Trường Đại Học Công Nghiệp TP.HCM

Khoa Công Nghệ Thông Tin

Tài liệu hướng dẫn sử dụng công nghệ mới:

Nghiên cứu các lỗ hổng bảo mật SQL Injection, Upload file

GVHD: ThS. Võ Ngọc Tấn Phước, ThS. Nguyễn Phúc Hưng

Chuyên ngành đào tạo: Hệ Thống Thông Tin

Lớp: DHHTTT15B

STT	HỌ VÀ TÊN	MSSV
1	Ngô Thanh Ngân	19477261
2	Nguyễn Hoàng Khang	19515421

TP HCM, ngày 18 tháng 10 năm 2022

Mục lục

CHƯƠNG I: UPLOAD FILE

I. Tổng quan

1. Giới thiệu

1.1. Khái niệm:

Lỗ hổng File Upload xảy ra khi máy chỉ web cho phép người dùng upload file lên hệ thống mà không xác thực đầy đủ những thứ như tên, loại, nội dung hoặc kích thước của chúng. Khi không có những hạn chế đối với file thì attacker có thể lợi dụng điều đó để thực thi các kịch bản tấn công, thậm chí là thực thi mã từ xa.

1.2. Web shell là gì?

Web sell là một tập lệnh độc hại cho phép attacker thực hiện các lệnh tùy ý trên máy chủ web từ xa chỉ bằng cách gửi các yêu cấp HTTP đến đúng điểm cuối. Nếu có thể upload web shell thành công, bạn có toàn quyền kiểm soát máy chủ. Một tập lệnh web sell thường chứa một backdoor, cho phép hacker truy cập từ xa và có thể điều khiển máy chủ truy cập internet bất cứu lúc nào.

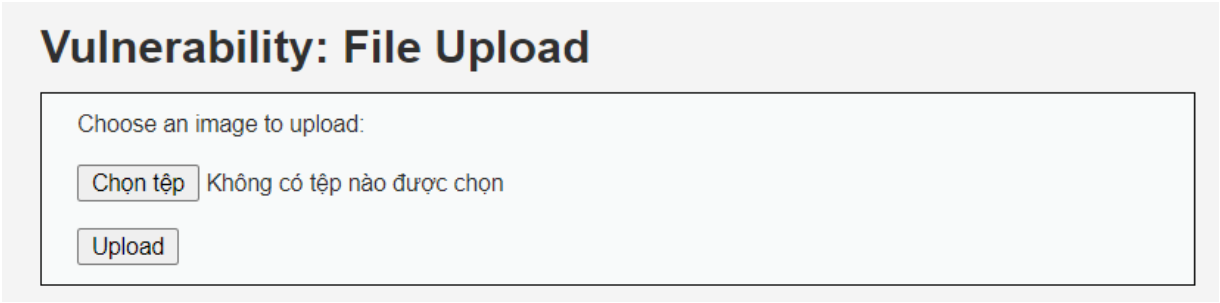
2. Khai thác lỗ hổng

2.1. Client-Side Filters

Client-Side Filters là một kiểu xác thực các yêu cầu khi được gửi lên máy chủ. Được sử dụng trên những trang web có sử dụng thuộc tính của JavaScript, VBScript hoặc HTML5. Các lập trình viên thường sử dụng kiểu xác nhận này để mang tới sự phản hồi nhanh hơn cho người dùng. Để bypass qua client-side filter, chúng ta có thể sử dụng các cách như:

- Tắt javascript của trình duyệt thông qua Developer Tools của Chrome, Firefox
- Giả mạo yêu cầu gửi lên (Proxify the application and tamper the request)
- Giả mạo dữ liệu bằng cách sử dụng firefox addon.

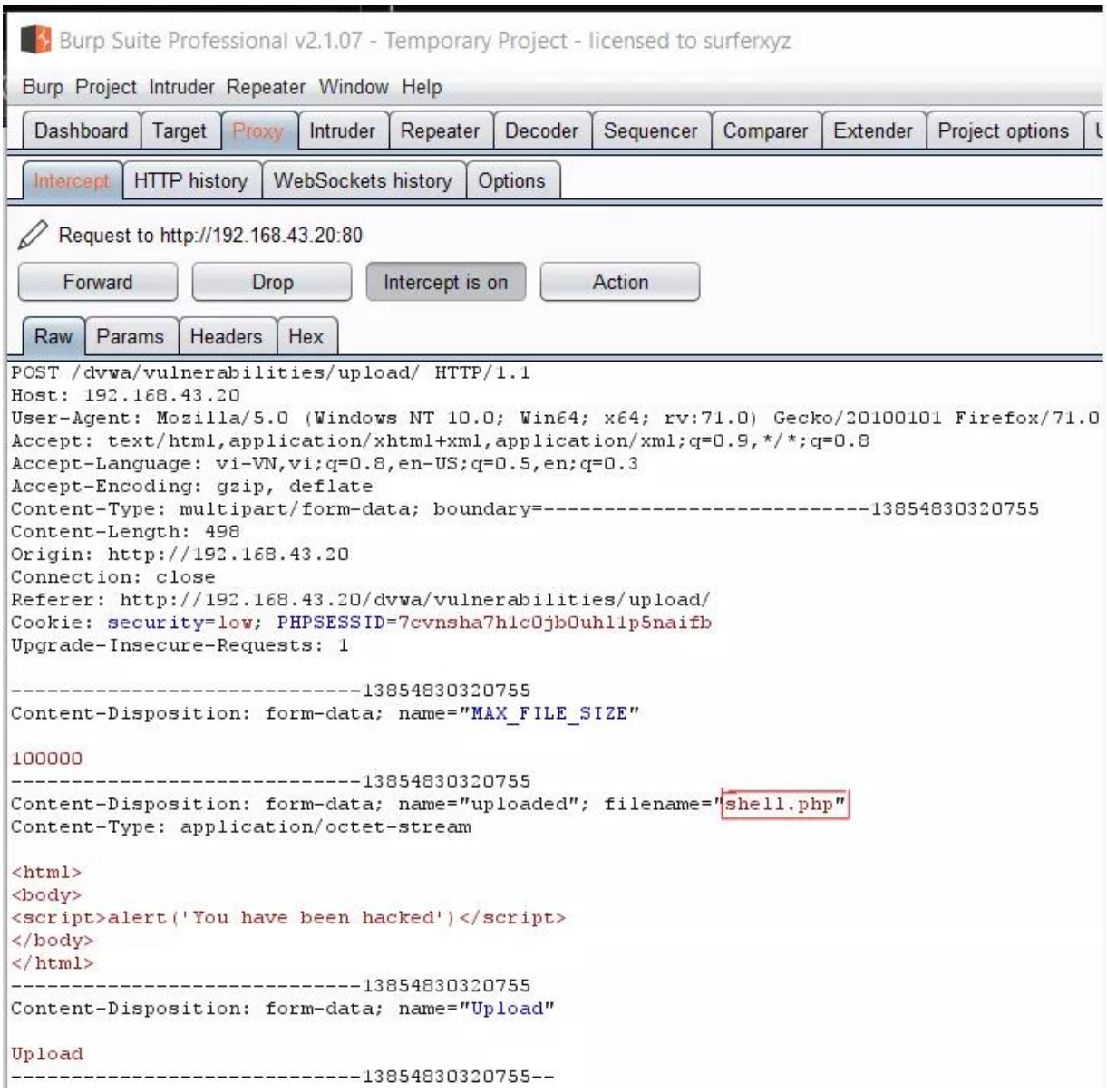
Ví dụ với giả mạo yêu cầu gửi lên, chúng ta có một trang web có thể tải ảnh lên:



Ở đây chúng ta có một file shell.php với nội dung bên trong như sau:

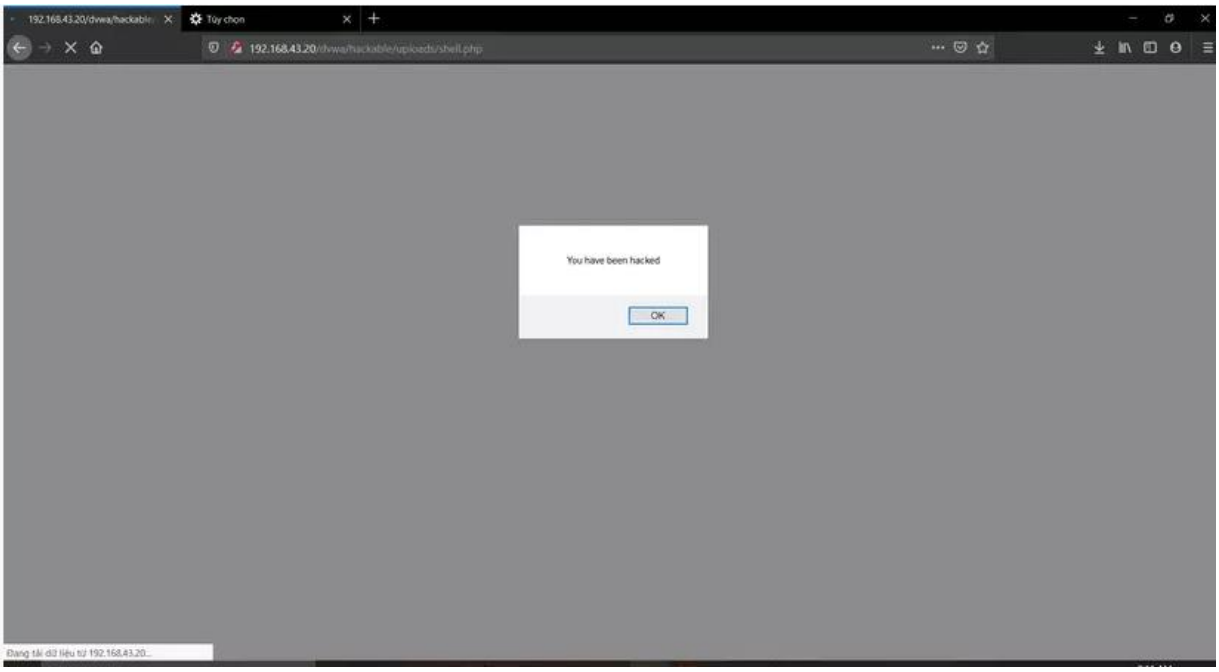
```
<html>
<body>
<script>alert('You have been hacked')</script>
</body>
</html>
```

Tuy nhiên nếu muốn đăng được shell này lên trang web chúng ta phải khiến nó nghĩ rằng chúng ta đã gửi lên đúng định dạng được yêu cầu. Thay đổi extension của file thành shell.php.jpg sau đó sử dụng burp suite để thay đổi request lên server đổi lại tên file thành shell.php:



Vậy là chúng ta đã upload thành công

Truy cập vào đường dẫn ta thu được kết quả



2.2. Content/type Vertification

Đây là kiểu xác thực mà nhà phát triển yêu cầu file upload trong trường hợp này bắt buộc phải là kiểu image thì mới được chấp thuận. Tuy nhiên, Content/type lại có thể thay đổi trước khi đến server cho nên chúng ta chỉ cần đổi từ type application/octet-stream sang image/ (kiểu định dang ảnh của bạn) ví dụ như là image/jpeg.

```
-----------------------------278431405823742
Content-Disposition: form-data; name="MAX_FILE_SIZE"

100000
-----------------------------278431405823742
Content-Disposition: form-data; name="uploaded"; filename="shell.php"
Content-Type: application/octet-stream

<html>
<body>
<script>alert('You have been hacked')</script>
</body>
</html>
-----------------------------278431405823742
Content-Disposition: form-data; name="Upload"

Upload
-----------------------------278431405823742--
```

2.3. The extensions Blacklisting

Như cái tên gọi của nó, blacklist tức là 1 danh sách đen các shell bị các nhà phát triển web chặn nhằm chống việc tải shell lên trang web. Tuy nhiên cũng giống như cái cách mà các nhà mạng viettel chặn các trang web đen, đó là chúng ta sẽ không bao giờ lọc được hết tất cả các trang web hay là các shell.

Ví dụ: Nhà phát triển lọc các trang php để không bị đẩy lên server

- Hacker có thể có thể đổi đuôi extension thành shell.php1, shell.php2, shell.php3, … Và thậm chí shell vẫn có thể chạy với các đuôi như .pl hoặc .cgi.
- Nếu tất cả các đuôi bạn thử đều đã nằm trong danh sách đen, chúng ta có thể check xem bộ lộc có phân biệt chữ hoa chữ thường không: shell.Php1, shell.PHP2
- Đôi khi, nhà phát triển có thể tạo 1 regex kiểm thử.jpg, vì vậy chúng ta có thể thử cách chồng extension như là shell.jpg.php.
- Tuy nhiên, nếu tất cả các bước trên đều thất bại, thì chúng ta vẫn còn 1 khả năng cuối cùng để tải shell lên bằng cách sử dụng .htaccess file. Tập tin .htaccess (hypertext accsess) là một file có ở thư mục gốc của các hosting và do apache quản lý, cấp quyền. File .htaccsess có thể điều khiển, cấu hình được nhiều thứ đa dạng các thông số, nó có thể thay đổi được các giá trị được set mặc định của apache.

2.4. The extentions Whitelisting

Trái ngược với blacklist, có một số trang web lại yêu cầu bạn bắt buộc phải sử dụng những extension được liệt kê trong whitelist như là .jpg, .jpeg, .gif, … Vậy làm như thế nào để vượt qua được nó:

- *Null Byte Injection* là một kỹ thuật khai thác trong đó sử dụng các ký tự null byte URL-encoded (ví dụ: 00%, hoặc 0x00 trong hex) được người dùng cung cấp. Một null byte trong các URL được đại diện bởi '00%', trong ASCII là một "" (giá trị null) trong quá trình lây nhiễm. Phần sau %00 sẽ được hiểu là giá trị null, là giá trị kết thúc của chuỗi nên tệp được tải lên với tên là shell.php *shell.php%00.jpg*.
- *Bypass using Double Extension*

Trong một số trường hợp, chúng ta có thể sử dụng shell.php.jpg, shell.php; .jpg, shell.php: jpg để thực thi lệnh shell, nhưng lổ hỗng này, thường là do cấu hình của webserver hoặc hệ điều hành. Lỗi của nhà phát triển ở đây, là khi chúng ta tải tệp tin lên mà tên của tệp tin không thay đổi dẫn đến lỗ hổng này.

- *Invalid Extension Bypass*

Ngoài ra, vẫn còn 1 số lỗi từ phía máy chủ như là nếu chúng ta sử dụng extension .test thì hệ điều hành sẽ không nhận ra. Cho nên chúng ta có thể tải lên tệp shell.php.test và hệ điều hành sẽ bỏ qua .test và thực thi shell.

2.5. Bypassing the Content Length and Malicious script checks

Đây là cách ít phổ biến nhất, đối với một số trang web nhất định và thường là khá ít, chúng ta sẽ phải đối mặt với kiểm tra độ dài của file upload. Với những web như vậy, chúng ta sẽ sử dụng những lệnh shell ngắn để bypass như là:

<?system ($GET [0]);

II. Cách khắc phục

- Giới hạn các loại tệp mà bạn cho phép tải lên ở những loại cần thiết cho trải nghiệm người dùng.
- Sử dụng bộ lọc whitelist để loại bỏ những extension đem lại rủi ro

- Ứng dụng sẽ thực hiện việc lọc và kiểm tra nội dung của file được tải lên và loại bỏ trực tiếp nếu phát hiện nguy cơ rủi ro.
- Giới hạn quyền của thư mục. Tức là tệp được tải lên sẽ không có bất kì quyền "thực thi" nào đối với ứng dụng, website và tự động loại bỏ nếu có.
- Luôn thực hiện input validation: Kiểm tra đồng thời tên file, content-type, header, file size mỗi khi thực hiện kiểm tra các file upload
- Phân quyền các thư mục upload, nếu là chức năng upload ảnh thì cần chặn quyền thực thi ở thư mục chứa ảnh.
- Tránh để lộ đường dẫn file được upload lên
- Đổi tên file trên server khi upload thành công, thực hiện hash đường dẫn file đã được upload để chống lại việc đoán được đường dẫn file.

III. Áp dụng vào web

Quanlydaotao > upload.php > ...

```php
<?php
  if ($_SERVER['REQUEST_METHOD'] !== 'POST')
  {
      echo "Phải Post dữ liệu";
      die;
  }
  if (!isset($_FILES["fileupload"]))
  {
      echo "Dữ liệu không đúng cấu trúc";
      die;
  }

  if ($_FILES["fileupload"]['error'] != 0)
  {
    echo "Dữ liệu upload bị lỗi";
    die;
  }
  $target_dir    = "../uploads/";
  $target_file   = $target_dir . basename($_FILES["fileupload"]["name"]);

  $allowUpload   = true;

  $imageFileType = pathinfo($target_file,PATHINFO_EXTENSION);

  $maxfilesize   = 800000;

  $allowtypes    = array('xlsx', 'csv');

```

```php
if(isset($_POST["submit"])) {
    $check = filesize($_FILES["fileupload"]["tmp_name"]);
    if($check !== false)
    {
        echo "Đây là file excel - " . @$check["mime"] . ".";
        $allowUpload = true;
    }
    else
    {
        echo "Không phải file ảnh.";
        $allowUpload = false;
    }
}
if (file_exists($target_file))
{
    echo "Tên file đã tồn tại trên server, không được ghi đè";
    $allowUpload = false;
}
if ($_FILES["fileupload"]["size"] > $maxfilesize)
{
    echo "Không được upload file lớn hơn $maxfilesize (bytes).";
    $allowUpload = false;
}

if (!in_array($imageFileType,$allowtypes ))
{
    echo "Chỉ được upload các định dạng xlsx và csv !";
    $allowUpload = false;
}


if ($allowUpload)
{
    if (move_uploaded_file($_FILES["fileupload"]["tmp_name"], $target_file))
    {
        echo "File ". basename( $_FILES["fileupload"]["name"]).
        " Đã upload thành công.";

        echo "File lưu tại " . $target_file;

    }
    else
    {
        echo "Có lỗi xảy ra khi upload file.";
    }
}
else
{
    echo "Không upload được file, có thể do file lớn, kiểu file không đúng ...";
}
?>
```

CHƯƠNG 2: SQL INJECTION

I. Tổng quan

1. Giới thiệu

1.1. Khái niệm

Sql Injection là một kỹ thuật chèn code được sử dụng trong tấn công các ứng dụng chứa dữ liệu (data-driven). Trong đó, các lệnh SQL độc hại được chèn vào trong một entry field để thực thi (chẳng hạn như để kết xuất nội dung Cơ sở dữ liệu cho hacker). SQL Injection phải khai thác lỗ hổng bảo mật trong phần mềm của ứng dụng.

SQL Injection chủ yếu được biết đến như một vector tấn công dành cho các trang web. Nhưng nó cũng có thể được dùng để tấn công bất kỳ loại cơ sở dữ liệu SQL nào. Các cuộc tấn công SQL Injection cho phép hacker giả mạo danh tính, xáo trộn dữ liệu. Hay gây ra các vấn đề như làm mất hiệu lực giao dịch, thay đổi số dư, tiết lộ hay phá hủy dữ liệu trên hệ thống. Thậm chí là làm dữ liệu không khả dụng hoặc trở thành admin của server cơ sở dữ liệu.

2. Các kiểu tấn công

2.1. In-band SQLi

Đây là phương thức tấn công SQL phổ biến nhất hiện nay. Điểm nổi bật nhất của In-band SQLi là kẻ xấu sẽ sử dụng cùng một kênh để tiến hành tấn công và thu thập dữ liệu đánh cắp được. In-band SQLi hiện có 2 biến thể thông dụng bao gồm:

- Error-based SQLi: Đầu tiên, kẻ tấn công sẽ cài một đoạn mã độc để hệ thống cơ sở dữ liệu báo lỗi. Sau đó hacker sẽ dùng dữ liệu thu thập được từ những thông báo này để truy xuất ra thông tin của cấu trúc cơ sở dữ liệu.
- Union-based SQLi: Bằng cách lợi dụng toán tử UNION SQL, hacker sẽ tiến hành hợp nhất các câu lệnh được tạo ra từ cơ sở dữ liệu để thu được một HTTP response. Trong response sẽ chứa thông tin riêng tư mà kẻ tấn công nhắm đến.

2.2. Inferential (Blind) SQLi là gì

Inferential SQLi có đặc tính blind vì hacker sẽ không thể thấy trực tiếp cách mà cuộc tấn công hoạt động. Kẻ tấn công không trực tiếp gây tổn hại đến cơ sở dữ liệu mà

sẽ gửi các data payload đến server. Những data payload này sẽ gây ảnh hưởng đến cơ sở dữ liệu của bạn và bạn buộc phải đưa ra những phản ứng công khai. Đây chính là điều hacker cần, họ nắm bắt những phản ứng này và đưa ra những phán đoán về cấu trúc cơ sở dữ liệu của bạn.

Inferential SQLi thường được thực thi chậm hơn vì nó cần đợi những phản ứng của server. Tuy nhiên, thiệt hại nó gây ra lại không vì thế mà bị hạn chế bớt đi. Có 2 biến thể của Inferential SQLi thường xuyên được sử dụng:

- Boolean: Đầu tiên, kẻ xấu sẽ gửi một câu truy vấn SQL đến cho server. Khi đó, cơ sở dữ liệu buộc phải gửi trả lại kết quả để trả lời cho câu lệnh này. Đáp án có thể là đúng hoặc sai. Dựa theo đáp án mà thông tin của HTTP response sẽ được chỉnh sửa đến khi đúng với thực tế. Vậy là hacker đã nắm được những thông tin xung quanh cấu trúc server.
- Time-based: Cách thức tấn công này cũng tương tự như Boolean. Tuy nhiên, thay vì đợi cơ sở dữ liệu đưa ra đáp án, hacker sẽ dùng những câu lệnh SQL làm server ngừng hoạt động trong vài giây. Sau đó từ mốc thời gian phản hồi tra ra được kết quả của các truy vấn. Như vậy, một HTTP response đã được tạo ra.

2.3. Out-of-band SQLi

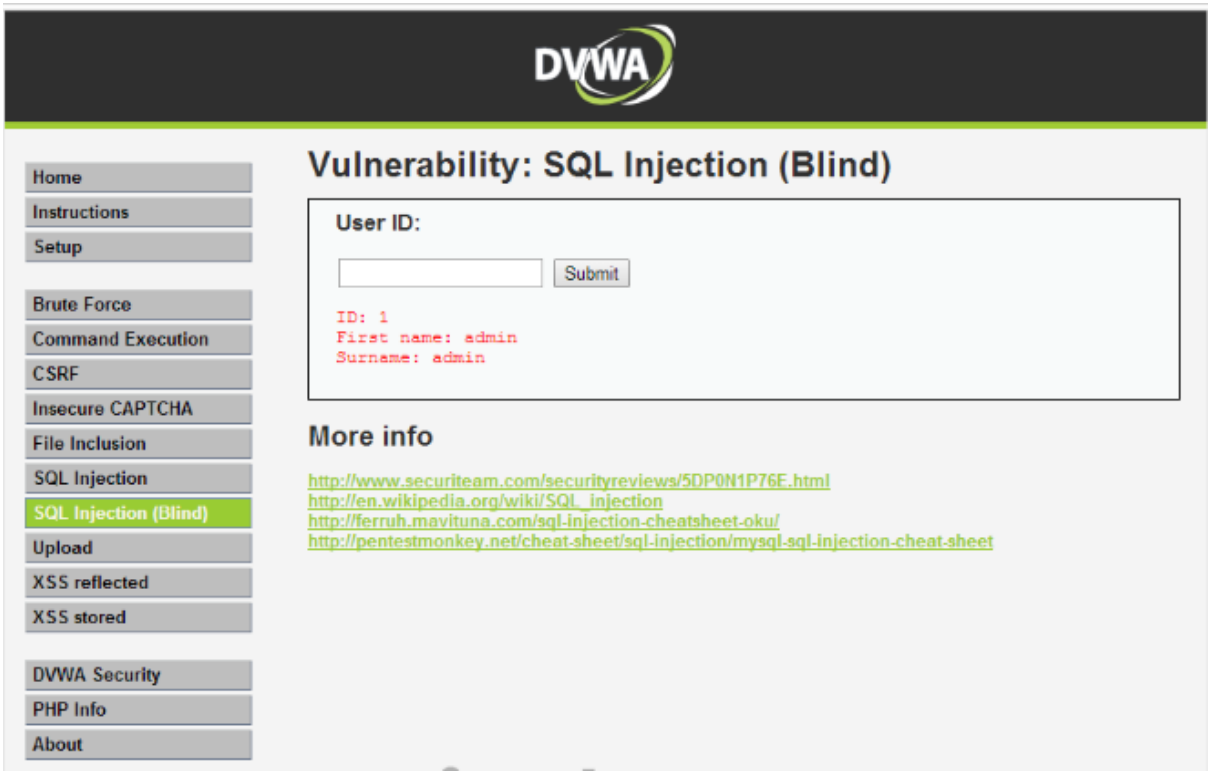
Nếu không thể thực hiện 2 phương pháp trên do server hoạt động quá chậm, không ổn định để tấn công hoặc hacker không có một kênh để đồng thời tấn công và thu thập kết quả thì sẽ kẻ xấu sẽ suy nghĩ đến phương án thứ ba là Out-of-band SQLi.

Tuy nhiên cách này cần có điều kiện: một số tính năng của server phải được kích hoạt. Hacker sẽ nắm lấy cơ hội server tạo ra DNS hay HTTP request để thu lại được dữ liệu cho mình.

Ví dụ về cách sử dụng SQL Injection

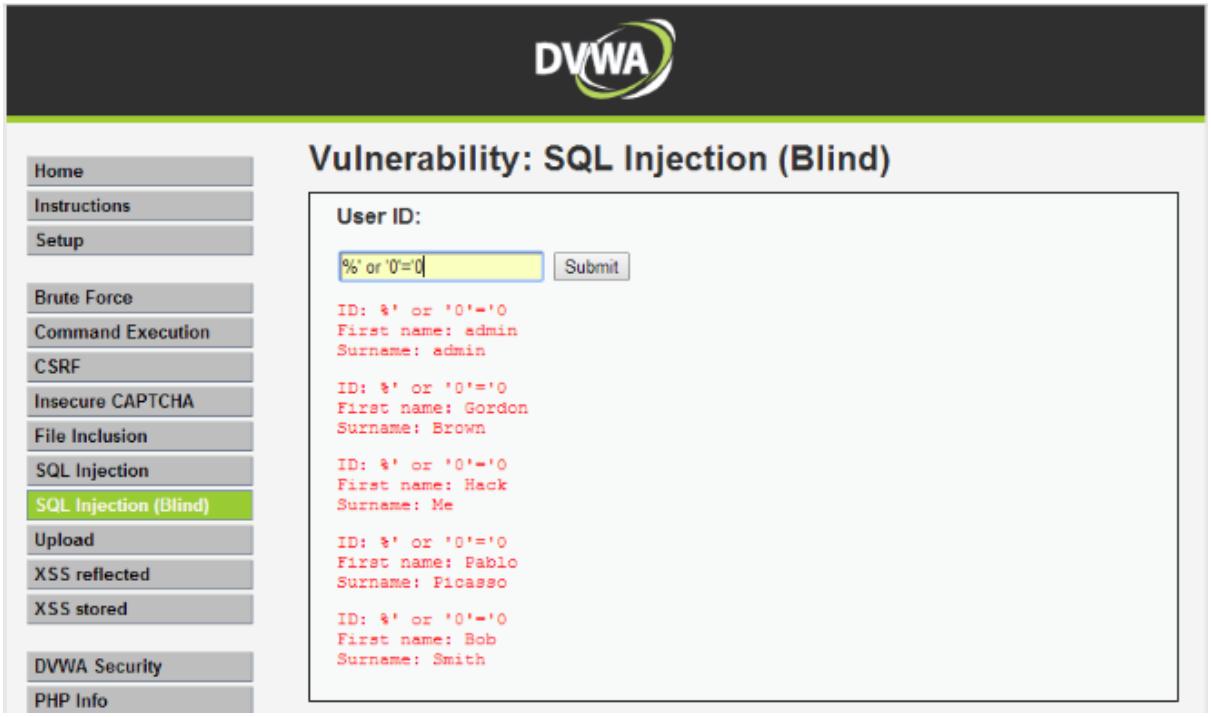
Tại hộp text này nếu chúng ta nhập ID (id=1 tới 5) thì sẽ hiển thị từng bản ghi trong CSDL vì đã được lập trình với câu lệnh sql truy vấn như sau:

$getid = "SELECT first_name, last_name FROM users WHERE user_id = '$id'";



Trường hợp 1:

ID tại text có giá trị: %’ or ‘0’ = ‘0



Vậy sẽ hiển thị tất cả các bản ghi trong CSDL.

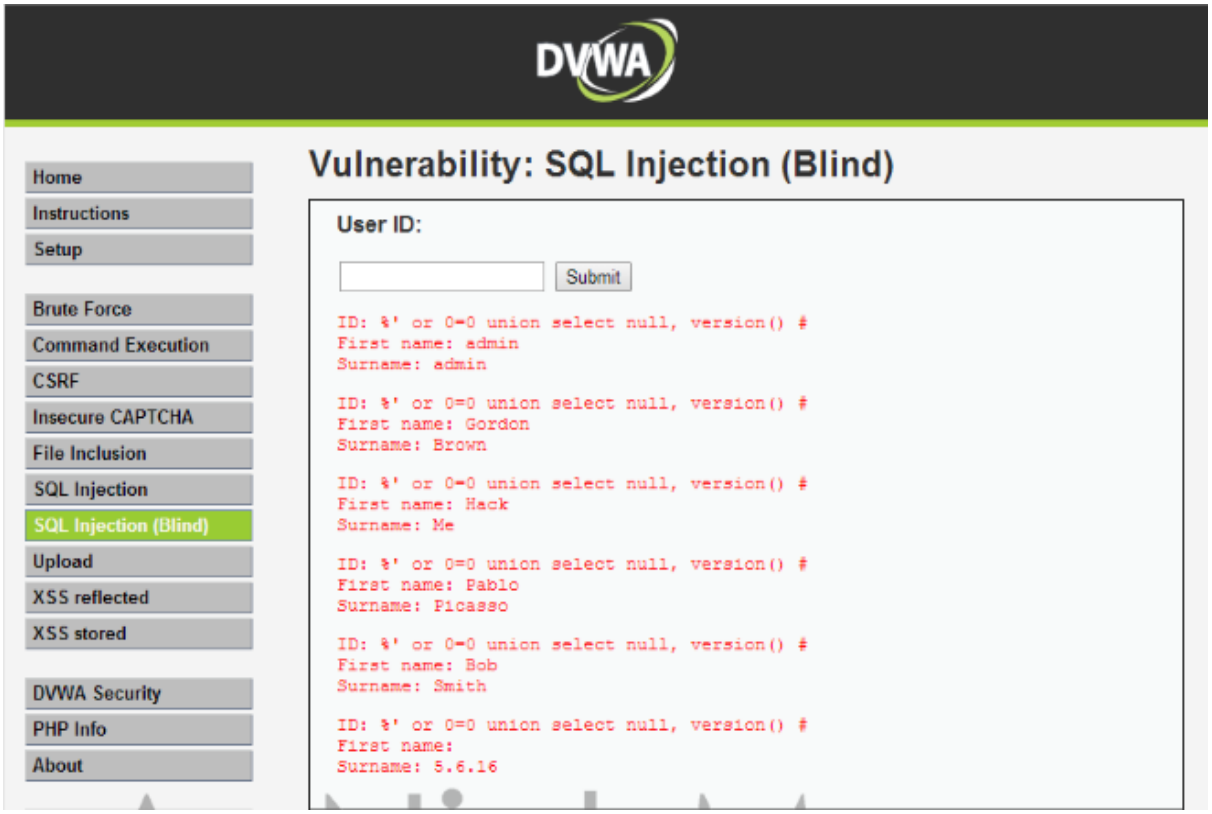
Vì ký tự %' có thể là không đúng với CSDL nào và '0'='0' là biểu thước luôn đúng (sẽ thực hiện vế này)

Như vậy câu truy vấn bây giờ sẽ là:

mysql> SELECT first_name, last_name FROM users WHERE user_id = '%' or '0'='0'

Trường hợp 2: Hiển thị phiên bản CSDL Mysql

Nhập tại Text: %' or 0=0 union select null, version() #

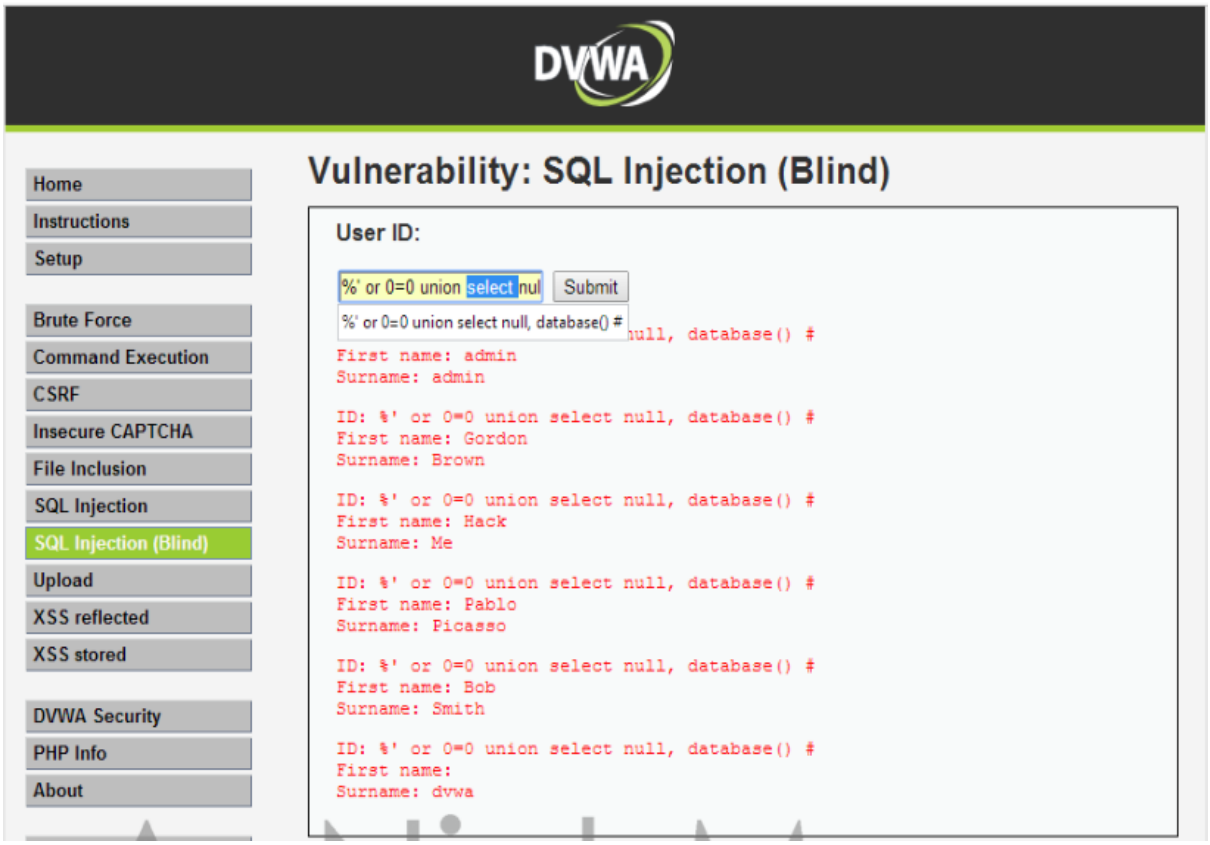


Dòng cuối cùng của thông tin chính là phiên bản CSDL

Vì hàm union: ghép chuỗi, hàm version() hiển thị phiên bản

Trường hợp 3: Hiển thị tên CSDL

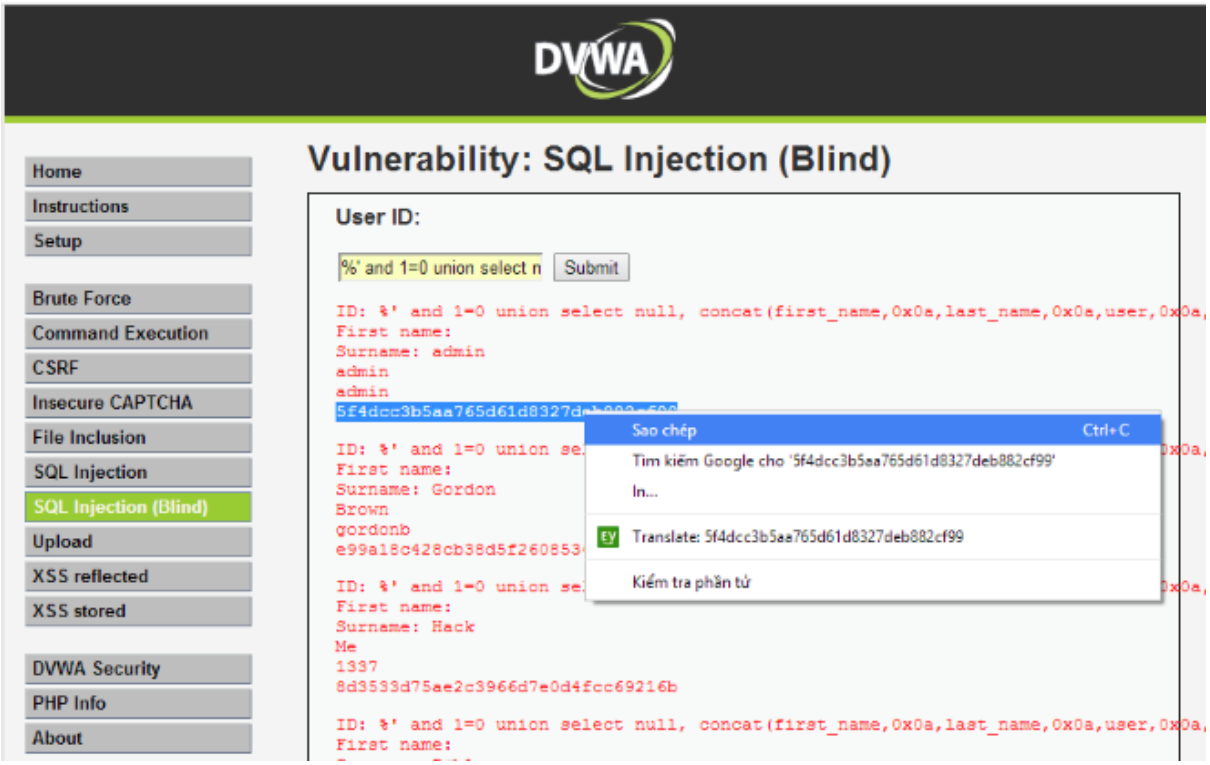
Nhập tại Text: %’ or 0=0 union select null, database() #



Hàm database() hiển thị tên CSDL sẽ được hiển thị ở bản ghi cuối cùng.

Trường hợp 4: Hiển thị nội dung các cột của bảng User trong CSDL

Nhập: %’ and 1=0 union select null, concat(firt_name,0x0a,lastname,0x0a,user,0x0a,password) from user #



II. Biện pháp khắc phục

- *Không tin tưởng kênh Input của người dùng*: Trong thực tế, mọi thông tin trên kênh Input mà người dùng nhập đều được đánh dấu “don’t trust and verify”. Điều này có nghĩa là một thông tin ngoại địa đều sẽ được coi là độc hại, trừ khi có bằng chứng ngược lại. Là một người quản lý server, bạn phải tỉnh táo và không được tin những dữ liệu này. Mọi thứ xâm nhập từ bên ngoài đều phải được quản lý sát sao, bao gồm cả văn bản, input ẩn, các chuỗi tham số truy vấn, cookie và tệp tải lên.
- *Xác nhận chuỗi các input ở phía máy chủ*: một trong những quá trình đảm bảo cho dữ liệu của người nhập vào hợp lệ, và có thể vô hiệu hóa bất kỳ lệnh độc hại tiềm ẩn nào khác đều có thể sử dụng trong chuỗi nhập. Các sửa đổi đơn giản này sẽ bảo vệ mã của bạn tránh khỏi các cuộc tấn công SQL Injection bằng biện pháp thêm ký tự thoát (\) vào trước dấu nháy đơn đã được kẻ tấn công thêm vào.

- *Sử dụng các câu lệnh tham số:* Một cách hiệu quả khác nữa để ngăn chặn các SQL Injection là sử dụng câu lệnh tham số. Việc tham số hóa các câu lệnh giúp cơ sở dữ liệu có thể phân biệt giữa mã và dữ liệu đầu vào. Chính vì thế, nó sẽ dễ dàng chặn lại các dữ liệu không được cung cấp tham số khi chúng muốn xâm nhập.
- *Mã hóa dữ liệu nhạy cảm:* Nó thường bao gồm mật khẩu, câu hỏi, câu trả lời về bảo mật, dữ liệu tài chính, thông tin y tế và những thông tin khác. Điều này sẽ đảm bảo cả khi tin tặc nắm trong tay dữ liệu của bạn thì chúng cũng không thể khai thác nó ngay lập tức và cho bạn thời gian để phát hiện ra sự vi phạm.

CHƯƠNG 4: KẾT LUẬN

Bảo mật web là nhiệm vụ vô cùng quan trọng đối với mỗi website trong quá trình sử dụng, vận hành trang web. Việc tìm hiểu các lỗ hổng và áp dụng để bảo mật website sẽ giúp đảm bảo hạ tầng, bảo đảm an toàn an ninh dữ liệu website giúp web hoạt động ổn định, tấn công. Tránh bị các hacker xâm nhập chiếm các thông tin người dùng, dữ liệu với các mục đích xấu khác nhau.