TỔNG LIÊN ĐOÀN LAO ĐỘNG VIỆT NAM
TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG
KHOA CÔNG NGHỆ THÔNG TIN

BÀI TIỂU LUẬN MÔN LẬP TRÌNH WEB VÀ ỨNG DỤNG

TÌM HIỂU WEB SECURITY

Người hướng dẫn: **Thầy VŨ ĐÌNH HỒNG**

Người thực hiện: **PHẠM ĐỨC MINH HIẾU – 52100796**

NGUYỄN VĂN HÀO – 52100792

Lớp : **21050201**

Khoá : **25**

THÀNH PHỐ HỒ CHÍ MINH, NĂM 2023

TỔNG LIÊN ĐOÀN LAO ĐỘNG VIỆT NAM
TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG
KHOA CÔNG NGHỆ THÔNG TIN

BÀI TIỂU LUẬN MÔN LẬP TRÌNH WEB VÀ ỨNG DỤNG

TÌM HIỂU WEB SECURITY

(Bìa phụ)

Người hướng dẫn: **Thầy VŨ ĐÌNH HỒNG**

Người thực hiện: **PHẠM ĐỨC MINH HIẾU – 52100796**

NGUYỄN VĂN HÀO – 52100792

Lớp : **21050201**

Khoá : **25**

THÀNH PHỐ HỒ CHÍ MINH, NĂM 2023

LỜI CẢM ƠN

Lời đầu tiên chúng em xin được phép cảm ơn quý thầy cô khoa Công nghệ Thông Tin trường Đại học Tôn Đức Thắng đã tạo mọi điều kiện cho chúng em tạo mọi điều kiện tốt nhất để chúng em được học tập môn Lập trình web và ứng dụng.

Chúng em cũng xin được gửi lời cảm ơn đến thầy Vũ Đình Hồng, thầy đã nhiệt tình giảng dạy, trang bị đầy đủ kiến thức để chúng em có thể hoàn thành bài tiểu luận giữa kỳ này.

Cuối cùng, do hạn chế về mặt kiến thức, kính mong thầy cô có thể bỏ qua những sai sót nhỏ và chỉ ra được những lỗi sai của chúng em trong bài tiểu luận này để những bài tiểu luận sau của chúng em được hoàn thiện hơn.

Một lần nữa chúng em xin chân thành cảm ơn thầy Vũ Đình Hồng và toàn thể quý thầy cô khoa Công Nghệ Thông Tin trường Đại học Tôn Đức Thắng.

CÔNG TRÌNH ĐƯỢC HOÀN THÀNH TẠI TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG

Chúng em xin cam đoan đây là sản phẩm đồ án của riêng chúng em và được hướng dẫn bởi ThS Vũ Đình Hồng. Các nội dung nghiên cứu, kết quả trong đề tài này là trung thực và chưa công bố dưới bất kỳ hình thức nào trước đây. Những số liệu trong các bảng biểu phục vụ cho việc phân tích, nhận xét, đánh giá được chính tác giả thu thập từ các nguồn khác nhau có ghi rõ trong phần tài liệu tham khảo.

Ngoài ra, trong tiểu luận còn sử dụng một số nhận xét, đánh giá cũng như số liệu của các tác giả khác, cơ quan tổ chức khác đều có trích dẫn và chú thích nguồn gốc.

Nếu phát hiện có bất kỳ sự gian lận nào nhóm chúng em xin hoàn toàn chịu trách nhiệm về nội dung Tiểu luận giữa kỳ lập trình web và ứng dụng của mình. Trường Đại học Tôn Đức Thắng không liên quan đến những vi phạm tác quyền, bản quyền do chúng em gây ra trong quá trình thực hiện (nếu có).

TP. Hồ Chí Minh, ngày 26 tháng 4 năm 2023

Tác giả

(ký tên và ghi rõ họ tên)

Phạm Đức Minh Hiếu

Nguyễn Văn Hào

PHẦN XÁC NHẬN VÀ ĐÁNH GIÁ CỦA GIẢNG VIÊN

Phần xác nhận của GV hướng dẫn

_____

_____

_____

_____

_____

_____

_____

Tp. Hồ Chí Minh, ngày tháng năm
(kí và ghi họ tên)

Phần đánh giá của GV chấm bài

_____

_____

_____

_____

_____

_____

_____

Tp. Hồ Chí Minh, ngày tháng năm
(kí và ghi họ tên)

TÓM TẮT

Với sự phát triển của thời đại công nghệ số, các trang web được thành lập ngày càng nhiều và sự đa dạng về dữ liệu là rất lớn. Điều này có lợi vì có thể các doanh nghiệp truy xuất được thông tin mà không mất nhiều thời gian. Tuy nhiên, việc phát triển công nghệ web này đòi hỏi phải có một sự bảo mật tốt để tránh việc bị đánh cắp thông tin từ bên thứ ba.

Trong bài tiểu luận này, chủ đề nghiên cứu web security của chúng em trình bày về cách thức mà hacker tấn công vào trang web và nêu lên một số biện pháp phòng tránh tình trạng bị đánh cắp hoặc sửa đổi thông tin cá nhân. Đảm bảo việc bảo mật thông tin cá nhân người dùng, các phương thức tấn công trang web bao gồm SQL Injection, XSS, DdoS, Brute Force Attack và một số phương thức khác. Đồng thời đưa ra một vài demo để minh họa cho bài tiểu luận thêm sinh động và rõ ràng hơn.

MỤC LỤC

DANH MỤC TỪ VIẾT TẮT

XSS: Cross-Site Scripting

CSRF: Cross-Site Request Forgery

DDoS: Distributed Denial of Service

HTTP: Hypertext transfer protocol

TCP/IP: Transmission Control Protocol / Internet Protocol

UDP: User Datagram Protocol

ICMP: Internet Control Message Protocol

LOIC: Low Orbit ION cannon

HOIC: High Orbit ION cannon

DANH MỤC HÌNH ẢNH

1. NHỮNG LỖ HỔNG THƯỜNG GẶP ĐỐI VỚI MỘT WEBSITE VÀ CÁCH THỨC HOẠT ĐỘNG

1.1. Giới thiệu:

Web Security là một chủ đề quan trọng và cực kỳ cần thiết trong thời đại công nghệ số hiện nay. Với sự phát triển mạnh mẽ của internet, các trang web và ứng dụng web đang trở thành mục tiêu tấn công của các hacker và tin tặc. Vì vậy, việc bảo vệ các trang web và hệ thống web trở thành một vấn đề rất quan trọng đối với các nhà phát triển và quản trị viên hệ thống.

Trong đề tài này, chúng ta sẽ tìm hiểu về các lỗ hổng bảo mật phổ biến trên các trang web và hệ thống web, cùng với các biện pháp phòng ngừa và giải pháp bảo mật để bảo vệ các trang web và hệ thống web khỏi các cuộc tấn công độc hại. Chúng ta sẽ tìm hiểu về các lỗ hổng bảo mật như SQL Injection, Cross-Site Scripting (XSS), Cross-Site Request Forgery (CSRF), và DDoS (Distributed Denial of Service),.. cùng với các giải pháp bảo mật để phòng tránh được các cuộc tấn công ấy.

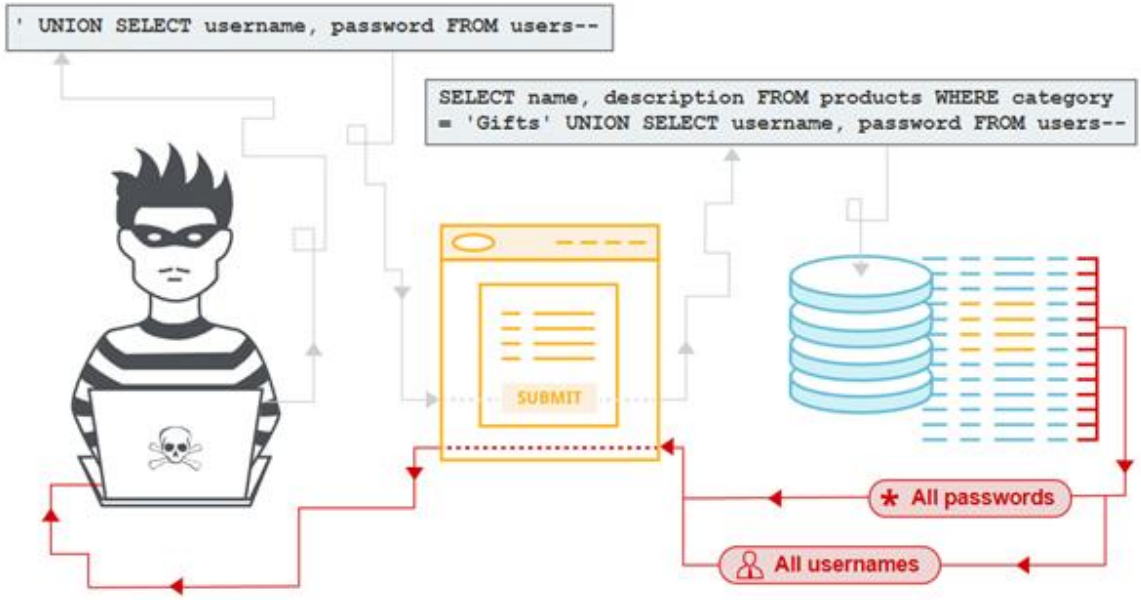
Việc nắm vững kiến thức về Web Security sẽ giúp chúng ta hiểu rõ hơn về các mối đe dọa bảo mật trên các trang web và hệ thống web và cung cấp các giải pháp bảo mật hiệu quả để bảo vệ chúng khỏi các cuộc tấn công độc hại.Tránh việc rò rỉ thông tin và mất dữ liệu hoặc bị đánh cắp dữ liệu. Sau đây là những kỹ thuật mà hacker thường dùng để tấn công vào một trang web.

1.2. Tấn công bằng SQL Injection:

SQL Injection là một kỹ thuật tấn công phổ biến nhằm khai thác các lỗ hổng bảo mật trong các ứng dụng web sử dụng ngôn ngữ SQL để truy vấn cơ sở dữ liệu. Tấn công này được thực hiện bằng cách chèn các câu lệnh SQL độc hại vào các trường nhập liệu trên trang web, từ đó thực hiện các hành động truy vấn, sửa đổi hoặc xóa dữ liệu trong cơ sở dữ liệu mà không được phép.

Các hacker thường sử dụng SQL Injection để truy cập thông tin nhạy cảm, như tài khoản ngân hàng, thông tin cá nhân của người dùng hoặc để phá hoại hệ thống bằng cách xóa hoặc thay đổi dữ liệu.

Cách thức hoạt động của SQL Injection là kẻ tấn công sử dụng các chuỗi đầu vào theo cấu trúc của ngôn ngữ truy vấn SQL để thực hiện các truy vấn không mong muốn. Khi một ứng dụng web không kiểm tra và sàng lọc đầu vào của người dùng, tấn công SQL Injection có thể thành công bằng cách chèn các truy vấn SQL độc hại vào các trường đầu vào của một biểu mẫu web hoặc các tham số URL.



Hình 1.2.1 Minh họa hacker tấn công bằng SQL Injection

Hậu quả của tấn công SQL Injection có thể rất nghiêm trọng, bao gồm mất dữ liệu, mất kiểm soát của cơ sở dữ liệu, việc tấn công được tiếp tục truy cập vào hệ thống và từ đó gây ra các tác động lớn hơn như thất thoát thông tin quan trọng hoặc tài nguyên của hệ thống.

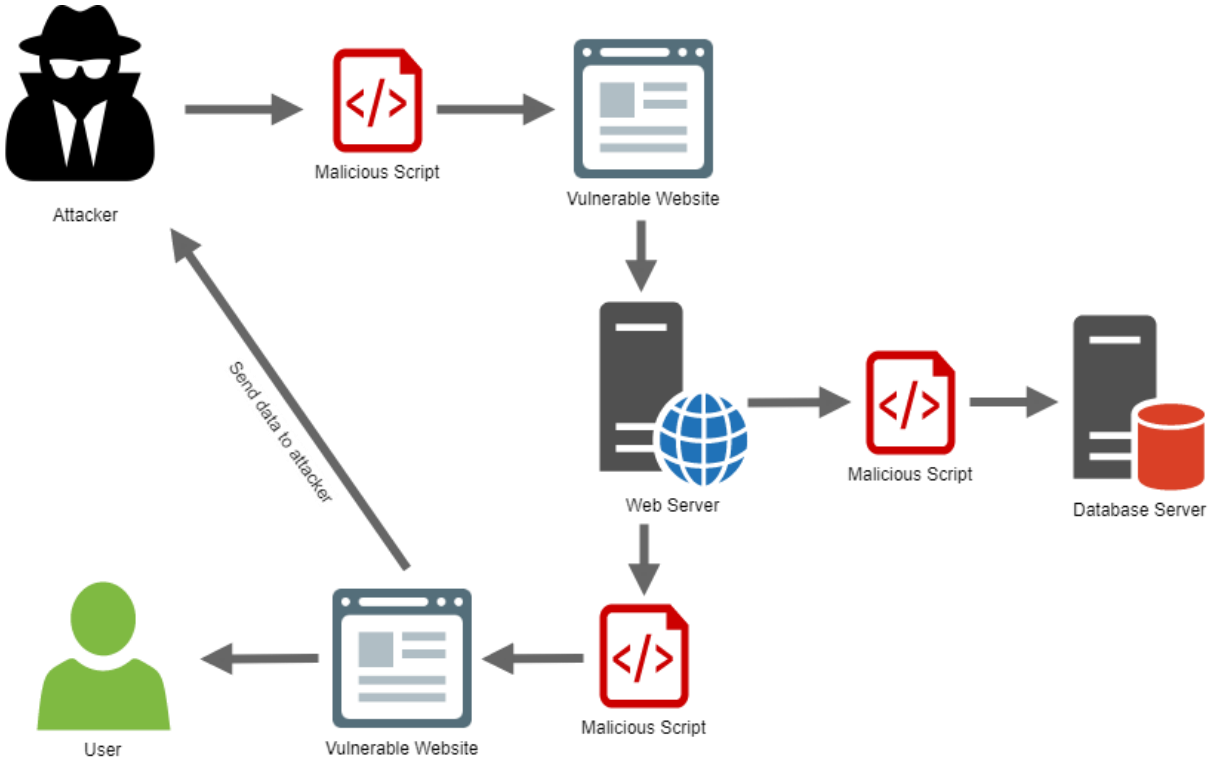
1.3. Tấn công bằng XSS (Cross Site Scripting):

XSS (Cross Site Scripting) là một lỗ hổng bảo mật trong các ứng dụng web, cho phép tin tặc chèn mã độc vào trang web và thực thi nó trên máy tính của người dùng khi họ truy cập trang đó. Các mã độc này có thể lấy được thông tin nhạy cảm của người dùng, thay đổi nội dung trang web hoặc thực hiện các hành động khác như đánh cắp cookie, tạo tài khoản giả mạo, hoặc thậm chí kiểm soát hoàn toàn trang web.

Có hai loại XSS: Reflected XSS và Stored XSS. Reflected XSS xảy ra khi mã độc được chèn vào trang web thông qua các tham số trong URL hoặc biểu mẫu và được thực thi ngay lập tức trên trình duyệt của người dùng. Stored XSS xảy ra khi mã độc được lưu trữ trên máy chủ và được thực thi khi người dùng truy cập trang chứa mã độc đó.

Các hacker sử dụng nhiều phương pháp để tấn công các trang web bị lỗ hổng XSS, bao gồm:

- Thêm mã độc vào các trường biểu mẫu có thể được lưu trữ trên máy chủ.
- Thêm đoạn mã độc vào các tài liệu được đưa ra trên trang web, chẳng hạn như trang giới thiệu sản phẩm hoặc trang giới thiệu công ty.
- Chèn mã độc vào các trang thông báo lỗi hoặc cảnh báo bằng cách sử dụng các ký tự đặc biệt hoặc các ký tự mã hóa.



Hình 1.3.1 Minh họa hacker tấn công bằng XSS

Hậu quả của tấn công XSS có thể rất nghiêm trọng, bao gồm mất dữ liệu, mất kiểm soát của trang web, việc tấn công được tiếp tục truy cập vào hệ thống và từ đó gây ra các tác động lớn hơn như thất thoát thông tin quan trọng hoặc tài nguyên của hệ thống.

1.4. Tấn công bằng Security Misconfiguration:

Security Misconfiguration là một trong những lỗ hổng bảo mật phổ biến trong các ứng dụng web. Lỗ hổng này xảy ra khi cấu hình hệ thống bị sai, cho phép kẻ tấn công tìm thấy các thông tin quan trọng về cấu hình hệ thống, các môi trường, tài khoản, các thư mục và tệp, cũng như các tham số và giá trị được sử dụng bởi các ứng dụng.

Các lỗ hổng Security Misconfiguration thường xảy ra do thiếu kiểm soát và quản lý cấu hình hệ thống, các lỗi lập trình, sử dụng các giá trị mặc định, cập nhật phần mềm và hệ điều hành không đầy đủ hoặc sử dụng các chính sách an ninh không chặt chẽ. Khi xảy ra lỗ hổng này, kẻ tấn công có thể khai thác để truy cập vào hệ thống, đánh cắp thông tin quan trọng, thay đổi các cài đặt hệ thống và thực hiện các hành động độc hại.

1.5. Tấn công bằng DdoS:

DdoS (Distributed denial of service) là cuộc tấn công từ chối dịch vụ phân tán nhắm mục tiêu vào các website và máy chủ qua việc gây gián đoạn các dịch vụ mạng. Khi một cuộc tấn công DDoS xảy ra, kẻ tấn công sẽ điều khiển các máy tính bị lây nhiễm để tạo ra một lượng lớn yêu cầu truy cập đến một trang web hoặc hệ thống cụ thể. Khi số lượng yêu cầu truy cập này vượt quá khả năng xử lý của hệ thống đó, hệ thống sẽ không thể phản hồi lại các yêu cầu từ người dùng hợp lệ và trang web hoặc hệ thống sẽ bị tắt nghẽn và không thể hoạt động. Trong đó có 3 loại DDoS phổ biến: Volume-based (sử dụng lưu lượng truy cập cao trong thời gian ngắn để làm tràn băng thông mạng), Protocol (tập trung vào việc khai thác các tài nguyên, dữ liệu của máy chủ), Application (tập trung vào các ứng dụng web, khai thác các lỗ hổng bảo mật để đạt được các mục đích mà hacker muốn).

Cách thức hoạt động của một cuộc tấn công DdoS bao gồm:

- Khi bị tấn công DDoS, một chuỗi bot hoặc botnet sẽ gây tràn một website hoặc dịch vụ bằng các yêu cầu HTTP và lưu lượng truy cập. Về cơ bản, trong một cuộc tấn công, nhiều máy tính sẽ tấn công một máy tính, khiến người dùng hợp pháp bị đẩy ra. Từ đó, dịch vụ có thể bị trì hoãn hay nói cách khác là bị gián đoạn trong một khoảng thời gian.
- Khi bị tấn công, tin tặc có thể đột nhập vào cơ sở dữ liệu của bạn và truy nhập vào mọi loại thông tin quan trọng. Cuộc tấn công DDoS có thể khai thác các lỗ hổng về bảo mật và nhắm mục tiêu tới bất kỳ điểm cuối nào có thể tiếp cận công khai qua internet.
- Cuộc tấn công từ chối dịch vụ có thể tồn tại trong nhiều giờ, hay thậm chí là nhiều ngày. Những cuộc tấn công mạng kiểu này cũng có thể gây ra nhiều gián đoạn trong suốt cuộc tấn công. Cả thiết bị cá nhân lẫn doanh nghiệp đều dễ bị ảnh hưởng từ đó.

Các phương thức tấn công DDoS:

- DDoS theo phương thức TCP / IP

Đây là loại tấn công lớn nhất và phổ biến nhất. Nó được thực hiện bằng cách gửi một lượng lớn các yêu cầu kết nối TCP (là giao thức truyền thông giữa 2 thiết bị mạng) đến máy chủ đích, khiến cho máy chủ đích sử dụng tài nguyên nhiều hơn so với quy định. Điều này dẫn đến việc giảm hiệu suất của máy chủ đích và dễ dàng tạo ra trường hợp từ chối dịch vụ.

- DDoS theo phương thức HTTP

Cách thức tấn công này tương tự như DDoS theo phương thức TCP / IP, nhưng được thực hiện bằng cách sử dụng yêu cầu HTTP thay vì yêu cầu kết nối TCP (HTTP truyền tải dữ liệu không đồng bộ trong khi TCP/IP truyền tải dữ liệu đồng bộ và cần nhiều thông tin quan trọng như trạng thái kết nối hay trạng thái của các gói tin). Điều này giúp tấn công tỏ ra nhẹ nhàng hơn, không chiếm nhiều tài nguyên hệ thống và do đó khó phát hiện .

- DDoS theo phương thức UDP

Tấn công bằng phương thức UDP được thực hiện bằng cách gửi một lượng lớn các gói tin UDP tới địa chỉ IP của máy chủ đích. Do đó, máy chủ đích sẽ không thể xác định được xem các gói tin được gửi từ đâu và trả lời lại chúng. Điều này dẫn đến việc giảm hiệu suất của máy chủ và có thể dẫn đến trường hợp từ chối dịch vụ.

- DDoS theo phương thức ICMP

Loại tấn công này được thực hiện bằng cách gửi các gói tin ICMP Echo Request đến máy chủ đích. Khi máy chủ đích nhận được các yêu cầu này, nó sẽ phản hồi với các gói tin ICMP Echo Reply. Tuy nhiên, khi số lượng yêu cầu gửi tăng lên, máy chủ đích sẽ không thể xử lý và trả lời lại đủ các yêu cầu này, dẫn đến trường hợp từ chối dịch vụ.

Một số công cụ thực hiện tấn công DdoS:

- LOIC

LOIC, viết tắt của cụm từ Low Orbit ION cannon, được biết đến là phần mềm mã nguồn mở lý tưởng cho các cuộc tấn công DDoS. Ngôn ngữ lập trình nên LOIC là C #, cho phép thực hiện gửi yêu cầu HTTP, UDP TCP tới máy chủ. Điểm ấn tượng khác của công cụ, bao gồm miễn phí kiểm tra hiệu suất của mạng; khởi tạo DDoS Attack trực tuyến xâm nhập vào bất kỳ website nào; không ẩn địa chỉ IP trong cả trường hợp máy chủ Proxy không hoạt động,…

- HOIC

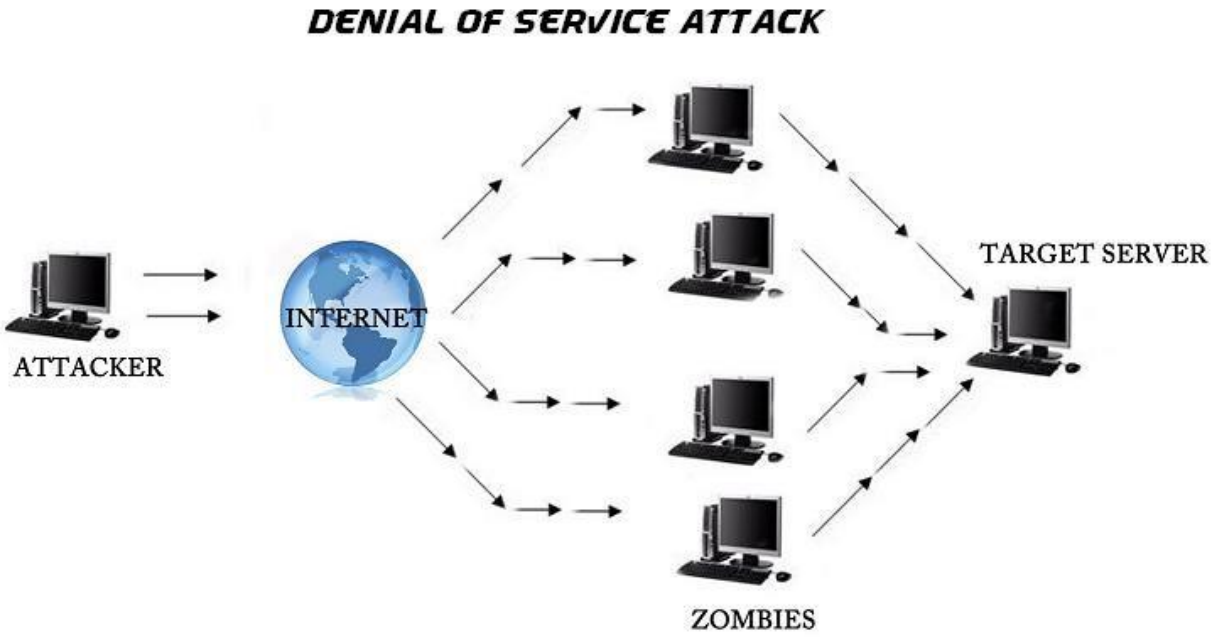
  HOIC (High Orbit ION cannon) là một công cụ miễn phí, cho phép tấn công đồng thời 256 trang web cùng lúc. Phương thức mà HOIC sử dụng là HTTP, được tối ưu trên cả Linux hoặc Mac OS và cho phép kiểm soát các cuộc tấn công với các mức độ tùy chỉnh.

- HTTP Unbearable Load King

  HTTP Unbearable Load King (HULK) cũng là một công cụ tấn công từ chối dịch vụ phân tán miễn phí, có thể xóa bộ nhớ cache của máy chủ, tạo lưu lượng mạng riêng biệt. Song, bản thân HULK cũng rất phù hợp trong công tác nghiên cứu của các nhà phát triển công nghệ, nhà bảo mật hệ thống.

- DDoSIM

  DDoSIM hay DDoS Simulator sẽ là lựa chọn phù hợp nếu bạn muốn biết khả năng xử lý của máy chủ trước các cuộc tấn công DDoS. Ngoài ra, nó cũng sở hữu khả năng tạo các kết nối TCP đến máy chủ mục tiêu một cách đầy đủ, gây tình trạng làm tràn ngẫu nhiên các kết nối TCP trên một cổng…



Hình 1.5.1 Minh họa một cuộc tấn công DdoS

Những hậu quả dễ thấy của một cuộc tấn công DDoS bao gồm:

- Hệ thống bị quá tải, không thể truy cập hay thực thi bất kỳ công việc nào.
- Máy tính hoạt động thiếu ổn định, bị giật lag bất thường.
- Hệ thống truy cập web bị sập, hỏng hoàn toàn.
- Truy cập nhiều lần cùng lúc vào một trang web nào đó với rất nhiều yêu cầu.
- Tệ hơn, dù ảnh hưởng đối với máy tính và hệ thống website trước DDoS Attack là gì đi nữa, nó chưa thể sánh những hệ lụy đối với doanh nghiệp, tổ chức đứng sau. Họ vừa phải gánh chịu chi phí khắc phục, sửa chữa hệ thống, vừa mất đi lượng người dùng lớn cả ngắn và dài hạn,… Thiệt hại sau cùng có thể lên đến hàng ngàn tỷ đồng.

1.6. Tấn công bằng Clickjacking

Clickjacking (còn được gọi là “UI redress attack”) là một thuật ngữ diễn tả việc lừa người sử dụng click chuột vào một liên kết đưa người dùng tới các trang web giả mạo hay tải xuống một phần mềm độc hại mà không hề hay biết từ đó người dùng có

nguy cơ bị đánh cắp thông tin cá nhân hay tài khoản đăng nhập hay thậm chí là kiểm soát máy tính của người dùng.

Cách thức hoạt động của Clickjacking rất đơn giản bằng việc hacker tạo ra một lớp bị ẩn (overlay) trên một trang web, với mục đích làm cho người dùng nhấp chuột vào vùng hiển thị của lớp đó mà không hề biết rằng họ đang thực hiện một hành động không mong muốn. Kẻ tấn công thường sử dụng các kỹ thuật như iframe, CSS và JavaScript để tạo ra lớp bị ẩn và đưa đối tượng hoặc liên kết độc hại vào trong vùng hiển thị đó. Điều này khiến cho thông tin của người sử dụng rơi vào trạng thái nguy hiểm nếu như thông tin cá nhân bị lộ ra hay thậm chí là tài khoản tín dụng bị đánh cắp.

Một số hậu quả nghiêm trọng mà Clickjacking đem lại như:

- Mất quyền kiểm soát: Clickjacking có thể khiến người dùng mất quyền kiểm soát trên trang web, khiến họ thực hiện những hành động không mong muốn mà không hề biết. Điều này có thể dẫn đến việc tải xuống phần mềm độc hại hoặc truy cập vào các trang web giả mạo.
- Mất thông tin cá nhân: Clickjacking có thể được sử dụng để thu thập thông tin cá nhân của người dùng, bao gồm tên đăng nhập và mật khẩu, số thẻ tín dụng và thông tin tài khoản khác.
- Rủi ro an ninh: Clickjacking có thể dẫn đến các rủi ro an ninh bảo mật, bao gồm mất tài khoản truy cập, tiết lộ thông tin quan trọng và bị tấn công bởi các phần mềm độc hại.
- Thiệt hại tài chính: Clickjacking có thể dẫn đến thiệt hại tài chính cho các tổ chức, bao gồm mất thông tin tài khoản và bị chiếm đoạt tài khoản để thực hiện các giao dịch không mong muốn.

1.7. Tấn công bằng CSRF (Cross-site Request Forgery)

CSRF hay Cross-site Request Forgery là một hình thức tấn công website thông qua việc sử dụng Cookies, buộc người dùng thực hiện những yêu cầu không mong muốn

bằng cách sử dụng quyền chứng thực người dùng đến một website giả mạo mà không cần sự cho phép của người dùng.

Cách thức hoạt động:

- Khi người dùng đăng nhập vào một trang web và nhận được một cookie đăng nhập. Cookie này chứa thông tin xác thực để cho phép người dùng truy cập vào các tài nguyên được bảo vệ trên trang web đó. Kẻ tấn công tạo ra một trang web độc hại và chèn vào đó một yêu cầu đến trang web đích thông qua thẻ ảnh, thẻ liên kết, thẻ script,…Và khi người dùng truy cập vào trang web độc hại và có tương tác nhất định trong trang web đó thì vô tình yêu cầu không mong muốn từ các tương tác đó được gửi đến trang web đích và được gửi đến máy chủ đích kèm theo Cookie đăng nhập được đính kèm trong yêu cầu. Vì máy chủ đích không thể phân biệt được đó có phải là yêu cầu hợp lệ của người dùng hay từ hacker nên sẽ thực hiện các yêu cầu được gửi lên.

Một số hậu quả của CSRF mang lại như:

- Đánh cắp thông tin: Kẻ tấn công có thể sử dụng các yêu cầu CSRF để đánh cắp thông tin cá nhân, chẳng hạn như tên đăng nhập, mật khẩu, thông tin thẻ tín dụng, và các thông tin khác.
- Thay đổi hoặc xóa dữ liệu: CSRF có thể được sử dụng để thay đổi hoặc xóa các dữ liệu quan trọng, chẳng hạn như thông tin tài khoản ngân hàng, các hồ sơ cá nhân, hoặc các tài liệu quan trọng khác.
- Phá hoại hệ thống: Kẻ tấn công có thể sử dụng các yêu cầu CSRF để phá hoại hệ thống bằng cách thực hiện các hành động độc hại, chẳng hạn như xóa bỏ các tệp quan trọng hoặc làm cho hệ thống không hoạt động được.
- Lây nhiễm mã độc: CSRF có thể được sử dụng để lây nhiễm mã độc vào hệ thống của người dùng, đe dọa tới tính bảo mật của toàn bộ hệ thống.

1.8. Tấn công bằng Brute Force Attack

Brute Force Attack (tấn công vét cạn) là một kỹ thuật tấn công mạng mà kẻ tấn công sử dụng phương pháp thử để tìm ra mật khẩu hoặc mã xác thực bằng cách thử nhiều khả năng khác nhau cho đến khi tìm ra mật khẩu hoặc mã xác thực đúng. Kỹ thuật này được sử dụng để tấn công các hệ thống đăng nhập bảo mật, nơi người dùng phải nhập mật khẩu hoặc mã xác thực để truy cập vào tài khoản của họ. Phương pháp tấn công này vô cùng đơn giản mà không phải sử dụng bất kì kỹ thuật thông minh nào. Dựa trên cơ sở là toán học và xoay vòng số nên để tấn công hiệu quả sẽ cần phải tốn một khoảng thời gian để mở mật khẩu. Số thời gian chỉ cần tính bằng giây và nhanh hơn rất rất nhiều lần so với bộ óc của con người.

Cách thức hoạt động:

- Kẻ tấn công sử dụng phần mềm hoặc script để tự động thực hiện các yêu cầu đăng nhập bằng cách sử dụng các danh sách từ điển hoặc thử tất cả các khả năng ký tự có thể có. Quá trình thử và sai này được lặp đi lặp lại đến khi tìm ra mật khẩu đúng để truy cập vào tài khoản.
- Cụ thể, kẻ tấn công sẽ đưa ra các giả định về mật khẩu hoặc mã xác thực, chẳng hạn như độ dài, kiểu ký tự, hoặc kết hợp giữa chữ và số. Sau đó, phần mềm hoặc script sẽ thực hiện thử các khả năng ký tự một cách tự động để đăng nhập vào tài khoản.
- Nếu mật khẩu hoặc mã xác thực không đúng, phần mềm sẽ tiếp tục thử lại với các khả năng ký tự khác cho đến khi tìm ra mật khẩu đúng hoặc đạt được số lần thử đăng nhập tối đa. Khi kẻ tấn công tìm ra mật khẩu chính xác, họ có thể sử dụng nó để truy cập vào tài khoản và thực hiện các hành động độc hại như đánh cắp thông tin cá nhân hoặc tài sản của người dùng.

6 loại tấn công Brute Force phổ biến

- Tấn công Simple Brute Force: Sử dụng một cách tiếp cận có hệ thống để “đoán” username hay password, không cần dựa vào external logic.

- Tấn công Hybrid Brute Force: Bắt đầu từ external logic để xác định các tổ hợp password có khả năng thành công cao nhất. Sau đó tiếp cận với Simple Brute Force Attack để thử nhiều tổ hợp nhất có thể.
- Tấn công Dictionary: Đoán username hoặc password bằng cách sử dụng một từ điển các xâu hay cụm từ khả thi.
- Tấn công Rainbow table: Rainbow Table là một bảng được tính toán trước để so khớp với kết quả của các hàm hash. Nó có thể dùng để đoán một hàm có độ dài xác định và chứa một tập hợp kí tự cụ thể.
- Tấn công Reverse Brute Force: Sử dụng một password chung hay một tập hợp các password để thử với nhiều username khả thi. Loại tấn công này nhắm vào một mạng người dùng mà các hacker đã lấy được dữ liệu trước đó.
- Tấn công Credential Snuffing: sử dụng các cặp password-username đã biết trước, và thử chúng trên nhiều website khác nhau. Sở dĩ vì có không ít người dùng có thói quen sử dụng cùng một cặp password-username trên nhiều hệ thống khác nhau.

Hậu quả của Brute Force Attack có thể rất nghiêm trọng và đáng lo ngại. Khi kẻ tấn công tìm ra mật khẩu chính xác, họ có thể sử dụng nó để truy cập vào tài khoản của người dùng và thực hiện các hành động độc hại như đánh cắp thông tin cá nhân hoặc tài sản của người dùng, tấn công các hệ thống khác, hoặc phá hoại hệ thống.

Một số hậu quả của Brute Force Attack bao gồm:

- Đánh cắp thông tin cá nhân: Khi kẻ tấn công truy cập vào tài khoản của người dùng, họ có thể truy cập vào các thông tin cá nhân như thông tin tài khoản ngân hàng, thông tin thẻ tín dụng, thông tin cá nhân, và các thông tin nhạy cảm khác.
- Tấn công hệ thống khác: Khi kẻ tấn công truy cập được vào một hệ thống, họ có thể sử dụng tài khoản của người dùng để tấn công các hệ thống khác và gây ra thiệt hại cho các tổ chức khác.
- Phá hoại hệ thống: Kẻ tấn công có thể xóa hoặc thay đổi dữ liệu quan trọng trong hệ thống, gây ra sự cố hệ thống và làm ảnh hưởng đến hoạt động của tổ chức.

2. PHƯƠNG ÁN KHẮC PHỤC, BIỆN PHÁP PHÒNG TRÁNH

2.1. Giới thiệu giao thức HTTPS:

HTTPS (Hypertext Transfer Protocol Secure) là một giao thức mạng được sử dụng để bảo mật truyền thông và truy cập web an toàn. HTTPS được xây dựng trên nền tảng của giao thức HTTP, với sự bổ sung của một lớp bảo mật SSL/TLS (Secure Sockets Layer/Transport Layer Security) để đảm bảo an toàn trong quá trình truyền tải dữ liệu giữa các trình duyệt và máy chủ web. Khi một trình duyệt truy cập một trang web được bảo mật bằng HTTPS, trình duyệt sẽ xác thực chứng chỉ SSL/TLS và bắt đầu một phiên kết nối bảo mật với máy chủ web. Trong quá trình này, dữ liệu sẽ được mã hóa để đảm bảo rằng các thông tin nhạy cảm như mật khẩu, thông tin thẻ tín dụng và các thông tin khác không bị đánh cắp hoặc đánh giá thấp bởi các kẻ tấn công trên mạng.

Trong một thế giới ngày càng phụ thuộc vào Internet, việc bảo vệ dữ liệu trở thành một nhu cầu thiết yếu. HTTPS cung cấp nhiều lợi ích cho người dùng và các tổ chức trên mạng, bao gồm:

- Bảo mật dữ liệu: HTTPS đảm bảo rằng dữ liệu được mã hóa và an toàn trong quá trình truyền tải trên mạng, giúp ngăn chặn các kẻ tấn công trên mạng có thể đánh cắp thông tin và sử dụng sai mục đích.
- Tăng uy tín: Sử dụng HTTPS cung cấp một bằng chứng cho người dùng rằng trang web mà họ đang truy cập là chính xác và đáng tin cậy, giúp tăng độ tin cậy và tăng cường niềm tin của khách hàng vào tổ chức.
- Đáp ứng các yêu cầu bảo mật: Nhiều tổ chức, đặc biệt là các tổ chức tài chính, y tế cần tuân thủ các yêu cầu bảo mật nghiêm ngặt. Việc sử dụng HTTPS giúp đáp ứng các yêu cầu này, giúp các tổ chức tuân thủ các quy định và đảm bảo rằng thông tin của khách hàng được bảo vệ tốt nhất có thể.

Sử dụng HTTPS không chỉ được áp dụng cho các trang web bán hàng hoặc các tổ chức tài chính. Thực tế, mọi trang web đều nên sử dụng HTTPS để đảm bảo bảo mật dữ liệu của người dùng. Một số ứng dụng của HTTPS bao gồm:

- Thương mại điện tử: Các trang web bán hàng trực tuyến cần sử dụng HTTPS để đảm bảo an toàn cho thông tin của khách hàng khi họ thanh toán.
- Dịch vụ ngân hàng trực tuyến: Các trang web cung cấp dịch vụ ngân hàng trực tuyến cần sử dụng HTTPS để đảm bảo an toàn cho thông tin tài khoản của khách hàng.
- Các trang web chia sẻ thông tin nhạy cảm: Các trang web chia sẻ thông tin nhạy cảm như thông tin y tế hoặc thông tin cá nhân cần sử dụng HTTPS để đảm bảo an toàn cho thông tin này.
- Các trang web của chính phủ: Các trang web của chính phủ cần sử dụng HTTPS để đảm bảo an toàn cho thông tin liên lạc và thông tin nhạy cảm khác của người dân.

Trong tổng thể, HTTPS là một công nghệ rất quan trọng trong việc đảm bảo an toàn và bảo mật trên mạng. Việc sử dụng HTTPS không chỉ giúp bảo vệ thông tin của người dùng mà còn giúp các tổ chức tuân thủ các yêu cầu bảo mật nghiêm ngặt và tăng độ tin cậy của khách hàng đối với các trang web.

2.2. Cách phòng tránh SQL Injection:

Để phòng chống SQL Injection, có một số biện pháp cần được áp dụng:

- Sử dụng các công nghệ bảo mật như WAF (Web Application Firewall) để chặn các tấn công SQL Injection.
- Sử dụng các hàm bảo mật SQL như mysqli_real_escape_string() hoặc PDO để xử lý các truy vấn SQL và tránh bị chèn thêm các câu lệnh độc hại.
- Sử dụng cơ chế xác thực đầu vào (input validation) để đảm bảo rằng dữ liệu nhập vào là hợp lệ và không chứa các ký tự độc hại.

- Hạn chế quyền truy cập đến cơ sở dữ liệu, chỉ cho phép người dùng được truy cập đến các bảng và cột cần thiết.
- Sử dụng các phương pháp mã hóa dữ liệu như băm (hashing) hoặc mã hóa (encryption) để bảo vệ dữ liệu nhạy cảm trong cơ sở dữ liệu.

Kết luận, SQL Injection là một lỗ hổng bảo mật nghiêm trọng trong các ứng dụng web, có thể gây ra những thiệt hại lớn cho dữ liệu và hệ thống của ứng dụng web. Để phòng ngừa tấn công SQL Injection, cần thực hiện các biện pháp bảo mật như kiểm tra và sàng lọc đầu vào của người dùng, sử dụng các phương pháp mã hóa, áp dụng các bản vá bảo mật và thực hiện các kiểm tra bảo mật thường xuyên.

2.3. Cách phòng tránh XSS:

Để khắc phục XSS, các nhà phát triển web nên thực hiện các biện pháp sau:

- Kiểm tra và lọc dữ liệu nhập vào từ người dùng.
- Sử dụng các thư viện bảo mật để kiểm tra và lọc dữ liệu nhập vào.
- Sử dụng mã hóa HTML để chuyển đổi các ký tự đặc biệt thành các thẻ HTML tương ứng.
- Sử dụng HTTP-only cookie để ngăn chặn việc đánh cắp cookie.
- Cập nhật các bản vá bảo mật mới nhất để bảo vệ khỏi các lỗ hổng bảo mật đã biết.

Kết luận, XSS là một lỗ hổng bảo mật nghiêm trọng trong các ứng dụng web, có thể gây ra những thiệt hại lớn cho dữ liệu và hệ thống của ứng dụng web. Để phòng ngừa tấn công XSS, cần thực hiện các biện pháp bảo mật như kiểm tra và sàng lọc đầu vào của người dùng, sử dụng các phương pháp mã hóa, sử dụng các thư viện bảo mật và thực hiện các kiểm tra bảo mật thường xuyên.

2.4. Cách phòng tránh Security Misconfiguration:

Để ngăn chặn lỗ hổng Security Misconfiguration, cần phải thực hiện các biện pháp bảo mật đầy đủ và đảm bảo rằng cấu hình hệ thống và ứng dụng đang được quản lý chặt chẽ. Các biện pháp cần thực hiện bao gồm:

- Kiểm tra và cập nhật các phần mềm và hệ điều hành định kỳ để đảm bảo sử dụng phiên bản mới nhất.
- Chỉ cung cấp quyền truy cập và chức năng cần thiết cho người dùng và ứng dụng.
- Đảm bảo rằng các thông tin nhạy cảm được mã hóa trên hệ thống.
- Tắt các chức năng và dịch vụ không sử dụng và loại bỏ các giá trị mặc định.
- Thiết lập các chính sách bảo mật và kiểm tra cấu hình định kỳ để phát hiện lỗ hổng.
- Giám sát các hoạt động trên hệ thống để phát hiện và ngăn chặn các hành động độc hại.

Kết luận, Security Misconfiguration là một lỗ hổng bảo mật nghiêm trọng trong các ứng dụng web, có thể gây ra những thiệt hại lớn cho dữ liệu và hệ thống của ứng dụng web. Để phòng ngừa lỗ hổng này, cần thực hiện các biện pháp bảo mật như cài đặt các bản vá bảo mật, cấu hình an ninh cho hệ thống và các ứng dụng web, và thực hiện các kiểm tra bảo mật thường xuyên.

2.5. Cách phòng tránh các cuộc tấn công DdoS

Thực tế, dù hiểu rõ đặc điểm của DDoS là gì nhưng bất cứ ai cũng có thể trở thành nạn nhân của các cuộc tấn công mạng kiểu này. Việc ngăn chặn DDoS một cách tuyệt đối là không thể ở thời điểm hiện tại. Dù vậy, trong trường hợp không may mắn bị tin tặc nhắm đến, hãy tham khảo một số biện pháp xử lý sau:

- Sử dụng các giải pháp bảo mật mạng như tường lửa, IDS/IPS để ngăn chặn các cuộc tấn công từ các nguồn không xác định.
- Tăng cường chịu tải cho hệ thống như tối ưu hóa mã nguồn, tối ưu hóa cơ sở dữ liệu, sử dụng bộ nhớ cache.
- Loại bỏ các thiết bị máy tính nhiễm mã độc.
- Sử dụng các dịch vụ, phần mềm chống DDoS Attack (như Cloudflare, Akamai, Radware, Arbor Networks, F5 Networks).

- Liên lạc ngay với nhà cung cấp Internet (ISP) để nhận được những tư vấn, hướng dẫn về giải pháp tối ưu nhất.
- Liên hệ ngay với nhà cung cấp host để họ tạo "black hole" (lỗ đen), giúp hút traffic cho đến khi vấn đề được giải quyết và giảm thiệt hại có thể gánh chịu.
- Tìm đến các chuyên gia trong lĩnh vực để có được những lời khuyên hữu ích, giúp khắc phục hậu quả triệt để và lâu dài.

Tóm lại, DDoS là một kiểu tấn công mạng trong đó tin tặc sử dụng một loạt thiết bị điều khiển từ xa để gửi lưu lượng truy cập đến máy chủ hoặc mạng. Mục đích của cuộc tấn công này là làm quá tải máy chủ hoặc mạng để nó không thể xử lý các yêu cầu của người dùng.

DDoS có thể tàn phá mạng của bạn, từ việc gián đoạn dịch vụ cho đến các mối đe dọa về tính toàn vẹn của dữ liệu. Vì vậy, việc phòng chống DDoS là rất quan trọng để đảm bảo an toàn thông tin và sự ổn định của hệ thống mạng.

Phòng chống DDoS có thể đơn giản như giới hạn số lượng kết nối từ một địa chỉ IP hoặc sử dụng các công nghệ bảo mật mạng tiên tiến hơn như tường lửa và hệ thống phát hiện xâm nhập. Tuy nhiên, việc ngăn chặn DDoS không hề dễ dàng vì tin tặc có thể tấn công mạng của bạn và đánh lừa hệ thống bảo vệ của bạn theo nhiều cách. Vì vậy, nâng cao kiến thức và kỹ năng an ninh mạng của bạn là rất quan trọng để giảm thiểu rủi ro DDoS.

2.6. Cách phòng tránh các cuộc tấn công Clickjacking

Mặc dù Clickjacking gây ra rất nhiều thiệt hại nghiêm trọng tuy nhiên việc phòng chống Clickjacking rất dễ dàng thực hiện

- Yêu cầu người dùng xác nhận lại bằng cách hiển thị hộp thoại thông báo thao tác người dùng đã thực hiện yêu cầu và xác nhận.
- Đặt các đối tượng web vào các vị trí ngẫu nhiên gây khó khăn cho kẻ tấn công vì giao diện không ổn định.

- Thiết lập các chính sách trên trình duyệt yêu cầu các frame hiển thị với opacity > 0
- Sử dụng Javascript ngăn cản một trang web khác nhúng nội dung của trang đó vào iframe
- Sử dụng câu lệnh điều kiện kiểm tra trang web có nằm trong iframe hay không
- Chuyển hướng cửa sổ trình duyệt về trang web bị nhúng và iframe.

Tóm lại, Clickjacking là một kỹ thuật lừa đảo phổ biến trong việc tấn công các trang web. Khi bị clickjacking, người dùng sẽ bị đưa vào tình trạng click vào những nút bấm hoặc liên kết mà họ không muốn hoặc không biết rõ về chức năng của chúng. Kết quả là, họ có thể bị mất thông tin cá nhân hoặc gặp các vấn đề bảo mật nghiêm trọng.

Để tránh clickjacking, các nhà phát triển web cần áp dụng các biện pháp bảo vệ thích hợp, bao gồm sử dụng header X-Frame-Options để giới hạn khung của trang web chỉ được hiển thị trong một số tên miền cụ thể và sử dụng Content Security Policy (CSP) để kiểm soát việc tải lại các tài nguyên từ các nguồn không đáng tin cậy.

2.7. Cách phòng tránh các cuộc tấn công CSRF

Phía User

Để tránh trở thành nạn nhân của các cuộc tấn công CSRF nên thực hiện một số lưu ý sau:

- Nên đăng xuất khỏi các website quan trọng: Tài khoản ngân hàng, thanh toán trực tuyến, các mạng xã hội, gmail… khi đã thực hiện xong giao dịch.
- Nên login vào một máy riêng và không cho người thứ 2 tiếp xúc với máy đó.
- Không nên click vào các đường dẫn mà bạn nhận được qua email, qua facebook … Khi bạn đưa chuột qua 1 đường dẫn, phía dưới bên trái của trình duyệt thường có địa chỉ website đích, bạn nên lưu ý để đến đúng trang mình muốn.
- Không lưu các thông tin về mật khẩu tại trình duyệt của mình. Không nên chọn các phương thức "đăng nhập lần sau", "lưu mật khẩu" …

- Trong quá trình thực hiện giao dịch hay vào các website quan trọng không nên vào các website khác, có thể chứa các mã khai thác của kẻ tấn công.

Phía Server

Cho đến nay vẫn chưa có biện pháp nào có thể phòng chống triệt để CSRF. Sau đây là một vài kĩ thuật sử dụng.

- Sử dụng captcha, các thông báo xác nhận: Captcha được sử dụng để nhận biết đối tượng đang thao tác với hệ thống là con người hay không. Các thao tác quan trọng như "đăng nhập" hay là "chuyển khoản", "thanh toán" thường là hay sử dụng captcha. Những chức năng quan trọng như reset mật khẩu, xác nhận thay đổi info của account cũng nên gửi url qua email đã đăng ký để người dùng có thể click vào xác nhận.
- Sử dụng csrf_token: token này sẽ thay đổi liên tục trong phiên làm việc, và khi thay đổi thông tin gửi kèm thông tin token này. Nếu token được sinh ra và token được gửi lên không trùng nhau thì loại bỏ request.
- Sử dụng cookie riêng biệt cho trang admin: Nên để trang quản trị ở một subdomain riêng để chúng không dùng chung cookies với front end của sản phẩm.
- Kiểm tra IP: Một số hệ thống quan trọng chỉ cho truy cập từ những IP được thiết lập sẵn, hoặc chỉ cấp phép truy cập quản trị qua IP local hoặc VPN.

Tóm lại CSRF (Cross-Site Request Forgery) là một kỹ thuật tấn công mạng phổ biến nhằm vào các ứng dụng web, đặc biệt là những ứng dụng có tính năng quản lý tài khoản và thanh toán trực tuyến. Tấn công này sử dụng việc lừa người dùng truy cập vào một trang web độc hại, từ đó thực hiện các yêu cầu trái phép tới các trang web khác, gây ra các hậu quả nghiêm trọng như đánh cắp thông tin tài khoản, tiền tệ hoặc thực hiện các hành động độc hại.

Để phòng chống tấn công CSRF, cần sử dụng các biện pháp như sử dụng mã định danh phiên và CSRF token, sử dụng HTTP method POST, tắt cookie của một trang web

trong trình duyệt của người dùng, cài đặt các cơ chế bảo mật như CSP và kiểm tra và xác thực dữ liệu nhập vào từ người dùng.

Tuy nhiên, việc triển khai các biện pháp phòng chống CSRF không thể đảm bảo 100% an toàn cho các ứng dụng web. Do đó, người dùng cần cẩn trọng khi truy cập vào các trang web không rõ nguồn gốc hoặc không tin cậy, không chia sẻ thông tin tài khoản và mật khẩu trên các trang web không được xác thực và luôn cập nhật các phần mềm bảo mật để bảo vệ máy tính của mình khỏi các mối đe dọa của tấn công mạng.

2.8. Cách phòng tránh các cuộc tấn công Bruce Force Attack

Để phòng chống Brute Force Attack chúng ta cần đảm bảo sử dụng các mật khẩu có tính bảo mật cao:

- Không sử dụng thông tin có thể dễ dàng tìm được trên mạng (chẳng hạn như họ, tên).
- Càng nhiều ký tự càng tốt.
- Kết hợp số, chữ cái, ký tự đặc biệt.
- Sử dụng password khác nhau cho các tài khoản.
- Tránh lặp lại một khuôn mẫu chung.

Đối với quản trị viên, sau đây là một số cách để bảo vệ người dùng khỏi tấn công Brute Force:

- Lockout privacy: Có thể khóa các tài khoản sau nhiều lần đăng nhập thất bại và unlock lại sau này.
- Delay: Khóa tài khoản sau một số lần đăng nhập thất bại nhất định, đồng thời làm cho thời gian delay sau mỗi lần đăng nhập dài hơn.
- Captcha: Các công cụ như reCAPTCHA yêu cầu người dùng hoàn thành các tác vụ đơn giản để có thể đăng nhập vào hệ thống. Mặc dù người dùng thực có thể dễ dàng hoàn thành, những công cụ brute force attack sẽ không thể nào làm được.

- Yêu cầu mật khẩu mạnh: Admin có thể bắt buộc người dùng sử dụng mật khẩu đủ dài và đủ phức tạp. Bên cạnh đó, ta cũng có thể yêu cầu người dùng thay đổi mật khẩu định kỳ.
- Xác thực hai yếu tố: Sử dụng nhiều yếu tố khác nhau để xác thực danh tính người dùng và cấp quyền truy cập vào tài khoản.

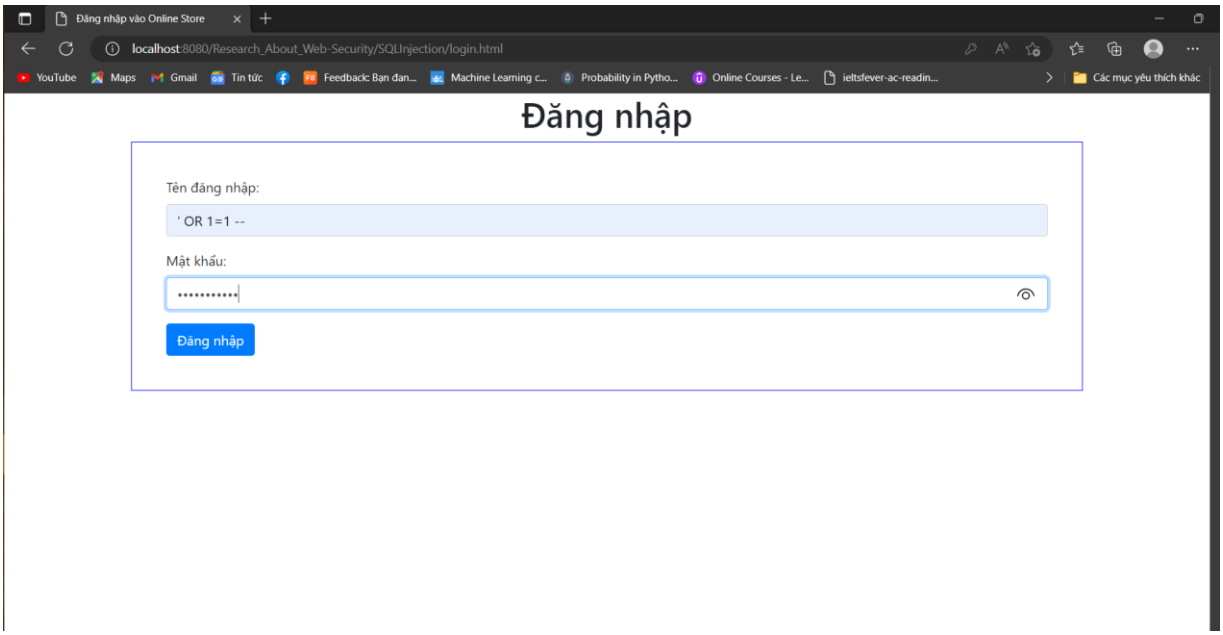
Tóm lại, Brute Force Attack là một trong những phương pháp tấn công thường được sử dụng để đánh cắp thông tin và xâm nhập vào các hệ thống của người dùng hoặc tổ chức. Phương pháp này dựa trên việc thử các mật khẩu khác nhau cho đến khi tìm ra mật khẩu đúng để truy cập vào tài khoản.

Brute Force Attack có thể gây ra những hậu quả nghiêm trọng như đánh cắp thông tin cá nhân, tấn công các hệ thống khác và phá hoại hệ thống. Vì vậy, để bảo vệ tài khoản của mình hoặc tổ chức, người dùng cần sử dụng các biện pháp bảo mật như đặt mật khẩu phức tạp, sử dụng các cơ chế xác thực hai yếu tố và đổi mật khẩu thường xuyên. Đồng thời, các tổ chức cần triển khai các biện pháp bảo mật để ngăn chặn các cuộc tấn công Brute Force và giữ an toàn cho hệ thống của mình.

3. TRIỂN KHAI MỘT SỐ WEBSITE PHÒNG TRÁNH CÁC LỖI BẢO MẬT THƯỜNG GẶP

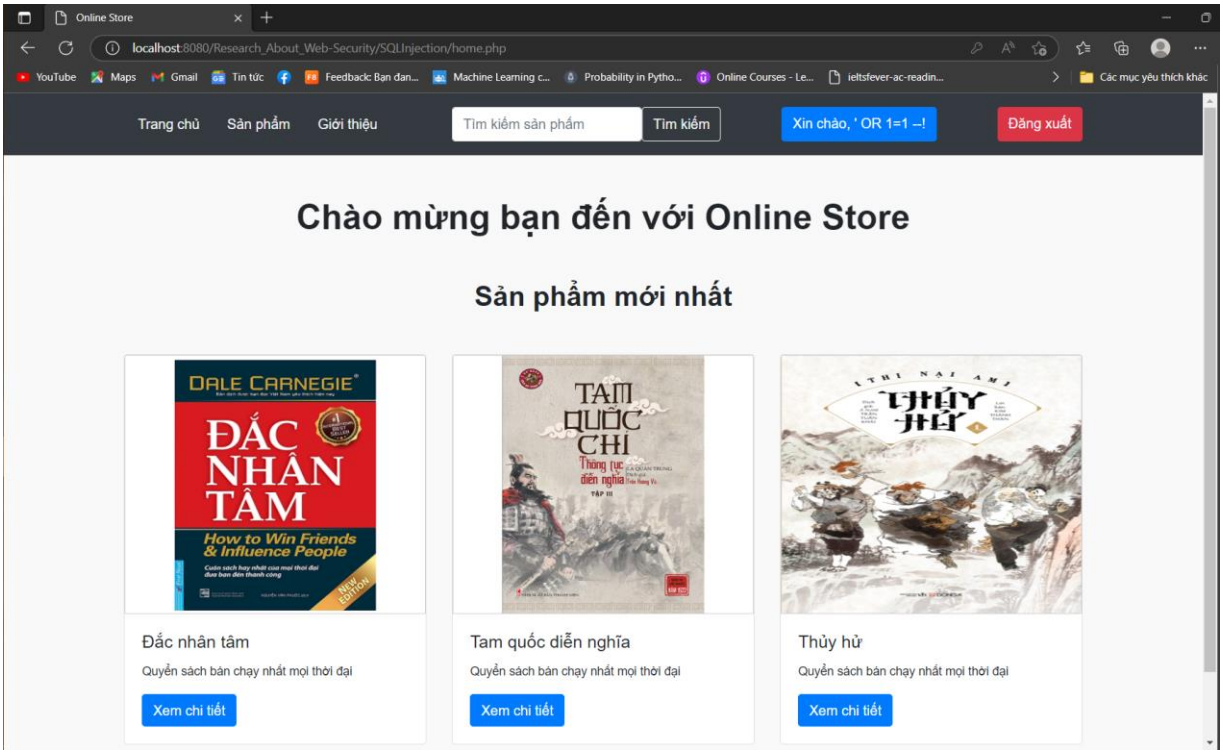
3.1. Trang web bị tấn công SQL Injection và cách khắc phục

Mô tả tình huống: Một trang web bán hàng dưới đây bị tấn công SQL Injection, hacker lợi dụng cơ sở dữ liệu bị tất công bằng cách để tên đăng nhập và mật khẩu đều là “' OR 1=1 --”. Sau đó kẻ này đăng nhập thành công vào hệ thống trang chủ và có thể đánh cắp một vài thông tin khác.



Hình 3.1.1 Màn hình đăng nhập

Sau khi nhấn đăng nhập, hacker đã vào được hệ thống:



Hình 3.1.2 Giao diện trang chủ sau khi bị hacker xâm nhập

Xác định lí do bị tấn công SQL Innjection: do trang login.php yêu cầu đăng nhập truyền đến file action.php, mà action.php sử dụng dữ liệu được truyền từ form login.php để truy vấn cơ sở dữ liệu mà không có bất kỳ kiểm tra hoặc xử lý bảo mật nào. Điều này đã dẫn đến lỗ hổng bảo mật SQL Injection.

```
<body>
    <div class="container">
        <h1>Đăng nhập</h1>
        <div class="margin">
            <form method="post" action="action.php">
                <div class="form-group">
                    <label for="username">Tên đăng nhập:</label>
                    <input type="text" class="form-control" name="username" placeholder="Enter username">
                </div>
                <div class="form-group">
                    <label for="password">Mật khẩu:</label>
                    <input type="password" class="form-control" name="password" placeholder="Enter password">
                </div>
                <button type="submit" class="btn btn-primary">Đăng nhập</button>
            </form>
        </div>

    </div>
```

Hình 3.1.3 Mã nguồn Form đăng nhập

SQLInjection > action.php

```php
<?php
    // Lấy thông tin đăng nhập từ form
    $username = $_POST['username'];
    $password = $_POST['password'];

    // Kết nối đến cơ sở dữ liệu
    $conn = mysqli_connect('localhost', 'root', '', 'SQL_Injection');

    // Kiểm tra thông tin đăng nhập
    $query = "SELECT * FROM users WHERE username='$username' AND password='$password'";
    // die($query);
    $result = mysqli_query($conn, $query);

    session_start();
    if(mysqli_num_rows($result) > 0) {
      // Đăng nhập thành công
      $_SESSION['username'] = $username;
      header("Location: home.php");
      exit();
    } else {
      // Đăng nhập thất bại
      echo "Invalid username or password.";
      $_SESSION['username'] = null; // hoặc unset($_SESSION['username']);
      header("Location: login.php");
      exit();
    }

    // Đóng kết nối
    mysqli_close($conn);
?>
```

Hình 3.1.4 Mã nguồn xử lý form đăng nhập

Câu lệnh truy vấn trên không kiểm tra bảo mật mà dùng trực tiếp biến $username và $password được lấy từ form login.html, và lúc này khi hacker chèn các ký tự đặc biệt vào các trường dữ liệu đăng nhập, như dấu nháy đơn ('), chúng có thể thực hiện các câu truy vấn SQL không mong muốn. Ví dụ, nếu kẻ tấn công chèn chuỗi ' OR 1=1 -- vào trường đăng nhập tên đăng nhập, câu truy vấn SQL sẽ trở thành:

```
SELECT * FROM users WHERE username= '' OR 1=1 --' AND password = '';
```

Hình 3.1.5 Câu lệnh SQL bị tấn công

Trong trường hợp này, điều kiện kiểm tra tên đăng nhập sẽ trả về giá trị luôn đúng (1=1), điều này có nghĩa là câu truy vấn SQL sẽ trả về tất cả các bản ghi trong bảng users, cho phép kẻ tấn công đọc được thông tin người dùng và thậm chí có thể thực hiện các hành động không mong muốn trên cơ sở dữ liệu như thêm, sửa, xóa dữ liệu.

Vì vậy để tăng tính bảo mật cho trang web và không bị tấn công bằng SQL Injection, chúng em sử dụng file solve.php:

SQLInjection > solve.php

```php
<?php
    // Lấy thông tin đăng nhập từ form
    $username = $_POST['username'];
    $password = $_POST['password'];

    // Kết nối đến cơ sở dữ liệu
    $dsn = 'mysql:host=localhost;dbname=SQL_Injection';
    $user = 'root';
    $pass = '';
    $options = array(
      PDO::ATTR_ERRMODE => PDO::ERRMODE_EXCEPTION
    );
    try {
      $conn = new PDO($dsn, $user, $pass, $options);
    } catch (PDOException $e) {
      echo "Kết nối cơ sở dữ liệu thất bại: " . $e->getMessage();
      exit();
    }
    // Kiểm tra thông tin đăng nhập
    $stmt = $conn->prepare("SELECT * FROM users WHERE username=:username AND password=:password");
    $stmt->bindParam(':username', $username);
    $stmt->bindParam(':password', $password);
    $stmt->execute();

    session_start();
    if($stmt->rowCount() > 0) {
      // Đăng nhập thành công
      $_SESSION['username'] = $username;
      header("Location: home.php");
      exit();
    } else {
      // Đăng nhập thất bại
      $_SESSION['username'] = null;
      header("Location: login.php?error=1");
      exit();
    }
    // Đóng kết nối
    $conn = null;
?>
```

Hình 3.1.6 Mã nguồn phòng tránh SQL Injection sử dụng PDO

Trong file solve.php này, chúng em sử dụng PDO và prepared statements để bảo vệ ứng dụng khỏi lỗ hổng SQL Injection. Các câu truy vấn được thực hiện bằng cách sử dụng các biến định danh và các tham số thực hiện chuẩn bị trước đó. Việc bindParam sẽ tránh được tình trạng dòng chuỗi ' OR 1=1 -- chạy vào câu truy vấn. Lúc này hacker sẽ không thể đăng nhập vào hệ thống và phá hoại thông tin. Lúc này ở form login.php, chúng ta sẽ đổi action là solve.php

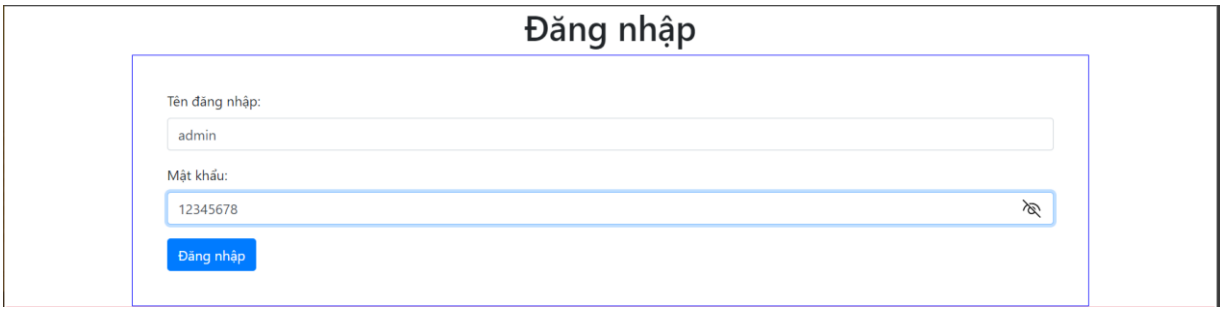
```
<body>
    <div class="container">
        <h1>Đăng nhập</h1>
        <div class="margin">
            <form method="post" action="solve.php">
                <div class="form-group">
                    <label for="username">Tên đăng nhập:</label>
                    <input type="text" class="form-control" name="username" placeholder="Enter username">
                </div>
                <div class="form-group">
                    <label for="password">Mật khẩu:</label>
                    <input type="password" class="form-control" name="password" placeholder="Enter password">
                </div>
                <button type="submit" class="btn btn-primary">Đăng nhập</button>
            </form>
        </div>

    </div>
    <?php
    if(isset($_GET['error']) && $_GET['error'] == 1) {
        echo "<div class='alert alert-danger' role='alert'>Đăng nhập thất bại. Vui lòng kiểm tra lại tên đăng nhập và mật khẩu.</div>";
    }
    ?>
    <!-- Thêm đường dẫn JavaScript của Bootstrap -->
    <script src="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/js/bootstrap.min.js"></script>
</body>
```

Hình 3.1.7 Thay đổi lại form đăng nhập

Ta tiến hành đăng nhập lại:

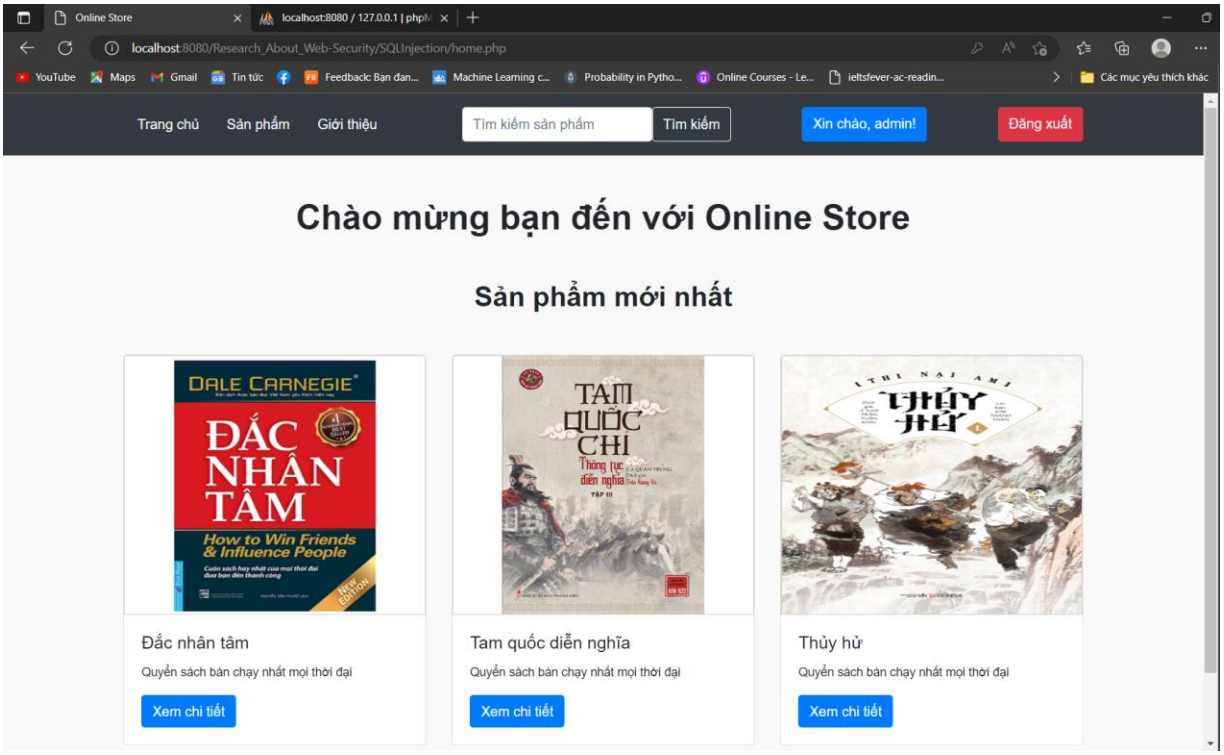
Hình 3.1.8 Hacker không thể tấn công bằng cách đăng nhập vào trang web

Phải dùng đúng tài khoản tồn tại trong database để đăng nhập:



Hình 3.1.9 Dùng tài khoản đã tồn tại để đăng nhập

Thông báo đăng nhập thành công và chuyển đến trang chủ:

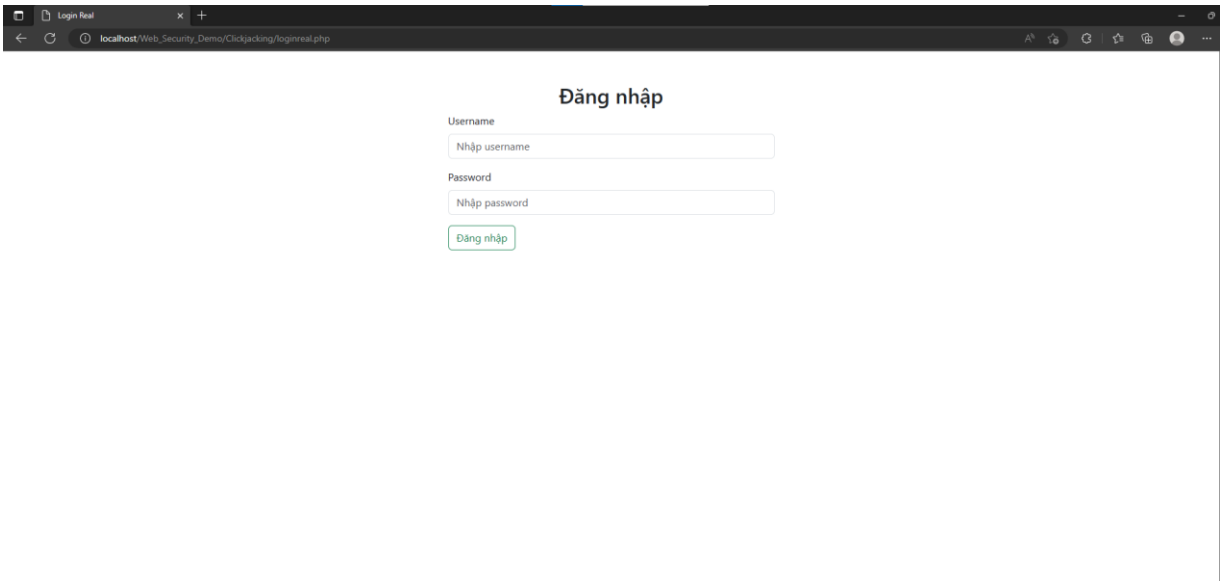


Hình 2.1.10 Đăng nhập bằng admin thành công

Kết luận, ta thấy hacker không thể dùng SQL Injection để đăng nhập vào hệ thống nữa, vậy là ta đã bảo mật được trang web này. Tuy nhiên đây chỉ là một tình huống cơ bản, vẫn có nhiều cách khác để hacker tấn công vào trang web này vì vậy chúng ta phải tăng cường thêm nhiều biện pháp để tăng tính bảo mật cho trang web trên.

3.2. Triển khai trang web bị tấn công Clickjacking và cách khắc phục

Ở đây chúng em tạo ra một trang web login với giao diện sau:

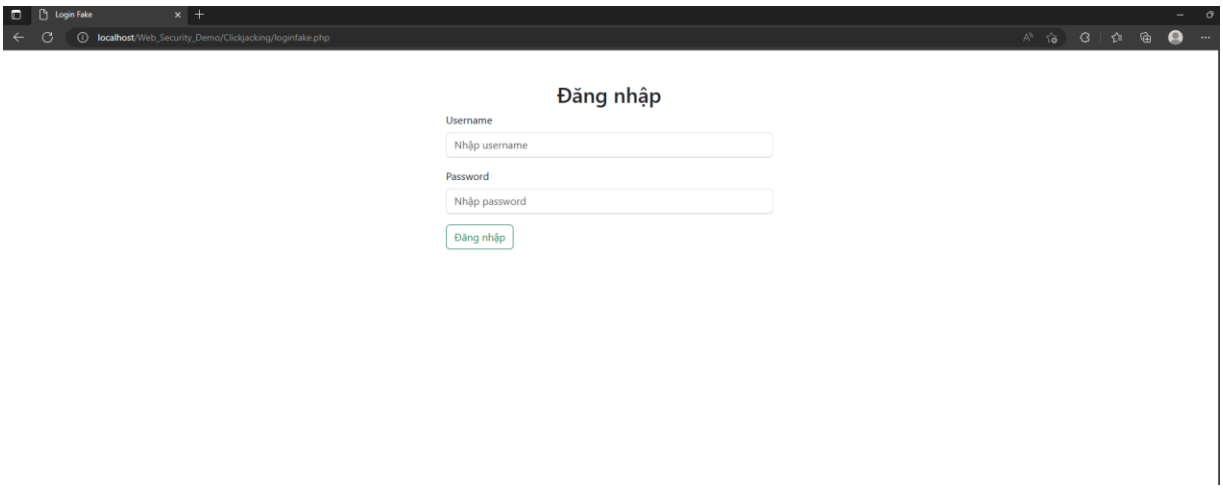


Hình 3.2.1 Giao diện trang web đăng nhập thật

Và em tạo ra một trang web giả mạo loginfake với cặp thẻ iframe

```
<iframe src="loginreal.php" width="100%" height="500px"></iframe>
```

Cặp thẻ này sẽ lấy giao diện của trang login thật và đưa vào trang login giả mạo với giao diện như sau:



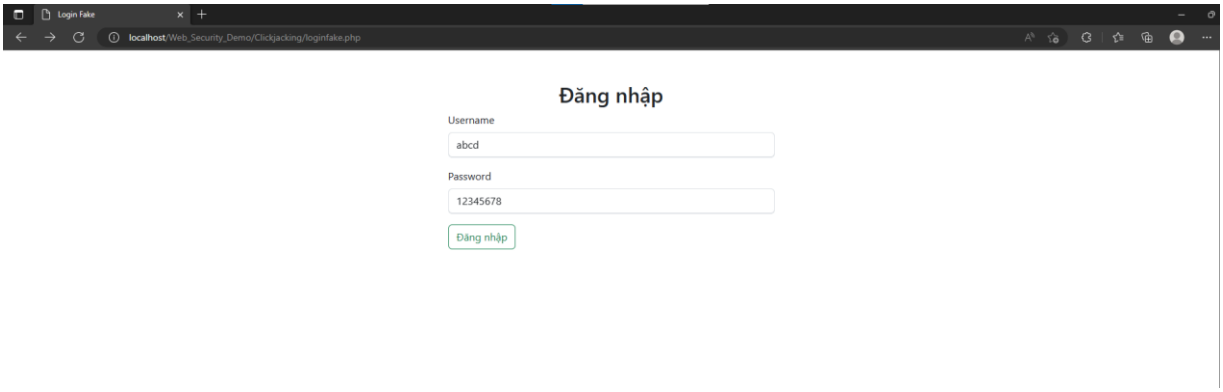
Hình 3.2.2 Giao diện trang web đăng nhập giả mạo

Tuy nhiên trong giao diện giả mạo này em đã tạo nên 2 cặp thẻ input username và password đè lên cặp thẻ username và password thật và đặt trong một thẻ form để gửi yêu cầu đến trang đích mà chúng em mong muốn

```
<body>
    <iframe src="loginreal.php" width="100%" height="500px"></iframe>
    <div class="form-container">
        <form action="democlickjacking.php" method="get">
            <input type="text" class="form-control username" id="exampleFormControlInput1" name="username" placeholder="Nhập username">
            <input type="text" class="form-control password" id="exampleFormControlInput1" name="password" placeholder="Nhập password">
            <button type="submit" class="btn btn-outline-success login">Đăng nhập</button>
        </form>
    </div>
</body>
```

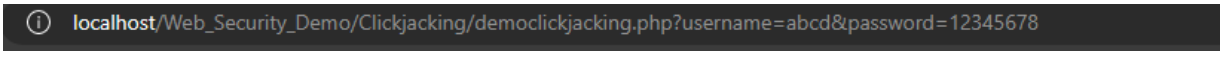
Hình 3.2.3 Đoạn mã nguồn dùng thẻ iframe và overlay thẻ để đánh lừa người dùng

Sau khi dụ người dùng truy cập vào trang login giả mạo và người dùng nhập thông tin tài khoản thì tất cả thông tin sẽ bị lộ và có thể bị đánh cắp dữ liệu hoặc các thông tin quan trọng



Hình 3.2.4 Thông tin đăng nhập khi người dùng nhập vào

Sau khi nhấn vào đăng nhập thông tin trong form sẽ được gửi đến trang đích mà hacker mong muốn, trong trường hợp này đó là trang democlickjacking.php



Hình 3.2.5 Thông tin username và password bị lộ

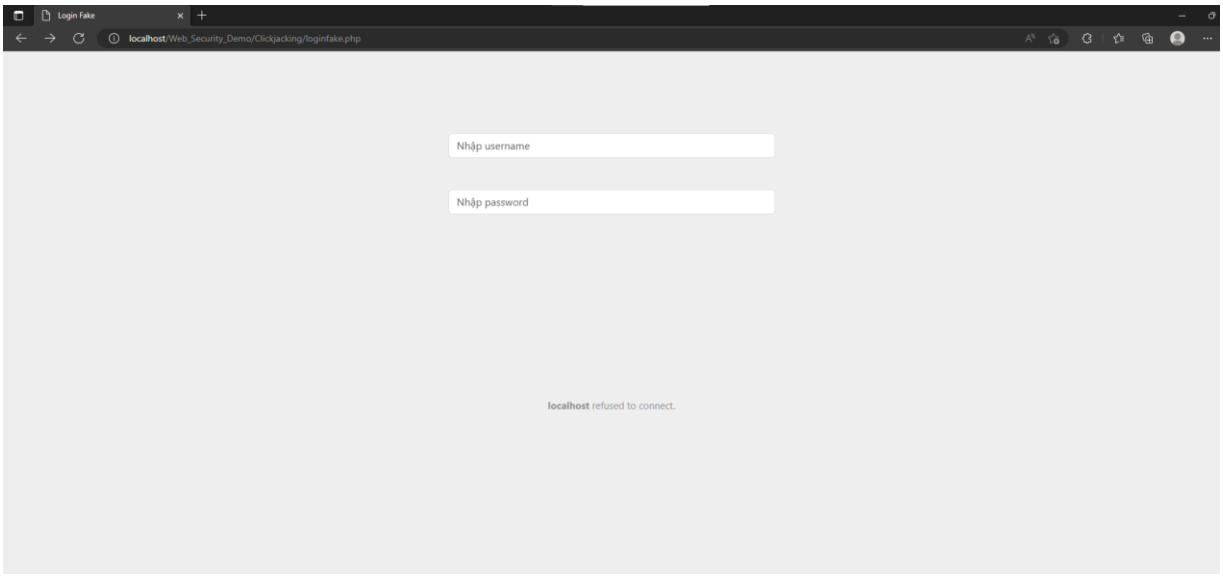
Như chúng ta có thể thấy, thông tin của username và password đã bị lộ trên đường dẫn url và bị hacker lấy được dữ liệu.

Để phòng trách Clickjacking rất đơn giản, chúng em chỉ cần thêm một cặp thẻ là có thể ngăn chặn các trang web giả mạo khác.

```php
<?php
    header("X-Frame-Options: DENY");
?>
```

Hình 3.2.6 Đoạn mã nguồn khắc phục tình trạng Clickjacking

Kết quả sau khi cặp thẻ này:

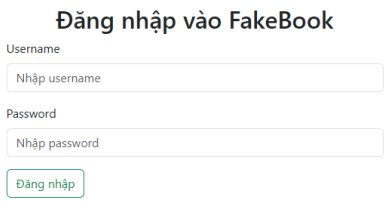


Hình 3.2.7 Kết quả sau khi dùng đoạn mã nguồn để khắc phụ Clickjacking

Có thể thấy được giao diện của trang login thật không thể bị giả mạo từ đó người dùng có thể nhận biết được trang web mình truy cập vào là không chính xác từ đó tránh được tình trạng bị đánh cắp tài khoản.

3.3. Triển khai trang web bị tấn công CSRF và cách khắc phục

Đầu tiên, em sẽ tạo ra một trang Fakebook news nhỏ mà người dùng có thể đăng trạng thái mỗi khi đăng nhập vào trang và cần đăng nhập để có thể truy cập vào



Hình 3.3.1 Giao diện trang đăng nhập vào FakeBook

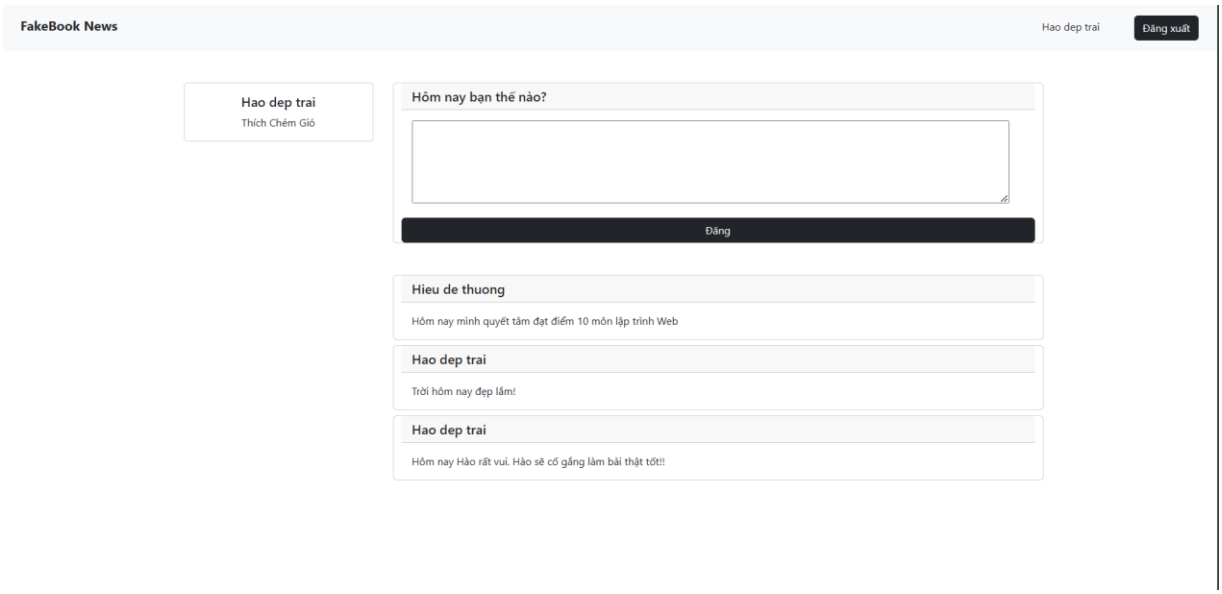
Tại đây người dùng cần nhập thông tin tài khoản và mật khẩu để có thể truy cập vào FakeBook. Sau khi đăng nhập thành công sẽ hiển thị giao diện FakeBook News để người dùng có thể chia sẻ trạng thái, cảm xúc của mình.

```
<form action="" method="post">
    <div class="row mb-5">
        <div class="card">
                <h5 class="card-header">Hôm nay bạn thế nào?</h5>
                <div class="card-body">
                    <textarea name="post-content" cols="120" rows="5"></textarea>
                </div>
                <button type="submit" name="post_news" class="btn btn-dark">Đăng</button>
        </div>
    </div>
</form>
```

Hình 3.3.2 Form đăng post khi chưa có token

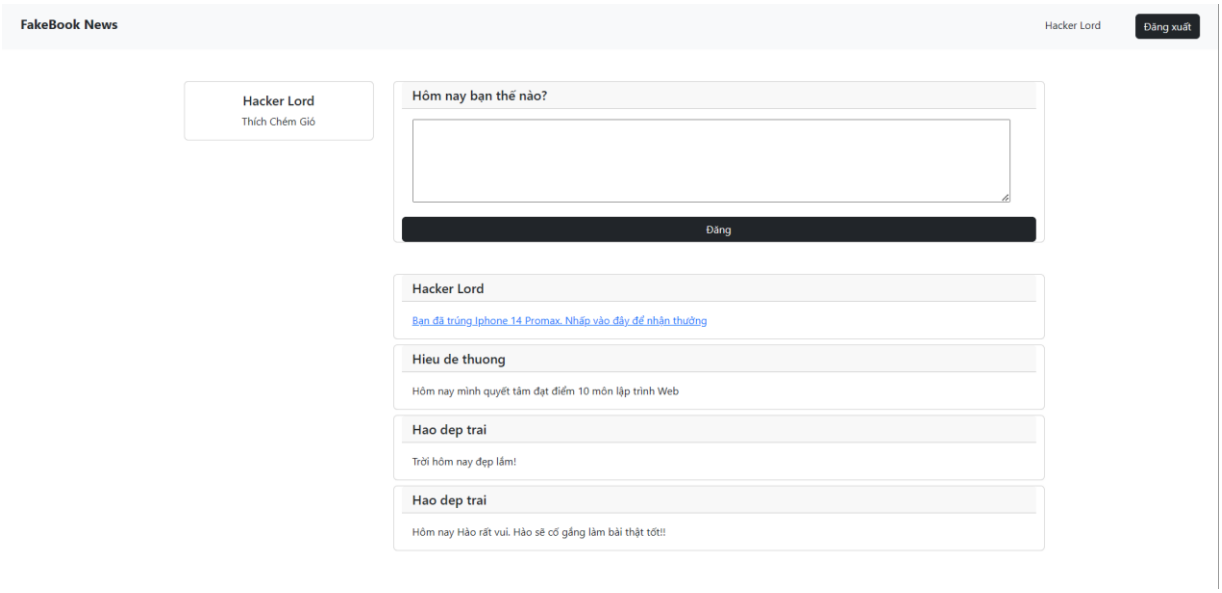
```
//Tạo post chưa có token
public function add_post($usname, $post_content){
    global $conn;
    $stm = $conn->prepare("insert into posts(username,post_content) values(?,?)");
    $stm->bind_param('ss',$usname, $post_content);
    $stm->execute();
}
```

Hình 3.3.3 Mã nguồn tạo post khi chưa có token



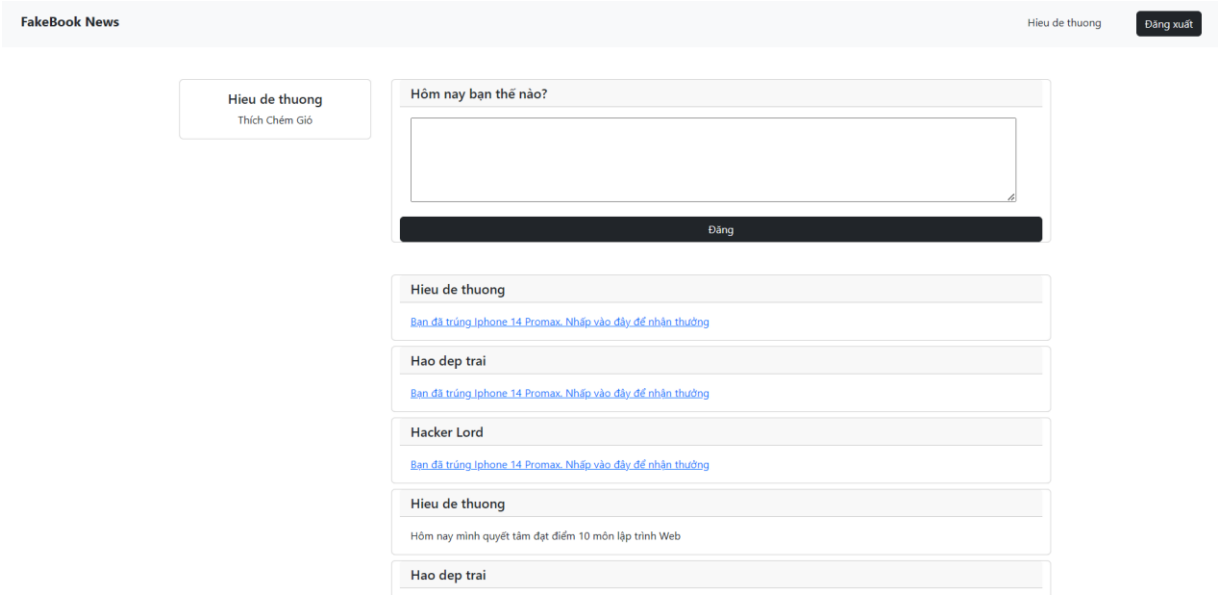
Hình 3.3.4 Giao diện của trang web FakeBook

Tuy nhiên trong một hôm gió táp phong ba trái gió trở trời, một tên Hacker Lord đã truy cập vào trang web là đăng lên một đường link với nội dung là “Bạn đã trúng Iphone 14 Promax. Hãy nhấp vào đây để nhận thưởng”.



Hình 3.3.5 Hình ảnh đường link bị hacker phát tán

Nhẹ dạ cả tin nhiều người dùng đã nhấn vào link trang web và từ đó phát tán mã độc nguy hiểm mà không hề hay biết.



Hình 3.3.6 Hình ảnh các đường link bị phát tán

Để khắc phục tình trạng này em đã tạo ra một token và thêm vào mỗi khi đăng bài post mới, và kiểm tra xem nếu token được gửi lên form có trùng khớp với token được lưu trong session hay không, nếu trùng khớp thì bài được đăng lên, nếu không sẽ được cảnh báo bị tấn công CSRF.

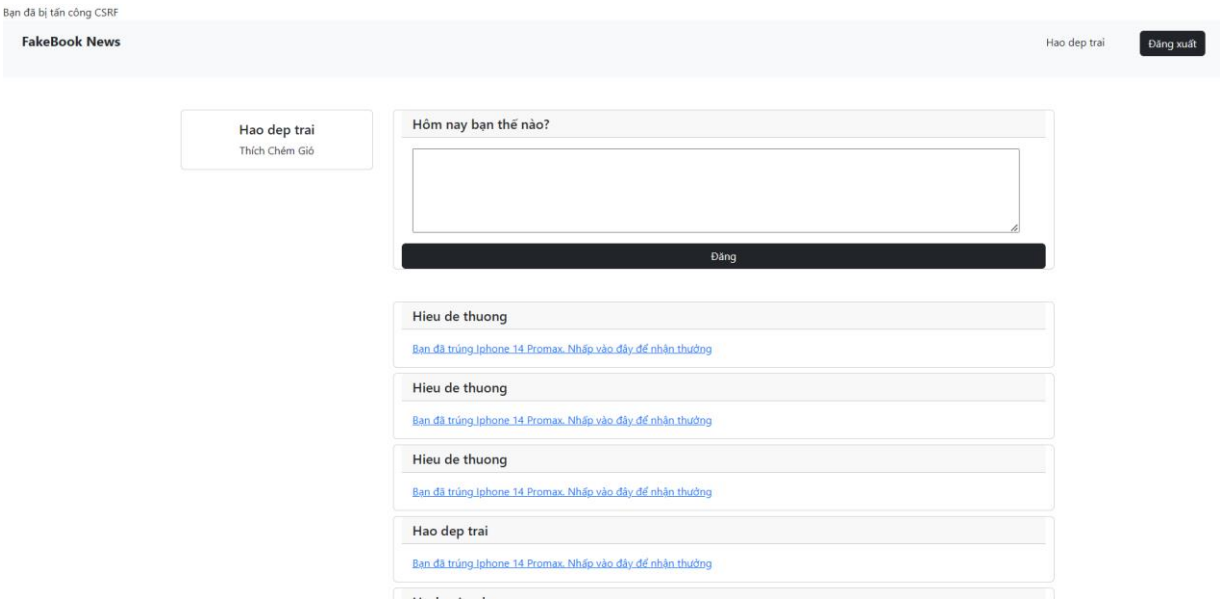
```
<form action="" method="post">
    <div class="row mb-5">
        <div class="card">
                <h5 class="card-header">Hôm nay bạn thế nào?</h5>
                <div class="card-body">
                    <textarea name="post-content" cols="120" rows="5"></textarea>
                    <input type="hidden" name="token" value='<?php echo $_SESSION['csrf_token'] ?>'>
                </div>
                <button type="submit" name="post_news" class="btn btn-dark">Đăng</button>
        </div>
    </div>
</form>
```

Hình 3.3.7 Form sau khi đã thêm thẻ hidden chứa giá trị token

```php
//tạo post khi có token
public function add_post($usname, $post_content,$ses){
    global $conn;
    $_SESSION['csrf_token'] = $ses;
    if(($_POST['token'])==$_SESSION['csrf_token']){
        $stm = $conn->prepare("insert into posts(username,post_content) values(?,?)");
        $stm->bind_param('ss',$usname, $post_content);
        $stm->execute();
    }
    else{
        echo "Bạn đã bị tấn công CSRF";
    }
}
```

Hình 3.3.8 Mã nguồn tạo post khi đã có token

Sau khi nhấp vào đường link sẽ hiện ra cảnh báo và đường link không còn được phát tán nữa



Hình 3.3.9 Hình cảnh báo khi nhấp vào đường link

Vậy là chúng em đã phần nào giải quyết 1 phần nhỏ kỹ thuật tấn công CSRF, tuy nhiên người dùng cần cẩn thận hơn trong việc click vào những đường link không chính xác hay những url không chính thống.

DANH MỤC TÀI LIỆU THAM KHẢO

Tiếng Anh

[1]. Stuttard, D., & Pinto, M. (2011). The Web Application Hacker's Handbook: Finding and Exploiting Security Flaws. Wiley.

[2]. Open Web Application Security Project. (2021). OWASP Top Ten Project. https://owasp.org/Top10/

[3]. Seitz, J. (2014). Black Hat Python: Python Programming for Hackers and Pentesters. No Starch Press.

[4]. Scambray, J., Shema, M., & Sima, C. (2010). Hacking Exposed Web Applications, Third Edition. McGraw-Hill.

[5]. Sullivan, B., & Liu, V. (2011). Web Application Security: A Beginner's Guide. McGraw-Hill.

[6]. Zalewski, M. (2011). The Tangled Web: A Guide to Securing Modern Web Applications. No Starch Press.

--Hết--