**CVE-2023-23488**

- Plugin : Paid Memberships Pro

- Phiên bản lỗi: <2.9.8

- Lỗi : Unauthenticated Blind time-based SQLi

a) Mô tả

- Paid Memberships Pro (PMPro) là một plugin WordPress được sử dụng để tạo và quản lý các khách hàng trả phí trên trang web của bạn. Plugin này cung cấp một loạt các tính năng và chức năng để tạo và quản lý chương trình thành viên trả phí trên trang web của bạn.

b) Hàm xử lý bị lỗi

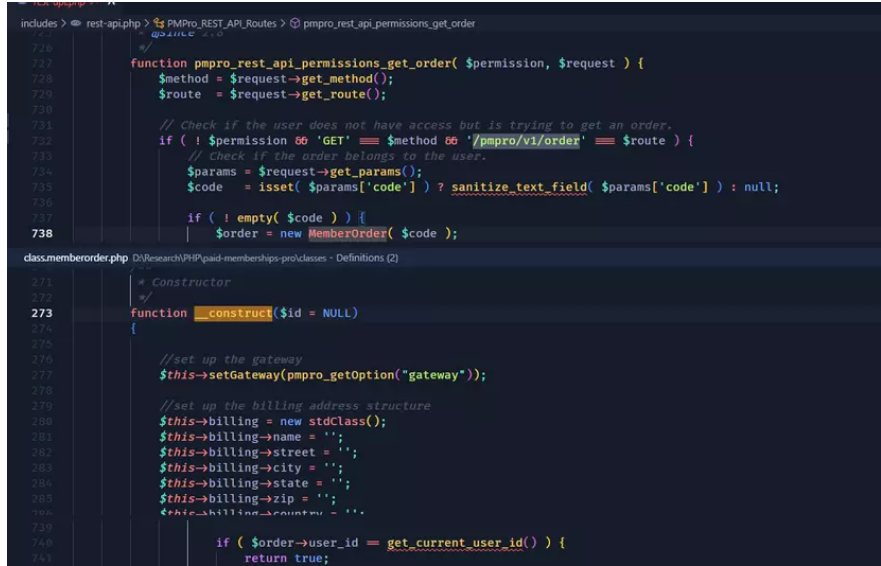
- Sử dụng REST API của Wordpress được đăng ký theo router



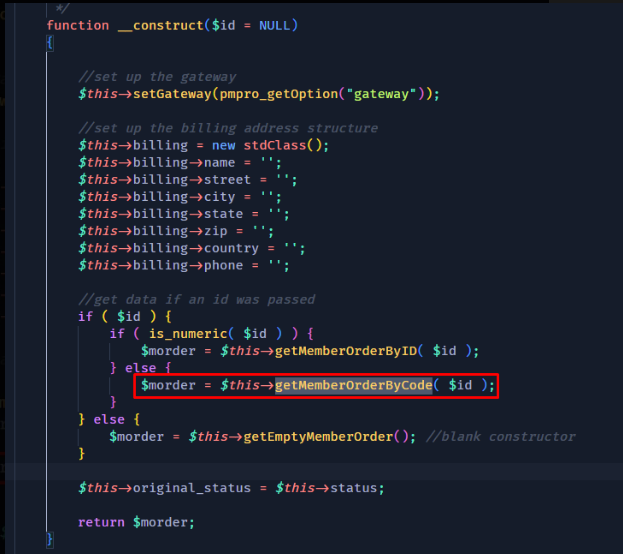
- nhảy vào function pmpro\_rest\_api\_permissions\_get\_order() với

GET /?rest\_route=/pmpro/v1/order HTTP/1.1

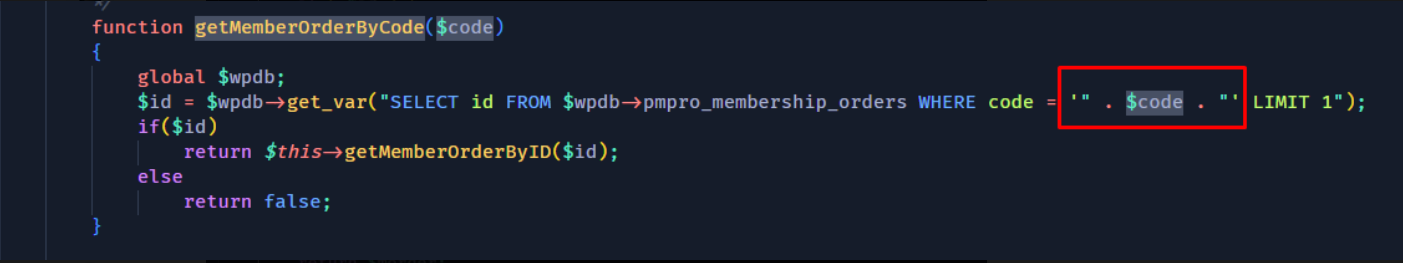
- Sau đó truyền thêm tham số code vào khởi tạo MemberOrder, nhảy trực tiếp đến \_\_construct



- Ở đây ta truyền giá trị code không phải là số sẽ nhảy sang function  getMemberOrderByCode($id)

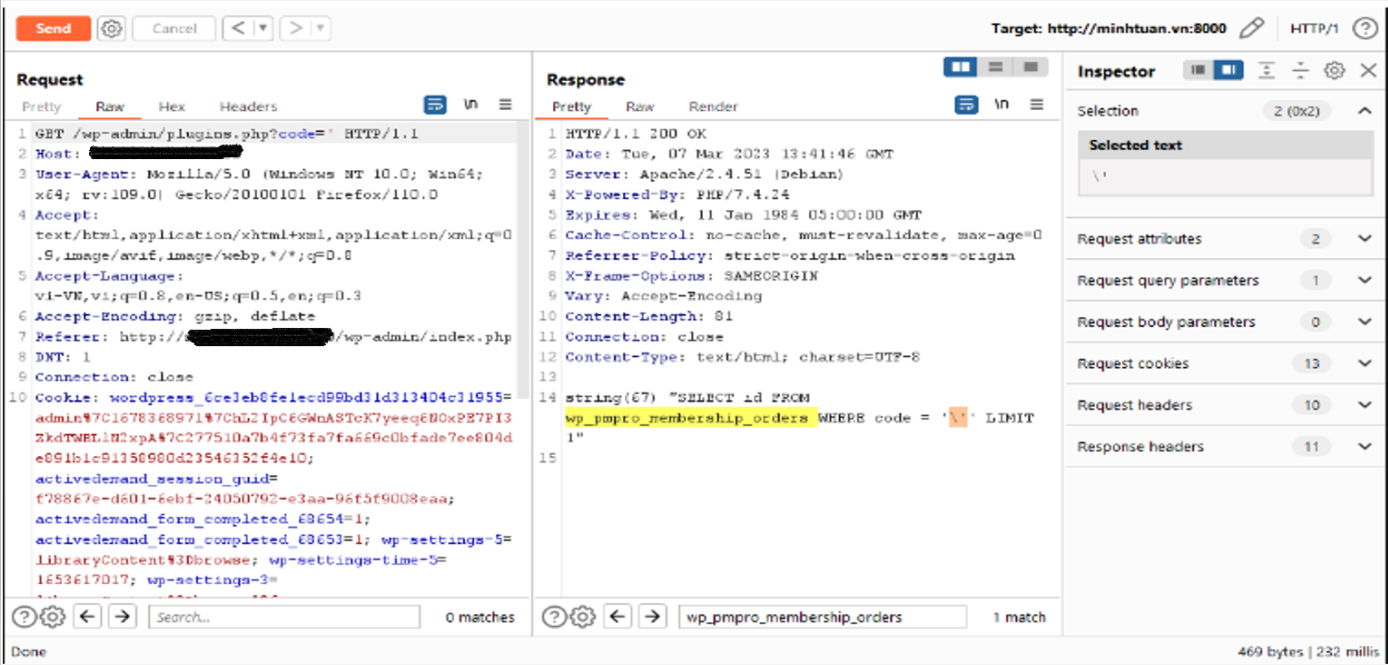


- Tại đây giá trị code được truyền vào hoàn toàn không được filter hay escape gì, tại đây chúng ta có thể inject SQL vào đây theo kiểu TimeBased



c) Khai thác

- Bắt yêu cầu từ burp

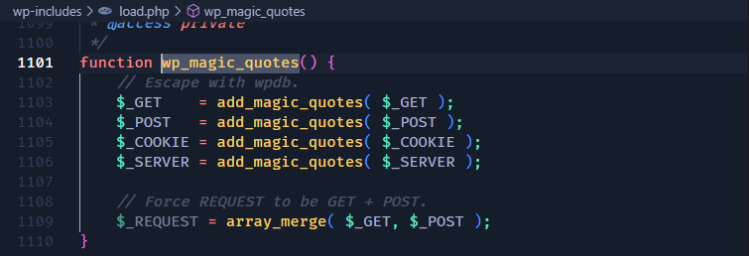


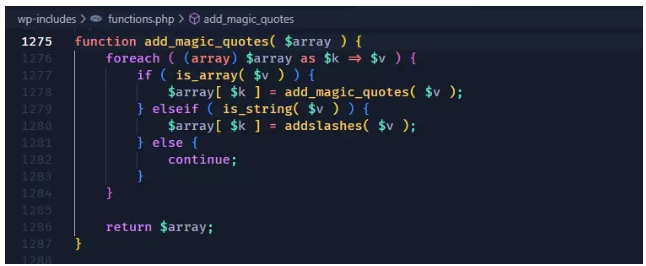
- Như đã thấy rằng giá trị $\_GET['code'] đã tự động thêm dấu \ từ đây ta không thể nào break ra khỏi dấu nháy đơn để thoát khỏi string

NOTE: WordPress sẽ tự động xử lý và áp dụng addslashes() cho các giá trị này để đảm bảo rằng chúng an toàn khi được sử dụng trong các câu lệnh SQL hoặc các tình huống khác. Hàm được gọi lên ngay từ wp-setting.php



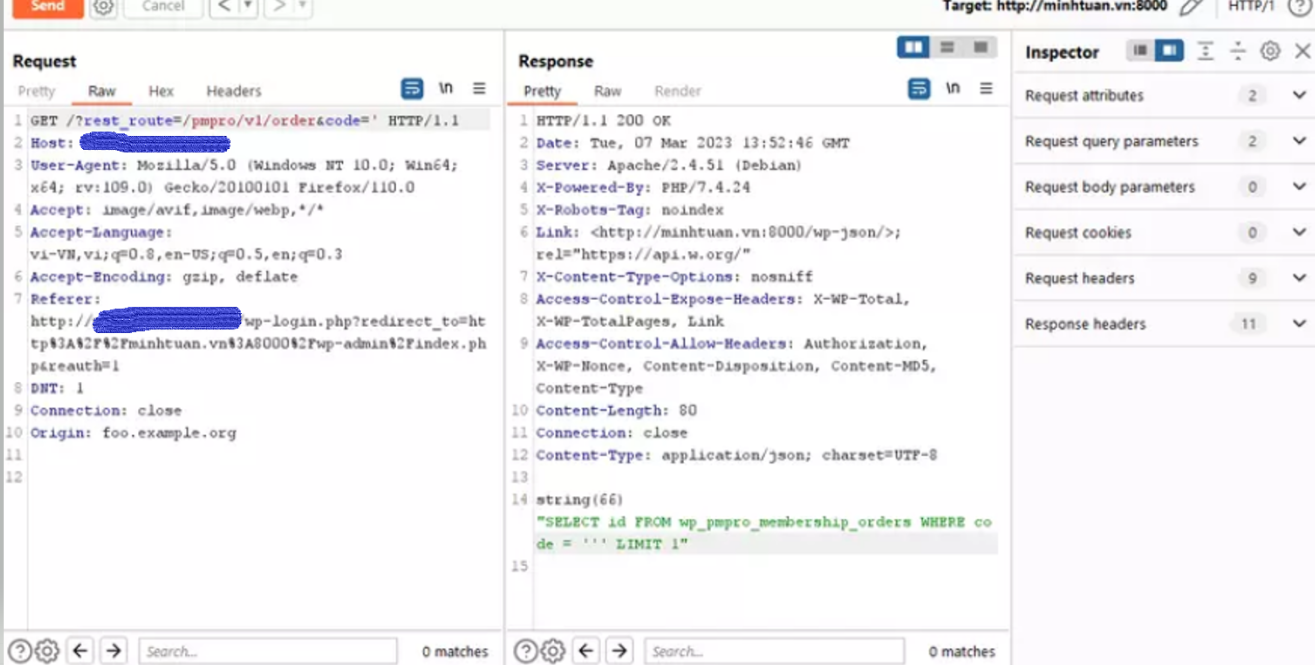
- Với ghi chú rõ ràng rằng sử dụng để escape cho wpdb



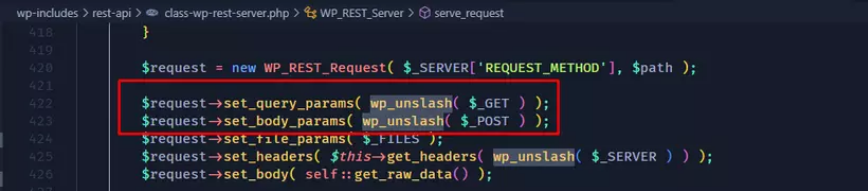


- với **mọi** $\_GET, $\_POST, $\_COOKIE, $\_SERVER khi truyền từ input người dùng thì đều sẽ bị thêm slash vào.

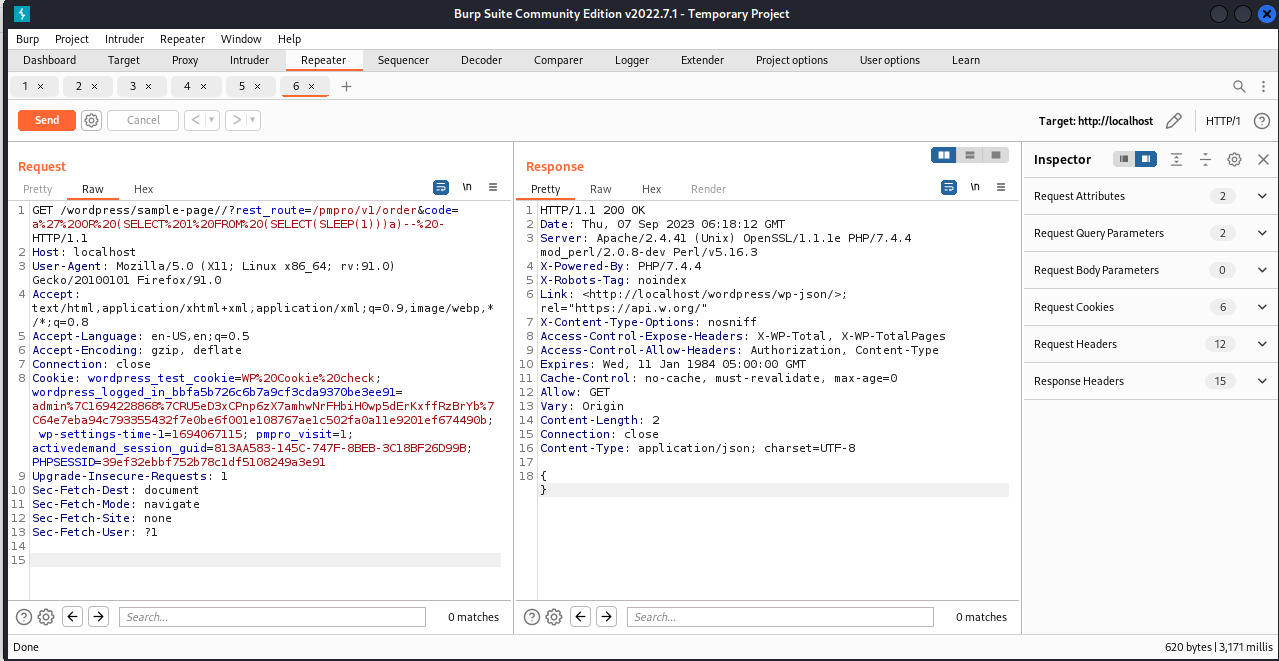
- Tuy nhiên, tại sao khi sử dụng REST API thì Wordpress lại xoá mất dấu slash đi. Chúng ta có thể break ra khỏi dấu nháy đơn, inject câu lệnh SQL tuỳ ý.



- Đoạn xử lý xóa dấu \

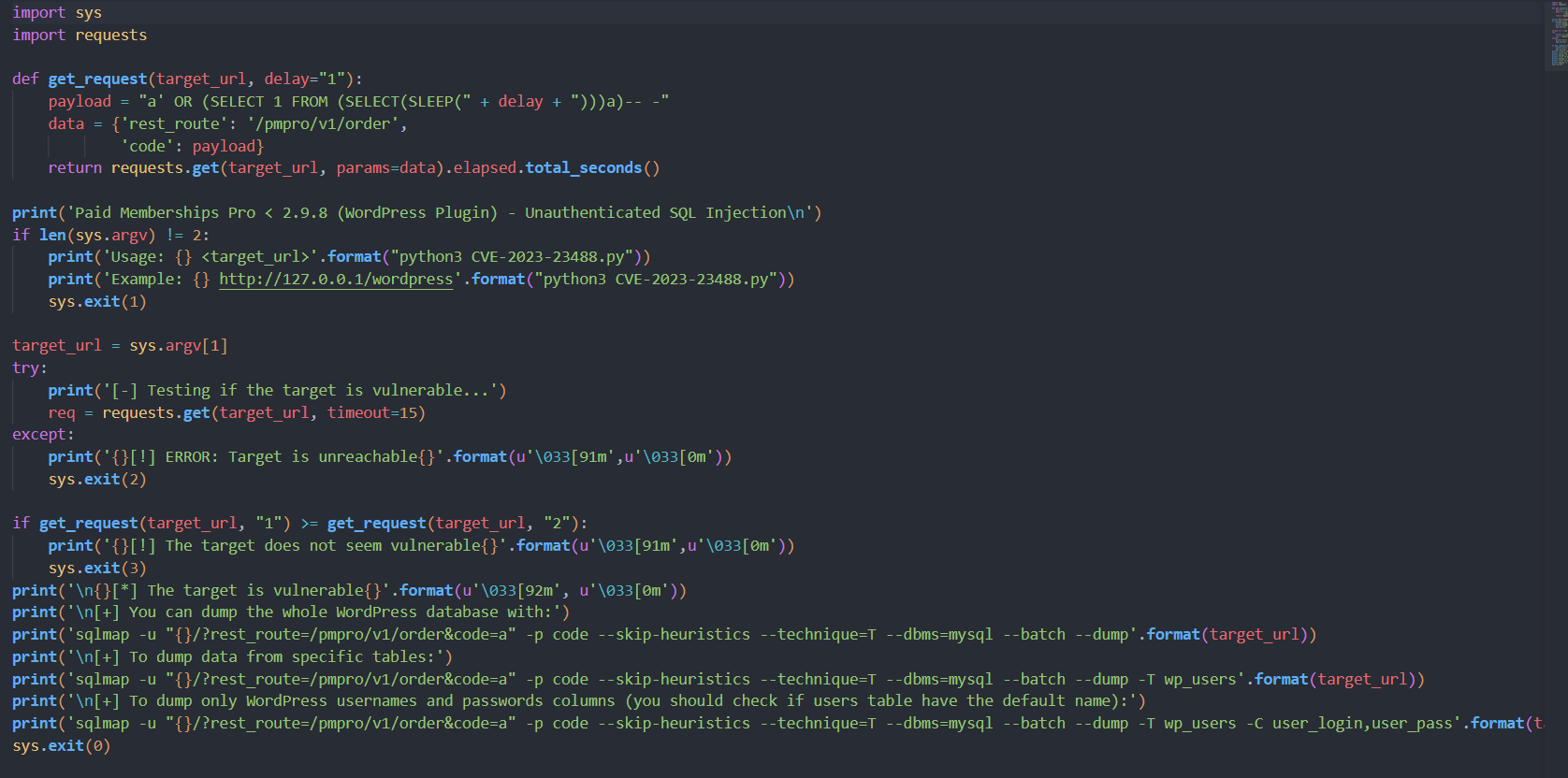


- Từ đó ta chèn payload timebased vào thấy thời gian phản hồi lớn hơn 1s



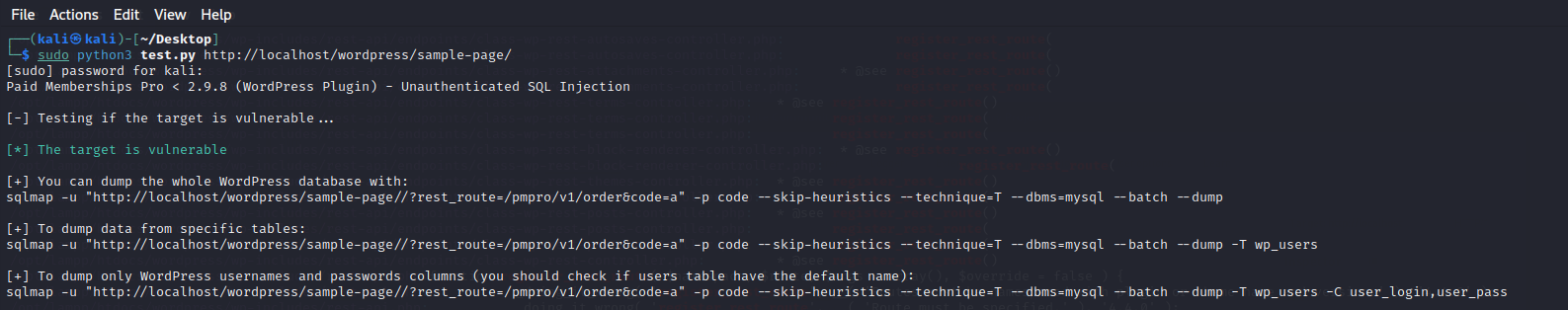
d) Khai thác bằng script

Mã khai thác



1. Đầu tiên, mã này sử dụng thư viện Python **requests** để thực hiện các yêu cầu HTTP đến trang web mục tiêu.
2. Hàm **get\_request(target\_url, delay="1")** được định nghĩa để tạo và gửi yêu cầu HTTP để tận dụng lỗ hổng SQL Injection. Biến **target\_url** là URL của trang web mục tiêu và **delay** là thời gian trễ mà bạn muốn chờ đợi trước khi nhận được phản hồi từ trang web.
3. Trong hàm **get\_request**, một chuỗi **payload** được tạo ra để thực hiện lỗ hổng SQL Injection. Payload này sẽ thực hiện một truy vấn SQL **SLEEP(delay)** để tạo ra thời gian trễ. Điều này được thực hiện thông qua tham số **code** trong yêu cầu HTTP.
4. Mã này cũng sử dụng **data** để xác định tham số **rest\_route** và **code** trong yêu cầu HTTP.
5. Đoạn mã kiểm tra xem trang web mục tiêu có bị lỗ hổng SQL Injection không bằng cách so sánh thời gian trễ của hai yêu cầu:
   * Yêu cầu đầu tiên gửi payload với **delay** là 1 giây.
   * Yêu cầu thứ hai gửi payload với **delay** là 2 giây. Nếu thời gian trễ của yêu cầu với **delay** 2 giây lớn hơn, điều này cho thấy có sự trễ trong việc thực thi truy vấn SQL, cho biết lỗ hổng SQL Injection.
6. Nếu lỗ hổng được tìm thấy, mã sẽ in ra thông báo cho biết trang web mục tiêu có lỗ hổng và cung cấp các hướng dẫn về cách sử dụng công cụ **sqlmap** để khai thác lỗ hổng này và truy cập dữ liệu cơ sở dữ liệu WordPress.

Chạy PoC



- Khi đó ta check được là phiên bản đang dùng <2.9.8 và có thể khai thác được lỗ hổng này

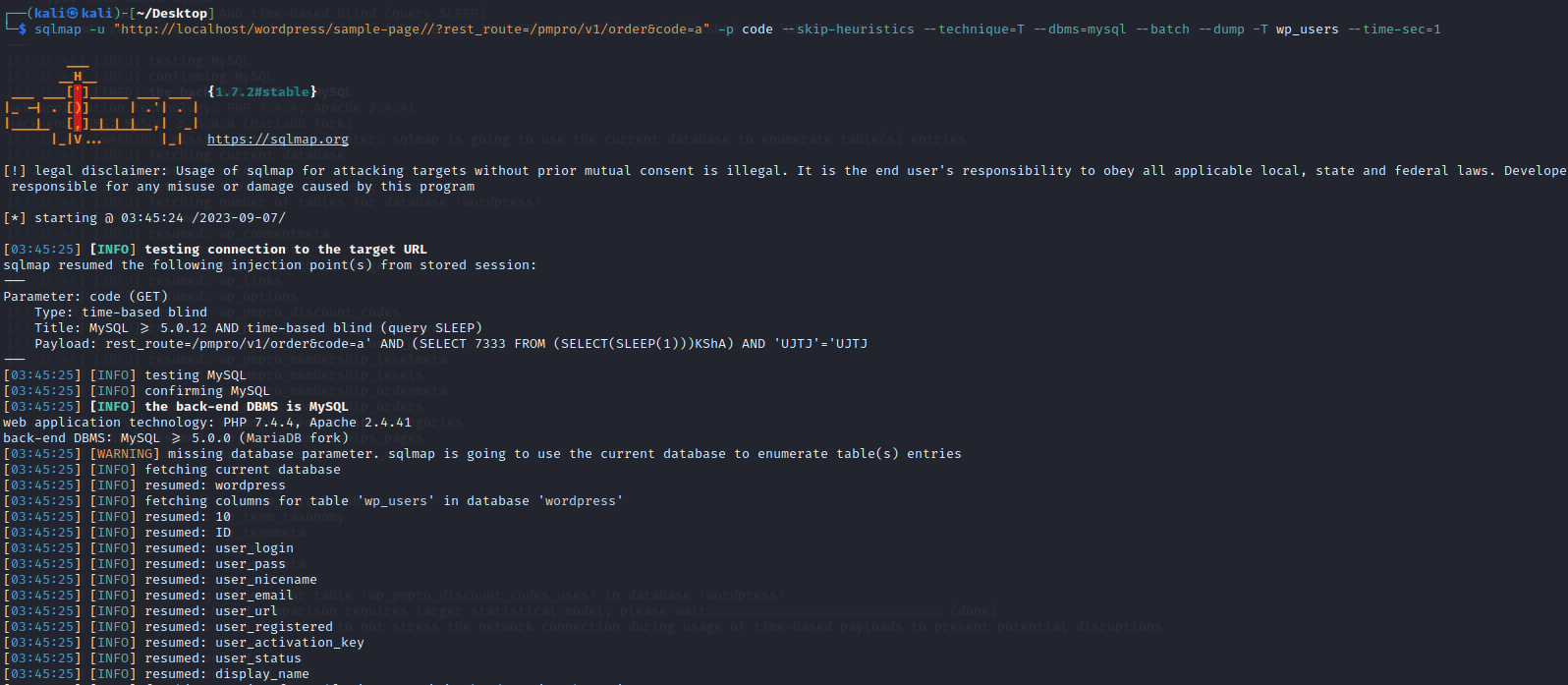
- Sau đó sẽ xuất ra các câu lệnh SQLMap để tiếp tục dump được db của chúng ta

B1: dump bảng





B2: dump cột trong bản wp\_user



B3: dump giá trị của các cột trong bản wp\_user

