

**BỘ GIÁO DỤC VÀ ĐÀO TẠO
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ ĐÔNG Á
KHOA: CÔNG NGHỆ THÔNG TIN**



BÁO CÁO THỰC TẬP

HỌC PHẦN: THỰC TẬP KỸ THUẬT

**ĐỀ TÀI: Nghiên cứu và triển khai tường lửa ModSecurity
trên máy chủ Web Apache Server**

Sinh viên thực hiện	Lớp	Khóa
Đinh Khắc Hoạt	DCCNTT11.10.3	11

Bắc Ninh, năm 2024

**BỘ GIÁO DỤC VÀ ĐÀO TẠO
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ ĐÔNG Á
KHOA: CÔNG NGHỆ THÔNG TIN**

BÁO CÁO THỰC TẬP

HỌC PHẦN: Thực tập kỹ thuật

Nhóm: 02

**Đề tài: Nghiên cứu và triển khai tường lửa ModSecurity
trên máy chủ Web Apache Server**

STT	Sinh viên thực hiện	Mã sinh viên	Điểm bằng số	Điểm bằng chữ
1	Đình Khắc Hoạt	20200184		

CÁN BỘ CHẤM 1

(Ký và ghi rõ họ tên)

CÁN BỘ CHẤM 2

(Ký và ghi rõ họ tên)

Mục lục

Mở đầu	1
Chương I: Cơ sở lý thuyết và phát biểu bài toán	3
1.1. Cơ sở lý thuyết của đề tài	3
1.1.1 Thông tin	3
1.1.2 Hệ thống và tài sản hệ thống	3
1.1.3 An toàn thông tin.....	3
1.1.4 Những mối đe dọa tới an toàn thông tin	3
1.1.5 Mục tiêu của an toàn thông tin.....	4
1.2. Phát biểu bài toán	5
1.2.1 Những phương pháp đảm bảo an toàn thông tin.....	5
1.2.2 Đảm bảo an toàn thông tin bằng thiết bị Firewall.....	6
Chương II: Máy chủ Web Apache và lỗ hổng website	7
2.1. Máy chủ Web Apache	7
2.2. Những lỗ hổng trên máy chủ web Apache	8
2.2.1 Tấn công từ chối dịch vụ (DoS).....	8
2.2.2 SQL Injection (SQLi).....	12
2.2.3 Cross Site Scripting (XSS).....	15
Chương III: Tường lửa ứng dụng web ModSecurity	17
3.1 Giới thiệu chung về tường lửa	17
3.1.1 Khái niệm tường lửa	17
3.1.2 Phân loại và tính năng	17
3.2 Tường lửa modSecurity	19
3.2.1 Lịch sử phát triển	19
3.2.2 Chức năng chính	20
3.2.3 Nguyên lý hoạt động của ModSecurity.....	21
3.2.4 Cấu trúc đặt quy tắc (rule).....	23
3.2.5 Cài đặt tường lửa ModSecurity	28
Chương IV: Bảo vệ máy chủ Web Apache bằng tường lửa ModSecurity	33
4.1 Công cụ sử dụng	33
4.1.1 DVWA	33
4.1.2 Kali linux.....	33
4.2 Tấn công và phòng thủ DOS	34

4.2.1 Tấn công DoS bằng công cụ slowloris.....	34
4.2.2 Tạo rule bảo vệ.....	36
4.3 Tấn công và phòng thủ SQL Injection.....	38
4.3.1 Tấn công SQLi	38
4.3.2 Tạo rule bảo vệ.....	43
4.4 Tấn công và phòng thủ XSS.....	45
4.4.1 Tấn công XSS	45
4.4.2 Tạo rule bảo vệ.....	49
Kết luận	51
Tài liệu tham khảo	51

Danh mục bảng biểu

Bảng 1 Đánh giá lỗ hổng SQL Injection	12
Bảng 2 Đánh giá lỗ hổng Cross-Site Scripting (XSS).....	15
Bảng 3 Mô tả thành phần của quy tắc ModSecurity	24
Bảng 4 Mô tả dạng dữ liệu của ModSec Rule.....	25
Bảng 5 Mô tả giá trị biến (Variable) của ModSec Rule	26
Bảng 6 Mô tả toán tử chuỗi của ModSec Rule.....	26
Bảng 7 Mô tả toán tử số học của ModSec Rule	27
Bảng 8 Mô tả hành động (Action) của ModSec Rule	27
Bảng 9 Mô tả hàm chuyển đổi của ModSec Rule	28
Bảng 10 Mô tả ModSec Rule chặn IP	37
Bảng 11 Mô tả ModSec Rule chặn tấn công SQLi	44
Bảng 12 Mô tả ModSec Rule chặn tấn công XSS.....	49

Danh mục hình ảnh

Hình 1 Cấu trúc bảo mật thông tin	4
Hình 2 Mô hình hoạt động Apache Web Server	7
Hình 3 Mô tả tấn công DoS (1)	9
Hình 4 Mô tả tấn công DoS (2)	10
Hình 5 Mô tả tấn công DOS (3)	10
Hình 6 Mô tả tấn công DoS (4)	11
Hình 7 Quy trình thực hiện của quy tắc trong ModSec.....	22
Hình 8 Kiểm tra phiên bản ubuntu	28
Hình 9 Tiến trình cập nhật ubuntu.....	29
Hình 10 Cài đặt ModSecurity2.....	29
Hình 11 Tùy chỉnh tệp cấu hình (1).....	30
Hình 12 Tùy chỉnh tệp cấu hình (2).....	30
Hình 13 Khởi động lại apache server	30
Hình 14 Kiểm tra cấu hình Modsecurity	31
Hình 15 Tùy chỉnh Core Rule Set (1).....	31
Hình 16 Tùy chỉnh Core Rule Set (2).....	31
Hình 17 Tùy chỉnh Core Rule Set (3).....	32
Hình 18 Tùy chỉnh Core Rule Set (4).....	32
Hình 19 Damn Vulnerable Web Application (DVWA).....	33
Hình 20 Kali linux	34
Hình 21 Mô phỏng tấn công DoS (1)	34
Hình 22 Mô phỏng tấn công DoS (2)	35
Hình 23 Mô phỏng tấn công DoS (3)	35
Hình 24 Kết quả tấn công Dos	36
Hình 25 File access.log.....	37
Hình 26 Cấu hình Rule Block IP	37
Hình 27 Kết quả thực thi	37
Hình 28 Mô phỏng tấn công SQL Injection (1)	38

Hình 29 Mô phỏng tấn công SQL Injection (2)	39
Hình 30 Mô phỏng tấn công SQL Injection (3)	39
Hình 31 Mô phỏng tấn công SQL Injection (4)	40
Hình 32 Mô phỏng tấn công SQL Injection (5)	40
Hình 33 Mô phỏng tấn công SQL Injection (6)	41
Hình 34 Mô phỏng tấn công SQL Injection (7)	42
Hình 35 Mô phỏng tấn công SQL Injection (8)	42
Hình 36 Hậu quả tấn công SQL Injection	43
Hình 37 Cấu hình Rule chặn SQL Injection.....	44
Hình 38 Kiểm thử SQL Injection	44
Hình 39 Kết quả kiểm thử	45
Hình 40 Mô phỏng tấn công XSS (1).....	45
Hình 41 Mô phỏng tấn công XSS (2).....	45
Hình 42 Mô phỏng tấn công XSS (3).....	46
Hình 43 Mô phỏng tấn công XSS (4).....	46
Hình 44 Mô phỏng tấn công XSS (5).....	47
Hình 45 Mô phỏng tấn công XSS (6).....	47
Hình 46 Mô phỏng tấn công XSS (7).....	48
Hình 47 Mô phỏng tấn công XSS (8).....	48
Hình 48 Cấu hình Rule chặn	50
Hình 49 Kiểm thử XSS.....	50
Hình 50 Kết quả kiểm thử	50

Danh mục chữ viết tắt

STT	Chữ viết tắt	Ý nghĩa
1	ModSec	Viết tắt của tường lửa ModSecurity
2	IDS - Intrusion Detection System	Hệ thống phát hiện xâm nhập
3	WAF Web Application Firewall	Tường lửa ứng dụng web
4	Dos Denial of Service	Tấn công từ chối dịch vụ
5	URL - Uniform Resource Locator	Hệ thống định vị tài nguyên hệ thống
6	OWASP - Open Web Application Security Project	Dự án bảo mật ứng dụng web mã nguồn mở

Mở đầu

Hiện nay các lĩnh vực trong công nghệ thông tin ngày càng được phát triển và mở rộng. Không chỉ trong quy mô quốc gia mà trên toàn cả quốc tế. Việc ứng dụng những thành tựu của công nghệ thông tin vào sản xuất và cuộc sống hằng ngày đã không thể thiếu trong thời buổi hiện nay. Cùng với việc mở rộng và ứng dụng của công nghệ thông tin cũng đã xuất hiện những hình thức tội phạm mới, tội phạm trên không gian mạng hay tội phạm số, lừa đảo qua mạng.

Xuất phát từ những mối đe dọa về tội phạm công nghệ cao này, ngành an ninh mạng đã được hình thành. Trong ngành này các kỹ sư hay các nhà khoa học không ngừng nghiên cứu và phát triển những tuyến phòng thủ để chống lại những cuộc tấn công từ không gian mạng. Có thể thấy Việt Nam ta cũng quan tâm và thực hiện một số biện pháp để giảm thiểu và ngăn ngừa những mối đe dọa này. Vào ngày 12 tháng 6 năm 2018, luật an ninh mạng đã được quốc hội nước Việt Nam thông qua và ban hành. Từ đó đến nay luật đã có thêm nhiều bổ sung và sửa đổi để phù hợp với xu hướng phát triển hiện tại.

Ngoài nhà nước, những doanh nghiệp tư nhân cũng thực hiện phòng chống những mối đe dọa và các cuộc tấn công bằng cách trang bị những thiết bị chuyên dụng như firewall để đảm bảo an toàn mạng hay hệ thống kiểm soát xâm nhập (IDS-Intrusion Detection System). Ngoài ra còn thực hiện đào tạo an ninh mạng cho nhân viên nâng cao nhận thức về những mối nguy hiểm trên mạng.

Trên mạng đang có rất nhiều sản phẩm của công nghệ thông tin, điển hình ta có thể nhận thấy rõ nhất là những website. Những website đang trở nên thông dụng và phổ biến nó cũng trở thành một mục tiêu để cho tội phạm mạng khai thác và tấn công. Vì website của một doanh nghiệp hay nhà nước luôn được công khai ngoài internet rộng lớn. Những mối đe dọa là không thể lường trước.

Từ những mối đe dọa và nhu cầu cho một thiết bị, một ứng dụng có thể giảm thiểu rủi ro bảo mật cho một website. Những tường lửa ứng dụng Web (WAF-Web Application Firewall) ra đời, nó có thể là thiết bị phần cứng được cài đặt trong hệ thống mạng hoặc một phần mềm chạy trên Web Server.

Xuất phát từ nhu cầu và ứng dụng thực tiễn của một tường lửa website. Thêm vào đó em đang đang học môn học thực tập kỹ thuật. Em xin giành thời gian của môn học này để thực hiện đề tài nghiên cứu và triển khai hệ thống tường lửa web dựa trên firewall mã nguồn mở ModSecurity. Đây cũng là lần đầu làm đề tài về hệ thống thông tin nên bài làm sẽ còn nhiều thiếu sót mong được sự chỉ bảo của thầy hướng dẫn môn “Nguyễn Đức Thiện” để có thể hoàn thành bài làm và hoàn thành môn học một cách chính chu và hoàn thiện nhất.

Em xin chân thành cảm ơn thầy!

Chương I: Cơ sở lý thuyết và phát biểu bài toán

1.1. Cơ sở lý thuyết của đề tài

1.1.1 Thông tin

Thông tin là một bộ phận quan trọng và là tài sản thuộc quyền sở hữu của các tổ chức. Sự thiệt hại và lạm dụng thông tin không chỉ ảnh hưởng đến người sử dụng hoặc các ứng dụng mà nó còn gây ra các hậu quả tai hại cho toàn bộ tổ chức đó. Thêm vào đó sự ra đời của Internet đã giúp cho việc truy cập thông tin ngày càng trở nên dễ dàng hơn.

1.1.2 Hệ thống và tài sản hệ thống

Hệ thống là một tập hợp các máy tính bao gồm các thành phần, phần cứng, phần mềm và dữ liệu làm việc được tích lũy qua thời gian.

Hệ thống thông tin bao gồm:

- Phần cứng
- Phần mềm
- Dữ liệu
- Các truyền thông giữa các máy tính của hệ thống
- Môi trường làm việc
- Con người

1.1.3 An toàn thông tin

An toàn thông tin là hành động ngăn cản, phòng ngừa sự sử dụng, truy cập, tiết lộ, chia sẻ, phát tán, ghi lại hoặc phá hủy thông tin chưa có sự cho phép. Ngày nay vấn đề an toàn thông tin được xem là một trong những quan tâm hàng đầu của xã hội, có ảnh hưởng rất nhiều đến hầu hết các ngành khoa học tự nhiên, kỹ thuật, khoa học xã hội và kinh tế.

1.1.4 Những mối đe dọa tới an toàn thông tin

Có 3 hình thức chủ yếu đe dọa đối với hệ thống:

Phá hoại: kẻ thù phá hỏng thiết bị phần cứng hoặc phần mềm hoạt động trên hệ thống.

Sửa đổi: Tài sản của hệ thống bị sửa đổi trái phép. Điều này thường làm cho hệ thống không làm đúng chức năng của nó. Chẳng hạn như thay đổi mật khẩu, quyền người dùng trong hệ thống làm họ không thể truy cập vào hệ thống để làm việc.

Can thiệp: Tài sản bị truy cập bởi những người không có thẩm quyền. Các truyền thông thực hiện trên hệ thống bị ngăn chặn, sửa đổi.

Các đe dọa đối với một hệ thống thông tin có thể đến từ ba loại đối tượng như sau:

- Các đối tượng từ ngay bên trong hệ thống (insider), đây là những người có quyền truy cập hợp pháp đối với hệ thống.
- Những đối tượng bên ngoài hệ thống (hacker, cracker), thường các đối tượng này tấn công qua những đường kết nối với hệ thống như Internet chẳng hạn.
- Các phần mềm chạy trên hệ thống.

1.1.5 Mục tiêu của an toàn thông tin

Ba mục tiêu chính của an toàn bảo mật thông tin:



Hình 1 Cấu trúc bảo mật thông tin

Tính bí mật (Confidentiality): Đảm bảo rằng thông tin không bị truy cập bất hợp pháp

- Thuật ngữ *privacy* thường được sử dụng khi dữ liệu được bảo vệ có liên quan tới các thông tin mang tính cá nhân.

Tính toàn vẹn (Integrity): Đảm bảo rằng thông tin không bị sửa đổi bất hợp pháp.

Tính sẵn dùng (availability): Tài sản luôn sẵn sàng được sử dụng bởi những người có thẩm quyền.

Thêm vào đó sự chính xác của thông tin còn được đánh giá bởi:

Tính xác thực (Authentication): Đảm bảo rằng dữ liệu nhận được chắc chắn là dữ liệu gốc ban đầu

Tính không thể chối bỏ (Non-repudiation): Đảm bảo rằng người gửi hay người nhận dữ liệu không thể chối bỏ trách nhiệm sau khi đã gửi và nhận thông tin.

1.2. Phát biểu bài toán

1.2.1 Những phương pháp đảm bảo an toàn thông tin

Để đảm bảo an toàn thông tin, yêu cầu sự kết hợp giữa các biện pháp kỹ thuật và không kỹ thuật. Một số biện pháp đảm bảo an toàn thông tin:

- **Chính sách bảo mật:** phát triển và thực thi các chính sách bảo mật chi tiết, bao gồm quy định về quản lý tài khoản, mật khẩu, quản lý rủi ro và quản lý quyền truy cập.
- **Bảo vệ dữ liệu:** sử dụng các biện pháp bảo vệ dữ liệu như mã hoá, tokenization và hashing để bảo vệ dữ liệu quan trọng khỏi việc truy cập trái phép.
- **Quản lý rủi ro:** xác định, đánh giá và quản lý các rủi ro bảo mật thông tin trong tổ chức và thực hiện các biện pháp để giảm thiểu các rủi ro này.
- **Cập nhật phần mềm:** thường xuyên cập nhật và vá các phần mềm và hệ điều hành để khắc phục các lỗ hổng bảo mật mới và giảm thiểu nguy cơ từ các cuộc tấn công.
- **Firewall và tường lửa mạng:** triển khai những firewall và tường lửa mạng để kiểm soát lưu lượng mạng và ngăn chặn các mối đe dọa từ bên ngoài.
- **Kiểm tra và theo dõi:** thực hiện các hoạt động kiểm tra và theo dõi thường xuyên để phát hiện và phản ứng nhanh chóng đối với các hoạt động không bình thường và những mối đe dọa mới.
- **Giáo dục và đào tạo:** đào tạo nhân viên về các nguy cơ bảo mật thông tin và các biện pháp bảo vệ và thúc đẩy ý thức về an toàn thông tin trong toàn bộ tổ chức.

- **Sao lưu và khôi phục dữ liệu:** thực hiện sao lưu dữ liệu thường xuyên và xây dựng kế hoạch phục hồi dữ liệu để đảm bảo khả năng phục hồi sau sự cố.
- **Kiểm tra định kỳ những lỗ hổng bảo mật:** thực hiện kiểm tra bảo mật thứ ba định kỳ để xác định các lỗ hổng bảo mật và đề xuất biện pháp cải thiện.
- **Tuân thủ pháp luật:** tuân thủ các quy định và yêu cầu pháp lý liên quan đến bảo mật thông tin, bao gồm việc bảo vệ dữ liệu cá nhân và tuân thủ các quy định về bảo mật thông tin doanh nghiệp.

1.2.2 Đảm bảo an toàn thông tin bằng thiết bị Firewall

Firewall (tường lửa) là một phần quan trọng trong việc đảm bảo an toàn thông tin trên mạng. Nó ngày càng phổ biến ngay cả trong những doanh nghiệp lớn, vừa và nhỏ. Những tính năng và chức năng của chúng để thực hiện đảm bảo an toàn thông tin:

- **Cấu hình các quy tắc:** xác định các quy tắc và quản lý lưu lượng mạng. Bao gồm địa chỉ IP, cổng, giao thức và loại dữ liệu nào cho phép hoặc bị từ chối.
- **Quản lý và theo dõi lưu lượng mạng:** theo dõi lưu lượng mạng đến và đi từ máy chủ, mạng nội bộ và Internet để phát triển và ngăn chặn các hoạt động không mong muốn hoặc không an toàn.
- **Tường lửa ứng dụng (Application Firewall)** Tường lửa ứng dụng có khả năng kiểm tra lưu lượng ở tầng ứng dụng và có thể phát hiện và ngăn chặn các tấn công như SQL Injection, Cross-Site Scripting (XSS) và các loại tấn công khác dựa trên nội dung.
- **Tạo mạng DMZ (Demilitarized Zone):** xây dựng một mạng DMZ để cô lập các dịch vụ công khai khỏi các mạng nội bộ nhạy cảm. Sử dụng firewall để kiểm soát truy cập ra và vào vùng DMZ.
- **Tích hợp với các hệ thống khác:** Firewall có thể tích hợp với những hệ thống khác để nâng cao hiệu quả phát hiện những mối đe dọa và bảo vệ hệ thống mạng khỏi các cuộc tấn công.

Chương II: Máy chủ Web Apache và lỗi hỏng website

2.1. Máy chủ Web Apache

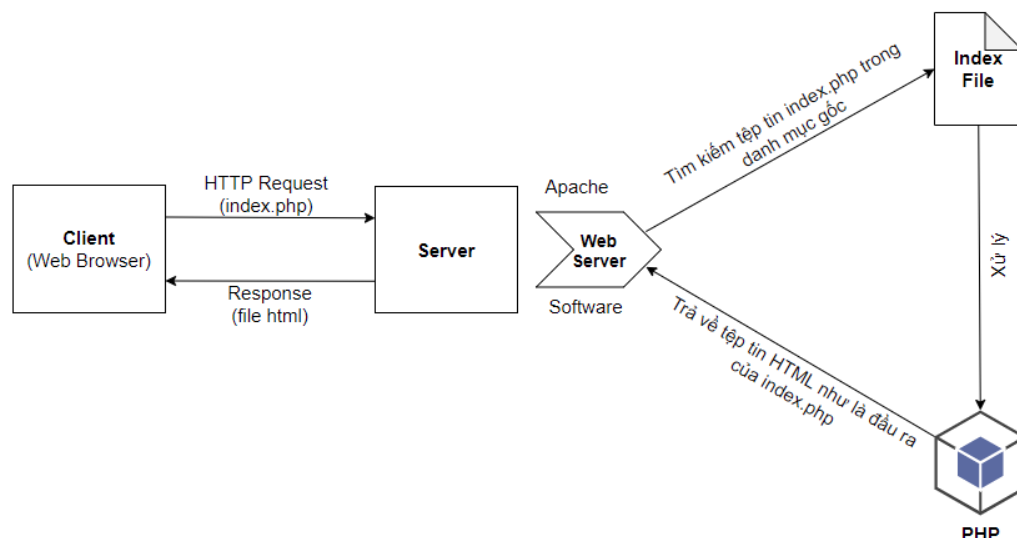
Apache là phần mềm web server miễn phí mã nguồn mở, chiếm tới 46% thị phần website trên toàn thế giới. Tên của Apache là Apache HTTP Server, được điều hành và phát triển bởi Apache Software Foundation.

Web Server là server vật lý hay là một máy chủ ảo cài phần mềm chuyên biệt để cung cấp dịch vụ Word Wide Web. Nhiệm vụ là đưa website lên internet. Để thực hiện, nó sẽ là trung gian giữa Server và client. Nó sẽ thực hiện truyền tải nội dung từ Server khi nhận được yêu cầu của client và hiển thị nó dưới hình thức là một website.

Web server xử lý các file dưới ngôn ngữ lập trình như là PHP, Python, Java, ... Nhiệm vụ quan trọng nhất của web server là điều hướng dữ liệu nhiều người dùng cùng một lúc để mỗi người với mỗi yêu cầu khác nhau sẽ ứng với trang web khác nhau.

Từ những ngôn ngữ lập trình website sẽ được hiển thị thành các file HTML trên trình duyệt để người dùng có thể thấy được. Như vậy một webserver chịu trách nhiệm giao tiếp giữa server-client.

Cách thức hoạt động



Hình 2 Mô hình hoạt động Apache Web Server

Apache là một phần mềm chạy trên server, nhiệm vụ của nó là thiết lập kết nối giữa server và trình duyệt của người dùng (Firefox, Google Chrome, Safari, ...). Chuyển các file

giữa chúng (cấu trúc hai chiều dạng client-server). Apache có thể chạy trên nhiều hệ điều hành khác nhau, hoạt động đa nền tảng trên cả server Unix và Windows.

Khi người dùng truy cập tới một trang web trên website. Brower sẽ tiếp nhận yêu cầu của người dùng sau đó chuyển chúng tới server. Apache sẽ gửi những file của trang web được yêu cầu của trang (Hình ảnh, âm thanh, chữ,...). Server và client giao tiếp với nhau thông qua giao thức HTTP và Apache chịu trách nhiệm cho tiến trình giữa client-server được mượt mà và bảo mật.

Apache là một nền tảng module có độ tùy biến cao. Modules cho phép Administrator tắt bật và thêm các tính năng. Apache có các modules cho bảo mật caching, URL rewriting, chứng thực mật khẩu. Thiết lập cấu hình server thông qua file ‘.htaccess’

Apache web server là lựa chọn ưu việt để vận hành một website ổn định và có thể tùy chỉnh linh hoạt. Tuy nhiên, nó cũng có một số điểm bất lợi mà bạn nên biết.

Ưu điểm:

- Phần mềm mã nguồn mở và miễn phí, kể cả cho mục đích thương mại.
- Phần mềm đáng tin cậy, ổn định.
- Được cập nhật thường xuyên, nhiều bản vá lỗi bảo mật liên tục.
- Linh hoạt vì có cấu trúc module.
- Dễ cấu hình, thân thiện với người mới bắt đầu
- Đa nền tảng (hoạt động được cả với server Unix và Windows).
- Hoạt động cực kỳ hiệu quả với WordPress sites.
- Có cộng đồng lớn và sẵn sàng hỗ trợ với bất kỳ vấn đề nào.

Nhược điểm:

- Gặp vấn đề hiệu năng nếu website có lượng truy cập cực lớn.
- Quá nhiều lựa chọn thiết lập có thể gây ra các điểm yếu bảo mật.

2.2. Những lỗ hổng trên máy chủ web Apache

2.2.1 Tấn công từ chối dịch vụ (DoS)

Tấn công từ chối dịch vụ (DoS- Denial of Service). Tấn công từ chối dịch vụ DoS là cuộc tấn công nhằm làm sập một máy chủ hoặc mạng, khiến người dùng khác không thể truy cập vào máy chủ/mạng đó. Kẻ tấn công thực hiện gửi ồ ạt nhiều lưu lượng mạng hoặc

gửi thông tin có thể kích hoạt sự cố máy chủ, hệ thống hoặc mạng mục tiêu, từ đó người dùng hợp pháp (nhân viên, thành viên, chủ tài khoản) không thể truy cập dịch vụ hay tài nguyên mong muốn.

Nạn nhân của tấn công DoS thường là máy chủ web của các tổ chức cao cấp như ngân hàng, doanh nghiệp thương mại, công ty truyền thông, trang báo, mạng xã hội,...

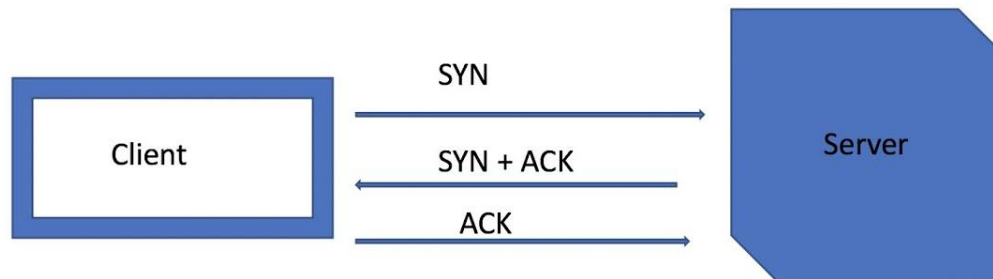


Hình 3 Mô tả tấn công DoS (1)

Những hậu quả của cuộc tấn công DoS có thể gây ra:

- Hệ thống, máy chủ bị DoS sẽ sập khiến người dùng không thể truy cập được.
- Doanh nghiệp bị ảnh hưởng có thể bị ảnh hưởng đến doanh thu và tài sản khi phải bỏ tiền để khắc phục sự cố.
- Khi cuộc tấn công xảy ra hệ thống sập sẽ ảnh hưởng làm gián đoạn công việc, ảnh hưởng tới hiệu suất công việc.
- Uy tín của công ty và doanh nghiệp có thể bị giảm khi những dịch vụ cung cấp bị gián đoạn.
- Phá hoại hoặc thay đổi thông tin cấu hình.

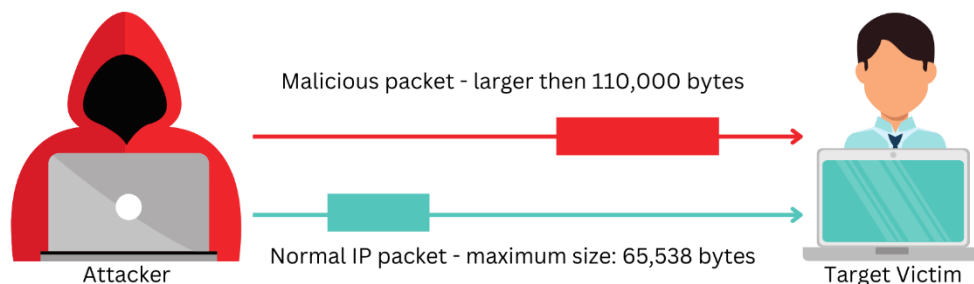
Một số kiểu và phương thức tấn công từ chối dịch vụ phổ biến hiện nay:



Hình 4 Mô tả tấn công DoS (2)

SYN Flood: SYN Flood khai thác điểm yếu trong chuỗi kết nối TCP, được gọi là quá trình bắt tay ba bước. Máy chủ sẽ nhận được một thông điệp đồng bộ (SYN) để bắt đầu "bắt tay". Máy chủ nhận tin nhắn bằng cách gửi cờ báo nhận (ACK) tới máy lưu trữ ban đầu, sau đó đóng kết nối. Tuy nhiên, trong một SYN Flood, kẻ tấn công sẽ gửi tin thông điệp (SYN), khi máy chủ gửi lại gói tin đồng bộ (SYN/ACK) thì máy kẻ tấn công sẽ không nhận gói tin này mà sẽ để máy chủ đợi. Như vậy thực hiện nhiều yêu cầu nhưng không phản hồi làm máy chủ quá tải.

Ping of Death attack



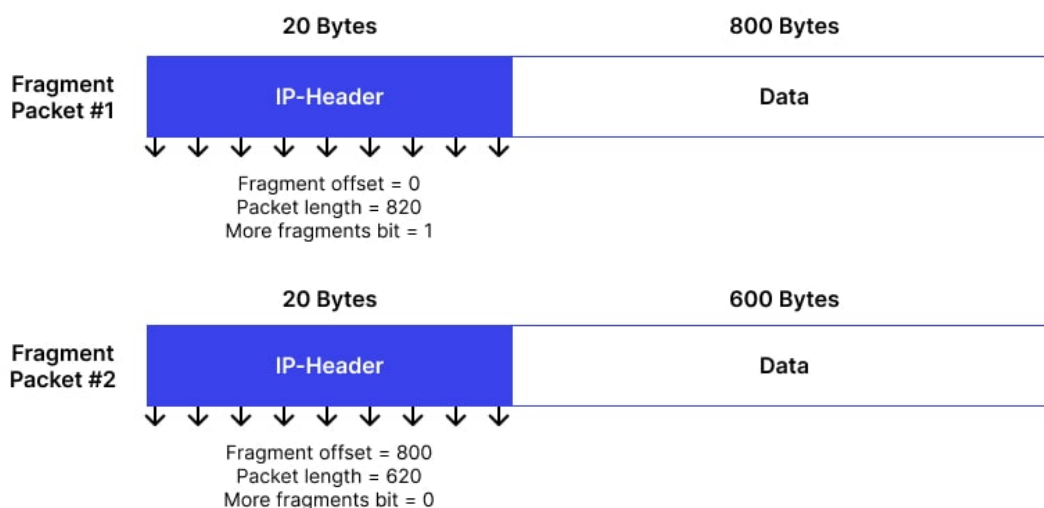
Hình 5 Mô tả tấn công DOS (3)

Ping of Death: Kiểu tấn công này dùng giao thức ICMP. Có 2 phần quan trọng trong ICMP packet là ICMP ECHO_REQUEST và ICMP ECHO_RESPONSE datagrams và thông thường dùng PING command để thi hành các hoạt động của ICMP. Khi 1

máy tính gửi ICMP ECHO_REQUEST đến 1 máy nào đó, nếu máy đó đang hoạt động thì nó sẽ gửi trả lại ICMP ECHO_RESPONSE.

Hacker dùng PING program để tạo nên kích thước lớn cho gói tin ICMP (gói gọn trong 1 IP packet), có nhiều cách để gửi ICMP datagrams mà packet mà chỉ bao gồm 8 bits ICMP header information, Hacker thường dùng PING program để gửi những packet lớn hơn 65536 bytes (vượt qua sự cho phép của TCP/IP).

Khi tấn công bằng Ping of Death một gói tin echo được gửi đi có kích thước lớn hơn kích thước cho phép là 65,536 bytes. Gói tin sẽ bị chia nhỏ ra thành các phần khi máy đích lắp ráp lại thì do gói tin quá lớn với buffer bên nhận nên hệ thống không thể quản lý nổi gây ra bị reboot hoặc bị treo.



Hình 6 Mô tả tấn công DoS (4)

Teardrop: Tất cả các dữ liệu chuyển đi trên mạng từ hệ thống nguồn đến hệ thống đích đều phải trải qua 2 quá trình sau: dữ liệu sẽ được chia ra thành các mảnh nhỏ ở hệ thống nguồn, mỗi mảnh đều phải có một giá trị offset nhất định để xác định vị trí của mảnh đó trong gói dữ liệu được chuyển đi. Khi các mảnh này đến hệ thống đích, hệ thống đích sẽ dựa vào giá trị offset để sắp xếp các mảnh lại với nhau theo thứ tự đúng như ban đầu. Ví dụ, có một dữ liệu gồm 4000 bytes cần được chuyển đi, giả sử rằng 4000 bytes này được chia thành 3 gói nhỏ(packet):

- packet thứ nhất sẽ mang các 1bytes dữ liệu từ 1 đến 1500
- packet thứ hai sẽ mang các bytes dữ liệu từ 1501 đến 3000

- packet thứ ba sẽ mang các bytes dữ liệu còn lại, từ 3001 đến 4000

Khi các packets này đến đích, hệ thống đích sẽ dựa vào offset của các gói packets để sắp xếp lại cho đúng với thứ tự ban đầu:

packet thứ nhất → packet thứ hai → packet thứ ba.

Trong tấn công Teardrop, một loạt gói packets với giá trị offset chồng chéo lên nhau được gửi đến hệ thống đích. Hệ thống đích sẽ không thể nào sắp xếp lại các packets này, nó không điều khiển được và có thể bị crash, reboot hoặc ngừng hoạt động nếu số lượng packets với giá trị offset chồng chéo lên nhau quá lớn.

2.2.2 SQL Injection (SQLi)

Loại lỗ hổng:	Server-Side
Cơ hội tìm thấy	Phổ biến; SQLi là một phần của lỗi Injection xếp hạng trên OWASP Top-10 Vulnerabilities
Mô tả	Lỗ hổng SQLi cho phép kẻ tấn công truy vấn cơ sở dữ liệu theo cách ngoài ý muốn, cho phép thao tác, thêm hoặc xóa dữ liệu.

Bảng 1 Đánh giá lỗ hổng SQL Injection

Lỗ hổng SQL Injection (SQLi) là một lỗ hổng bảo mật web cho phép kẻ tấn công can thiệp vào các truy vấn mà ứng dụng thực hiện đối với cơ sở dữ liệu của nó. Điều này cho phép kẻ tấn công có thể xem những dữ liệu mà thông thường chúng không thể truy xuất được. Nó có thể là dữ liệu về người dùng, hoặc bất cứ dữ liệu nào mà ứng dụng có thể truy cập. Trong nhiều trường hợp, một kẻ tấn công có thể chỉnh sửa hoặc xóa những dữ liệu này, gây ra những thay đổi liên tục đối với nội dung hoặc hành vi của ứng dụng.

Trong nhiều trường hợp, kẻ tấn công có thể lợi dụng phương thức tấn công này để thực hiện xâm nhập vào những máy chủ hoặc cơ sở hạ tầng của hệ thống. Nó có thể cho phép chúng thực hiện một cuộc tấn công từ chối dịch vụ.

❖ Nguyên nhân gây ra lỗi SQL Injection

Lỗi bắt nguồn từ mã nguồn của ứng dụng web chứ không phải từ phía database. Chính vì thế bất cứ thành phần nào của ứng dụng mà người dùng có thể tương tác được để điều khiển nội dung. Nguyên nhân gây ra lỗi:

Không kiểm tra dữ liệu đầu vào (input): Đây là dạng lỗi SQL Injection xảy ra khi thiếu đoạn mã kiểm tra dữ liệu đầu vào trong câu truy vấn SQL. Kết quả là người dùng cuối có thể thực hiện một số truy vấn không mong muốn đối với cơ sở dữ liệu của ứng dụng.

Xử lý không đúng trọng tâm: Lỗi SQL Injection dạng này thường xảy ra do lập trình viên định nghĩa đầu vào dữ liệu không rõ ràng. Hoặc thiếu bước kiểm tra và lọc kiểu dữ liệu đầu vào. Điều này có thể xảy ra khi một trường số được sử dụng trong truy vấn SQL nhưng lập trình viên lại thiếu bước kiểm tra dữ liệu đầu vào để xác minh kiểu của dữ liệu mà người dùng nhập vào có phải là số hay không.

Lỗi bảo mật bên trong máy chủ: Lỗ hổng có thể tồn tại trong chính phần mềm máy chủ cơ sở dữ liệu. Điều này cho phép một cuộc tấn công SQL Injection có thể thực hiện thành công dựa trên những ký tự Unicode không thông thường ngay cả khi dữ liệu đầu vào đã được kiểm soát.

❖ Ảnh hưởng của một cuộc tấn công SQL injection thành công.

Tác động của cuộc tấn công phụ thuộc vào dữ liệu được lưu trữ trong cơ sở dữ liệu dễ bị tấn công. Các cuộc tấn công điển hình bao gồm:

- **Bỏ qua xác thực:** Kẻ tấn công đạt được mục đích đăng nhập vào tài khoản không được phép, ví dụ: kẻ tấn công đăng nhập vào tài khoản người quản trị mà không phải đoán đúng mật khẩu.
- **Truy xuất dữ liệu:** Kẻ tấn công có thể tải xuống các phần hoặc tất cả cơ sở dữ liệu, có khả năng dẫn đến rò rỉ dữ liệu như: mật khẩu, thông tin thẻ tín dụng, thông tin cá nhân của người dùng,...
- **Thao tác với dữ liệu:** Kẻ tấn công có thể thay đổi dữ liệu lưu trữ trong cơ sở dữ liệu nhằm thu lợi ích từ những thông tin thay đổi.

- **Tấn công từ chối dịch vụ:** Kẻ tấn công có thể xoá một số bảng khiến các dịch vụ không hoạt động.

❖ Các biến thể tấn công khác nhau của SQLi

Error-based SQLi: là một lỗi kỹ thuật SQL Injection dựa trên các thông báo lỗi do máy chủ cơ sở dữ liệu đưa ra để lấy thông tin từ cơ sở dữ liệu. Những thông tin đó có thể là những thông tin nhạy cảm mà thông báo lỗi đó tiết lộ như phương thức xác thực, phiên bản và phần mềm sử dụng. Thông thường lỗ hổng này hiển thị những thông báo lỗi thay vì trả về dữ liệu. Những kẻ tấn công sẽ hiểu được và lợi dụng những lỗ hổng này. Điều này thường xảy ra khi thực hiện cấu hình cơ sở dữ liệu và lỗi thông báo.

- Thực hiện kiểm tra kết quả của lỗi dựa trên kết quả logic (true hoặc false).
- Kích hoạt những thông báo lỗi xuất dữ liệu được truy vấn trả về.

Union-based SQLi: là phương pháp tấn công SQL Injection dựa trên toán tử UNION để có thể kết hợp nhiều câu lệnh SELECT với nhau thành một câu lệnh duy nhất và đưa ra một kết quả. Từ khoá UNION cho phép thực hiện một hoặc nhiều truy vấn SELECT bổ sung và nối kết quả vào truy vấn ban đầu. Khi thực hiện khai thác lỗ hổng này cần thực hiện xác định số lượng cột của bảng truy vấn và kiểu dữ liệu của bảng truy vấn để thực hiện trích xuất thông tin.

Blind SQLi: Blind SQL injection xảy ra khi một ứng dụng website bị tấn công SQL injection, những phản hồi của HTTP không bao gồm kết quả liên quan đến câu lệnh truy vấn SQL hoặc chi tiết về những lỗi của cơ sở dữ liệu. Những công nghệ như tấn công union không hiệu quả với các lỗ hổng Blind SQL. Do những các khai thác đó dựa vào kết quả phản hồi của ứng dụng khi đưa những câu lệnh truy vấn vào. Nhưng vẫn có thể thực hiện khai thác lỗ hổng để truy cập dữ liệu trái phép bằng cách sử dụng nhiều các kỹ thuật khác nhau.

2.2.3 Cross Site Scripting (XSS)

Loại lỗ hổng:	Client-Side
Cơ hội tìm thấy	Rất cao; XSS là lỗ hổng bảo mật được xếp hạng trên OWASP Top-10 Vulnerabilities
Mô tả	Một lỗ hổng XSS cho phép kẻ tấn công thực thi những đoạn mã javascript trên trình duyệt của nạn nhân. Nó cho phép kẻ tấn công có thể xâm nhập hoàn toàn vào tài khoản nạn nhân.

Bảng 2 Đánh giá lỗ hổng Cross-Site Scripting (XSS)

Cross-Site Scripting (XSS) là một lỗ hổng bảo mật website cho phép kẻ tấn công thay đổi tương tác mà người dùng thực hiện với một trang web đáng tin cậy. XSS cho phép kẻ tấn công giả thành nạn nhân, để thực hiện bất kỳ hành động nào mà người dùng có thể thực hiện và có thể truy cập vào bất cứ dữ liệu nào của người dùng. Nếu người dùng có quyền truy cập trong ứng dụng, kẻ tấn công có thể kiểm soát tất cả chức năng và dữ liệu của ứng dụng.

❖ Nguyên nhân gây ra lỗ hổng XSS

Không kiểm tra dữ liệu đầu vào (input): Một trong những nguyên nhân chính gây ra lỗ hổng XSS là khi ứng dụng web không kiểm tra hoặc xác thực dữ liệu đầu vào từ người dùng. Điều này cho phép kẻ tấn công chèn mã JavaScript độc hại vào các trường hoặc tham số của URL mà sau đó được hiển thị trực tiếp trên trang web.

Sử dụng những hàm không an toàn trong HTML: Khi ứng dụng web hiển thị dữ liệu người dùng nhập vào mà không làm sạch hoặc xác thực dữ liệu này đúng cách, nó có thể dẫn đến việc chèn mã HTML hoặc JavaScript độc hại vào trang web.

Sử dụng những hàm không an toàn trong Javascript: Trong một số trường hợp, ứng dụng web có thể sử dụng các hàm JavaScript không an toàn như eval() để xử lý dữ liệu đầu vào từ người dùng. Điều này tạo ra một lỗ hổng lớn cho kẻ tấn công chèn và thực thi mã JavaScript độc hại.

Sử dụng cookie không an toàn: Nếu ứng dụng web lưu trữ dữ liệu không an toàn trong cookie và sau đó sử dụng nó mà không kiểm tra hoặc làm sạch, kẻ tấn công có thể chèn mã JavaScript độc hại vào cookie và gửi nó đến trình duyệt của người dùng.

Sử dụng các tham số trong URL không an toàn: Khi các tham số trong URL không được kiểm tra đúng cách, kẻ tấn công có thể chèn mã JavaScript độc hại vào URL và gửi nó đến người dùng qua email, tin nhắn hoặc các phương tiện truyền thông khác.

❖ Ảnh hưởng của một cuộc tấn công XSS thành công

Tùy thuộc vào bản chất của ứng dụng, những chức năng và dữ liệu kèm với trạng thái của người dùng bị xâm nhập.

- Kẻ tấn công có thể thực hiện cài đặt key-logger để đọc từng nút trên một domain (ví dụ: do thám mật khẩu của người dùng).
- Hành động thay mặt nạn nhân trên trang web (ví dụ: gửi tin nhắn, cập nhật thông tin).
- Truy xuất dữ liệu bí mật mà chỉ hiển thị đối với nạn nhân (ví dụ: thông tin xác thực, thông tin định danh cá nhân).
- Thay đổi giao diện trang web nhằm lừa nạn nhân thực hiện thêm các hành động không mong muốn.

❖ Các kiểu tấn công của XSS

Stored cross-site scripting: hay còn gọi là second-order hoặc persistent XSS xảy ra khi ứng dụng nhận được dữ liệu từ nguồn không tin cậy và bao gồm dữ liệu trong phản hồi HTTP theo cách không an toàn. Những đoạn mã JavaScript độc hại vào một trang web hoặc ứng dụng web được lưu trữ trên máy chủ. Khi người dùng truy cập vào trang web hoặc ứng dụng này, mã JavaScript sẽ được thực thi trong ngữ cảnh của trang web đó, có thể làm lộ thông tin nhạy cảm, thực hiện các hành động độc hại hoặc thậm chí kiểm soát tài khoản của người dùng.

Reflected cross-site scripting: là một cách để tấn công XSS trong đó kẻ tấn công chèn mã JavaScript độc hại vào một trang web hoặc ứng dụng web, và sau đó đưa link chứa mã JavaScript này đến cho người dùng. Khi người dùng nhấp vào liên kết đó, trình duyệt của họ sẽ thực thi mã JavaScript trong ngữ cảnh của trang web hoặc ứng dụng web đó, có thể làm lộ thông tin nhạy cảm, thực hiện các hành động độc hại hoặc kiểm soát tài khoản của người dùng.

Chương III: Tường lửa ứng dụng web ModSecurity

3.1 Giới thiệu chung về tường lửa

3.1.1 Khái niệm tường lửa

Tường lửa (Firewall) là một thành phần chính của hệ thống bảo mật mạng, được sử dụng để giữ cho mạng và hệ thống máy tính an toàn khỏi các mối đe dọa và tấn công từ bên ngoài. Mục tiêu chính của firewall là kiểm soát và giám sát lưu lượng mạng dựa trên một loạt các quy tắc và chính sách được xác định trước, để đảm bảo rằng chỉ những kết nối an toàn và hoạt động được phép mới được thông qua.

3.1.2 Phân loại và tính năng

a) Phân loại

Tùy thuộc vào tính năng, dữ liệu được kiểm tra mà một tường lửa được phân loại ra các dạng tường lửa khác nhau tùy thuộc vào mục đích sử dụng của chúng:

- **Tường lửa lọc gói (Packet Filtering Firewall):** Kiểm tra và quyết định xem một gói tin nên được chấp nhận hay từ chối dựa trên thông tin trong tiêu đề gói tin, như địa chỉ nguồn, địa chỉ đích, cổng và giao thức.
- **Tường lửa ứng dụng (Application Layer Firewall):** Hoạt động ở tầng ứng dụng của mô hình OSI và có thể lọc dựa trên thông tin chi tiết về ứng dụng đang chạy. Có thể xác định các ứng dụng, nó cho phép hoặc chặn lưu lượng của ứng dụng đi qua.
- **Tường lửa trạng thái (Stateful Firewall):** Theo dõi trạng thái của các kết nối mạng và quyết định xem gói tin nên được chấp nhận hay từ chối dựa trên thông tin về trạng thái của kết nối.

Nguyên tắc hoạt động: Firewall hoạt động dựa trên những quy tắc và chính sách. Nó thực hiện kiểm soát lưu lượng dựa trên những quy tắc và chính sách được xác định bởi người quản trị. Các quy tắc này xác định cách mà lưu lượng mạng được xử lý.

Vị trí cấu hình: Trong mạng lớn, firewall thường được triển khai ở các địa điểm cổng vào và ra của mạng để bảo vệ hệ thống và dữ liệu khỏi các mối đe dọa.

b) Tính năng

Firewall là một công cụ quan trọng để bảo vệ mạng máy tính khỏi những mối đe dọa từ bên ngoài và cả bên trong của hệ thống mạng. Những chức năng chính của một Firewall thường bao gồm:

- **Kiểm soát truy cập:** Firewall cho phép người quản trị mạng kiểm soát quyền truy cập vào và ra khỏi mạng, dựa trên các quy tắc được thiết lập trước. Những quy tắc hay rule là phần quan trọng nhất của một firewall nó như một luật lệ được đặt ra để kiểm soát và điều khiển những dữ liệu vào và ra.
- **Bộ lọc gói tin (Packet filtering):** Firewall có thể kiểm tra và lọc các gói tin dựa trên các thuộc tính như địa chỉ IP nguồn và đích, cổng đích, giao thức, và các tiêu chí khác. Những gói tin tiềm ẩn nhiều những mối nguy hiểm như mã độc kèm theo trong các tệp đi kèm. Ngoài ra những gói tin cũng bao gồm nhiều những thông tin hữu ích khác nhau, có thể sử dụng chúng để kiểm tra và trích xuất những thông tin hữu ích.
- **Proxy server:** Một số loại firewall cung cấp tính năng proxy, cho phép chuyển tiếp gói tin giữa các mạng, ẩn địa chỉ IP thực sự của người dùng và bảo vệ danh tính của họ.
- **Giám sát lưu lượng mạng:** Firewall có thể giám sát và ghi lại các hoạt động mạng, cung cấp thông tin chi tiết về lưu lượng truy cập và các sự kiện an ninh.
- **Phát hiện xâm nhập (Intrusion detection):** Một số firewall có khả năng phát hiện các hành vi xâm nhập bằng cách so sánh dữ liệu mạng với các chữ ký hoặc quy tắc được xác định trước.
- **VPN (Virtual Private Network):** Firewall có thể hỗ trợ các kết nối VPN, cho phép người dùng từ xa truy cập vào mạng nội bộ một cách an toàn.
- **Quản lý đường truyền (Bandwidth management):** Firewall có thể quản lý băng thông mạng bằng cách giới hạn hoặc ưu tiên lưu lượng truy cập từ các ứng dụng hoặc nguồn khác nhau.
- **Tường lửa ứng dụng web (Web Application Firewall - WAF):** Một số firewall có tính năng bảo vệ các ứng dụng web khỏi các cuộc tấn công như SQL injection, cross site scripting (XSS), và các lỗ hổng bảo mật khác.

- **Tích hợp với các giải pháp bảo mật khác:** Firewall thường có khả năng tích hợp với các giải pháp bảo mật khác như antivirus, antimalware, và IPS/IDS để tăng cường bảo vệ mạng.
- **Cập nhật định kỳ:** Firewall cần được cập nhật định kỳ để nhận diện và ngăn chặn các mối đe dọa mới và cập nhật các quy tắc an ninh.

3.2 Tường lửa modSecurity

3.2.1 Lịch sử phát triển

ModSecurity, đôi khi được gọi là Modsec, là tường lửa ứng dụng web nguồn mở (WAF). Ban đầu được thiết kế như một module cho Máy chủ HTTP Apache, nó đã phát triển để cung cấp một loạt các khả năng lọc phản hồi và yêu cầu giao thức HTTP cùng với các tính năng bảo mật khác trên một số nền tảng khác nhau bao gồm Máy chủ HTTP Apache, Microsoft IIS và Nginx. Đây là phần mềm miễn phí được phát hành theo giấy phép Apache 2.0.

Nền tảng này cung cấp ngôn ngữ cấu hình bằng những quy tắc được gọi là 'SecRules' để theo dõi, ghi nhật ký và lọc thời gian thực các giao thức HTTP dựa trên các quy tắc do người dùng xác định.

ModSecurity được phát triển lần đầu tiên bởi Ivan Ristić, anh ấy viết modul này với mục đích là giám sát lưu lượng ứng dụng trên Máy chủ HTTP Apache. Phiên bản đầu tiên được phát hành vào tháng 11 năm 2002 hỗ trợ Apache HTTP Server 1.3.x. Bắt đầu từ năm 2004, Ivan đã thành lập Thinking Stone để tiếp tục làm việc toàn thời gian cho dự án. Trong khi đang phát triển phiên bản 2.0, Thinkg Stone đã được Breach Security, một công ty bảo mật Mỹ-Israel, mua lại vào tháng 9 năm 2006. Ivan vẫn tiếp tục phát triển phiên bản 2.0, sau đó được phát hành vào tháng 10 năm 2006 tại hội nghị OWASP AppSec ở Seattle.

Ristić và Breach Security đã phát hành một bản phiên bản 2.5, với những thay đổi lớn về cú pháp vào tháng 2 năm 2008. Vào tháng 12 năm 2008, Ivan rời Breach để thành lập SSL Labs. Ngay sau khi Ivan rời Breach Security, Trustwave Holdings đã mua lại Breach vào tháng 6 năm 2010 và cấp phép lại ModSecurity theo giấy phép Apache. Quá trình phát triển vẫn tiếp tục và giấy phép mới cho phép tích hợp ModSecurity vào các sản phẩm khác

dễ dàng hơn. Kết quả là đã có nhiều sản phẩm thương mại khác nhau đã ứng dụng ModSecurity. Việc thay đổi giấy phép cũng khiến việc chuyển phần mềm trở nên dễ dàng hơn. Do đó, Microsoft cho phép ModSecurity hoạt động trên máy chủ web IIS vào tháng 8 năm 2012. Phiên bản cho máy chủ Nginx đã được phát hành tại Black Hat Briefings vào năm 2012.

Năm 2017 phát hành sách hướng dẫn về ModSec, do Christian Folini và Ivan Ristić viết. Nó bao gồm ModSecurity lên đến phiên bản 2.9.2.

Ban đầu là một module Apache, việc chuyển ModSecurity sang các nền tảng khác tốn nhiều thời gian và chi phí bảo trì cao. Do đó, việc viết lại hoàn chỉnh đã được bắt đầu vào tháng 12 năm 2015. Phiên bản mới này, libmodsecurity, thay đổi kiến trúc cơ bản, tách ModSecurity thành một công cụ độc lập giao tiếp với máy chủ web thông qua API. WAF dựa trên kiến trúc module này, được công bố sử dụng rộng rãi vào tháng 1 năm 2018, trở thành libmodsecurity (ModSecurity phiên bản 3.0) và hỗ trợ các trình kết nối cho Nginx và Apache.

Năm 2021, Trustwave Holdings, công bố kết thúc hỗ trợ (EOS – End of Sale) của Trustwave cho ModSecurity có hiệu lực từ ngày 1 tháng 8 năm 2021 và ngừng phát triển sản phẩm (EOL – End of Life) có hiệu lực từ ngày 1 tháng 7 năm 2024. Việc duy trì mã nguồn ModSecurity được trao cho cộng đồng nguồn mở.

3.2.2 Chức năng chính

ModSecurity là một bộ công cụ giám sát ứng dụng web thời gian thực, ghi nhật ký, gỡ lỗi và kiểm soát truy cập. Chức năng của nó được chia thành bốn nhóm chính:

❖ Phân tích cú pháp (Parsing)

ModSec sẽ thu thập nhiều dữ liệu nhất có thể. Các định dạng dữ liệu được phân tích bởi những bộ công cụ bảo mật phân tích cú pháp, từ đó trích xuất và phân tích dữ liệu để có thể sử dụng tới những bộ quy tắc.

❖ Bộ đệm (Buffering)

Trong cài đặt thông thường, những nội dung về yêu cầu và phản hồi của website sẽ được lưu vào bộ đệm. Nghĩa là những yêu cầu từ máy khách sẽ chuyển qua ModSec trước khi đến được server và tương tự những phản hồi cũng được đưa qua trước khi

quay trở lại trình duyệt người dùng. Bộ đệm là một tính năng quan trọng, vì đây là cách để đánh giá những yêu cầu và phản hồi trong thời gian thực. Nhược điểm là nó cần nhiều RAM để thực hiện lưu trữ request và toàn bộ thông tin phản hồi.

❖ Ghi nhật ký (Logging)

Chức năng này cho phép ghi lại những yêu cầu và phản hồi của website khi dữ liệu được chuyển qua ModSec. Những thông tin này hữu dụng cho việc điều tra những cuộc tấn công và là dữ liệu quan trọng để kết hợp với những hệ thống bảo mật khác để kiểm soát an ninh mạng.

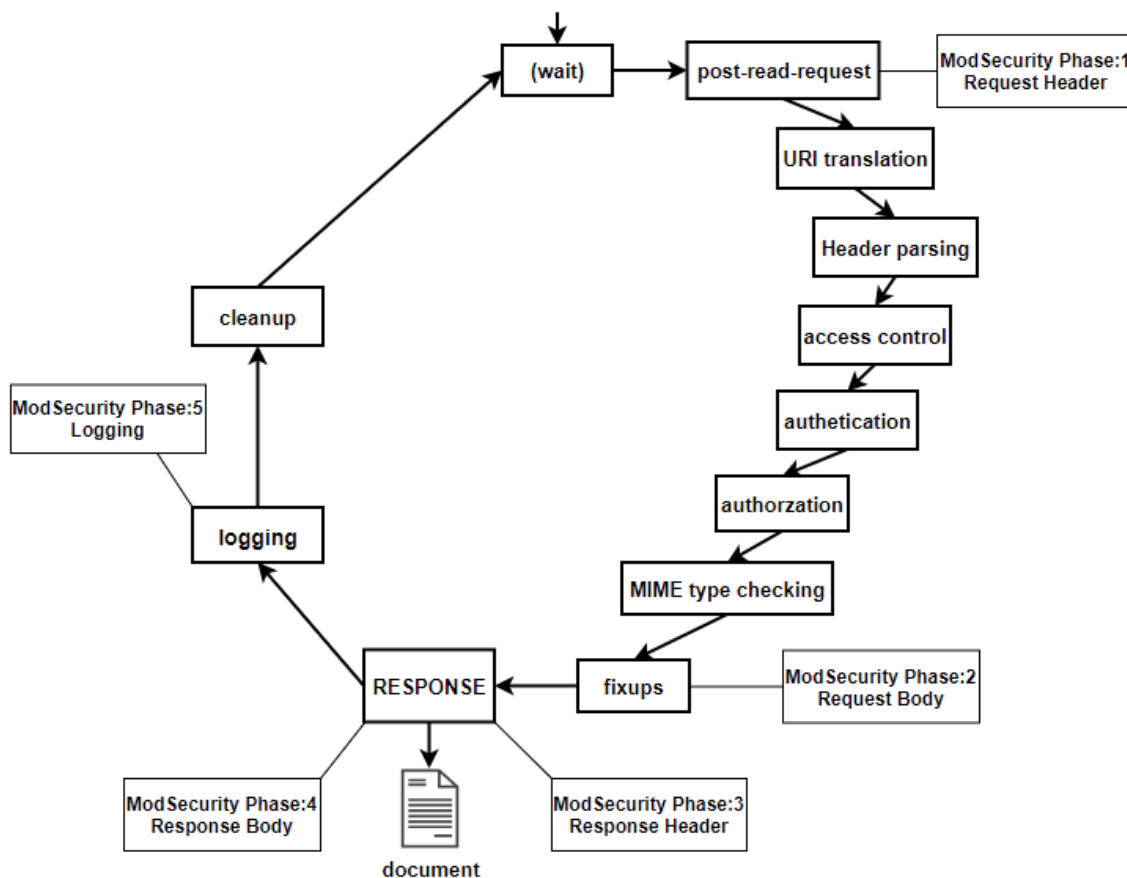
❖ Bộ công cụ quy tắc (Rule Engine)

Bộ công cụ quy tắc là phần quan trọng nhất để một firewall có thể thực hiện được chức năng của nó. Để hoạt động được nó cần những dữ liệu và chức năng của tất cả các thành phần khác. Khi bộ quy tắc hoạt động, những bit dữ liệu và những đoạn dữ liệu để chuẩn bị cho cuộc điều tra. Vào thời điểm đó, những quy tắc sẽ kiểm soát truy cập dữ liệu và thực hiện những hành động cần thiết.

3.2.3 Nguyên lý hoạt động của ModSecurity

Trong ModSecurity, mỗi quy trình kiểm tra dữ liệu được thực hiện qua 5 bước hay 5 giai đoạn. Mỗi giai đoạn lại có một chức năng nhất định, kết quả của giai đoạn trước làm dữ liệu đầu vào của giai đoạn sau. Năm giai đoạn đó bao gồm:

- Request headers (REQUEST_HEADERS)
- Request body (REQUEST_BODY)
- Response headers (RESPONSE_HEADER)
- Response body (RESPONSE_BODY)
- Logging (LOGGING)



Hình 7 Quy trình thực hiện của quy tắc trong ModSec

❖ Giai đoạn một – Request Headers (phase:1)

Đây là giai đoạn đầu tiên khởi đầu của ModSec. Sau khi máy chủ Apache bắt đầu xử lý yêu cầu người dùng, những quy tắc ở giai đoạn này sẽ đọc và phân tích các HTTP Request Header. Mục đích quan trọng của giai đoạn này là xem xét những HTTP Request Header độc hại trước khi máy chủ Apache nhận được và xử lý những đối số này. Giai đoạn này những quy tắc giúp xác định những tham số trên có được chấp nhận hay không.

❖ Giai đoạn hai – Request Body (phase:2)

Sau khi xử lý Request Header, tiếp theo là xử lý Request Body. Trong giai đoạn này, Apache đã nhận được những yêu cầu. Những quy tắc trong giai đoạn này chủ yếu liên quan tới những ứng dụng. ModSec hỗ trợ ba kỹ thuật mã hoá trong giai đoạn này, những kỹ thuật này được sử dụng bởi những máy chủ web hiện tại.

- application/x-www-form-urlencoded – for transferring form data

- multipart/form-data – for file transfers
- text/xml – for XML data

❖ **Giai đoạn ba – Response Headers (phase:3)**

Đây là giai đoạn mà những yêu cầu (request) được xử lý và gửi lại những dữ liệu được yêu cầu. Giống với những quy tắc của Request Header, những quy tắc này sẽ xem xét về Response Header trước khi gửi nội dung phản hồi đó cho người dùng. Những quy tắc xác định có quyết định những nội dung ở phần Response Body có phản hồi lại người dùng hay không.

❖ **Giai đoạn bốn – Response Body (phase:4)**

Khi một tín hiệu Response Header được chấp nhận và gửi tới bộ đệm để được trả về Response Body, những quy tắc sẽ được triển khai và thực thi để phân tích những nội dung được gửi lại từ những yêu cầu. Giai đoạn này có thể phân tích và đánh giá những thông tin về xác thực không thành công và những thông báo lỗi.

❖ **Giai đoạn năm – Logging**

Đây là bước cuối cùng trong năm bước của một chu kỳ để ModSecurity bảo vệ được máy chủ web Apache. Những quy tắc trong giai đoạn này được đặt ngay trước khi việc ghi lại nhật ký được diễn ra. Trong giai đoạn này những nhật ký do máy chủ web Apache tạo ra sẽ được phân tích. Quan trọng là trong giai đoạn này những quy tắc đã không còn tác động được tới bất kỳ kết nối nào vì giai đoạn này chỉ ghi lại những kết quả sau quá trình Request/Response diễn ra. Tại đây có ta hoàn toàn có thể kiểm tra được những thông tin của các phân tiêu đề mà giai đoạn 3 và giai đoạn 4 không thể hiện.

3.2.4 Cấu trúc đặt quy tắc (rule)

Trong một hệ thống tường lửa, rule (hay còn gọi là quy tắc) đóng một vai trò cốt lõi như những bộ lọc, kiểm soát lưu lượng truy cập mạng một cách hiệu quả và an toàn. Những quy tắc giúp xác định lưu lượng được phép đi quan mạng, loại nào bị chặn và loại nào cần được ghi chép lại để theo dõi. Dựa vào những quy tắc này hệ thống có thể nhận dạng cảnh báo hoặc ngăn chặn các cuộc tấn công từ bên ngoài vào hệ thống để đảm bảo an toàn cho hệ thống công nghệ thông tin.

Tương tự như những tường lửa khác, modsecurity cũng sử dụng hệ thống quy tắc nhất định để thực hiện việc kiểm tra lưu lượng, từ đó đưa ra quyết định tới lưu lượng mạng đó. Cho phép thông quan, cảnh báo chặn hay chỉ thực hiện ghi lại nhật ký.

Cấu trúc của một quy tắc trong tường lửa modSecurity:

SecRule VARIABLES "OPERATOR" "[TRANSFORMATION_FUNCTIONS, ACTIONS]"

Từ quy tắc trên có thể thấy được cấu trúc để tạo một quy tắc trong modSecurity

Thành phần	Ý nghĩa
SecRule	Phương thức được sử dụng tạo và xác định một quy tắc trong modSecurity
VARIABLES	Chỉ định một hoặc nhiều điểm dữ liệu từ HTTP request mà quy tắc sẽ kiểm tra.
OPERATOR	Xác định điều kiện logic mà các biến(variables) phải đáp ứng để kích hoạt quy tắc.
TRANSFORMATION_FUNCTIONS	Danh sách các hàm (cách nhau bởi dấu phẩy) thao tác với dữ liệu biến trước khi so sánh.
ACTIONS	Xác định những gì ModSecurity sẽ làm nếu điều kiện của quy tắc được đáp ứng.

Bảng 3 Mô tả thành phần của quy tắc ModSecurity

Variables: trong tường lửa, những biến (hay variables) được sử dụng để xác định chính xác nơi mà quá trình xử lý dữ liệu của giao thức HTTP diễn ra. Một trong những tính năng của ModSec là nó sẽ xử lý dữ liệu giao thức HTTP ở dạng thô trở nên dễ dàng hơn để những quy tắc có thể tập trung vào kiểm tra logic.

Dạng dữ liệu	Miêu tả	Ví dụ
Biến thông thường (Regular variables)	Bao gồm duy nhất một phần thông tin hoặc một chuỗi.	REQUEST_METHOD
Tập nhiều biến (collections)	Tập hợp những biến thông thường.	ARGS_NAME

Tập nhiều biến chỉ đọc (Read-only collections)	Tập dữ liệu chứa những biến thông thường không thể sửa đổi hoặc thay thế nên chỉ có thể đọc.	ARGS
Đọc/Ghi tập biến (Read/Write collections)	Tập dữ liệu dựa trên những biến có thể thay đổi được để có thể đọc và ghi lại biến đó.	IP, TX

Bảng 4 Mô tả dạng dữ liệu của ModSec Rule

Một số biến và ý nghĩa khi sử dụng của biến đó:

Giá trị biến	Miêu tả	Ví dụ
ARGS	Đại diện cho tất cả những đối số yêu cầu.	ARGS:username @eq "admin"
ARGS_NAMES	Tập hợp những đối số được cách nhau bởi dấu phẩy.	ARGS_NAMES:/* "@rx(?:get post)"
REQUEST_HEADERS	Ghi lại tất cả tiêu đề liên quan đến yêu cầu.	REQUEST_HEADERS:User-Agent
REQUEST_URI	Ghi lại đường dẫn URL được yêu cầu	REQUEST_URI @pm "/login"
REQUEST_METHOD	Ghi lại phương thức HTTP được sử dụng trong yêu cầu	REQUEST_METHOD @eq "POST"
RESPONSE_HEADERS	Ghi lại tất cả tiêu đề phản hồi	RESPONSE_HEADERS:Server "@rx ^Apache.+"
RESPONSE_BODY	Ghi lại nội dung phản hồi	RESPONSE_BODY "@rx<title>phpinfo\(\)</title>

RESPONSE_STATUS	Ghi lại mã phản hồi	RESPONSE_STATUS @eq "404 Not Found"
-----------------	---------------------	-------------------------------------

Bảng 5 Mô tả giá trị biến (Variable) của ModSec Rule

Toán tử (Operator): những toán tử được sử dụng để thực hiện những phép toán như so sánh, đối chiếu để đạt được những hiệu quả khi thực hiện tạo một quy tắc. Để tăng tính chuyên biệt và hiệu quả, những toán tử được phân ra thành các loại khác nhau để thực hiện các phép toán.

Đặc điểm của một toán tử

- + Bắt đầu với ký tự @
- + Luôn nằm ở vị trí sau SecRule và các biến
- + Luôn có một khoảng trắng giữa thành phần phía trước với toán tử
- + Với mỗi toán tử rõ ràng cần thêm dấu ngoặc kép quanh mã thông báo.

Một số loại toán tử bao gồm toán tử chuỗi, toán tử số học.

Toán tử chuỗi		
Toán tử	Miêu tả	Ví dụ
@beginsWith	Bắt đầu với	REQUEST_URI @beginsWith "/admin/"
@contains	Bao gồm	User-Agent @contains "sqlmap"
@endsWith	Kết thúc với	REQUEST_URI @endswith ".exe"
@rx	So khớp với mẫu biểu thức chính quy	REQUEST_URI @rx "\.php(๓๓<\\)??"
@pm	Kết hợp song song	REQUEST_URI @pm "^/login(\$ /)"
@pmFromFile	Kết hợp song song, thường từ một tệp tin	REQUEST_HEADERS:User-Agent @pmFromFile "/path/to/banned_user_agents.txt"
@streq	Giá trị chuỗi bằng	ARGS_NAMES @streq "Username"
@within	Ở trong	ARGS:score @within 1,10

Bảng 6 Mô tả toán tử chuỗi của ModSec Rule

Toán tử số học		
Toán tử	Miêu tả	Ví dụ
@eq	Bằng	REQUEST_METHOD @eq GET
@ge	Lớn hơn hoặc bằng	ARGS:id @ge 1
@gt	Lớn hơn	TX:CONTENT_LENGTH @gt 1024000
@le	Nhỏ hơn hoặc bằng	ARGS:age @le 18
@lt	Nhỏ hơn	HTTP_VERSION @lt 1.1

Bảng 7 Mô tả toán tử số học của ModSec Rule

Hành động (Actions) quy định hành động sẽ thực hiện của tường lửa khi có dữ liệu khớp với quy tắc đã định. Để tăng tính linh hoạt của tường lửa một số quy tắc được chỉ định bao gồm:

Hành động	Miêu tả
allow	Dừng xử lý một hoặc nhiều giai đoạn còn lại.
block	Chỉ định quy tắc muốn chặn.
deny	Chặn quá trao đổi với một trang báo lỗi.
drop	Đóng kết nối.
pass	Không chặn, qua tới quy tắc tiếp theo.
proxy	Chặn những yêu cầu từ web server.
redirect	Chuyển hướng yêu cầu tới một web server khác.

Bảng 8 Mô tả hành động (Action) của ModSec Rule

Hàm chuyển đổi dữ liệu (Transformation Functions) trong ModSecurity được sử dụng để thao tác với dữ liệu văn bản được trích xuất từ những yêu cầu như (header, URI, đối số) trước khi dữ liệu đó được sử dụng để kiểm tra với những quy tắc. Điều này sẽ giúp những quy tắc trở nên linh động và hiệu quả hơn.

Chức năng	Mô tả
lowercase	Chuyển đổi dữ liệu về chữ thường.
urlDecode	Chuyển những ký tự mã hoá của url thành những ký tự thông thường.

none	Giữ nguyên dữ liệu gốc được trích xuất từ những yêu cầu để khớp với những quy tắc.
removeWhitespace	Xoá khoảng trắng.
replaceNulls	Thay thế ký tự Null.
removeNulls	Xoá ký tự Null.
htmlEntityDecode	Giải mã các ký tự được mã hóa dưới dạng thực thể HTML.
compressWhitespace	Chuyển đổi bất kỳ ký tự khoảng trắng nào (0x20, \f, \t, \n, \r, \v, 0xa0) thành khoảng trắng (ASCII 0x20), nén nhiều ký tự khoảng trắng liên tiếp thành một.
urlDecodeUni	Mã hoá chuỗi đầu vào bằng mã hoá URL, có hỗ trợ mã hóa %u dành riêng cho Microsoft.

Bảng 9 Mô tả hàm chuyển đổi của ModSec Rule

3.2.5 Cài đặt tường lửa ModSecurity

ModSecurity có thể được tải và cài đặt từ mã nguồn hoặc gói phiên bản có sẵn trên những hệ điều hành khác nhau. Hầu hết những phiên bản hiện nay đã có các gói cài đặt trên các hệ điều hành để dễ dàng cài đặt và cấu hình. Các bước để thực hiện cấu hình từ những gói cài đặt trên hệ điều hành ubuntu cụ thể là phiên bản ubuntu 22.04.

Bước 1: Kiểm tra phiên bản của hệ thống sử dụng câu lệnh **lsb_release -a**

```
root@modsec:/etc/modsecurity/crs# lsb_release -a
No LSB modules are available.
Distributor ID: Ubuntu
Description:   Ubuntu 22.04.2 LTS
Release:      22.04
Codename:     jammy
```

Hình 8 Kiểm tra phiên bản ubuntu

Phiên bản ubuntu server hiện tại đang sử dụng phiên bản 22.04

Bước 2: Tiến hành cập nhật danh sách gói để đảm bảo có những phiên bản mới nhất trong kho lưu trữ

sudo apt update

```

root@modsec:~# sudo apt update
Hit:1 http://vn.archive.ubuntu.com/ubuntu jammy InRelease
Hit:2 http://vn.archive.ubuntu.com/ubuntu jammy-updates InRelease
Hit:3 http://vn.archive.ubuntu.com/ubuntu jammy-backports InRelease
Hit:4 http://vn.archive.ubuntu.com/ubuntu jammy-security InRelease
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
94 packages can be upgraded. Run 'apt list --upgradable' to see them.
root@modsec:~# _

```

Hình 9 Tiến trình cập nhật ubuntu

Tiến hành cài đặt ModSecurity bằng cách thực hiện câu lệnh

sudo apt install libapache2-mod-security2

Apache là máy chủ sử dụng những module để hoạt động, kích hoạt module

sudo a2enmod security2

Khởi động lại máy chủ Apache để thực hiện thay đổi.

Service apache2 restart

```

root@modsec:~# sudo apt install libapache2-mod-security2
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
libapache2-mod-security2 is already the newest version (2.9.5-1).
0 upgraded, 0 newly installed, 0 to remove and 94 not upgraded.
root@modsec:~#

```

Hình 10 Cài đặt ModSecurity2

Bước 3: Kích hoạt cấu hình ModSecurity

Sau khi cài đặt, ModSecurity không được kích hoạt mặc định. Để kích hoạt nó, cần thay đổi tên tệp cấu hình mặc định.

**sudo mv /etc/modsecurity/modsecurity.conf-recommended
/etc/modsecurity/modsecurity.conf**

Chỉnh sửa tệp cấu hình ModSecurity để điều chỉnh các cài đặt cơ bản

sudo nano /etc/modsecurity/modsecurity.conf

Tìm phần SecRuleEngine và thay đổi giá trị đó thành On để cho phép quá trình xử lý của những quy tắc:

Chuyển SecRuleEngine DetectionOnly → SecRuleEngine On

```
GNU nano 6.2 /etc/modsecurity/modsecurity.conf
# -- Rule engine initialization -----
# Enable ModSecurity, attaching it to every transaction. Use detection
# only to start with, because that minimises the chances of post-installation
# disruption.
#
SecRuleEngine DetectionOnly_

# -- Request body handling -----
# Allow ModSecurity to access request bodies. If you don't, ModSecurity
# won't be able to see any POST parameters, which opens a large security
# hole for attackers to exploit.
#
SecRequestBodyAccess On
```

Hình 11 Tùy chỉnh tệp cấu hình (1)

Tìm kiếm tới dòng 186 thay đổi đoạn mã để ModSecurity biết thông tin đưa vào nhật ký kiểm tra.

SecAuditLogParts ABDEFHIJZ → SecAuditLogParts ABCEFHIJKZ

```
SecAuditEngine On
SecAuditLogRelevantStatus "^(?:5|4(?:04))"
# Log everything we know about a transaction.
SecAuditLogParts ABCEFHIJKZ
# Use a single file for logging. This is much easier to look at, but
# assumes that you will use the audit log only occasionally.
#
SecAuditLogType Serial
SecAuditLog /var/log/apache2/modsec_audit.log
```

Hình 12 Tùy chỉnh tệp cấu hình (2)

Sau đó khởi động lại Apache để khởi động lại cấu hình.

Service apache2 restart

```
root@modsec:/# service apache2 restart
root@modsec:/# _
```

Hình 13 Khởi động lại apache server

Bước 5: Kiểm tra phiên bản ModSecurity cài đặt

apt-cache show libapache2-mod-security2

```

Package: libapache2-mod-security2
Source: modsecurity-apache
Version: 2.9.8~pre-20230105-ubuntu0.22.04.1
Architecture: amd64
Maintainer: Alberto Gonzalez Iniesta <agi@inittab.org>
Installed-Size: 1223
Depends: libxml2 (>= 2.9.0), libapr1 (>= 1.2.7), libaprutil1 (>= 1.4.0), libc6 (>= 2.34), libcurl3-gnutls (>= 7.16.2), libfuzzy2 (>= 2.13), liblua5.1-0, libpcre3, libyajl2 (>= 2.0.4), apache2-api-20120211, apache2-bin (>= 2.4.16)
Recommends: modsecurity-crs
Breaks: libapache2-modsecurity (<< 2.7.7-1~)
Replaces: libapache2-modsecurity (<< 2.7.7-1~)
Homepage: http://www.modsecurity.org/

```

Hình 14 Kiểm tra cấu hình Modsecurity

Bước 6: Cài đặt OWASP Core Rule Set (CRS). Tải phiên bản mới nhất của OWASP CRS từ Github và giải nén tập rule.

Tải rule: *wget https://github.com/coreruleset/coreruleset/archive/v3.3.5.tar.gz*

Giải nén tập tin: *unzip v3.3.5.tar.gz*

```

root@modsec:/home/modsec# ls
coreruleset-3.3.5  v3.3.5.zip
root@modsec:/home/modsec#

```

Hình 15 Tùy chỉnh Core Rule Set (1)

Chuyển tập tới mục lưu trữ rule để modsecurity có thể lấy thông tin của những rule

sudo mv coreruleset-3.3.5/ /etc/apache2/modsecurity-crs/

Chuyển tới tập coreruleset-3.3.5 để chỉnh sửa tập cấu hình

cd /etc/apache2/modsecurity-crs/coreruleset-3.3.5/

```

RESPONSE-950-DATA-LEAKAGES.conf
RESPONSE-951-DATA-LEAKAGES-SQL.conf
RESPONSE-952-DATA-LEAKAGES-JAVA.conf
RESPONSE-953-DATA-LEAKAGES-PHP.conf
RESPONSE-954-DATA-LEAKAGES-IIS.conf
RESPONSE-959-BLOCKING-EVALUATION.conf
RESPONSE-980-CORRELATION.conf
RESPONSE-999-EXCLUSION-RULES-AFTER-CRS.conf.example
restricted-files.data
restricted-upload.data
scanners-headers.data
scanners-urls.data
scanners-user-agents.data
scripting-user-agents.data
sql-errors.data
unix-shell.data
windows-powershell-commands.data
root@modsec:/etc/modsecurity/rules# _

```

Hình 16 Tùy chỉnh Core Rule Set (2)

Đổi tên file crs-setup.conf.example thành crs-setup.conf

sudo mv crs-setup.conf.example crs-setup.conf

```
root@modsec:/etc/modsecurity# ls
crs          modsecurity.conf  rules
localrules.conf  modsecurity.conf-recommended.dpkg-dist  unicode.mapping
root@modsec:/etc/modsecurity# _
```

Hình 17 Tùy chỉnh Core Rule Set (3)

Chỉnh sửa tệp tin security2.conf

sudo nano /etc/apache2/mods-enabled/security2.conf

Thêm đường dẫn tới tệp rule đã cài đặt

```
<IfModule security2_module>
    # Default Debian dir for modsecurity's persistent data
    SecDataDir /var/cache/modsecurity

    # Include all the *.conf files in /etc/modsecurity.
    # Keeping your local configuration in that directory
    # will allow for an easy upgrade of THIS file and
    # make your life easier
    Include /etc/modsecurity/*.conf
    Include /etc/modsecurity/crs/*.conf
    # Include OWASP ModSecurity CRS rules if installed
    IncludeOptional /etc/modsecurity/rules/*.conf
    # Include /usr/share/modsecurity-crs/*.load
</IfModule>
```

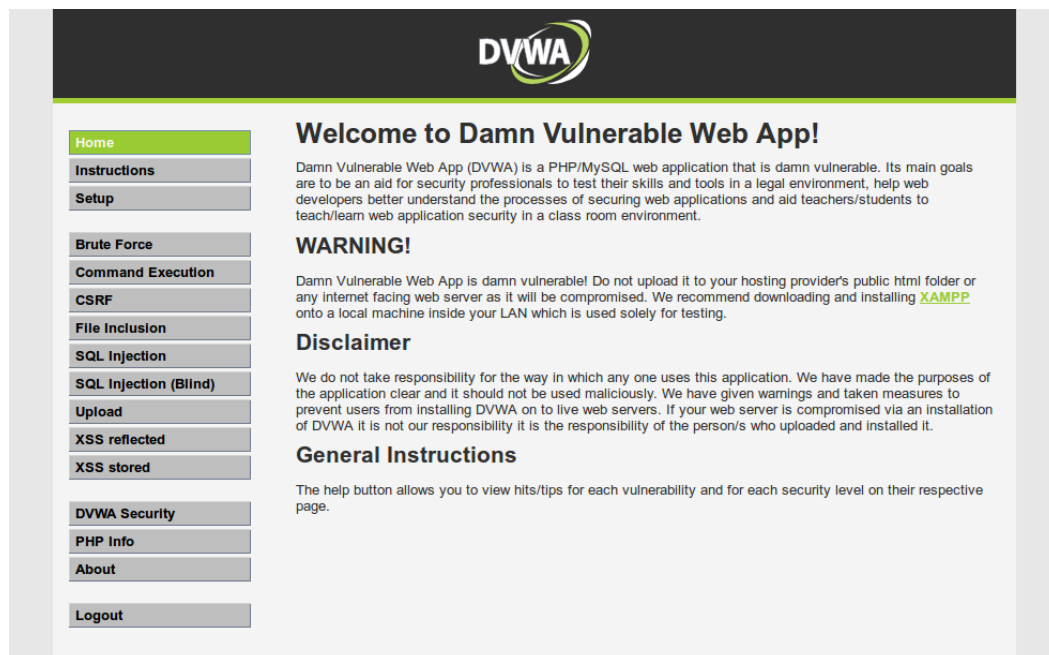
Hình 18 Tùy chỉnh Core Rule Set (4)

Chương IV: Bảo vệ máy chủ Web Apache bằng tường lửa ModSecurity

4.1 Công cụ sử dụng

4.1.1 DVWA

Damn Vulnerable Web Application (DVWA) là một ứng dụng mã nguồn PHP/MySQL tập hợp sẵn các lỗi logic về bảo mật ứng dụng web trong mã nguồn PHP. Lỗi logic khi lập trình có thể áp dụng đối với các loại ngôn ngữ lập trình nhằm giảm thiểu khả năng tạo ra lỗ hổng bảo mật từ tư duy lập trình chưa cẩn thận.



Hình 19 Damn Vulnerable Web Application (DVWA)

4.1.2 Kali linux

Kali Linux là một bản phân phối Linux dựa trên Debian. Mục tiêu của nó đơn giản là: tập hợp nhiều công cụ kiểm tra bảo mật và thâm nhập tốt nhất có thể trong một môi trường hệ điều hành. Kali được sinh ra với mục đích như vậy nên bạn có thể tìm thấy nhanh gọn nhiều công cụ mã nguồn mở để thực hiện các quy trình kiểm thử (pentest), tấn công, hacking để tiết kiệm thời gian cho những chuyên gia bảo mật.



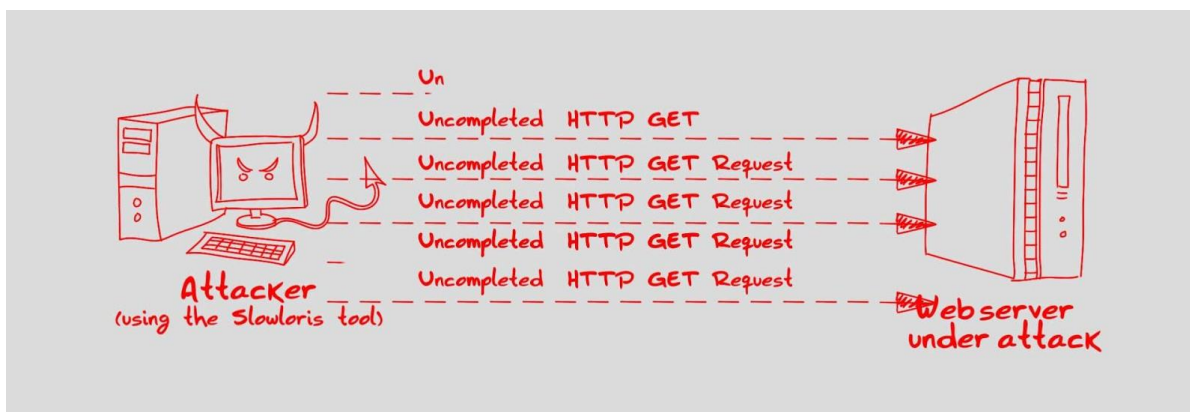
Hình 20 Kali linux

4.2 Tấn công và phòng thủ DOS

4.2.1 Tấn công DoS bằng công cụ slowloris

Tấn công từ chối dịch vụ là thực hiện gửi hàng loạt những yêu cầu tới máy chủ để thực hiện chiếm băng thông của máy chủ làm gián đoạn việc sử dụng của người dùng. Sau khi gửi hàng loạt yêu cầu làm máy chủ không kịp xử lý sẽ dẫn tới việc website bị quá tải dẫn tới một số dịch vụ hoặc cả trang web không hoạt động.

Sử dụng công cụ slowloris để liên tục gửi những request. Công cụ slowloris là công cụ thực hiện kiểm thử tấn công DoS một trang Web. Công cụ này sẽ gửi những gói tin HTTP với nhiều User-Agent với nhiều giá trị khác nhau và những tham số ngẫu nhiên không xác định để chiếm băng thông của trang web mục tiêu.



Hình 21 Mô phỏng tấn công DoS (1)

Khi thực hiện tấn công công cụ sẽ gửi những gói tin với những giá trị ngẫu nhiên để thực hiện chiếm đoạt băng thông.

Khi kiểm tra access.log của máy chủ web Apache có thể thấy những giá trị yêu cầu này.

```
192.168.6.167 - - [13/May/2024:05:20:40 +0000] "GET /?621 HTTP/1.1" 408 482 "-" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_11_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/53.0.2785.143 Safari/537.36"
192.168.6.167 - - [13/May/2024:05:20:40 +0000] "GET /?1575 HTTP/1.1" 408 482 "-" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_11_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/53.0.2785.143 Safari/537.36"
192.168.6.167 - - [13/May/2024:05:20:40 +0000] "GET /?1344 HTTP/1.1" 408 482 "-" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_11_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/53.0.2785.143 Safari/537.36"
192.168.6.167 - - [13/May/2024:05:20:40 +0000] "GET /?265 HTTP/1.1" 408 482 "-" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_11_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/53.0.2785.143 Safari/537.36"
192.168.6.167 - - [13/May/2024:05:20:40 +0000] "GET /?869 HTTP/1.1" 408 482 "-" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_11_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/53.0.2785.143 Safari/537.36"
192.168.6.167 - - [13/May/2024:05:20:40 +0000] "GET /?885 HTTP/1.1" 408 482 "-" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_11_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/53.0.2785.143 Safari/537.36"
192.168.6.167 - - [13/May/2024:05:20:41 +0000] "GET /?1043 HTTP/1.1" 408 482 "-" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_11_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/53.0.2785.143 Safari/537.36"
192.168.6.167 - - [13/May/2024:05:20:42 +0000] "GET /?496 HTTP/1.1" 408 482 "-" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_11_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/53.0.2785.143 Safari/537.36"
192.168.6.167 - - [13/May/2024:05:20:42 +0000] "GET /?1587 HTTP/1.1" 408 482 "-" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_11_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/53.0.2785.143 Safari/537.36"
192.168.6.167 - - [13/May/2024:05:20:43 +0000] "GET /?1686 HTTP/1.1" 408 482 "-" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_11_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/53.0.2785.143 Safari/537.36"
```

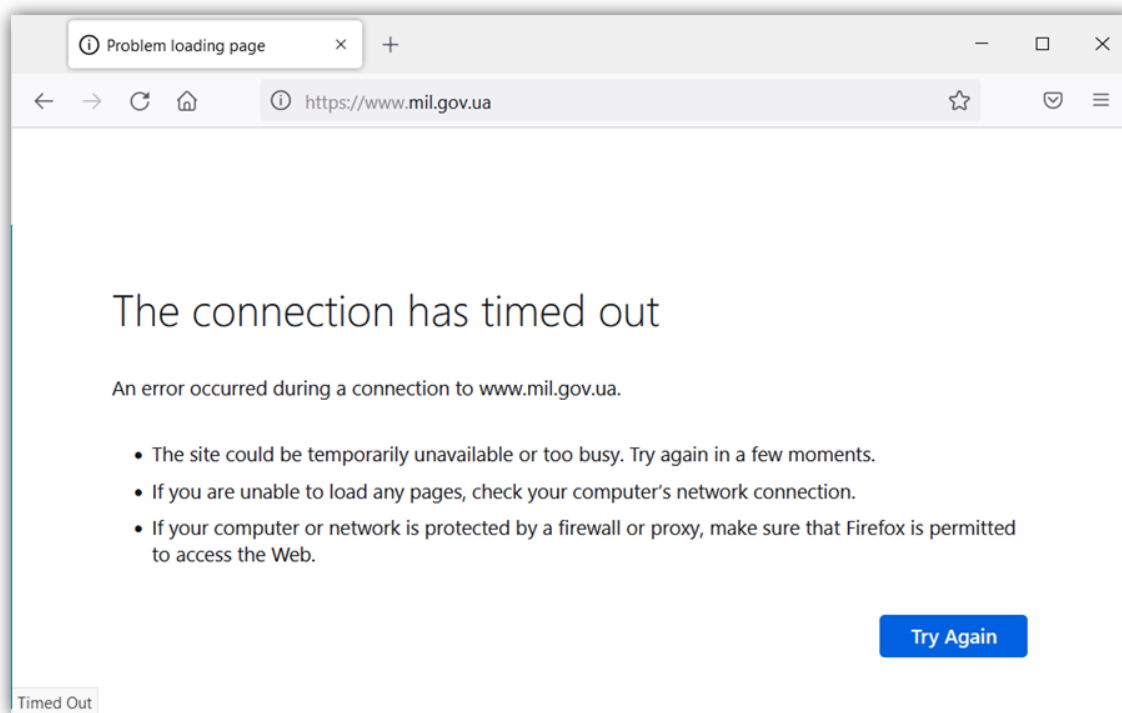
Hình 22 Mô phỏng tấn công DoS (2)

Ngoài gửi những yêu cầu đó, những giá trị đó được gửi liên tục. Khi kiểm tra số lượng log được lưu trong quá trình khi tấn công DoS xảy ra nhận thấy số lượng truy cập tăng lên một cách nhanh chóng và đáng kể trong một thời gian ngắn. Số lượng này có thể gấp nhiều lần số lượng của log. Đây cũng là một dấu hiệu nhận biết được một cuộc tấn công DoS.

```
root@modsec:/var/log/apache2# cat access.log | wc -l
1783
root@modsec:/var/log/apache2# cat access.log | wc -l
4181
```

Hình 23 Mô phỏng tấn công DoS (3)

Ảnh hưởng của cuộc tấn công ảnh hưởng tới những người dùng khi họ không thể truy cập website khi bị chiếm hết băng thông HTTP. Người dùng sẽ không thể truy cập vào một số dịch vụ hoặc toàn bộ trang web đang bị tấn công.



Hình 24 Kết quả tấn công Dos

4.2.2 Tạo rule bảo vệ

Bản chất của những cuộc tấn công DoS là gửi liên tục những gói tin để chiếm băng thông của những người dùng khác. Khiến cho những người dùng khác không thể truy cập vào những dịch vụ của trang web.

Để ngăn chặn và hạn chế một cuộc tấn công DoS có thể thực hiện hạn chế truy cập với những người dùng đáng nghi khi thực hiện gửi nhiều yêu cầu trong một thời gian ngắn. Để ngăn chặn có thể thực hiện chặn luôn IP đó để người dùng sở hữu IP không thể truy cập trang web. Cấu trúc của một đoạn quy tắc bảo vệ chặn yêu cầu từ một IP:

SecRule REMOTE_ADDR “^192\.168\.6\.154\$” “phase:1, id:10001, log, deny, status:403”

Thành phần	Giá trị	Miêu tả
Variable	REMOTE_ADDR	Giá trị là địa chỉ truy cập
Operator	^192\.168\.6\.154\$	Biểu thức thể hiện đây là địa chỉ IP 192.168.6.154
Tranform Function		
ID	10001	ID của rule là 10001

Action	Deny	Khi khớp mới điều kiện rule sẽ kích hoạt chặn yêu cầu từ IP.
Option	Phase1: Log Status:403	Giai đoạn một của quá trình kiểm tra Thực hiện ghi lại nhật ký Hiển thị trạng thái 403 khi truy cập

Bảng 10 Mô tả ModSec Rule chặn IP

Kiểm tra IP của máy kẻ tấn công bằng cách kiểm tra tệp nhật ký của máy chủ Apache kiểm tra tệp access.log phát hiện bất thường.

```
192.168.6.167 - - [14/May/2024:04:46:55 +0000] "GET /?142 HTTP/1.1" 400 483 "-" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_11_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/53.0.2785.143 Safari/537.36"
192.168.6.167 - - [14/May/2024:04:46:55 +0000] "GET /?76 HTTP/1.1" 400 483 "-" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_11_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/53.0.2785.143 Safari/537.36"
192.168.6.167 - - [14/May/2024:04:46:55 +0000] "GET /?943 HTTP/1.1" 400 483 "-" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_11_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/53.0.2785.143 Safari/537.36"
192.168.6.167 - - [14/May/2024:04:46:55 +0000] "GET /?1077 HTTP/1.1" 400 483 "-" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_11_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/53.0.2785.143 Safari/537.36"
192.168.6.167 - - [14/May/2024:04:46:57 +0000] "GET /?1220 HTTP/1.1" 400 483 "-" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_11_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/53.0.2785.143 Safari/537.36"
192.168.6.167 - - [14/May/2024:04:47:00 +0000] "GET /?1206 HTTP/1.1" 400 483 "-" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_11_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/53.0.2785.143 Safari/537.36"
192.168.6.167 - - [14/May/2024:04:47:00 +0000] "GET /?1996 HTTP/1.1" 400 483 "-" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_11_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/53.0.2785.143 Safari/537.36"
```

Hình 25 File access.log

Cấu hình quy tắc trong tệp cấu hình

```
# Test rules
# Block IP
SecRule REMOTE_ADDR "^192\.168\.6\.167$" "phase:1, id:10001, deny, status:403, msg:'IP blocked'"
```

Hình 26 Cấu hình Rule Block IP

Khởi động lại máy chủ và kiểm tra phản hồi. Thấy được IP đã bị chặn bởi ModSecurity.

Forbidden

You don't have permission to access this resource.

Apache/2.4.52 (Ubuntu) Server at 192.168.6.171 Port 80

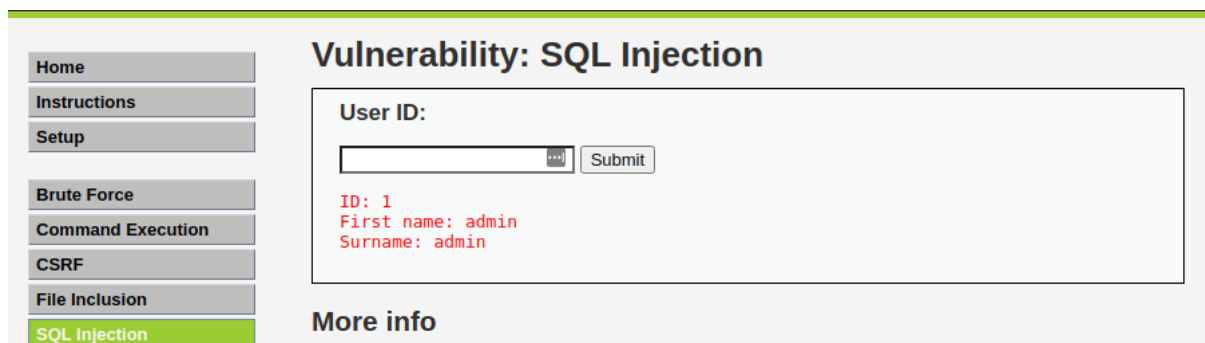
Hình 27 Kết quả thực thi

4.3 Tấn công và phòng thủ SQL Injection

4.3.1 Tấn công SQLi

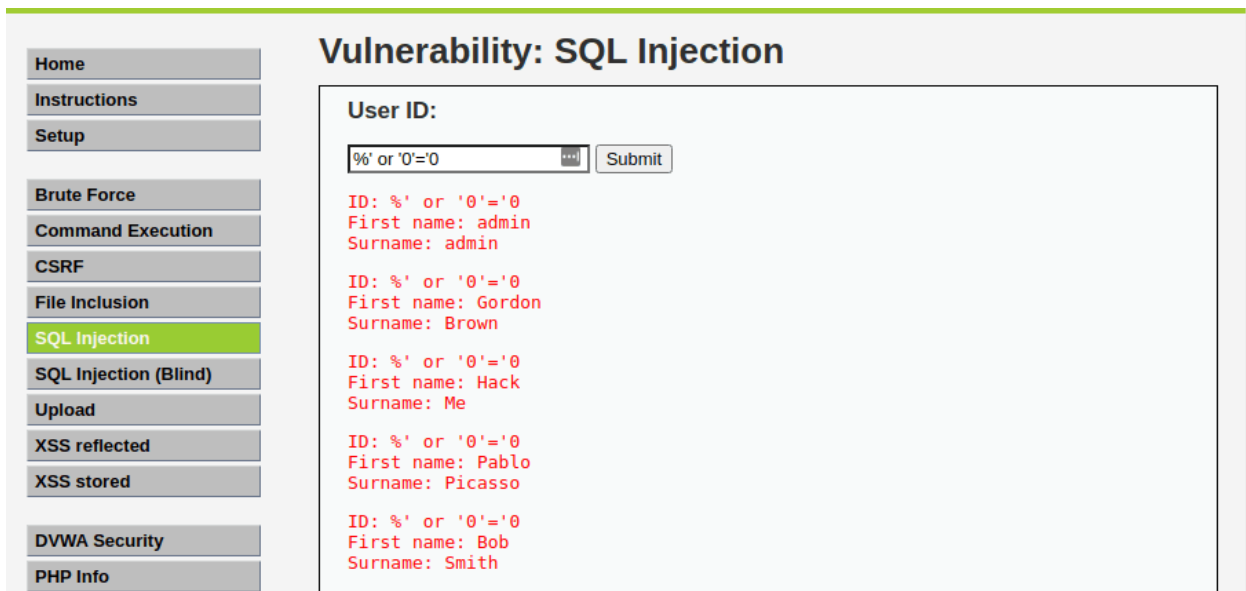
Bản chất của lỗ hổng SQL Injection là có thể chen và thực thi mã SQL mà không bị kiểm soát. Khi nhập thông tin ID của một người dùng ta sẽ được tên và họ của họ. Bản chất của câu lệnh SQL trong tác vụ này sử dụng:

SELECT first_name, last_name FROM users WHERE user_id = '1';



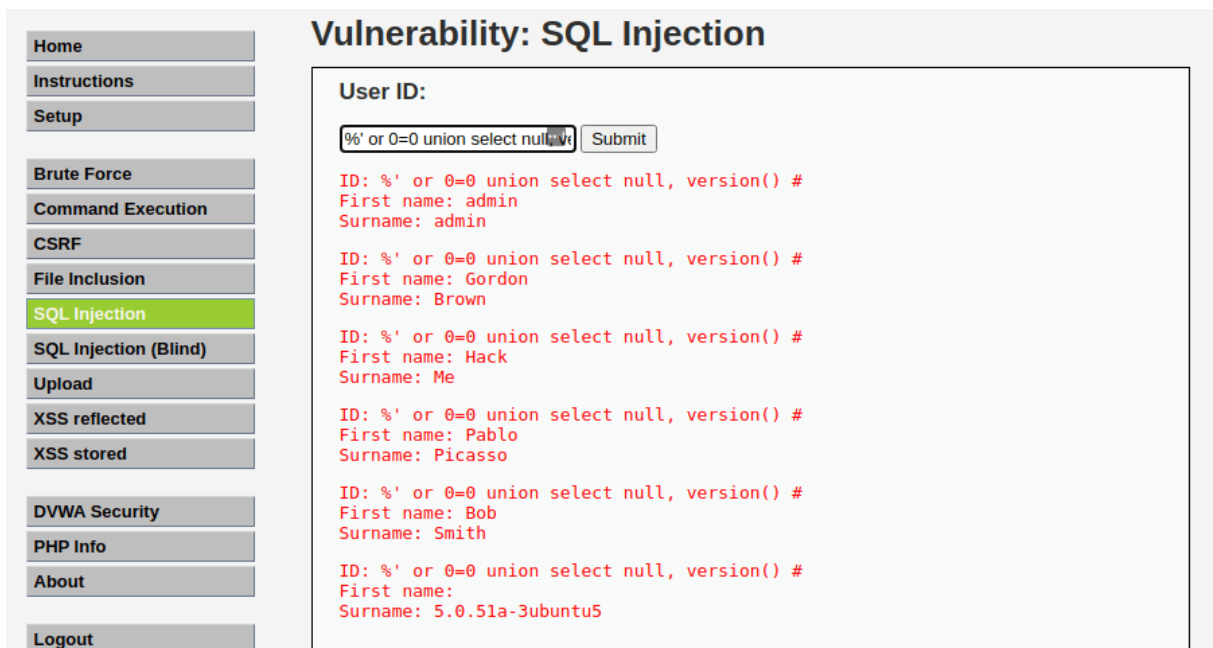
Hình 28 Mô phỏng tấn công SQL Injection (1)

Thử thực hiện kiểm tra với một vài ký tự đặc biệt: ***%'*** or ***'1'='1'***. Thấy được toàn bộ thông tin của những người dùng. Như vậy khi thêm vào đoạn mã ký tự trên câu lệnh SQL đã trở thành: ***SELECT first_name, last_name FROM users WHERE user_id = '%'*** or ***'1'='1'***; Khi user_id = “%” hệ thống sẽ nhận user_id đó là sai và or (hoặc), với or điều kiện trước thực hiện sai, hệ thống sẽ kiểm tra điều kiện phía sau. Khi ta để điều kiện đó là 1=1 thì điều kiện này luôn đúng, như vậy câu lệnh đó sẽ lấy toàn bộ thông tin về first_name và last_name từ bảng user.



Hình 29 Mô phỏng tấn công SQL Injection (2)

Khi kiểm tra tới đây thấy được có tồn tại lỗ hổng SQL Injection. Thực hiện khai thác thêm thông tin và đánh giá mức độ nguy hiểm của lỗi này. Thực hiện tiếp với việc khai thác thêm thông tin về phiên bản phần mềm sử dụng Union-base SQL để thực hiện khai thác thông tin bằng chuỗi: *%'' or 0=0 union select null, version() #*. Phần *version()* giúp hiển thị thông tin về phiên bản hiện tại của hệ thống.



Hình 30 Mô phỏng tấn công SQL Injection (3)

Ngoài ra có thể kiểm tra thông tin về người dùng cơ sở dữ liệu bằng đoạn mã khai thác: *%' or 0=0 union select null, user() #*. Chức năng user() cho phép cơ sở dữ liệu hiển thị thông tin người dùng.

The screenshot shows the DVWA interface with a sidebar on the left containing navigation links: Home, Instructions, Setup, Brute Force, Command Execution, CSRF, File Inclusion, SQL Injection (highlighted), SQL Injection (Blind), Upload, XSS reflected, XSS stored, DVWA Security, PHP Info, About, and Logout. The main content area is titled 'Vulnerability: SQL Injection' and features a 'User ID:' label and a text input field containing the payload *%' or 0=0 union select null, user() #*. A 'Submit' button is next to the input. Below the input, the results of the query are displayed in red text, showing the first name and surname for various user IDs.

ID	First name	Surname
%' or 0=0 union select null, user() #	admin	admin
%' or 0=0 union select null, user() #	Gordon	Brown
%' or 0=0 union select null, user() #	Hack	Me
%' or 0=0 union select null, user() #	Pablo	Picasso
%' or 0=0 union select null, user() #	Bob	Smith
%' or 0=0 union select null, user() #		root@localhost

Hình 31 Mô phỏng tấn công SQL Injection (4)

Hiển thị thông tin về cơ sở dữ liệu hiện tại: *%' or 0=0 union select null, user() #*

This screenshot is similar to the previous one, showing the DVWA interface with the same sidebar. The main content area is titled 'Vulnerability: SQL Injection'. The 'User ID:' label and the text input field are present, but the input field is empty. The 'Submit' button is next to it. Below the input, the results of the query are displayed in red text, showing the first name and surname for various user IDs, but the database name is not visible.

ID	First name	Surname
%' or 0=0 union select null, database() #	admin	admin
%' or 0=0 union select null, database() #	Gordon	Brown
%' or 0=0 union select null, database() #	Hack	Me
%' or 0=0 union select null, database() #	Pablo	Picasso
%' or 0=0 union select null, database() #	Bob	Smith
%' or 0=0 union select null, database() #		dvwa

Hình 32 Mô phỏng tấn công SQL Injection (5)

Hiển thị toàn bộ thông tin bảng trong information_schema, nó là bảng dữ liệu lưu thông tin của những bảng, những cột và những thông tin về cơ sở dữ liệu trong cơ sở dữ liệu MySQL.

Đoạn mã khai thác sử dụng: *%' and 1=0 union select null, table_name from information_schema.tables #*

Vulnerability: SQL Injection

User ID:

ID: %' and 1=0 union select null, table_name from information_schema.tables #
First name:
Surname: CHARACTER_SETS

ID: %' and 1=0 union select null, table_name from information_schema.tables #
First name:
Surname: COLLATIONS

ID: %' and 1=0 union select null, table_name from information_schema.tables #
First name:
Surname: COLLATION_CHARACTER_SET_APPLICABILITY

ID: %' and 1=0 union select null, table_name from information_schema.tables #
First name:
Surname: COLUMNS

ID: %' and 1=0 union select null, table_name from information_schema.tables #
First name:
Surname: COLUMN_PRIVILEGES

ID: %' and 1=0 union select null, table_name from information_schema.tables #
First name:
Surname: KEY_COLUMN_USAGE

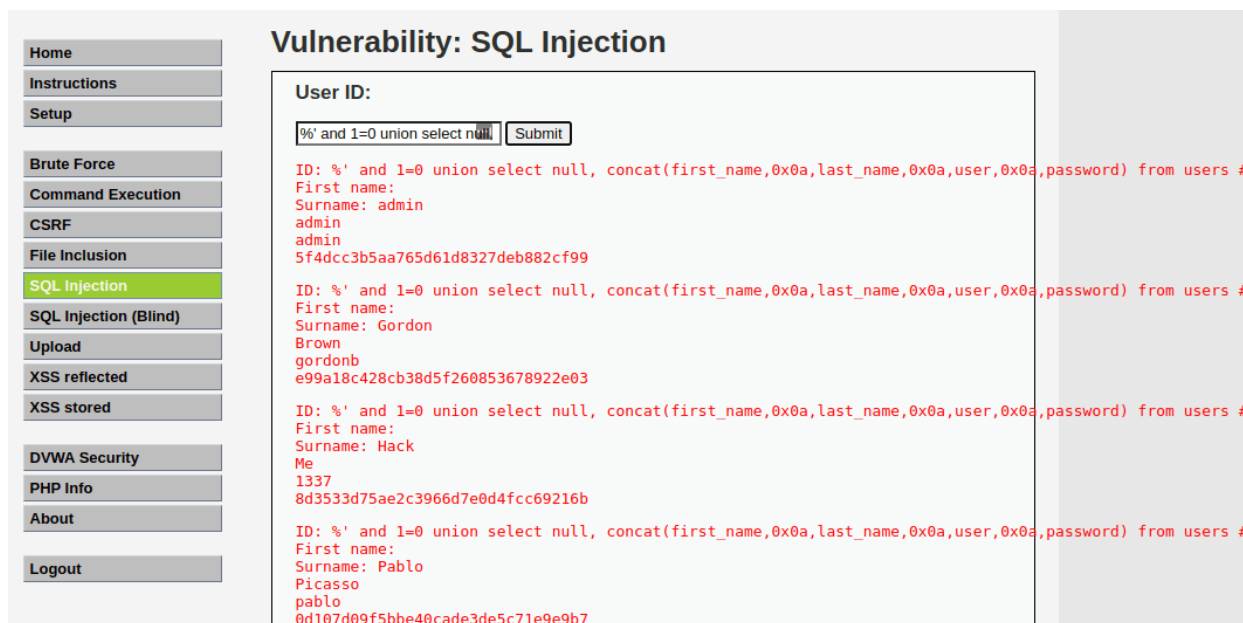
ID: %' and 1=0 union select null, table_name from information_schema.tables #
First name:

Hình 33 Mô phỏng tấn công SQL Injection (6)

Từ thông tin của bảng dữ liệu information_schema, tìm kiếm và khai thác những thông tin về những bảng dữ liệu chứa thông tin nhạy cảm, cụ thể ở đây là tìm kiếm bản liên quan đến thông tin người dùng.

Đoạn mã khai thác sử dụng: *%' and 1=0 union select null, table_name from information_schema.tables where table_name like 'user%'*#

Kết quả thu được thông tin về tài khoản và mật khẩu của người dùng.



Vulnerability: SQL Injection

User ID:

ID: %' and 1=0 union select null, concat(first_name,0x0a,last_name,0x0a,user,0x0a,password) from users #
First name:
Surname: admin
admin
admin
5f4dcc3b5aa765d61d8327deb882cf99

ID: %' and 1=0 union select null, concat(first_name,0x0a,last_name,0x0a,user,0x0a,password) from users #
First name:
Surname: Gordon
Brown
gordonb
e99a18c428cb38d5f260853678922e03

ID: %' and 1=0 union select null, concat(first_name,0x0a,last_name,0x0a,user,0x0a,password) from users #
First name:
Surname: Hack
Me
1337
8d3533d75ae2c3966d7e0d4fcc69216b

ID: %' and 1=0 union select null, concat(first_name,0x0a,last_name,0x0a,user,0x0a,password) from users #
First name:
Surname: Pablo
Picasso
pablo
0d107d09f5bbe40cade3de5c71e9e9b7

Hình 36 Hậu quả tấn công SQL Injection

4.3.2 Tạo rule bảo vệ

Từ mô phỏng tấn công có thể thấy được những đoạn mã độc hại để khai thác tấn công trang web sử dụng những câu lệnh SQL để thực hiện. Từ đó để phát hiện và ngăn chặn bắt thường các rule trong tường lửa cần phát hiện được những thông tin hay đoạn mã bao gồm những ký tự thường được sử dụng để khai thác trong cơ sở dữ liệu như select, delete, union, drop, insert into, like, or, alter, ‘, “, #, --. Khi phát hiện những ký tự đặc biệt này tường lửa sẽ thực hiện chặn những thông tin này tới máy chủ web.

Cấu trúc quy tắc bảo vệ cuộc tấn công SQL Injection:

```
SecRule ARGS "(select/union/from/where/having/order by/group by/#/';/|--)" \
    "id:1002,|
    t:none,|
    t:urlDecodeUni,|
    t:htmlEntityDecode,|
    t:compressWhiteSpace,|
    deny,|
    status:403, log,|
    msg:'SQL Injection Attempt' "
```

Thành phần	Giá trị	Miêu tả
Variable	ARGS	Giá trị thực hiện yêu cầu của người dùng
Operator	(select union from where having order by group by # ' ; --)	Những ký tự dùng để so sánh và đối chiếu với những yêu cầu của người dùng
id	1002	ID của quy tắc
Transform Function	t:none t:urlDecodeUni t:htmlEntityDecode t:compressWhiteSpace	Giữ nguyên dữ liệu yêu cầu. Giải mã những ký tự mã hoá HTML, xoá khoảng trắng.
Action	deny	Khi có yêu cầu khớp với quy tắc thực hiện chặn
Option	status :403 log msg: 'SQL Injection'	Hiện thị trạng thái 403 Có ghi lại nhật ký Tin nhắn ghi lại là SQL Injection

Bảng 11 Mô tả ModSec Rule chặn tấn công SQLi

Cấu hình quy tắc trong tệp cấu hình

```
# Block SQL Injection
SecRule ARGS "(select|union|from|where|having|order by|group by|#|'|;|--)" \
  "id:1002,\
  t:none,\
  t:urlDecodeUni,\
  t:htmlEntityDecode,\
  t:compressWhiteSpace,\
  deny,\
  status:403,\
  log,\
  msg:'SQL injection attempt'"
```

Hình 37 Cấu hình Rule chặn SQL Injection

Kiểm thử tấn công SQL Injection bằng ký tự #

Vulnerability: SQL Injection

User ID:

Hình 38 Kiểm thử SQL Injection

Forbidden

You don't have permission to access this resource.

Apache/2.4.52 (Ubuntu) Server at 192.168.6.171 Port 80

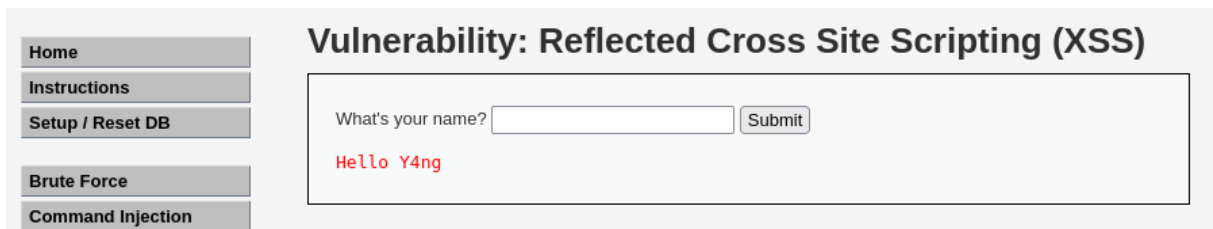
Hình 39 Kết quả kiểm thử

4.4 Tấn công và phòng thủ XSS

4.4.1 Tấn công XSS

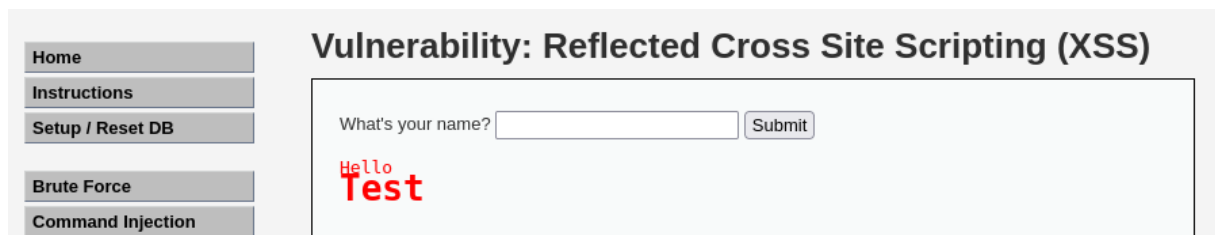
Bản chất của XSS là thực hiện chèn mã javascript độc hại vào trang website. Khi lập trình viên sử dụng những hàm và những đoạn mã không an toàn. Tuy vậy XSS có nhiều biến thể khác nhau như XSS reflected hay XSS stored.

Phát hiện dấu hiệu của lỗ hổng XSS khi thực hiện thay đổi giá trị của tham số đầu vào và xem xét đến những phản hồi của máy chủ tới những phản hồi đó để đưa ra kết luận. Khi thực hiện nhập tên vào ô và nhấn enter lập tức xuất hiện dòng chữ 'Hello + tên vừa nhập'.



Hình 40 Mô phỏng tấn công XSS (1)

Thực hiện kiểm tra với một đoạn mã HTML cơ bản `<h1>Test</h1>` và nhận thấy kết quả trả về bên dưới có sự thay đổi trong kích thước của chữ đã nhập. Như vậy có thể hoàn toàn thêm những đoạn mã khác để thực hiện kiểm tra lỗi XSS.



Hình 41 Mô phỏng tấn công XSS (2)

Mã nguồn của trang web cũng thay đổi theo dữ liệu đầu vào.

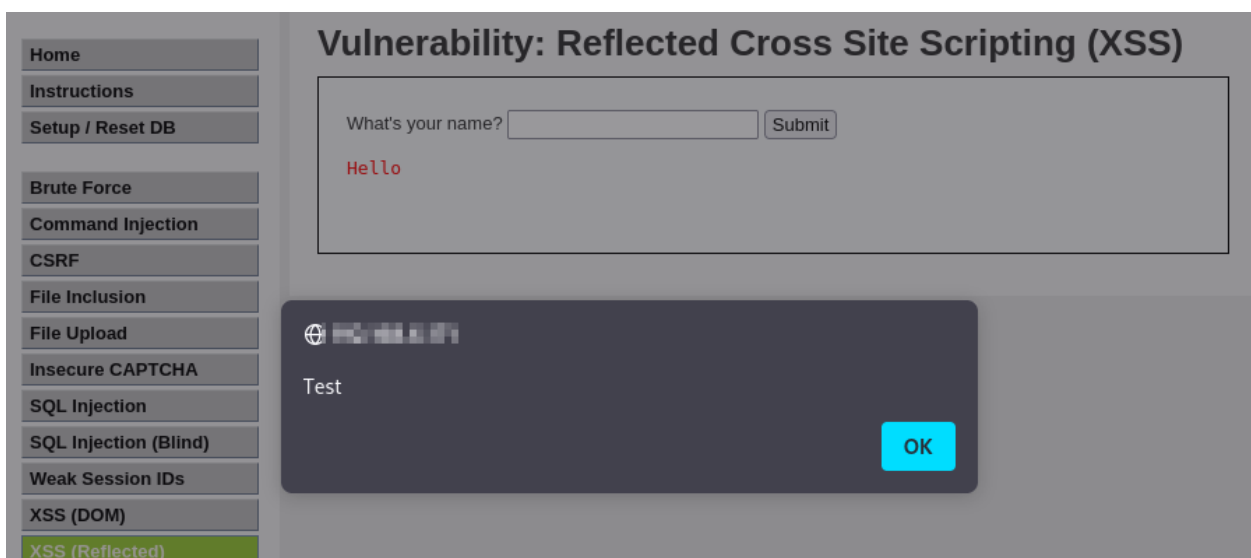
```

61 <div class="body_padded">
62   <h1>Vulnerability: Reflected Cross Site Scripting (XSS)</h1>
63
64   <div class="vulnerable_code_area">
65     <form name="XSS" action="#" method="GET">
66       <p>
67         What's your name?
68         <input type="text" name="name">
69         <input type="submit" value="Submit">
70       </p>
71     </form>
72     <pre>Hello <h1>Test<h1></pre>
73   </div>
74 </div>

```

Hình 42 Mô phỏng tấn công XSS (3)

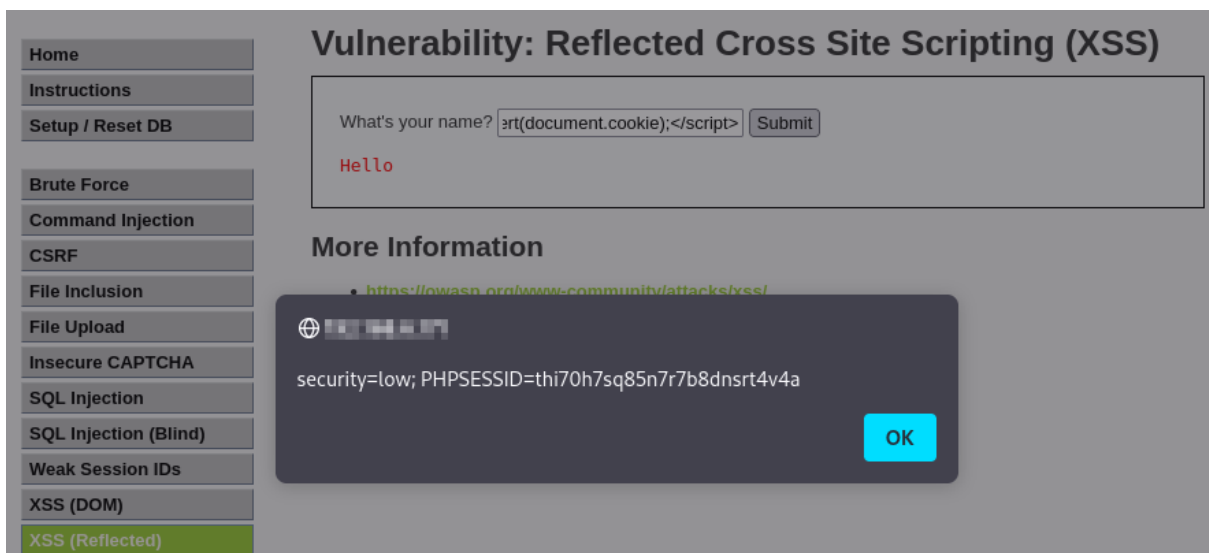
Khi nhận thấy trang web cho phép việc chèn những mã code từ đầu vào dữ liệu thực hiện kiểm tra với một đoạn mã Javascript cơ bản: `<script>alert('Test');</script>` đoạn code hiển thị một thông báo.



Hình 43 Mô phỏng tấn công XSS (4)

Ngoài hiển thị thông báo javascript tồn tại một số hàm để lấy những thông tin nhạy cảm phục vụ cho những lập trình viên khi phát triển phần mềm. Nhưng khi nó bị lợi dụng vào tay những kẻ tấn công nó có thể gây ra lộ thông tin. Với đoạn mã sau người dùng có thể lấy cookie của người dùng khi kẻ tấn công khai thác lỗi này.

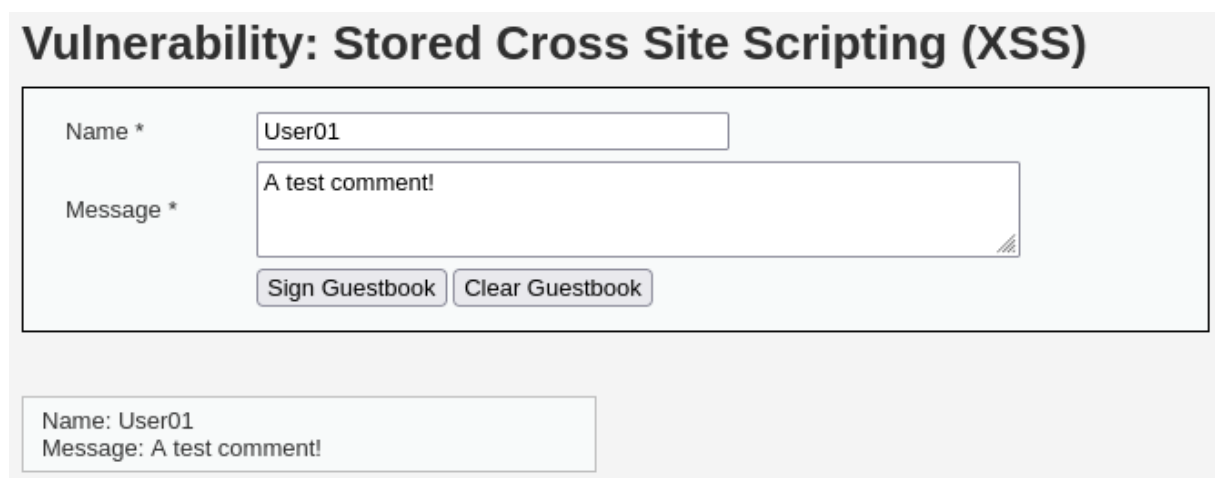
Đoạn mã thực hiện: `<script>alert(document.cookie);</script>`



Hình 44 Mô phỏng tấn công XSS (5)

Trong một số trường hợp có những đoạn văn bản được lưu trữ trong cơ sở dữ liệu mà không thực hiện lọc dữ liệu đầu vào có thể dẫn đến việc sẽ có thể khiến kẻ tấn công khai thác những lỗ hổng để chèn mã độc javascript vào trang web.

Một mẫu bình luận với hai trường dữ liệu cần nhập vào để thực hiện bình luận. Ví dụ ta thử bình luận: Name: User01, Message: A test comment!



Hình 45 Mô phỏng tấn công XSS (6)

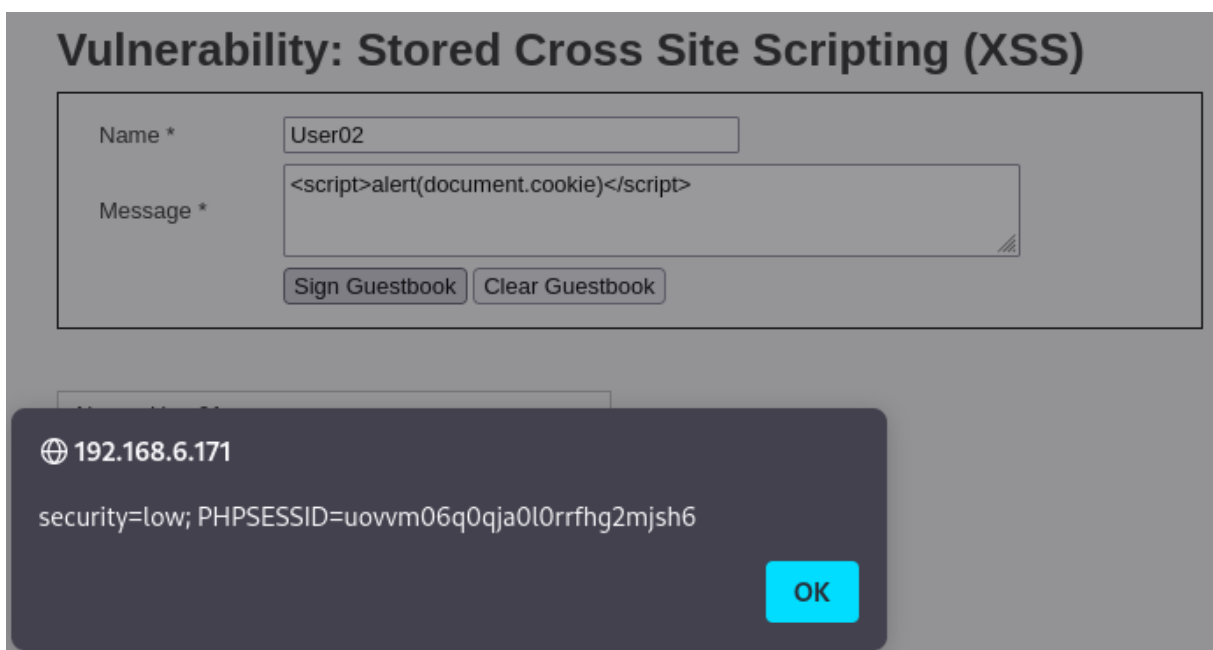
Thu được kết quả là bình luận đã được lưu lại và khi đăng nhập bằng tài khoản khác cũng sẽ thấy những bình luận này. Thực hiện thay đổi bình luận chút bằng cách thêm những thẻ HTML và hai trường nhập thông tin.

Với trường Name kiểm thử với đoạn mã `<h1>A</h1>` còn lại với trường Message chèn đoạn mã `<h1>Test XSS</h1>` và kiểm tra kết quả.



Hình 46 Mô phỏng tấn công XSS (7)

Thấy được thay đổi trong quá trình chèn thêm đoạn mã vào trong hai trường dữ liệu như vậy dự đoán có tồn tại lỗ hổng XSS. Kiểm thử với những đoạn mã javascript với Name là User2 với Message: `<script>alert(document.cookie)</script>` đoạn mã này lấy cookie hiện tại của người dùng đăng nhập. Thay vì sử dụng riêng lẻ thì những lỗi này thường kết hợp với những lỗ hổng bảo mật khác để tăng hiệu quả.



Hình 47 Mô phỏng tấn công XSS (8)

4.4.2 Tạo rule bảo vệ

Một tấn công XSS sử dụng những đoạn mã javascript để thực hiện khai thác và tấn công hệ thống website. Dựa vào một số đặc điểm của ngôn ngữ lập trình javascript, thực hiện chặn là lọc những yêu cầu có những ký tự có chứa mã code. Một số ký tự và thẻ thường xuất hiện trong những cuộc tấn công XSS. Một số thẻ như <script>, </script>, alert, document, cookie, >, <, :, #. Tạo quy tắc để chặn những ký tự nhạy cảm này từ người dùng.

Cấu trúc của quy tắc bảo vệ cuộc tấn công XSS:

SecRule ARGS "(<script></script>|script|alert|document|cookie|<|>|;|&|'|/|#)" \

id:1003,

t:none,

t:urlDecodeUni,

t:htmlEntityDecode,

t:compressWhiteSpace,

deny,

status:403, log,

msg:'XSS Attack'"

Thành phần	Giá trị	Miêu tả
Variable	ARGS	Giá trị thực hiện yêu cầu
Operator	(<script></script> script alert document cookie < > ; & ' / #)	Những ký tự dùng để so sánh và đối chiếu với những yêu cầu của người dùng
id	1003	ID của quy tắc
Transform Function	t:none t:urlDecodeUni t:htmlEntityDecode t:compressWhiteSpace	Giữ nguyên dữ liệu yêu cầu. Giải mã những ký tự mã hoá HTML, xoá khoảng trắng.
Action	deny	Chặn yêu cầu
Option	status :403 log msg: 'XSS Attack'	Hiện thị trạng thái 403 Có ghi lại nhật ký Tin nhắn ghi lại là XSS Attack

Bảng 12 Mô tả ModSec Rule chặn tấn công XSS

Cấu hình quy tắc trong tệp cấu hình

```
#Block XSS
SecRule ARGS "(<script>|</script>|script|alert|document|cookie|<|>|;|&|'|#)" \
  "id:1003,\
  t:none,\
  t:urlDecodeUni,\
  t:htmlEntityDecode,\
  t:compressWhiteSpace,\
  deny,\
  status:403,\
  log,\
  msg:'XSS Attack'"
```

Hình 48 Cấu hình Rule chặn

Thực hiện kiểm thử tấn công XSS bằng đoạn mã

Vulnerability: Reflected Cross Site Scripting (XSS)

What's your name?

Hình 49 Kiểm thử XSS

Forbidden

You don't have permission to access this resource.

Apache/2.4.52 (Ubuntu) Server at 192.168.6.171 Port 80

Hình 50 Kết quả kiểm thử

Kết luận

Hiện nay các hình thức tấn công vào máy chủ web ngày càng tinh vi và phức tạp hơn, việc bảo vệ website bằng firewall chỉ là một phần trong một quy trình để thực hiện bảo mật thông tin. Bảo mật thông tin là cả một quá trình và gồm nhiều bước khác nhau để đảm bảo được an toàn dữ liệu. Bắt đầu từ người dùng và quan trọng hơn đó là người quản trị hệ thống đảm bảo những yếu tố, quy tắc an toàn thông tin.

Để đảm bảo an toàn thông tin phải bắt đầu từ người dùng sau đó mới tới những hệ thống hỗ trợ như hệ thống phát hiện xâm nhập IDS hay hệ thống firewall. Đó chỉ là công cụ hỗ trợ phát hiện và ngăn chặn những mối nguy hại mà ta đã biết.

Bài làm đã trình bày được những thông tin về lỗ hổng bảo mật web hay thế nào là một tường lửa ứng dụng web nói chung và tường lửa ModSecurity nói riêng. Tuy vậy bài làm cũng còn nhiều hạn chế về kiến thức và về những thông tin tìm hiểu được. Để làm chủ một công nghệ là một vấn đề lâu dài và cần tới thời gian kiểm thử lâu dài. Những kiến thức trong bài chỉ đạt được ở mức cơ bản và giới thiệu, bài làm được thực hiện trong môi trường thử nghiệm nên không tránh được việc áp dụng vào thực tế còn hạn chế.

Phương hướng cho tương lai sẽ tiếp tục tìm hiểu và nghiên cứu công nghệ bảo mật website, mở rộng tìm kiếm kiến thức và tài liệu để quản trị tốt những công nghệ sử dụng. Đảm bảo sự bảo mật của một trang web.

Tài liệu tham khảo

Tài liệu:

ModSecurity Handbook – Ivan Ristic

ModSecurity Rule Writing Workshop – Ivan Ristic

Trang web

ModSecurity Wiki : <https://github.com/owasp-modsecurity/ModSecurity/wiki/>

Mã nguồn ModSecurity: <https://github.com/owasp-modsecurity/ModSecurity>

Mã nguồn DVWA: <https://github.com/digininja/DVWA>

Kali linux: <https://www.kali.org/>

Apache http Server Documentation: <https://httpd.apache.org/docs/>