what character breaks syntax? **Yes, -.**

- Take a screenshot showing completion of the level that includes your OdinId



7. sql-injection/union-attacks (1)

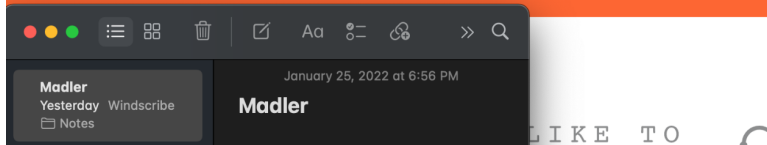
determine-number-of-columns

- How many columns does the products table contain? **3**
- Take a screenshot showing completion of the level that includes your OdinId
-

Congratulations, you solved the lab!

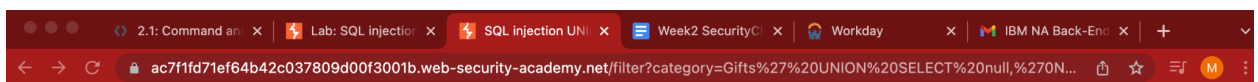
[Share your skills!](#)

[Continue learning >>](#)



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

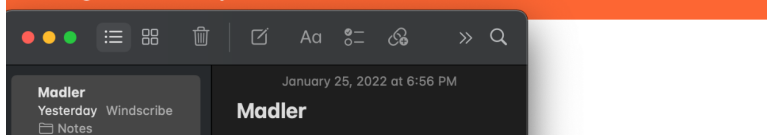
8. sql-injection/union-attacks (2)



Congratulations, you solved the lab!

[Share your skills!](#)

[Continue learning >>](#)



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

9. sql-injection/union-attacks (3)

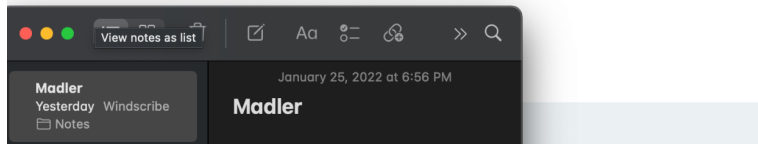
WebSecurity Academy SQL injection UNION attack, retrieving data from other tables LAB Solved

Back to lab description >>

Congratulations, you solved the lab!

Share your skills! Continue learning >>

Home | My account | Log out



10. sql-injection/examining-the-database (1)

WebSecurity Academy SQL injection attack, querying the database type and version on MySQL and Microsoft LAB Solved

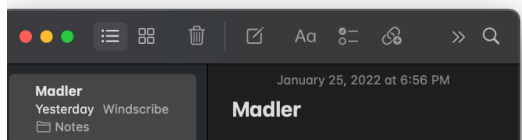
Back to lab description >>

Congratulations, you solved the lab!

Share your skills! Continue learning >>

Congratulations, you solved the lab!

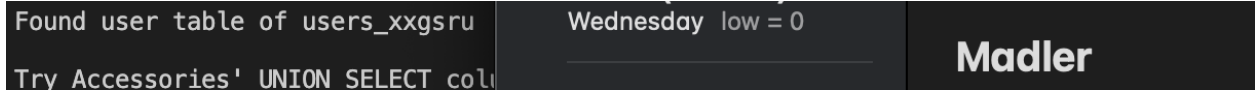
Share your skills! Continue learning >>

A screenshot of a Notepad application window, similar to the one above. It shows a note titled 'Madler' dated 'January 25, 2022 at 6:56 PM'. The interface includes a title bar, menu bar, and a sidebar with a list of notes.

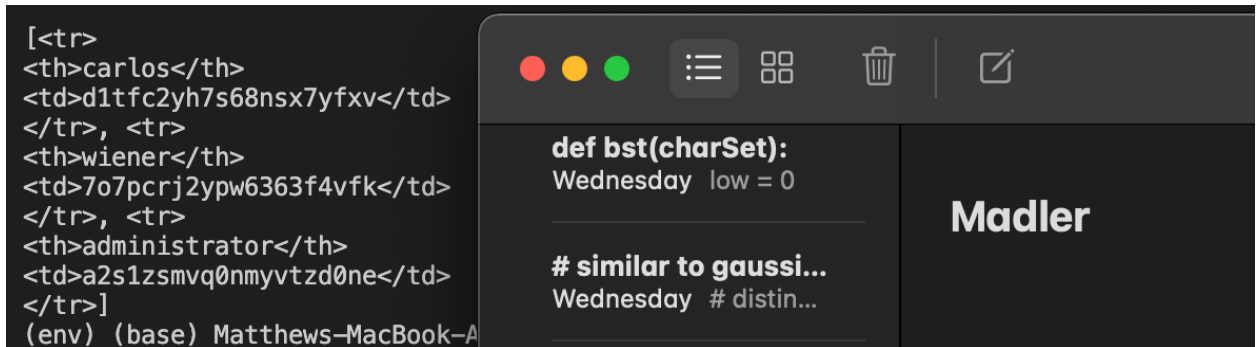
<https://ac911f231ef74013c0d45cba00870007.web-security-academy.net/filter?category=Accessories' UNION SELECT @@version, NULL->

11. sql-injection/examining-the-database (2)

- Take a screenshot of the results showing the name of the user table



- Take a screenshot of the results showing the column names of the user table
- Take a screenshot of the results showing all of the users and their passwords for the site.
-



- Take a screenshot showing completion of the level that includes your OdinId

