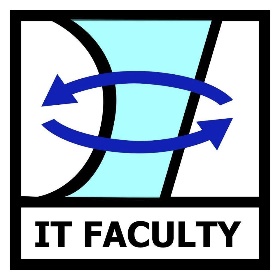


**ĐẠI HỌC ĐÀ NẴNG**

**TRƯỜNG ĐẠI HỌC BÁCH KHOA**

**KHOA CÔNG NGHỆ THÔNG TIN**

**🙟🕮🙝**



**ĐỒ ÁN MÔN HỌC**

**LẬP TRÌNH MẠNG**

**Đề tài:**

**IPTABLES VÀ XÂY DỰNG FIREWALL TRÊN LINUX**

**Giảng viên hướng dẫn: ThS. Trần Hồ Thuỷ Tiên**

**Sinh viên thực hiện: Lê Đình Sinh**

**Mã số sinh viên: 102150295**

**Lớp sinh hoạt: 15TCLC2**

**Đà Nẵng, tháng 12 năm 2018**

**LỜI CẢM ƠN**

Với những kiến thức nền tảng bắt đầu còn hạn hẹp, em không thể tự bắt đầu nghiên cứu mà không nhờ một sự trợ giúp dù lớn hay nhỏ nào cả. Trong suốt thời gian bắt đầu nhận đồ án đến nay, em cũng như các bạn trong cùng nhóm đồ án môn học **Lập trình mạng** đều nhận được sự ân cần, chia sẻ những kinh nghiệm quý báu trong chuyên môn đến từ cô Trần Hồ Thủy Tiên.

Một lần nữa, em xin gửi lời cảm ơn sâu sắc nhất đến cô Trần Hồ Thủy Tiên đã tận tụy bên chúng em để cùng nhau nghiên cứu các đề tài trong đồ án môn học lần này.

**Đà Nẵng, ngày 28 tháng 12 năm 2018**

**Sinh viên thực hiện**

**Lê Đình Sinh**

**MỤC LỤC**

[MỞ ĐẦU 1](#_Toc533540570)

[Bối cảnh chọn đề tài 1](#_Toc533540571)

[Mục tiêu của đồ án 1](#_Toc533540572)

[Cấu trúc của bài báo cáo 1](#_Toc533540573)

[CHƯƠNG I: IPTABLES VÀ NGUYÊN LÝ TƯỜNG LỬA CƠ BẢN 2](#_Toc533540574)

[1.1 Tìm hiểu nguyên lí hoạt động của tường lửa: 2](#_Toc533540575)

[1.1.1 Tường lửa (firewall) là gì? 2](#_Toc533540576)

[1.1.2 Lý do cần tường lửa với các máy tính có nối mạng 2](#_Toc533540577)

[1.1.3 Các loại tường lửa phổ biến hiện nay 2](#_Toc533540578)

[1.1.4 Nhiệm vụ chính của tường lửa 3](#_Toc533540579)

[1.1.5 Nguyên lý hoạt động của tường lửa 4](#_Toc533540580)

[1.1.6 Tưởng lửa trong giao thức TCP/IP 5](#_Toc533540581)

[1.2 Giới thiệu về iptables. 6](#_Toc533540582)

[1.2.1 Các bảng trong iptables 6](#_Toc533540583)

[1.2.2 Các chain trong iptables 7](#_Toc533540584)

[1.2.3 Các rules trong chain 10](#_Toc533540585)

[1.2.4 Các trạng thái của kết nối 11](#_Toc533540586)

[1.3 Các kỹ thuật lập lệnh cơ bản trong iptables 11](#_Toc533540587)

[1.3.1 iptables và Netfilter 11](#_Toc533540588)

[1.3.2 Cách đổi địa chỉ IP động (dynamic NAT) 12](#_Toc533540589)

[1.3.3 Đổi địa chỉ IP và đóng giả IP (NAT and IP masquerade): 12](#_Toc533540590)

[1.4 Một số câu lệnh, tham số cơ bản trong lập lệnh iptables 13](#_Toc533540591)

[1.4.1 Các tham số switching 13](#_Toc533540592)

[1.4.2 Các dịch vụ có sẵn trong Linux 14](#_Toc533540593)

[CHƯƠNG II: PHÂN TÍCH MỘT SỐ PHƯƠNG THỨC TẤN CÔNG 16](#_Toc533540594)

[2.1 IP spoofing (Giả mạo IP) 16](#_Toc533540595)

[2.1.1 Lý thuyết của giả mạo IP 16](#_Toc533540596)

[2.1.2 Nguyên lý của giả mạo IP 16](#_Toc533540597)

[2.1.3 Sử dụng giả mạo IP để làm gì 17](#_Toc533540598)

[2.1.4 Nhận biết một gói tin đã được giả mạo 17](#_Toc533540599)

[2.1.5 Kết luận 17](#_Toc533540600)

[2.2 Stealth scan TCP/SYN (Tấn công tràn SYN) 18](#_Toc533540601)

[2.2.1 Tấn công từ chối dịch vụ (DoS) là gì? 18](#_Toc533540602)

[2.2.2 Lý thuyết về quét tràn SYN 19](#_Toc533540603)

[2.2.2 Kết luận về quét SYN 21](#_Toc533540604)

[2.2 Ping flood (Tràn ping) 21](#_Toc533540605)

[2.2.1 Nguyên lý của tràn ping 21](#_Toc533540606)

[2.2.3 Một cuộc tấn công điển hình 22](#_Toc533540607)

[2.2.4 Các vấn đề thuộc giao thức ICMP 23](#_Toc533540608)

[2.3 Tấn công từ chối dịch vụ qua HTTP (HTTP DoS) 23](#_Toc533540609)

[2.3.1 Nguyên lý tấn công từ chối dịch vụ qua HTTP (HTTP Flood) 23](#_Toc533540610)

[2.3.2 Vấn đề toàn cầu với HTTP DDos 24](#_Toc533540611)

[CHƯƠNG III: THỬ NGHIỆM CÁC Ý TƯỞNG PHÒNG THỦ 27](#_Toc533540612)

[3.1 Phòng thủ tấn công giả mạo IP 27](#_Toc533540613)

[3.1.1 Ý tưởng 27](#_Toc533540614)

[3.1.2 Cài đặt 27](#_Toc533540615)

[3.2 Phòng thủ tấn công tràn SYN 28](#_Toc533540616)

[3.2.1 Ý tưởng 28](#_Toc533540617)

[3.2.1 Lập lệnh mẫu: 29](#_Toc533540618)

[3.3 Phòng thủ tấn công tràn ICMP 29](#_Toc533540619)

[3.3.1 Chọn lựa và giới hạn ICMP 29](#_Toc533540620)

[3.3.2 Cản ICMP hoặc cản ICMP “một nửa” 30](#_Toc533540621)

[3.4 Phòng thủ tấn công HTTP DdoS 31](#_Toc533540622)

[3.4.1 Ý tưởng phòng thủ 31](#_Toc533540623)

# MỞ ĐẦU

## Bối cảnh chọn đề tài

Chúng ta đang sống trong thời đại của cuộc cách mạng Internet kết nối toàn cầu. Mạng Internet đã mở ra những cơ hội vô cùng to lớn cho con người trong công cuộc hành trình tìm kiếm tri thức, nhưng đồng thời cũng phát sinh một vấn đề quan trọng hơn đó là đảm bảo sự an toàn của người sử dụng trên không gian mạng công khai đó. Trong vài năm trở lại đây xu hướng tấn công có chủ đích (APT) đang diễn biến hết sức phức tạp trên diện rộng. Đây là hình thức tấn công tinh vi và rất khó phát hiện do kẻ tấn công sử dụng các kỹ thuật mới để ẩn nấp và những cuộc tấn công này nhằm vào những người dùng hay các hệ thống quan trọng nhằm đánh cắp thông tin, phá hoại hệ thống và có thể xem là mối rủi ro nguy hiểm thường trực hiện nay trên Internet không chỉ ở Việt Nam và trên thế giới. Không nằm ngoài xu thế đó thì đây vẫn là xu hướng chính và cần tiếp tục được quan tâm và chú trọng trong năm 2016.

Vì vậy, việc chọn đề tài **IPTABLES VÀ XÂY DỰNG FIREWALL TRÊN LINUX** trong thời điểm bùng nổ về Internet kết nối toàn cầu mang lại rất nhiều kiến thức về an toàn thông tin, đồng thời tăng khả năng sử dụng các công cụ hỗ trợ, mức độ hiểu biết về các phương thức tấn công cơ bản hiện nay, từ đó tạo tiền đề cho việc nghiên cứu chuyên sâu vào các giá trị cốt lõi của chúng.

## Mục tiêu của đồ án

Giúp người đọc phần nào hình dung được nguyên lý hoạt động của một chương trình Firewall điển hình, cùng với những phương thức tấn công căn bản và cách ngăn chặn chúng. Người dùng có thể tự cài đặt Firewall này vào hệ thống Web-server của mình, từ đó phát triển lên một hệ thống tốt hơn và qui mô hơn.

## Cấu trúc của bài báo cáo

Bài báo cáo được chia thành 4 chương gồm:

* Chương I: iptables và nguyên lý một tường lửa cơ bản.
* Chương II: Phân tích một số phương thức tấn công thông thường.
* Chương III: Thử nghiệm cài đặt các ý tưởng phòng thủ
* Chương IV: Kết luận và hướng mở rộng cho đề tài.

Kèm theo trong báo cáo là phần các tài liệu tham khảo, phụ lục và code chương trình trong đĩa CD được đính kèm theo.

Trong quá trình nghiên cứu và làm báo cáo, chắc chắn còn những thiếu sót nhất định. Em rất mong nhận được sự góp ý, chỉnh sửa đến từ hội đồng chấm đồ án.

Em xin chân thành cảm ơn và chúc buổi bảo vệ diễn ra thành công tốt đẹp.

# CHƯƠNG I: IPTABLES VÀ NGUYÊN LÝ TƯỜNG LỬA CƠ BẢN

## 1.1 Tìm hiểu nguyên lí hoạt động của tường lửa:

### 1.1.1 Tường lửa (firewall) là gì?

Tường lửa là một hệ thống an ninh mạng, dựa trên cả phần cứng lẫn phần mềm của thiết bị, sử dụng một bộ các quy tắc điểm kiểm soát traffic vào ra trong khi máy chủ đang hoạt động trên mạng. Tường lửa hoạt động như một rào chắn giữa mạng an toàn và mạng không an toàn. Nó kiểm soát các truy cập đến nguồn lực của mạng thông qua một mô hình kiểm soát chủ động. Nghĩa là, chỉ những traffic phù hợp với chính sách (policy) được định nghĩa trong tường lửa mới được truy cập vào mạng, mọi traffic khác đều bị từ chối.

### 1.1.2 Lý do cần tường lửa với các máy tính có nối mạng

Mỗi máy tính trực tuyến lại có một chữ ký điện tử riêng, được gọi là Internet Protocol address (hay còn gọi là địa chỉ IP). Nếu không có tường lửa hỗ trợ, nó giống như việc mở toang hết cửa nhà ra và mời khách lạ vào. Họ sẽ mặc nhiên được khám phá từng ngóc ngách, lấy từng thông tin riêng tư của bạn. Firewall dễ hiểu là cửa nhà, đúng chìa khoá (policy) thì được vào. Tất nhiên, vẫn có những kẻ vào mà không cần chìa.

Firewall có thể là thiết bị phần cứng hoặc chương trình phần mềm chạy trên host bảo đảm hoặc kết hợp cả hai. Trong mọi trường hợp, nó phải có ít nhất hai giao tiếp mạng, một cho mạng mà nó bảo vệ, một cho mạng bên ngoài. Firewall có thể là gateway hoặc điểm nối liền giữa hai mạng, thường là một mạng riêng và một mạng công cộng như là Internet. Các firewall đầu tiên là các router đơn giản.

### 1.1.3 Các loại tường lửa phổ biến hiện nay

#### a) Tường lửa cá nhân (Personal Firewall):

Loại này được thiết kế để bảo vệ một máy tính trước sự truy cập trái phép từ bên ngoài. Bên cạnh đó thì tường lửa cá nhân còn được tích hợp thêm tính năng như theo dõi các phần mềm chống virus, phần mềm chống xâm nhập để bảo vệ dữ liệu. Một số tường lửa cá nhân thông dụng như: Microsoft Internet connection firewall, Symantec personal firewall, Cisco Security Agent…. Loại tường lửa này thì thích hợp với cá nhân bởi vì thông thường họ chỉ cần bảo vệ máy tính của họ, thường được tích hợp sẵn trong máy tính Laptop, máy tính PC..

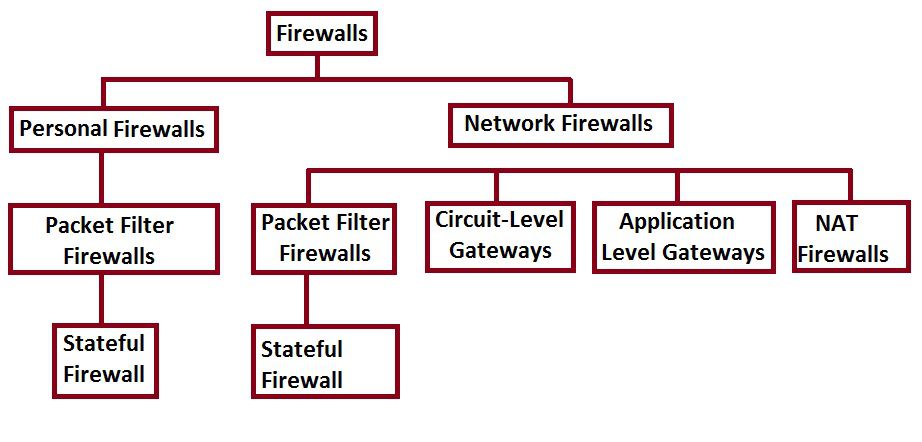
#### b) Tường lửa mạng (Network Firewall):

Được thiết kế ra để bảo vệ các host trong mạng trước sự tấn công từ bên ngoài. Chúng ta có các Appliance-Based network Firewalls như Cisco PIX, Cisco ASA, Juniper NetScreen firewall, Nokia firewalls, Symantec’s Enterprise Firewall. Hoặc một số ví dụ về Software-Base firewalls include Check Point’s Firewall, Microsoft ISA Server, Linux-based IPTables.

Điểm khác nhau giữa 2 loại tường lửa là tường lửa cá nhân chỉ bảo vệ cho một máy tính duy nhất, còn tưởng lửa mạng sẽ bảo vệ toàn cục cho các máy ở trong mạng đó.

Trong đó, hệ thống tường lửa mạng được cấu tạo bởi các thành phần chính như sau:

* Bộ lọc Packet (Packet- Filtering Router).
* Cổng ứng dụng ( đó là Application-Level Gateway hay Proxy Server).
* Cổng mạch (Circuite Level Gateway).



Hình 1 Các thành phần chính trong tường lửa

### 1.1.4 Nhiệm vụ chính của tường lửa

Tường lửa hỗ trợ máy tính kiểm soát luồng thông tin giữa intranet và internet, nó sẽ quyết định dịch vụ nào từ bên trong được phép truy cập ra bên ngoài, những người nào bên ngoài được phép truy cập vào bên trong hệ thống, hay là giới hạn truy cập những dịch vụ bên ngoài của những người bên trong hệ thống, mình lấy ví dụ như giới hạn trang Facebook, tất cả những người trong hệ thống sẽ không thể truy cập vào được mạng xã hội này. Sau đây là một số nhiệm vụ chính của tường lửa:

* Cho phép hoặc vô hiệu hóa các dịch vụ truy cập ra bên ngoài, đảm bảo thông tin chỉ có trong mạng nội bộ.
* Cho phép hoặc vô hiệu hóa các dịch vụ bên ngoài truy cập vào trong.
* Phát hiện và ngăn chặn các cuộc tấn công từ bên ngoài.
* Hỗ trợ kiểm soát địa chỉ truy cập (bạn có thể đặt lệnh cấm hoặc là cho phép).
* Kiểm soát truy cập của người dùng.
* Quản lý và kiểm soát luồng dữ liệu trên mạng.
* Xác thực quyền truy cập.
* Hỗ trợ kiểm soát nội dung thông tin và gói tin lưu chuyển trên hệ thống mạng.
* Lọc các gói tin dựa vào địa chỉ nguồn, địa chỉ đích và số Port ( hay còn cổng), giao thức mạng.
* Người quản trị có thể biết được kẻ nào đang cố gắng để truy cập vào hệ thống mạng.
* Tường lửa hoạt động như một proxy trung gian.
* Bảo vệ tài nguyên của hệ thống bởi các mối đe dọa bảo mật.
* Cân bằng tải: người dùng có thể sử dụng nhiều đường truyền internet cùng một lúc, việc chia tải sẽ giúp đường truyền internet ổn định hơn rất nhiều.
* Tính năng lọc ứng dụng cho phép ngăn chặn một số ứng dụng mà người dùng muốn. Ví dụ như Facebook Messenger, Skype, Zalo…

Một firewall khảo sát tất cả các luồng lưu lượng giữa hai mạng để xem nó có đạt chuẩn hay không. Nếu nó đạt, nó được định tuyến giữa các mạng, ngược lại nó bị hủy. Một bộ lọc firewall lọc cả lưu lượng ra lẫn lưu lượng vào. Nó cũng có thể quản lý việc truy cập từ bên ngoài vào nguồn tài nguyên mạng bên trong. Nó có thể được sử dụng để ghi lại tất cả các cố gắng để vào mạng riêng và đưa ra cảnh báo nhanh chóng khi kẻ thù hoặc kẻ không được phân quyền đột nhập.

Firewall có thể lọc các gói dựa vào địa chỉ nguồn, địa chỉ đích và số cổng của chúng. Điều này còn được gọi là lọc địa chỉ. Firewall cũng có thể lọc các loại đặc biệt của lưu lượng mạng. Điều này được gọi là lọc giao thức bởi vì việc ra quyết định cho chuyển tiếp hoặc từ chối lưu lượng phụ thuộc vào giao thức được sử dụng, ví dụ HTTP, FTP hoặc Telnet. Firewall cũng có thể lọc luồng lưu lượng thông qua thuộc tính và trạng thái của gói.

Nếu có thể giữ kẻ phá hoại trong một thời gian, người quản trị có thể lần theo dấu vết của họ. Ví dụ, có thể dùng lệnh finger để theo vết kẻ tấn công hoặc tạo tập tin “bẫy mồi” để họ phải mất thời gian truyền lâu, sau đó theo vết việc truyền tập tin về nơi của kẻ tấn công qua kết nối Internet.

### 1.1.5 Nguyên lý hoạt động của tường lửa

Firewall hoạt động chặt chẽ với giao thức TCP/IP, vì giao thức này làm việc theo thuật toán chia nhỏ các dữ liệu nhận được từ các ứng dụng trên mạng, hay nói chính xác hơn là các dịch vụ chạy trên các giao thức (Telnet, SMTP, DNS, SMNP, NFS …) thành các gói dữ liệu (data packets) rồi gán cho các packet này những địa chỉ có thể nhận dạng, tái lập lại ở đích cần gửi đến, do đó các loại Firewall cũng liên quan rất nhiều đến các packet và những con số địa chỉ của chúng.

Bộ lọc packet cho phép hay từ chối mỗi packet mà nó nhận được. Nó kiểm tra toàn bộ đoạn dữ liệu để quyết định xem đoạn dữ liệu đó có thỏa mãn một trong số các luật lệ của lọc packet hay không. Các luật lệ lọc packet này là dựa trên các thông tin ở đầu mỗi packet (header), dùng để cho phép truyền các packet đó ở trên mạng. Bao gồm:

• Địa chỉ IP nơi xuất phát (Source)  
• Địa chỉ IP nơi nhận ( Destination)  
• Những thủ tục truyền tin (TCP, UDP, ICMP, IP tunnel …)  
• Cổng TCP/UDP nơi xuất phát  
• Cổng TCP/UDP nơi nhận  
• Dạng thông báo ICMP  
• Giao diện packet đến  
• Giao diện packet đi

Nếu packet thỏa các luật lệ đã được thiết lập trước của Firewall thì packet đó được chuyển qua, nếu không thỏa thì sẽ bị loại bỏ. Việc kiểm soát các cổng làm cho Firewall có khả năng chỉ cho phép một số loại kết nối nhất định được phép mới vào được hệ thống mạng cục bộ. Cũng nên lưu ý là do việc kiểm tra dựa trên header của các packet nên bộ lọc không kiểm soát được nội dụng thông tin của packet. Các packet chuyển qua vẫn có thể mang theo những hành động với ý đồ ăn cắp thông tin hay phá hoại của kẻ xấu. Trong các phần sau chúng ta sẽ cùng tìm hiểu các kỹ thuật để vượt tường lửa.

### 1.1.6 Tưởng lửa trong giao thức TCP/IP

Firewall hoạt động ở các lớp khác nhau sử dụng các chuẩn khác nhau để hạn chế lưu lượng. Lớp thấp nhất mà firewall hoạt động là lớp 3. Trong mô hình OSI đây là lớp mạng. Trong mô hình TCP/IP đây là lớp IP (Internet Protocol). Lớp này có liên quan tới việc định tuyến các gói tới đích của chúng. Ở lớp này, một firewall có thể xác định rằng một gói từ một nguồn đáng tin cậy, nhưng không xác định gói chứa những gì.

Ở lớp transport, firewall biết một ít thông tin về gói và có thể cho phép hoặc từ chối truy cập dựa vào các tiêu chuẩn. Ở lớp ứng dụng, firewall biết nhiều về những gì đang diễn ra và có sự lựa chọn trong việc gán quyền truy cập.

#### 1.1.6 Những nhược điểm và hạn chế của Firewall

Không cái gì là toàn diện cả, tuy tường lửa cung cấp nhiều tính năng hữu ích để bảo vệ người dùng, song nó vẫn tồn tại rất nhiều nhược điểm.

Tường lửa không thể bảo vệ các mối nguy hiểm từ bên trong mạng nội bộ nếu không có sự ngăn chặn từ chính người quản trị mạng đó.

Tường lửa không có đủ thông minh để có thể đọc và hiểu từng loại thông tin và tất nhiên là nó không thể biết được đâu là nội dung tốt và đâu là nội dung xấu. Đơn thuần tường lửa chỉ hỗ trợ chúng ta ngăn chặn sự xâm nhập của những nguồn thông tin không mong muốn nhưng phải xác định rõ các thông số địa chỉ.

Tường lửa không thể ngăn chặn các cuộc tấn công nếu như cuộc tấn công đó không “đi qua” nó. Ví dụ cụ thể đó là nó không thể chống lại một cuộc tấn công từ một đường ngoại tuyến, hoặc là sự dò rỉ thông tin do dữ liệu bị sao chép bất hợp pháp.

Bên cạnh đó, nó cũng không thể chống lại các cuộc tấn công bằng dữ liệu (data-drivent attack). Khi có một số ứng dụng hay phần mềm.. được chuyển qua thư điện tử (ví dụ như Gmail, Yahoo mail…), nó có thể dễ dàng vượt qua và vào trong mạng được bảo vệ.

Cuối cùng, tường lừa không thể làm nhiệm vụ rà quét virus trên các dữ liệu được chuyển qua nó, do tốc độ làm việc, sự xuất hiện liên tục của các virus mới và do có rất nhiều cách để mã hóa dữ liệu để có thể thoát khỏi khả năng kiểm soát của tường lửa

## 1.2 Giới thiệu về iptables.

iptables là một tường lửa ứng dụng lọc gói dữ liệu rất mạnh, miễn phí và có sẵn trên Linux. iptables cho phép người quản trị Linux cấu hình cho phép/chặn luồng dữ liệu đi qua mạng. iptables có thể đọc, thay đổi, chuyển hướng hoặc hủy các gói tin đi tới/đi ra dựa trên các tables, chains và rules. Mỗi một table sẽ có nhiều chain chứa các rule khác nhau quyết định cách thức xử lý gói tin (dựa trên giao thức, địa chỉ nguồn, đích….).

Để đi sâu vào cách thức hoạt động của iptables, ta cần phải hiểu rõ về các khái niệm như table, chain và rule được mô tả bên dưới.

### 1.2.1 Các bảng trong iptables

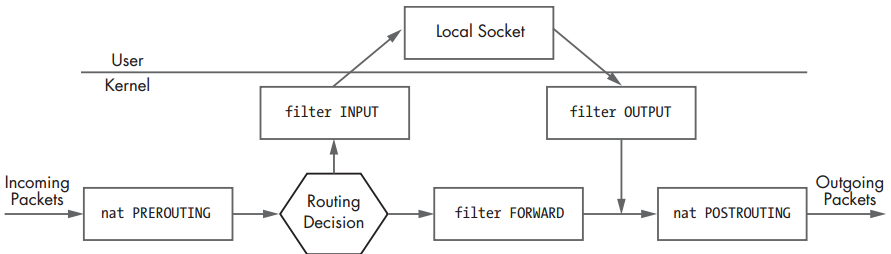
#### a) Filter table

Filter là bảng được dùng nhiều nhất trong iptables. Bảng này dùng để quyết định xem có nên cho một gói tin tiếp tục đi tới đích hoặc chặn gói tin này lại (lọc gói tin). Đây là chức năng chính yếu nhất của iptables, nếu các lệnh không khai báo bảng đích thì mặc định sẽ là bảng Filter.

#### b) NAT table

Bảng NAT được dùng để NAT (Network Address Translation – Phiên dịch địa chỉ mạng), khi các gói tin đi vào bảng này, gói tin sẽ được kiểm tra xem có cần thay đổi và sẽ thay đổi địa chỉ nguồn, đích của gói tin như thế nào.

Bảng này được sử dụng khi có một gói tin từ một connection mới gởi đến hệ thống, các gói tin tiếp theo của connection này sẽ được áp rule và xử lý tương tự như gói tin đầu tiên mà không cần phải đi qua bảng NAT nữa.



Hình 2 Sơ đồ sử lý gói tin trong bảng NAT và Filter

#### c) Mangle Table

Bảng mangle dùng để điều chỉnh một số trường trong IP header như TTL (Time to Live), TOS (Type of Serivce) dùng để quản lý chất lượng dịch vụ (Quality of Serivce)… hoặc dùng để đánh dấu các gói tin để xử lý thêm trong các bảng khác.

#### d) Raw Table

Theo mặc định, iptables sẽ lưu lại trạng thái kết nối của các gói tin, tính năng này cho phép iptables xem các gói tin rời rạc là một kết nối, một session chung để dễ dàng quản lý. Tính năng theo dõi này được sử dụng ngay từ khi gói tin được gởi tới hệ thống trong bảng raw.

Với bảng raw, ta có thể bật/tắt tính năng theo dõi này đối với một số gói tin nhất định, các gói tin được đánh dấu NOTRACK sẽ không được ghi lại trong bảng connection tracking nữa.

#### e) Security Table

Bảng security dùng để đánh dấu policy của SELinux lên các gói tin, các dấu này sẽ ảnh hưởng đến cách thức xử lý của SELinux hoặc của các máy khác trong hệ thống có áp dụng SELinux. Bảng này có thể đánh dấu theo từng gói tin hoặc theo từng kết nối.

### 1.2.2 Các chain trong iptables

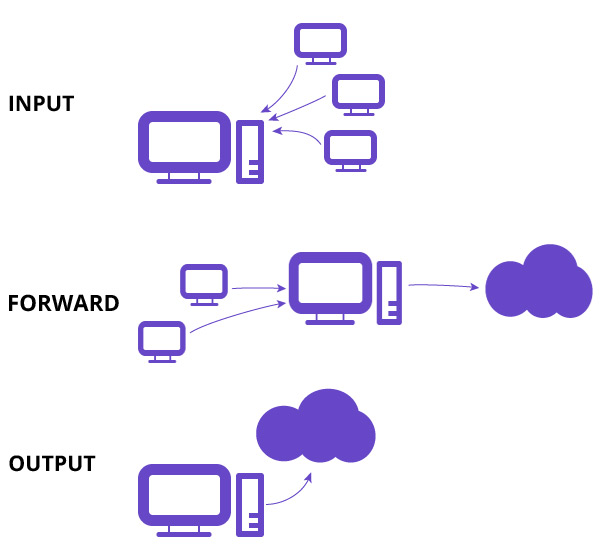
Mỗi một table đều có một số chain của riêng mình, sau đây là bảng cho biết các chain thuộc mỗi table.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Tables/Chain | PREROUTING | INPUT | FORWARD | OUTPUT | POSTROUTING |
| Raw |  |  |  |  |  |
| mangle |  |  |  |  |  |
| nat (DNAT) |  |  |  |  |  |
| filter |  |  |  |  |  |
| sercurity |  |  |  |  |  |
| nat (SNAT) |  |  |  |  |  |

Bảng 1 Các chain trong iptables

Giới thiệu về các chain:

* **INPUT**: Chain này dùng để kiểm soát hành vi của những các kết nối tới máy chủ. Ví dụ một user cần kết nối SSH và máy chủ, iptables sẽ xét xem IP và port của user này có phù hợp với một rule trong chain INPUT hay ko.
* **FORWARD**: Chain này được dùng cho các kết nối chuyển tiếp sang một máy chủ khác (tương tự như router, thông tin gởi tới router sẽ được forward đi nơi khác). Ta chỉ cần định tuyến hoặc NAT một vài kết nối (cần phải forward dữ liệu) thì ta mới cần tới chain này.
* **OUTPUT**: Chain này sẽ xử lý các kết nối đi ra ngoài. Ví dụ như khi ta truy cập google.com, chain này sẽ kiểm tra xem có rules nào liên quan tới http, https và google.com hay không trước khi quyết định cho phép hoặc chặn kết nối.
* **PREROUTING**: Header của gói tin sẽ được chỉnh sửa tại đây trước khi việc routing được diễn ra.
* **POSTROUTING**: Header của gói tin sẽ được chỉnh sửa tại đây sau khi việc routing được diễn ra.



Hình 3 Nhiệm vụ chính của 3 chain cơ bản trong iptables

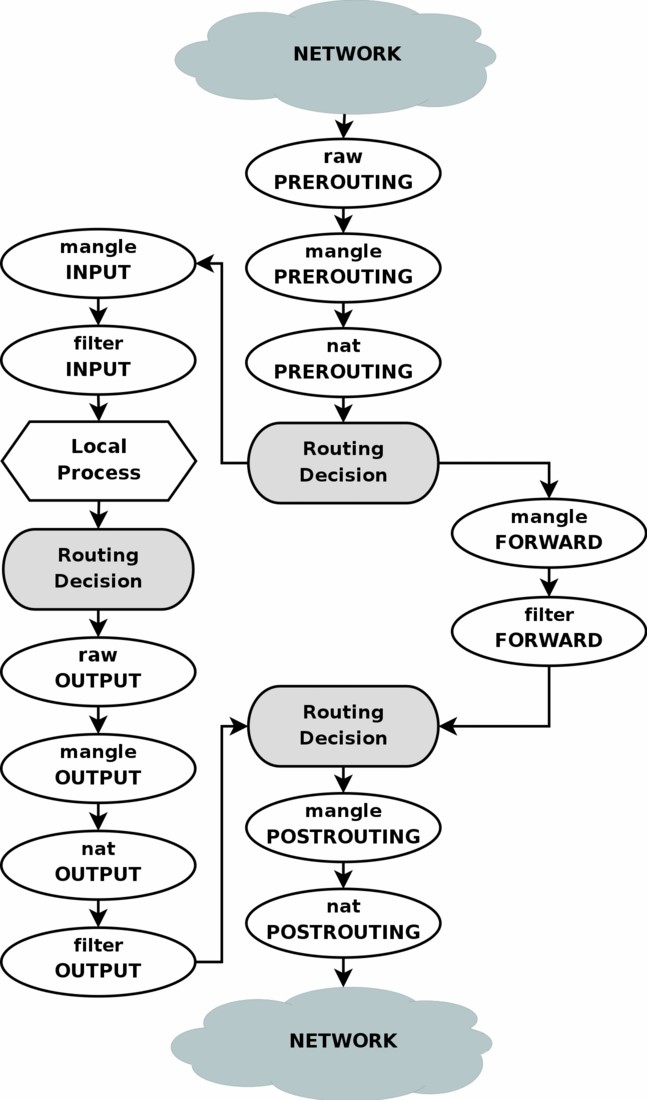
Mặc định thì các chain này sẽ không chứa bất kỳ một rule nào, tuy nhiên mỗi chain đều có một policy mặc định nằm ở cuối chain, policy này có thể là **ACCEPT** hoặc **DROP**, chỉ khi gói tin đã đi qua hết tất cả các rule ở trên thì gói tin mới gặp phải policy này.

Ngoài ra, thứ tự gói tin di chuyển giữa các chain sẽ có hơi khác tùy vào tình huống:

* Gói tin được gởi đến máy chủ:
  + ***PREROUTING → INPUT.***
* Gói tin được forward đến một máy chủ khác:
  + ***PREROUTING → FORWARD → POSTROUTING.***
* Gói tin được máy chủ hiện tại gởi ra ngoài:
  + ***OUTPUT → POSTROUTING.***

Đường đi chính của gói tin trong iptables được bao quát như sau:

* Gói dữ liệu đi vào mạng, được kiểu tra ở ***mangle table* PREROUTING** chain (nếu cần).
* Tiếp theo ***nat table PREROUTING*** sẽ kiểm tra dữ liệu có cần DNAT hay không. DNAT sẽ thay đổi địa chỉ đích của gói dữ liệu.
* Nếu gói dữ liệu đi vào một mạng được bảo vệ, thì nó sẽ được lọc bởi chain **FORWARD** của ***filter table***. Nếu cần thì gói dữ liệu sẽ được SNAT trong **POSTROUTING** chain để thay đổi địa chỉ nguồn trước khi vào mạng khác.
* Nếu gói dữ liệu được định hướng đi vào firewall, nó sẽ được kiểm tra bởi **INPUT** chain của ***mangle table****.*
* Nếu gói dữ liệu qua được các kiểm tra trong chain **INPUT** của ***filter table*** thì sẽ được vào bên trong chương trình server của tường lửa.
* Khi tường lửa cần gửi dữ liệu ra ngoài, gói dữ liệu được dẫn đi và qua kiểm tra bởi chain **OUTPUT** trong ***mangle table*** (nếu cần).
* Tiếp đến, gói dữ liệu sẽ được kiểm tra bởi chain **OUTPUT** trong ***nat table*** để xem có cần phải DNAT hay không. Trong khi đó **OUTPUT** chain của ***filter table*** sẽ kiểm tra xem dữ liệu đó có được phép gửi ra ngoài hay không.
* Cuối cùng, trước khi gói dữ liệu được chuyển đi, SNAT và QoS sẽ được kiểm tra lại trong chain **POSTROUTING** ở ***nat table***.



Hình 4 Đường đi của gói tin trong các chain của iptables

### 1.2.3 Các rules trong chain

Các rule là tập điều kiện và hành động tương ứng để xử lý gói tin (ví dụ ta sẽ tạo một rule chặn giao thức FTP, drop toàn bộ các gói tin FTP được gởi đến máy chủ). Mỗi chain sẽ chứa rất nhiều rule, gói tin được xử lý trong một chain sẽ được so với lần lượt từng rule trong chain này.

Cơ chế kiểm tra gói tin dựa trên rule vô cùng linh hoạt và có thể dễ dàng mở rộng thêm nhờ các extension của IPtables có sẵn trên hệ thống. Rule có thể dựa trên protocol, địa chỉ nguồn/đích, port nguồn/đích, card mạng, header gói tin, trạng thái kết nối… Dựa trên những điều kiện này, ta có thể tạo ra một tập rule phức tạp để kiểm soát luồng dữ liệu ra vào hệ thống.

Mỗi rule sẽ đươc gắn một hành động để xử lý gói tin, hành động có thể là:

* ***ACCEPT***: gói tin sẽ được chuyển tiếp sang bảng kế tiếp.
* ***DROP***: gói tin/kết nối sẽ bị hủy, hệ thống sẽ không thực thi bất kỳ lệnh nào khác.
* ***REJECT***: gói tin sẽ bị hủy, hệ thống sẽ gởi lại 1 gói tin báo lỗi ICMP – Destination port unreachable
* ***LOG***: gói tin khớp với rule sẽ được ghi log lại.
* ***REDIRECT***: chuyển hướng gói tin sang một proxy khác.
* ***MIRROR***: hoán đổi địa chỉ IP nguồn, đích của gói tin trước khi gởi gói tin này đi.
* ***QUEUE***: chuyển gói tin tới chương trình của người dùng qua một module của kernel.

### 1.2.4 Các trạng thái của kết nối

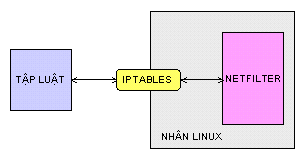
Đây là những trạng thái mà hệ thống connection tracking (module conntrack của iptables) theo dõi trạng thái của các kết nối:

* **NEW**: Khi có một gói tin mới được gởi tới và không nằm trong bất kỳ connection nào hiện có, hệ thống sẽ khởi tạo một kết nối mới và gắn nhãn NEW cho kết nối này. Nhãn này dùng cho cả TCP và UDP.
* **ESTABLISHED**: Kết nối được chuyển từ trạng thái ***NEW*** sang ***ESTABLISHED*** khi máy chủ nhận được phản hồi từ bên kia.
* **RELATED**: Gói tin được gởi tới không thuộc về một kết nối hiện có nhưng có liên quan đến một kết nối đang có trên hệ thống. Đây có thể là một kết nối phụ hỗ trợ cho kết nối chính, ví dụ như giao thức FTP có kết nối chính dùng để chuyển lệnh và kết nối phụ dùng để truyền dữ liệu.
* **INVALID**: Gói tin được đánh dấu***INVALID*** khi gói tin này không có bất cứ quan hệ gì với các kết nối đang có sẵn, không thích hợp để khởi tạo một kết nối mới hoặc đơn giản là không thể xác định được gói tin này, không tìm được kết quả trong bảng định tuyến.
* **UNTRACKED**: Gói tin có thể được gắn hãn ***UNTRACKED*** nếu gói tin này đi qua bảng raw và được xác định là không cần theo dõi gói này trong bảng connection tracking.
* **SNAT**: Trạng thái này được gán cho các gói tin mà địa chỉ nguồn đã bị NAT, được dùng bởi hệ thống connection tracking để biết khi nào cần thay đổi lại địa chỉ cho các gói tin trả về.
* **DNAT**: Trạng thái này được gán cho các gói tin mà địa chỉ đích đã bị NAT, được dùng bởi hệ thống connection tracking để biết khi nào cần thay đổi lại địa chỉ cho các gói tin gởi đi.

Các trạng thái này giúp người quản trị tạo ra những rule cụ thể và an toàn hơn cho hệ thống.

## 1.3 Các kỹ thuật lập lệnh cơ bản trong iptables

### 1.3.1 iptables và Netfilter



Hình 5 Liên kết giữa iptables và Netfilter

iptables nằm ngoài nhân. iptables chịu trách nhiệm giao tiếp giữa người dùng và Netfilter để đẩy các luật của người dùng vào cho Netfiler xử lí. Netfilter tiến hành lọc các gói dữ liệu ở mức IP. Netfilter làm việc trực tiếp trong nhân, nhanh và không làm giảm tốc độ của hệ thống.

### 1.3.2 Cách đổi địa chỉ IP động (dynamic NAT)

NAT động là là phương thức NAT cho phép chuyển đổi toàn bộ địa chỉ IP Private sang một số lượng địa chỉ IP Public để ra bên ngoài Internet. Tùy thuộc vào số lượng IP Public được dùng là bao nhiêu thì tại mỗi phiên làm việc sẽ có bấy nhiêu máy được giao tiếp ra bên ngoài Internet.

Mỗi NAT động được chọn một địa chỉ IP bên ngoài để sử dụng khi kết nối với các mạng bên ngoài, ánh xạ 1: 1 này vẫn tồn tại miễn là kết nối diễn ra. Điều này là phổ biến đối với các mạng DMZ nơi có một lượng địa chỉ IP bên ngoài cần được ánh xạ vào bên trong các địa chỉ IP.

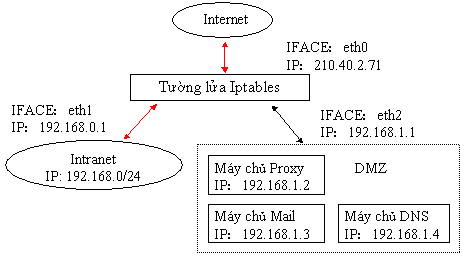
Ví dụ khi sử dụng trên iptables:  
**$iptables** -t nat -A POSTROUTING -o eth0 -s 192.168.0.0/24 -j SNAT --to-source 123.123.123.0/24

Cần chú ý rằng, phạm vi của dãy IP ngoài phải nằm chung một mạng, ta cũng có thể chia các phạm vi ra bằng các sử dụng tham số **-to.**

### 1.3.3 Đổi địa chỉ IP và đóng giả IP (NAT and IP masquerade):

#### a) SNAT

Với static NAT thì sự chuyển đổi packet giữa hai network, giữa nguồn và địa chỉ đến trở nên đơn giản và nhất định, các điều kiện về trạng thái kết nối không cần phải giử lại. Nó chỉ cần nhìn vào mỗi IP packet khi chuyển đổi, các thông tin về mapping đều không cần thiết. Static NAT sử dụng khi số lượng IP trong LAN bằng số lượng NAT-IP.



Hình 6 Mô tả cơ chế làm việc của NPAT

Để tạo kết nối “transparent” cho mạng LAN 192.168.0.1 với Internet, kỹ thuật NPAT sẽ đổi IP nguồn cho các packet ra card mạng eth0 là 210.40.2.71. Khi nhận được packet vào từ Internet, iptables sẽ tự động đổi IP đích 210.40.2.71 thành IP đích tương ướng trong mạng LAN 192.168.0/24

#### b) IP Masquerade

Đây là một kĩ thuật khác trong NAT. Dạng NAT này hay còn được gọi với một cái tên NPAT (Network Port Address Translation), với dạng NAT này tất cả các IP trong mạng LAN được dấu dưới một địa chỉ NAT-IP, các kết nối ra bên ngoài đều được tạo ra giả tạo tại NAT trước khi nó đến được địa chỉ internet.

Trong ví dụ này, tất cả lưu lượng truy cập đến một địa chỉ IP cụ thể nào, không chỉ tới một cổng đặc biệt, được phiên dịch sang một máy chủ trên mạng con được bảo vệ. Vì các tường lửa đều có hơn một địa chỉ IP, ta nên sử dụng Masquerade, nó sẽ buộc giả mạo địa chỉ IP của giao diện chính và không phải là một trong các bí danh địa chỉ IP tường lửa có thể có. Thay vào đó, sử dụng SNAT để chỉ định bí danh địa chỉ IP được sử dụng cho các kết nối được khởi xướng bởi tất cả các máy chủ khác trong mạng lưới bảo vệ.

## 1.4 Một số câu lệnh, tham số cơ bản trong lập lệnh iptables

### 1.4.1 Các tham số switching

|  |  |
| --- | --- |
| Switching | Ý nghĩa |
| -t <table> | Tùy chọn này chỉ định bảng mà packets hoạt động. Nếu bảng không được định nghĩa trước thì dùng bảng **filter** |
| -j <target> | Nhảy đến một chain nào đó khi packet khớp với điều kiện hiện tại |
| -A (append) | Nối thêm rule vào chain |
| -F (Flush) | Xoá hết rule trong bảng đã chọn |
| -p <protocol\_type> | Giao thức của rule hoặc của packet để kiểm tra |
| -s <ip-address> | Địa chỉ IP nguồn |
| -d <ip-address> | Địa chỉ IP đích |
| -i <interface-name> | Tên interface mà thông qua đó packets được nhận |
| -o <interface-name> | Tên của interface mà thông qua đó đưa packets ra |

Bảng 2 Các tham số cơ bản trong iptables

### 1.4.2 Các dịch vụ có sẵn trong Linux

Để iptables hoạt động tốt cần có sự hỗ trợ của các dịch vụ có sẵn trong Linux, dưới đây là một vài thông số thiết đặt được sử dụng:

echo 1 > /proc/sys/net/ipv4/tcp\_syncookies

Bật chức năng chống DDOS SYN qua syncookies của Linux

echo 0 > /proc/sys/net/ipv4/conf/all/accept\_source\_route

Dòng lệnh dùng để vô hiệu hoá định tuyến nguồn.

Định tuyến nguồn là một cơ chế Giao thức Internet cho phép gói IP mang thông tin, danh sách địa chỉ, cho bộ định tuyến biết đường dẫn mà gói phải đi. Ngoài ra còn có một tùy chọn để ghi lại các bước nhảy khi tuyến đường đi qua. Danh sách các bước nhảy được thực hiện, được xem như là "bản ghi tuyến đường" cung cấp đích đến với đường dẫn trở về nguồn. Điều này cho phép nguồn (máy chủ gửi) chỉ định tuyến một cách lỏng lẻo mà bỏ qua các bảng định tuyến của một số hoặc tất cả các bộ định tuyến. Nó có thể cho phép người dùng chuyển hướng lưu lượng mạng cho các mục đích độc hại. Do đó, định tuyến dựa trên nguồn nên bị vô hiệu hoá.

echo 0 > /proc/sys/net/ipv4/conf/all/accept\_redirects

Đối với một máy chủ không định tuyến thì nên tắt service này đi (Theo Linux kernel references), máy chủ ở đây hoặc là một web-server hoặc một PC muốn hệ thống của mình cứng cáp hơn.

echo 1 > /proc/sys/net/ipv4/conf/all/rp\_filter

Resever Path Filtering, là một cơ chế được nhân Linux chấp nhận, cũng như hầu hết các thiết bị mạng ngoài kia để kiểm tra xem địa chỉ nguồn gói nhận có thể định tuyến được hay không, chức năng chính của bộ định tuyến là định tuyến các gói từ nơi này đến nơi khác.

* Nếu nó có thể định tuyến thông qua giao diện mà nó đến, thì máy sẽ chấp nhận gói đó.
* Nếu nó không thể định tuyến thông qua giao diện mà nó đã xuất hiện, thì máy sẽ bỏ gói đó.

Việc bật services này trong bối cảnh lưu lượng truy cập đọc hại và tấn công trên internet ngày càng tăng là cần thiết. Khi được bật, nó sẽ chăm sóc cho việc định cấu hình các tuyến đường trên máy Linux hoặc bộ định tuyến vật lý tốt hơn.

echo 1 > /proc/sys/net/ipv4/icmp\_echo\_ignore\_broadcasts

Messages ICMP echo là các messages được sử dụng bởi công cụ dòng lệnh "ping". Bằng việc kích hoạt dịch vụ này, tường lửa sẽ không hồi đáp đến những ai muốn "ping" đến các địa chỉ broadcast (như là 255.255.255.255 hoặc 192.168.1.255 trong 192.168.1.0/24 subnet) để tìm tất cả các host trong network hoặc subnet cùng một lúc.

Tùy chọn sysctl này không có bất kỳ ảnh hưởng nào đến việc có thể phản hồi các lệnh unicast được gửi trực tiếp đến địa chỉ IP unicast của máy chính. Ngoài ra, tuỳ chọn này chỉ dành riêng cho ICMP **echo** broadcasts, nên sẽ không có tác động nào đến các mục đích sử dụng ICMP khác ngoài **echo**

echo 1 > /proc/sys/net/ipv4/ip\_forward

Một bộ định tuyến có nhiều giao diện mạng. Nếu lưu lượng truy cập đến trên một giao diện khớp với mạng con của giao diện mạng khác, thì bộ định tuyến sẽ chuyển tiếp lưu lượng đó đến giao diện mạng khác.

Kích hoạt ip\_forward bảo hệ thống Linux của bạn thực hiện việc này. Để nó có ý nghĩa, bạn cần hai giao diện mạng (bất kỳ 2 hoặc nhiều thẻ NIC có dây, thẻ Wifi hoặc chipset, liên kết PPP qua modem 56k hoặc nối tiếp, v.v.).

# CHƯƠNG II: PHÂN TÍCH MỘT SỐ PHƯƠNG THỨC TẤN CÔNG

## 2.1 IP spoofing (Giả mạo IP)

### 2.1.1 Lý thuyết của giả mạo IP

Giả mạo địa chỉ IP là một kỹ thuật được tin tặc sử dụng để ngụy trang địa chỉ IP của họ thành một IP khác và có quyền truy cập vào thông tin nhạy cảm hoặc truy cập các dịch vụ riêng tư trên các mạng đã được cấp quyền trước. Trong một số trường hợp đặc biệt, họ có thể tạo các gói IP chứa địa chỉ IP nguồn khác đi và cách này giúp ẩn địa chỉ gốc của họ hoặc mạo danh hệ thống máy tính khác.

Về lý thuyết thì có vẻ giả mạo IP khá khó khăn, nhưng điều này hoàn toàn có thể xảy ra nếu tin tặc có quyền truy cập vào router của mạng cục bộ.

### 2.1.2 Nguyên lý của giả mạo IP

#### a) Nguyên nhân chính

Tất cả các gói tin trong giao thức IP phải có phần ***header***, phần này chứa địa chỉ IP của người gửi và một vài thông tin khác. Bằng một cách nào đó, phần ***header*** này bị thay đổi, tạo ra một gói tin IP giả. Khi sự thay đổi diễn ra, người nhận vẫn một mực tin rằng gói tin này đến từ một nguồn an toàn.

Vậy có phải phần ***header*** là chìa khoá để giả mạo IP thành công?

Bắt nguồn từ các vấn đề thiết kế của bộ giao thức TCP/IP, giao thức IP hoạt động trên tầng mạng (tầng 4) của mô hình OSI. Do tính chất phi kết nối của mô hình OSI, không hề có một thông tin gì về trạng thái của việc giao dịch IP – liệu gói tin có đi đến đích được hay không?

#### b) **Giả mạo IP trên giao thức TCP**

Giao thức TCP có một giải pháp để xác nhận việc phân phối gói bằng cách dùng số thứ tự (sequence number) và số xác nhận (acknowledgment number) để đảm bảo phân phối gói phù hợp. Nhờ đó, trạng thái của giao dịch được xử lý và theo dõi.

TCP sử dụng phương thức cổ điển, bắt tay ba bước, để thiết lập kết nối, Khi **client** muốn kết nối đến **server**, **server** phải mở một cổng được chỉ định trước để thiết lập kết nối. Quá trình mở cổng được gọi với cái tên: *mở bị động (passive open)* bởi vì server sẽ chỉ nghe kết nối trên cổng được mở. Tiếp theo, **client** sẽ bắt đầu *mở chủ động,* và để hoàn tất kết nối, quá trình bắt tay ba bước được diễn ra ở trạng thái này.

Việc mở và gửi gói tin TCP với mọi địa chỉ IP tương đối dễ, thế nhưng, gói tin hồi đáp SYN/ACK sẽ chỉ được định tuyến đến IP nguyên bản (original IP), có nghĩa là hồi đáp không thể nào đến được tin tặc dù họ đã mạo danh địa chỉ. Và cũng vì điều này, IP phản hồi sẽ không gửi hồi đáp ACK để xác nhận gói tin của máy chủ và kết nối sẽ không bao giờ được khởi tạo.

Trên TCP, không thể giả mạo được địa chỉ trong khi quá trình bắt tay diễn ra.

#### c) Giả mạo IP trên giao thức UDP

Việc giả mạo gói tin thông qua giao thức UDP cũng có một vấn đề khác khi không hề có một giao tiếp bắt tay nào. Các hệ thống phát video và nhạc thời gian thực sử dụng UDP, nơi mà việc drop một gói tin được ưu ái hơn việc phải chờ đợi một gói tin nào đó bị chậm trễ. UDP là một giao thức không kết nối dựa trên thông báo, nơi giao tiếp đạt được bằng cách truyền thông tin theo một hướng từ nguồn đến đích mà không cần xác minh sự sẵn sàng hoặc trạng thái của người nhận. **Không có xác nhận, thời gian chờ, kể cả truyền lại gói tin.**

### 2.1.3 Sử dụng giả mạo IP để làm gì

Giả mạo IP thường được dùng cho việc tấn công từ chối dịch vụ, kể cả tấn công từ chối phân tán, nhờ vào việc các địa chỉ bị thay đổi khó lọc hơn vì mỗi gói tin giả mạo dường như được đưa đến từ các địa chỉ khác nhau. Trong trường hợp này, đa số tin tặc không nhắm đến việc có nhận được phản hồi hay không. Máy chủ sẽ bị áp đảo tài nguyên vì số lượng yêu cầu quá lớn, đến mức việc tấn công được máy chủ xem như là các truy cập hợp pháp và cố gắng dành tài nguyên cho nó.

Kẻ tấn công có thể chọn ngẫu nhiên các địa chỉ IP từ không gian địa chỉ IP chung hoặc chúng có thể sử dụng các công cụ tạo IP tinh vi để lọc các địa chỉ không thể thực hiện được hoặc không được sử dụng bởi bất kỳ ai.

Nhưng không chỉ địa chỉ IP, các trang web cũng có thể bị giả mạo. Tin tặc có thể sử dụng logo, hình ảnh, màu sắc và phông chữ hợp pháp để trông giống như một dịch vụ khác và thay đổi một trang web để có vẻ thực tế và đáng tin cậy (kỹ thuật fishing và pharming).

### 2.1.4 Nhận biết một gói tin đã được giả mạo

Các ISP (Nhà cung cấp dịch vụ Internet) và nhà cung cấp VPN có thể thực hiện một số bước nhất định chống giả mạo IP vì việc phát hiện gói giả mạo thường chỉ có thể được thực hiện gần với nguồn của lưu lượng mạng. Ví dụ, bộ định tuyến đầu tiên trên đường dẫn có khả năng phát hiện ra rằng địa chỉ nguồn trong gói không có thuộc về bất kỳ mạng nội bộ nào mà nó biết và do đó có thể loại bỏ nó. Điều này được gọi là lọc hướng ra (**egress filtering)**. Tuy nhiên, việc kiểm tra này chỉ có thể thực hiện được bên trong hoặc phần rìa của mạng. Nó thường chỉ được thực hiện bởi tường lửa và cần được cấu hình cẩn thận để tránh tác dụng phụ.

Ngoài ra, còn có một kỹ thuật tương tự như lọc hướng ra, là lọc xâm nhập (**ingress filtering).** Nó được sử dụng để đảm bảo rằng các gói đến thực sự là từ các mạng mà chúng đã kết nối từ trước. Kỹ thuật này chỉ hoạt động nếu thiết bị lọc được cấu hình nhận biết mạng được kết nối.

### 2.1.5 Kết luận

Với việc phân tích về cách thức giả mạo IP, ta có thể trách nhiệm lớn nhất trong việc ngăn chặn nó thuộc về các nhà cung cấp dịch vụ và VPN. Nhưng dưới phương diện một người dùng cuối, việc tự tìm hiểu và viết ra một vài quy tắc nhỏ để đảm bảo cho hệ thống vấn là điều cần thiết.

## 2.2 Stealth scan TCP/SYN (Tấn công tràn SYN)

### 2.2.1 Tấn công từ chối dịch vụ (DoS) là gì?

Cuộc tấn công từ chối dịch vụ (DoS) là một cuộc tấn công nhằm tắt máy hoặc mạng, khiến nó không thể truy cập được đối với tập người dùng của nó. Các cuộc tấn công DoS thực hiện điều này bằng cách làm ngập mục tiêu với lưu lượng truy cập hoặc gửi thông tin gây ra sự cố.

Trong cả hai trường hợp, cuộc tấn công DoS đều tước quyền sử dụng hợp pháp của người dùng (tức là nhân viên, thành viên hoặc chủ tài khoản) của dịch vụ hoặc tài nguyên mà họ mong đợi.

Nạn nhân của các cuộc tấn công DoS thường nhắm là các máy chủ web của các tổ chức cao cấp như ngân hàng, thương mại và các công ty truyền thông, hoặc các tổ chức chính phủ và thương mại. Mặc dù các cuộc tấn công DoS thường không dẫn đến việc đánh cắp hoặc mất thông tin quan trọng hoặc các tài sản khác, chúng có thể khiến nạn nhân phải trả rất nhiều thời gian và tiền bạc để xử lý.

Có 2 phương pháp chính trong tấn công DoS:

* Tấn công gây ngập lụt (flood attack).
* Tấn công phá vỡ (crash attack).

Tấn công tràn xảy ra lúc mà hệ thống nhận quá nhiều lưu lượng cho máy chủ xử lý (kể cả sử dụng bộ đệm), làm cho máy chủ ì ạch và hậu quả là có thể dừng lại. Tấn công tràn có các dạng chính như sau:

* **Buffer overflow attack**: Tấn công làm tràn bộ đệm, một kiểu tấn công thường thấy nhất. Kẻ tấn công sẽ gửi lưu lượng truy cập lớn hơn lượng tối đa mà hệ thống có thể xử lý. Nó bao gồm các cuộc tấn công được liệt kê bên dưới, ngoài các cuộc tấn công khác được thiết kế để khai thác các lỗi cụ thể đối với các ứng dụng hoặc mạng nhất định.
* **ICMP Flood**: Tấn công tràn ICMP, tận dụng các thiết bị mạng được định cấu hình sai bằng cách gửi các gói **ping** giả mạo đến mọi máy tính trên mạng mục tiêu, thay vì chỉ một máy cụ thể.
* **SYN Flood:** gửi yêu cầu kết nối với máy chủ, nhưng không bao giờ hoàn thành việc bắt tay. Quá trình tấn công diễn ra cho đến khi không còn một cổng nào khả dụng cho người dùng trên hệ thống.

Các cuộc tấn công DoS khác chỉ đơn giản là khai thác các lỗ hổng khiến hệ thống hoặc dịch vụ đích bị sập. Một dạng khác của DoS là DDoS, tấn công từ chối dịch vụ phân tán. Một cuộc tấn công DDoS xảy ra khi nhiều hệ thống phối hợp một cuộc tấn công DoS được đồng bộ hóa vào một mục tiêu. Sự khác biệt cơ bản là thay vì bị tấn công từ một địa điểm, mục tiêu bị tấn công từ nhiều địa điểm cùng một lúc.

### 2.2.2 Lý thuyết về quét tràn SYN

Quét SYN là tùy chọn quét mặc định và phổ biến nhất vì nhiều lý do (hầu hết là chính đáng). Nó có thể được thực hiện nhanh chóng, quét hàng nghìn cổng mỗi giây trên một mạng mà không bị cản trở bởi các tường lửa mặc định.

Một ví dụ lệnh để xem cách thức quét SYN, công cụ sử dụng là nmap

krad# **nmap -p22,113,139 scanme.nmap.org**

Starting Nmap ( http://nmap.org )

Nmap scan report for scanme.nmap.org (64.13.134.52)

PORT STATE SERVICE

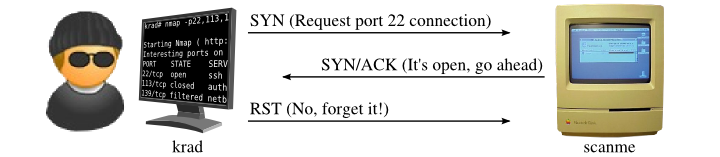
22/tcp open ssh

113/tcp closed auth

139/tcp filtered netbios-ssn

Nmap done: 1 IP address (1 host up) scanned in 1.35 seconds

Quét SYN tương đối kín đáo và lén lút vì nó không bao giờ hoàn thành kết nối TCP (theo quy tắc bắt tay 3 bước). SYN scan quét ở cấp độ packet, cùng xem ví dụ tấn công vào port 22 như bên dưới:



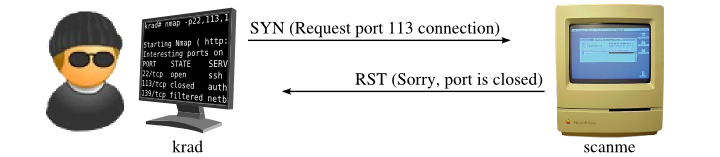
Hình 7 Tấn công SYN scan ở cổng 22

***nmap*** bắt đầu gửi một packet TCP với cờ SYN đã được set (tham chiếu đến ip header) và gửi đến port 22. Đây là bước đầu tiên trong quá trình bắt tay 3 bước mà mọi kết nối hợp pháp đều phải trải qua. Khi cổng mục tiêu (22) mở, **scanme** thực hiện bước thứ 2 bằng cách gửi phản hồi cờ SYN và ACK. Trong một kết nối bình thường, **krad** phải hoàn thành bước thứ 3 bằng việc gửi một packet ACK chấp nhận SYN/ACK.

***nmap*** thì không làm vậy, vì phản hồi SYN/ACK đã phản hồi rằng port đã mở. Nếu ***nmap*** hoàn thành được kết nối, sau đó nó phải chịu trách nhiệm đóng lại. Việc này yêu cầu một quy trình bắt tay mới, sử dụng packet FIN thay vì SYN. Vậy ACK không phải là một ý tưởng tốt, nhưng vẫn cần phải sử dụng nó trong một số tình huống. Nếu như SYN/ACK hoàn toàn bị bỏ qua, **scanme** sẽ giả sử rằng nó đã bị drop và tiếp tục gửi lại.

Câu trả lời thích hợp ở tình huống này, vì chúng ta không muốn tạo một kết nối đầy đủ, là một gói RST như được hiển thị trong biểu đồ. Việc này giống như thông báo cho **scanme** quên kết nối đã thử. ***nmap*** có thể gửi packet RST này một cách dễ dàng, nhưng nó thực sự không cần phải làm hoặc không muốn làm như vậy. Hệ điều hành chạy trên **krad** cũng nhận SYN/ACK, nhưng điều này nằm ngoài mong đợi vì ***nmap*** đã tạo ra đầu dò SYN cho chính nó. Vì vậy, hệ điều hành đáp ứng với các SYN / ACK không mong đợi với một gói RST. Mọi gói RST được mô tả trong phần SYN scan này đều có một ACK bit set vì chúng luôn được gửi trong gói packet phản hồi. Điều này dẫn đến bit đó không được hiển thị rõ ràng cho RST packets bởi vì quá trình bắt tay 3 bước sẽ không bao giờ được hoàn tất. SYN scan đôi lúc còn được biết đến với tên gọi half-open scanning, nghĩa là làm cho cổng mục tiêu mở ra và không bao giờ đóng.

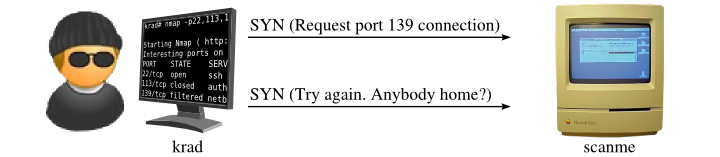
Hình dưới mô tả quá trình ***nmap*** xác nhận cổng 113 đã đóng. Việc này thậm chí còn dễ dàng hơn trường hợp mở kết nối. Bước đầu tiên khá giống nhau, ***nmap*** gửi một xác thực SYN đến **scanme**. Nhưng thay vì nhận SYN/ACK trở lui, nó trả về một RST. Việc gửi phản hồi RST xem như cổng đã đóng, không cần phải lưu ý đến giao tiếp của cổng này nữa.



Hình 8 **nmap** đóng cổng 113 bằng RTS packet

Cuối cùng, **krad** cho chúng thấy làm thế nào mà một port được lọc xuất hiện trong ***nmap*** ở hình 8, SYN khơi mào được gửi đầu tiên như thường lệ, nhưng ***nmap*** không thấy reply nào. Phản hồi có thể được gửi chậm. Từ phản hồi trước (hoặc theo mức timming quy định sẵn), ***nmap*** biết được cần phải đợi bao lâu là tuần tự từ bỏ những packet nhận được. Một cổng không phản hồi thường được lọc qua (thường là chặn bởi firewall của thiết bị, hoặc có thể host đã chết).

***nmap*** cũng có thể quét cổng được lọc, ví dụ là cổng 139 ở trên



Hình 9 **nmap** quét cổng đã được lọc (filtered)

Với cổng đã được lọc, ***nmap*** sẽ gán trạng thái của cổng nếu nhận được một số lỗi ICMP nhất định. Bảng dưới cho thấy trạng thái của cổng mà ***nmap*** sẽ gán.

|  |  |
| --- | --- |
| Thăm dò trả về | Trạng thái được gán |
| Hồi đáp TCP SYN/ACK | Open |
| Hồi đáp TCP RST | Closed |
| Không hồi đáp, kể cả đã truyền lại | Filtered |
| Không bắt được ICMP (mã lỗi 1, 2, 3, 9, 10, 13) | Filtered |

Bảng 3 Trạng thái **nmap** trả về cho cổng

### 2.2.2 Kết luận về quét SYN

Để hiểu hơn về quét SYN, trong lúc quét ta gán thêm tham số switching –packet-trace

krad# **nmap -d --packet-trace -p22,113,139 scanme.nmap.org**

Starting Nmap ( http://nmap.org )

SENT (0.0130s) ICMP krad > scanme echo request (type=8/code=0) ttl=52 id=1829

SENT (0.0160s) TCP krad:63541 > scanme:80 A iplen=40 seq=91911070 ack=99850910

RCVD (0.0280s) ICMP scanme > krad echo reply (type=0/code=0) iplen=28

We got a ping packet back from scanme: id = 48821 seq = 714 checksum = 16000

massping done: num\_hosts: 1 num\_responses: 1

Initiating SYN Stealth Scan against scanme.nmap.org (scanme) [3 ports] at 00:53

SENT (0.1340s) TCP krad:63517 > scanme:113 S iplen=40 seq=10438635

SENT (0.1370s) TCP krad:63517 > scanme:22 S iplen=40 seq=10438635

SENT (0.1400s) TCP krad:63517 > scanme:139 S iplen=40 seq=10438635

RCVD (0.1460s) TCP scanme:113 > krad:63517 RA iplen=40 seq=0 ack=10438636

RCVD (0.1510s) TCP scanme:22 > krad:63517 SA iplen=44 seq=75897108 ack=10438636

SENT (1.2550s) TCP krad:63518 > scanme:139 S iplen=40 seq=10373098 win=3072

The SYN Stealth Scan took 1.25s to scan 3 total ports.

Nmap scan report for scanme.nmap.org (64.13.134.52)

PORT STATE SERVICE

22/tcp open ssh

113/tcp closed auth

139/tcp filtered netbios-ssn

Nmap done: 1 IP address (1 host up) scanned in 1.40 seconds

Quét SYN từ lâu đã được gọi là quét lén vì nó tinh vi hơn quét kết nối TCP (TCP connect scan), đây là loại quét phổ biến nhất trước khi ***nmap*** được phát hành. Mặc dù có biệt danh đó, nhưng đừng chủ quan vào việc quét SYN mặc định mà không bị phát hiện thông qua các mạng được quản lý. Các hệ thống phát hiện xâm nhập được triển khai rộng rãi và thậm chí tường lửa cá nhân hoàn toàn có khả năng phát hiện các lần quét SYN mặc định, và người dùng với lượng kiến thức đủ tốt có thể tự cấu hình một tường lửa đơn giản nhưng vô cùng mạnh mẽ.

## 2.2 Ping flood (Tràn ping)

### 2.2.1 Nguyên lý của tràn ping

Tràn ping, còn được gọi là ngập ICMP, là một cuộc tấn công từ chối dịch vụ (DoS) phổ biến trong đó kẻ tấn công đánh sập máy tính nạn nhân bằng cách áp đảo nó bằng các yêu cầu tiếng ICMP echo, còn được gọi là ping.

Cuộc tấn công với mục đích làm ngập mạng mạng nạn nhân bằng việc gửi các gói tin yêu cầu đến, với mong muốn rằng mạng sẽ phản hồi với số lượng gói trả lời bằng nhau, từng gói một. Một quá trình diễn ra lâu dài như vậy, tất cả tài nguyên trên máy nạn nhân sẽ bị chiếm, kể cả kênh INPUT và OUTPUT của mạng, dẫn đến việc từ chối dịch vụ.

### 2.2.3 Một cuộc tấn công điển hình

Hiện nay, công cụ phổ biến nhất cho quá trình tấn công tràn ping là ***hping3***, ***hping3*** được cài sẵn trong môi trường Kali Linux và có thể cài thêm vào các hệ điều hành Unix khác.

Thông thường, các yêu cầu ping được sử dụng để kiểm tra khả năng kết nối của hai máy tính bằng cách đo thời gian khứ hồi từ khi yêu cầu ICMP được gửi đến khi nhận được phản hồi ICMP. Tuy nhiên, trong một cuộc tấn công, chúng được sử dụng để làm quá tải mạng đích với các gói dữ liệu này.

Việc thực hiện một trận lũ ping phụ thuộc vào việc những kẻ tấn công có biết địa chỉ IP của mục tiêu của chúng hay không. Do đó, các cuộc tấn công có thể được chia thành ba loại, dựa trên mục tiêu và cách giải quyết địa chỉ IP đích của nó.

* **A targeted local disclosed:** Một địa chỉ cục bộ được tiết lộ, kẻ tấn công cần có quyền truy cập vật lý (biết mật khẩu, cắm dây LAN) vào máy tính để khám phá địa chỉ IP của nó.
* **A router disclosed:** Một bộ định tuyến bị tiết lộ, cách thức này nhắm vào các bộ định tuyến để phá vỡ kết nối của các máy trên mạng. Nhưng nó cũng phụ thuộc vào việc kẻ tấn công có biết địa chỉ cục bộ của router hay không.
* **A blind ping flood:** Sử dụng những công cụ ngoài để mở đóng (unconver) địa chỉ IP của máy đích để tấn công.

Nếu một lúc nào đó ta bắt buộc phải sử dụng ***hping3***, thì dưới đây là những lệnh cơ bản để khơi mào một cuộc tấn công:

* ***-n***, là số lần mà request sẽ gửi đến máy đích.
* ***-l***, là tổng khối lượng dữ liệu gửi đến máy đích.
* ***-t***, sử dụng như một vòng lặp cho đến khi nào máy đích bị ***timeout.***

Lưu ý rằng để lũ ping được duy trì lâu dài, máy tính tấn công phải có quyền truy cập vào băng thông nhiều hơn nạn nhân. Điều này giới hạn khả năng thực hiện một cuộc tấn công DoS, đặc biệt là đối với một mạng lớn, nơi mà băng thông của máy đích và máy tấn công gần như là ngang nhau.

Ngoài ra, một cuộc tấn công từ chối dịch vụ phân tán (DDoS) được thực hiện với việc sử dụng botnet đang thịnh hành hơn tại thời điểm hiện tại, với khả năng áp đảo tài nguyên của máy đích tốt hơn.

### 2.2.4 Các vấn đề thuộc giao thức ICMP

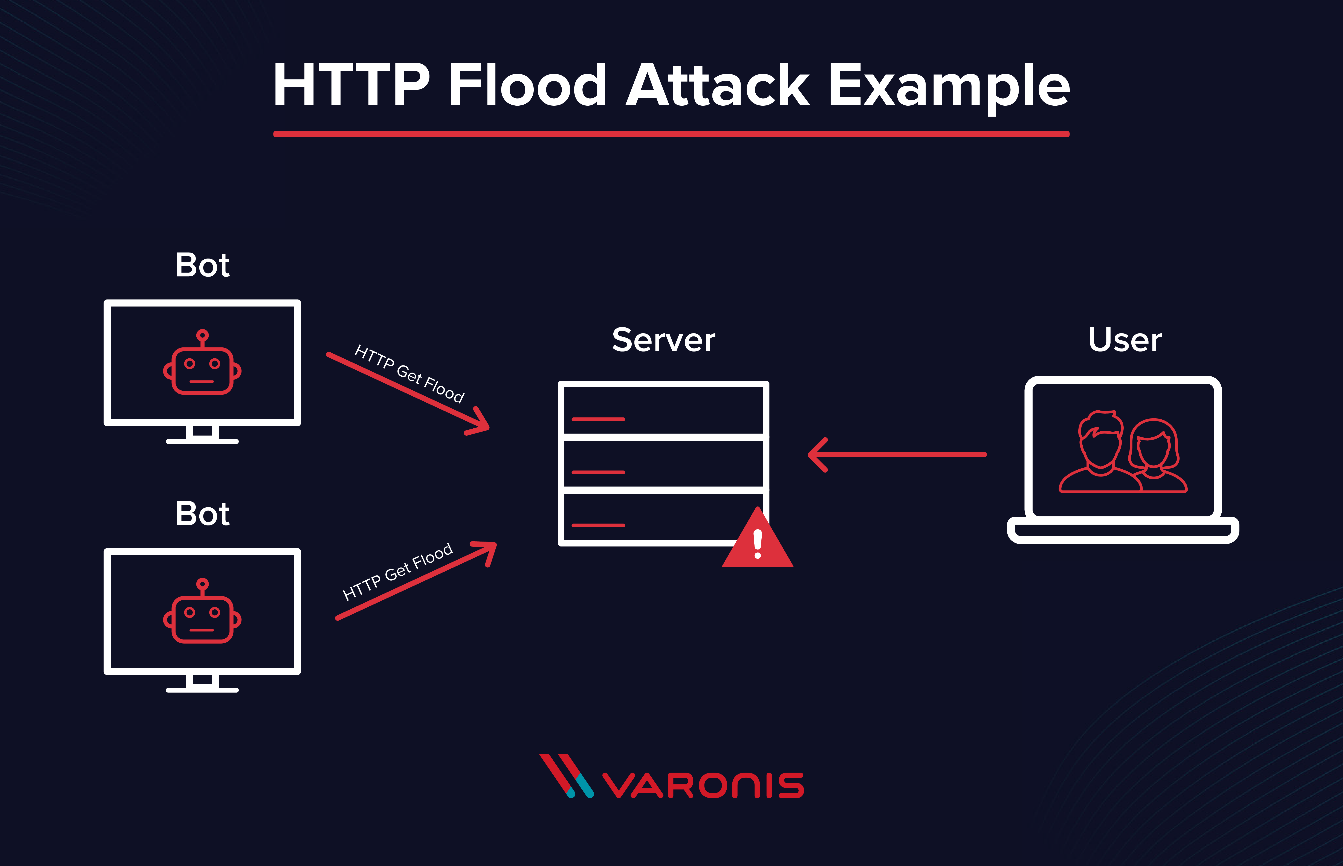
Giao thức ICMP là một giao thức rất tiện dụng trong các giềng mối hoạt động của mạng. Tuy nhiên, nó cũng là phương tiện căn bản dùng trong các quy trình tìm hiểu và tấn công. Có những loại ICMP không nên dùng trên mạng công cộng nếu bảo mật là vấn đề hàng đầu vì những loại ICMP này tiết lộ quá nhiều thông tin quan trọng của hệ thống chúng ta muốn bảo vệ. Hơn nữa, có những loại ICMP còn là phương tiện để đưa hệ thống chúng ta muốn bảo vệ vào tình trạng nguy hiểm. Các vấn đề chính về việc xử lý ICMP cho máy chủ sẽ được đề cập chi tiết ở chương III.

## 2.3 Tấn công từ chối dịch vụ qua HTTP (HTTP DoS)

### 2.3.1 Nguyên lý tấn công từ chối dịch vụ qua HTTP (HTTP Flood)

Còn được gọi là **tấn công phân tán từ chối dịch vụ qua lớp dịch vụ (Application layer Distributed Denial of Service).** Các cuộc tấn công DDoS của lớp ứng dụng nhằm làm cạn kiệt tài nguyên của mục tiêu và phá vỡ quyền truy cập vào trang web hoặc dịch vụ của mục tiêu. Kẻ tấn công sử dụng một con bot với những yêu cầu phức tạp gửi lên server làm server cố gắng dùng nhiều tài nguyên để xử lý. Thông thường, nó có thể yêu cầu truy cập cơ sở dữ liệu hoặc một băng thông tải xuống lớn. Nếu mục tiêu nhận được vài triệu yêu cầu trong một thời gian ngắn, nó có thể nhanh chóng bị quá tải và bị chậm lại khi cố gắng xử lý, hoặc nặng hơn, bị khóa dịch vụ hoàn toàn.

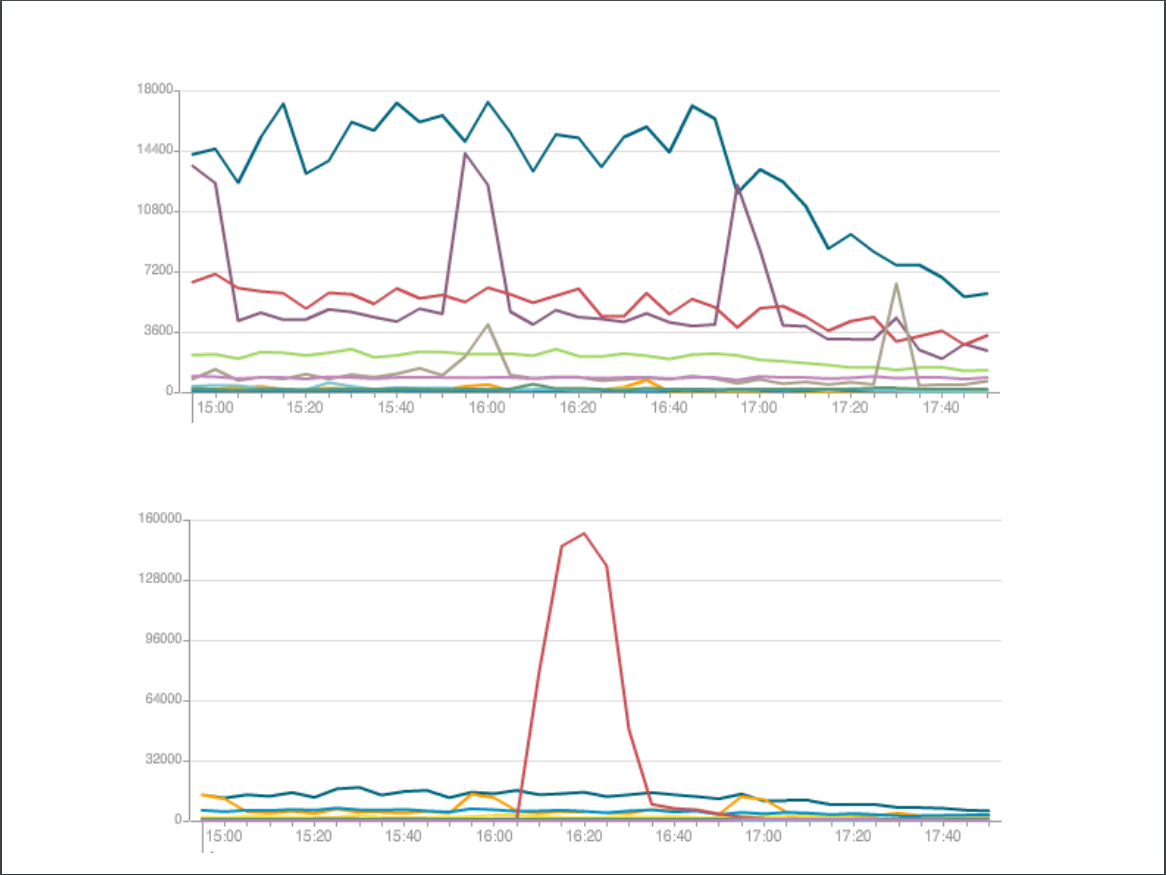
Tấn công tràn HTTP sử dụng các yêu cầu HTTP với mục đích làm vỡ máy chủ, các yêu cầu có thể là ***GET,*** lấy dữ liệu hoặc ***POST,*** chỉnh sử dữ liệu. Điển hình nhất là việc cố gắng gửi yêu cầu *refresh* trang, với lượng truy cập vài nghìn đến vài triệu máy, yêu cầu sẽ tăng lên theo cấp số nhân và việc server bị vỡ là điều không thể tránh khỏi.



Hình 10 Ví dụ về một cuộc tấn công tràn HTTP

### 2.3.2 Vấn đề toàn cầu với HTTP DDos

Hãy cùng xem biểu đồ lưu lượng cho một web-server khi bị tấn công DDoS



Hình Biểu đồ lưu lượng trên Server khi bị tấn công

Biểu đồ ở trên là lưu lượng bình thường của server, biểu đồ dưới là lưu lượng của server khi bị tấn công dữ dội.

Chỉ trong vòng 10 phút, từ lúc 16h10 đến 16h20, lượng **dữ liệu ra vào** đã tăng đột biến lên gấp gần 10 lần lúc bình thường. Nhưng đồng thời, chỉ trong vòng chưa tới 20 phút, quản trị viên đã kiểm soát được vụ tấn công này, và đưa toàn bộ hệ thống trở lại tình trạng bình thường.

Vậy tại sao quản trị viên lại có thể kịp thời nhận ra điều đó, và vấn đề là có phải mỗi quản trị viên có đủ nhận thức cho một tấn công hay không?

#### a) Nguyên tắc giám sát an ninh mạng

Chắc rằng bất kì một quản trị viên nào cũng tự mình đặt ra câu hỏi: ***Chuyện gì đang diễn ra vậy? Tại sao hệ thống đang bình thường lại cứng đơ, và khách hàng thì liên tục phàn nàn?***

Đây là một câu hỏi đặt ra vấn đề lớn nhất của an ninh mạng nói chung và phòng thủ chống HTTP DDoS nói riêng. Đơn giản vì chúng ta không thể bảo vệ được một hệ thống nếu chúng ta không biết được trạng thái của hệ thống đó. Và chúng ta chỉ có thể biết được trạng thái của hệ thống đó nếu chúng ta theo dõi nó thường xuyên, nói cách khác, ta phải biết được tất cả các hoạt động diễn ra trên hệ thống.

Thử nhìn vào hoạt động của một khách sạn. Để đảm bảo an ninh, người ta phải đặt camera theo dõi ở khắp nơi. Các camera này chắc hẳn sẽ đưa hình ảnh về một địa điểm tập trung, nơi có các chuyên viên theo dõi 24/7 để kịp thời phát hiện và đối phó với các sự cố an ninh.

Tương tự như thế, muốn đảm bảo an ninh thông tin chúng ta cũng phải tiến hành theo dõi 24/7. Nhưng trong thực tế, rất ít tổ chức có một **hệ thống giám sát an ninh** như thế. Để bảo vệ **hệ thống mạng** của mình, các doanh nghiệp và các tổ chức công thường triển khai các thiết bị như **tường lửa**, phần mềm **chống và diệt virus**, thiết bị **phát hiện xâm nhập**, thiết bị **ngăn chặn xâm nhập**. Rõ ràng họ nghĩ rằng, các thiết bị này đảm bảo an ninh mạng cho họ nên họ mới đầu tư nhiều tiền của để triển khai chúng.

#### b) Giám sát an ninh mạng trong thực tế

Thực tế hầu hết những người giữ quyền quyết định đầu tư cho **an toàn thông tin** thường hay hành động theo thị trường. Ví dụ như cách đây vài năm, tường lửa là tối ưu nhất. Hầu hết ai cũng đầu tư làm hệ thống tường lửa nên chúng ta cũng phải làm tường lửa. Sau đó, các giải pháp phát hiện xâm nhập lên ngôi. Điển hình là chuẩn ISO 27001.

Thực tế, ta không thể khẳng định rằng tường lửa, thiết bị phát hiện xâm nhập, chuẩn ISO không có tác dụng. Nhưng câu hỏi là: *tại sao quá nhiều lớp bảo vệ như vậy nhưng vẫn bị xâm nhập và tấn công?**Liệu ISO 27001 hay tường lửa có giúp bạn khắc phục được một vụ tấn công từ chối dịch vụ trong vòng 20′? Rồi khi đã bị xâm nhập, có thiết bị đắt tiền hay tiêu chuẩn nào giúp quý vị biết được hệ thống của quý vị bị xâm nhập khi nào, tại sao và như thế nào hay không?*

**Chỉ có con người mới có khả năng làm việc đó.** Nghĩa là, chúng ta cần các chuyên gia giám sát hệ thống có chuyên môn cao.

Tại sao chúng ta cần phải có chuyên gia, tại sao tự bản thân các thiết bị hay các tiêu chuẩn không thể bảo vệ hệ thống mạng? Bởi vì những kẻ tấn công rất thông minh, không thể dự đoán và rất có thể có động lực cao nhất là khi thương mại điện tử phát triển như bây giờ. Máy móc và quy trình không thể ngăn chặn được họ, chắc chắn là như thế. Máy móc chắc chắn sẽ thua khi chiến đấu với não người. Đó là lý do chúng ta cần con người, cần những chuyên gia, để biến **an ninh mạng** thành một cuộc chiến cân sức hơn giữa người và người, thay vì giữa máy và người.

#### c) Vậy thực sự giám sát an ninh mạng là gì?

Câu hỏi đặt ra là các chuyên gia an ninh mạng cần gì để có thể phát hiện và xử lý các sự cố an ninh mạng cũng như xây dựng các kế hoạch phòng thủ? Câu trả lời chỉ có một: **tất cả dữ liệu mà chúng ta có thể thu thập được trên hệ thống mạng trong khi sự cố xảy ra!**

Thuật ngữ**giám sát an ninh mạng** được chính thức định nghĩa vào năm 2002 và về cơ bản nó **gồm 3 bước: thu thập dữ liệu, phân tích dữ liệu**và**leo thang thông tin.**

Để **thu thập dữ liệu**, chúng ta sẽ sử dụng các phần mềm hay giải pháp có sẵn trên thị trường để thu thập dữ liệu ghi dấu hoạt động của các máy chủ, thiết bị mạng, phần mềm ứng dụng, cơ sở dữ liệu…Nguyên tắc của thu thập dữ liệu là thu thập càng nhiều càng tốt, với mục tiêu là chúng ta phải có đầy đủ thông tin về trạng thái, log file của tất cả các thành phần trong hệ thống cần phải bảo vệ. Nguyên tắc “thà giết lầm còn hơn bỏ sót” có thể áp dụng ở đây.

Nếu phần mềm có thể giúp chúng ta làm công việc thu thập dữ liệu, thì để phân tích dữ liệu và ra quyết định, như đã nói ở trên, chúng ta cần có chuyên gia, bởi chỉ có chuyên gia mới có thể hiểu rõ ngữ cảnh của dữ liệu mà phần mềm đã thu thập được. Điều tối quan trọng trong phân tích dữ liệu là ngữ cảnh.

Quy trình giúp cho chúng ta **leo thang thông tin**. **Leo thang thông tin** là việc các chuyên gia báo cáo lên trên cho những người có quyền quyết định những vấn đề mà họ cho là quan trọng, cần phải điều tra thêm. Những người có quyền quyết định là những người có đủ thẩm quyền, trách nhiệm và năng lực để quyết định cách đối phó với các sự cố ANTT tiềm tàng. Không có **leo thang thông tin**, công việc của các chuyên gia sẽ trở thành vô ích.

# CHƯƠNG III: THỬ NGHIỆM CÁC Ý TƯỞNG PHÒNG THỦ

## 3.1 Phòng thủ tấn công giả mạo IP

### 3.1.1 Ý tưởng

Thông thường ta sẽ không muốn nhận IP đến từ các nguồn riêng, vậy ta có thể thiết đặt luật iptables để chặn các gói tin giả có nguồn từ các mạng con riêng tư (cục bộ).

Nhóm private IP cho nội mạng, đôi khi còn gọi là “non-routable IP”, ám chỉ các IP này không thể định tuyến ra ngoài mạng công cộng (thực tế chúng vẫn có thể định tuyến được, nhưng chỉ trong nội mạng). Các nhóm private IP này gồm có:

* Lớp A:
  + 0.0.0.0/8
  + 127.0.0.0/8
  + 10.0.0.0/8
* Lớp B: 172.16.0.0/12
* Lớp C: 192.168.0.0/16
* Lớp D: 224.0.0.0/4
* Lớp E: 240.0.0.0/5
* Link Local: 169.254.0.0/16
* Test Net: 192.0.2.0/24

Ta sẽ cần chỉnh sửa lại thông tin của nguồn IP vào, nên việc sử dụng bảng **mangle** sẽ là thích hợp nhất.

### 3.1.2 Cài đặt

Đa phần giao diện địa chỉ loopback hiện nay được sử dụng là không gian IP 127.0.0.0/8. Ta có được tập quy tắc như sau

**$iptables** -t mangle -A PREROUTING -s 224.0.0.0/3 -j DROP

**$iptables** -t mangle -A PREROUTING -s 169.254.0.0/16 -j DROP

**$iptables** -t mangle -A PREROUTING -s 172.16.0.0/12 -j DROP

**$iptables** -t mangle -A PREROUTING -s 192.0.2.0/24 -j DROP

**$iptables** -t mangle -A PREROUTING -s 192.168.0.0/16 -j DROP

**$iptables** -t mangle -A PREROUTING -s 10.0.0.0/8 -j DROP

**$iptables** -t mangle -A PREROUTING -s 0.0.0.0/8 -j DROP

**$iptables** -t mangle -A PREROUTING -s 240.0.0.0/5 -j DROP

**$iptables** -t mangle -A PREROUTING -s 127.0.0.0/8 ! -i lo -j DROP

Với quy tắc lọc như trên, người dùng có thể từ chối các địa chỉ đến trên giao diện sai tại hoặc gần nguồn lưu lượng, nhưng trên máy chủ lưu trữ cuối cùng, nó rất khó phát hiện giả mạo vì mỗi gói đến sẽ được kiểm tra.

## 3.2 Phòng thủ tấn công tràn SYN

### 3.2.1 Ý tưởng

Đầu tiên, phải tạo một chain để chống lại stealth scan, ta đặt tên cho chain đó là STEALTH\_SCAN.

Tiếp theo, nhắc iptables ghi log để dễ quản lý.

Rule mặc định cho chain phải là **DROP.**

Tiến hành thêm các rule vào cho chain INPUT để kiểm soát packets vào:

**Bước 1**:

* Vì stealth scan quét dựa trên giao thức TCP nên rule của chain sẽ là TCP.
* Set flags nào cần được xem xét, ở đây ta cần xem cờ SYN với tham số cờ ACK phải được set trước đó. Vậy ta sẽ dùng tham số switching --tcp-flags SYN,ACK.
* Nếu 2 cờ SYN/ACK trên khớp, tham chiếu đến module state để kiểm tra trạng thái của nó, dùng tham số switching SYN,ACK -m state để chuyển đến module state.
* Ở module state, kiểm tra xem packet đó có khởi tạo một connection mới hoặc liên kết với một kết nối mà không thấy packets ở cả 2 hướng bằng tham số **NEW**.
* Nếu khớp với điền kiện, dùng tham số -j để nhảy về chain vừa khởi tạo, tăng counter của rule lên để packet sẽ được xử lý ở rule tiếp theo.
* DROP packet (policy chính của chain INPUT).

**Bước 2:** Sau khi packet đã đi qua rule 1, nghĩa là nó không tạo một connection mới hoặc không liên kết với một connection mà không thấy ở cả 2 hướng, nhưng không có nghĩa packet đó không có vấn đề, ta cần xét rule tiếp theo:

* Append thêm rule vào chain INPUT, vẫn sử dụng giao thức TCP nhưng xét 2 cờ SYN,FIN. Nếu cờ FIN đã được set, nghĩa là **krad** ở lúc ban đầu đang yêu cầu một kết nối mới mà **scanme** lại nghĩ rằng kết nối đã mất. Rule này sẽ thông báo rằng kết nối vừa tạo không đầy đủ và nhảy về chain STEALTH\_SCAN để tiếp tục xét.
* DROP packet (policy chính của chain INPUT).

**Bước 3:** Có thể **krad** sẽ không đóng kết nối, thay vào đó nó sẽ gửi một gói RTS thay vì FIN, ở bước 3, ta cần chắc chắn rằng **krad** có gửi RTS hay không bằng cách thay bộ cờ xem xét từ SYN,FIN sang SYN,RST và nhảy về chain STEALTH\_SCAN.

* DROP packet nếu nó có RTS.

**Bước 4:** Xét tất cả các flag có được tạo hay chưa trong điều kiện các flag SYN,RST,ACK,FIN,URG chưa tạo.

* Nếu có, đó là packet được **krad** gửi đến, lập tức nhảy về chain STEALTH\_SCAN và drop nó.

**Bước 5:** Đến lúc này ta đã cơ bản chống lại được STEALTH\_SCAN, nhưng cần chắc chắn rằng không có cờ FIN nào được tạo khi RTS chưa được tạo, tương tự điều ấy với cặp cờ ACK,FIN ACK,PSH, ACK,URG.

### 3.2.1 Lập lệnh mẫu:

**$iptables** -N STEALTH\_SCAN

**$iptables** -A STEALTH\_SCAN -j LOG --log-prefix "stealth\_scan\_attack: "

**$iptables** -A STEALTH\_SCAN -j DROP

**$iptables** -A INPUT -p tcp --tcp-flags SYN,ACK SYN,ACK -m state --state NEW -j STEALTH\_SCAN

**$iptables** -A INPUT -p tcp --tcp-flags SYN,FIN SYN,FIN -j STEALTH\_SCAN

**$iptables** -A INPUT -p tcp --tcp-flags SYN,RST SYN,RST -j STEALTH\_SCAN

**$iptables** -A INPUT -p tcp --tcp-flags ALL SYN,RST,ACK,FIN,URG -j STEALTH\_SCAN

**$iptables** -A INPUT -p tcp --tcp-flags FIN,RST FIN,RST -j STEALTH\_SCAN

**$iptables** -A INPUT -p tcp --tcp-flags ACK,FIN FIN -j STEALTH\_SCAN

**$iptables** -A INPUT -p tcp --tcp-flags ACK,PSH PSH -j STEALTH\_SCAN

**$iptables** -A INPUT -p tcp --tcp-flags ACK,URG URG -j STEALTH\_SCAN

## 3.3 Phòng thủ tấn công tràn ICMP

### 3.3.1 Chọn lựa và giới hạn ICMP

ICMP có 15 loại và mỗi loại có ít nhất một code khác nhau. Riêng ICMP loại 3 có đến 15 code khác nhau. Vậy, chúng ta nên chọn và giới hạn ICMP nào?

Sự chọn lựa này mang tính cá nhân vì mỗi người có cách nhìn khác nhau về ICMP. Riêng tôi, ICMP 0, 3, 4, 8 và 11 nên được dùng, số còn lại không nên cho phép ra vào vì chúng mang những tính chất ảnh hưởng đến vấn đề bảo mật cho máy chủ. Sau khi đã chọn loại ICMP được dùng, ta sẽ tiến hành viết luật cho chúng

Đầu tiên phải cho iptables biết rằng các ICMP thuộc loại cho phép được đi vào hệ thống, dùng đoạn lập lệnh sau:

OK\_ICMP="0 3 4 8 11"

for item in $OK\_ICMP; do

$**iptables** -A INPUT -i $IF -s $NET -p icmp --icmp-type $item -j ACCEPT

$**iptables** -A OUTPUT -o $IF -s $IP -p icmp --icmp-type $item -j ACCEPT

Đoạn lặp trên thiết lập nhóm luật xử lý giao thức ICMP cho phép các loại ICMP trong biến $OK\_ICMP. Thật ra nhóm luật trên chỉ mới giới hạn loại ICMP được dùng nhưng chưa có bất cứ cơ chế nào kiểm soát lượng lưu thông ICMP ra vào. Bởi vậy, muốn vững hơn thì nên đưa vào -m limit để tạo nên mức kiểm soát cụ thể:

**$iptables** -A INPUT -i $IF -s $NET -p icmp --icmp-type $item -m limit --limit 1/s --limit-burst 1 -j ACCEPT

**$iptables** -A OUTPUT -o $IF -s $IP -p icmp --icmp-type $item -m limit --limit 1/s --limit-burst 1 -j ACCEPT

-m limit trên áp đặt giá trị “rate” rất khắt khe: chỉ tiếp nhận một gói ICMP trong mỗi giây. Với giới hạn này, các cuộc dội ICMP (ICMP flood) gần như vô tác dụng. Để tiết kiệm tài nguyên hơn, ta có thể nâng “rate” lên ở mức 5/s hoặc 10/s nhưng đừng quá lâu.

### 3.3.2 Cản ICMP hoặc cản ICMP “một nửa”

ICMP được dùng làm bước đầu cho những cuộc khám phá một host / network. Có rất nhiều công cụ rà cổng (**port scan**) dựa trên hồi báo của ICMP để quyết định các bước khám phá kế tiếp. Những thủ thuật tìm hiểu “chữ ký hệ thống” (**system foot-print**) cũng trông cậy rất nhiều vào ICMP. Chỉ với rate 1/s cũng chưa đủ làm hệ thống cứng cáp hơn, ta cần thêm rất nhiều kỹ thuật.

#### a) Cản ICMP

Mức độ cản ở đây chỉ dừng lại ở mức độ cản không cho các gói ICMP khởi tạo và đi vào từ bên ngoài. Máy chủ có thể khởi tạo các gói ICMP (trong giới hạn các loại ICMP cho phép thuộc biến $OK\_ICMP) và các máy bên ngoài chỉ có thể “trả lời” các gói ICMP máy chủ tạo ra. Chức năng -m state một lần nữa hữu dụng cho trường hợp này:

$**iptables** -A INPUT -i $IF -s $NET -p icmp --icmp-type $item -m state --state ESTABLISHED -m limit --limit 1/s --limit-burst 1 -j ACCEPT

$**iptables** -A OUTPUT -o $IF -s $IP -p icmp --icmp-type $item -m state --state NEW,ESTABLISHED -m limit --limit 1/s --limit-burst 1 -j ACCEPT

TA có thể thấy gói tin đi vào xuyên qua chuỗi INPUT chỉ có thể được tiếp nhận ở tình trạng ESTABLISHED nhưng gói tin đi ra xuyên qua chuỗi OUTPUT thì có thể được tiếp nhận ở cả tình trạng NEW và ESTABLISHED. -m state hỗ trợ cho -m limit trong trường hợp này tạo nên các luật rất khắt khe cho ICMP. Có những quan điểm cho rằng quá khắt khe với ICMP không tiện dụng cho các hoạt động mạng, vì vậy, lựa chọn thắt chặt hay không là do quyết định của cá nhân người sử dụng.

#### b) Cản ICMP “một nửa”

**$iptables** -A INPUT -i $IF -s $NET -p icmp --icmp-type $item -m length 42:43 -m limit --limit 1/s --limit-burst 1 -j ACCEPT

**$iptables** -A OUTPUT -o $IF -s $IP -p icmp --icmp-type $item -m length 42:43 -m limit --limit 1/s --limit-burst 1 -j ACCEPT

Thông thường, ping gởi đi một gói dữ liệu nào đó để host được ping theo mặc định. Nếu firewall được ấn định như trên, chỉ có gói tin ICMP nào có chiều dài trong khoảng 42 đến 43 bytes thì mới tiếp nhận. Điều này có nghĩa, khi một ai đó thử ping theo mặc định trên MS-DOS prompt hoặc trên một \*nix console chắc chắn sẽ không có kết quả vì không thoả mãn kích thước gói tin đã ấn định. Tính “mở một nửa” nằm ở kích thước cụ thể của gói tin. Chỉ có bạn biết kích thước gói tin là bao nhiêu để ping vào máy chủ thành công (đây chính là tính “mở”); đối với mọi người dùng khác họ sẽ không ping vào máy chủ thành công vì hầu hết họ dùng kích thước gói tin theo mặc định (đây chính là tính “đóng”).

Có thể có những cá nhân kiên nhẫn dò từng kích thước gói tin hoặc viết một đoạn lập lệnh để thực hiện quy trình dò này nhưng chuyện này chỉ xảy ra nếu cá nhân ấy nghi ngờ máy chủ ấn định kích thước gói tin cụ thể. Đối với người dùng bên ngoài, khả năng cản hoàn toàn và cản “một nửa” không khác gì nhau vì đơn giản không thể “ping” ngay từ đầu. Với các ứng dụng kiểm soát ICMP trên, mối đe doạ của các dạng tấn công dựa trên ICMP hầu như vô hiệu hoá.

## 3.4 Phòng thủ tấn công HTTP DdoS

### 3.4.1 Ý tưởng phòng thủ

#### a) Khó khăn sẽ gặp phải

Với môi trường internet bên ngoài:

Việc phân biệt lưu lượng tấn công và lưu lượng truy cập bình thường là khó khăn, đặc biệt là trong trường hợp tấn công lớp ứng dụng như botnet thực hiện cuộc tấn công tràn HTTP chống lại máy chủ nạn nhân. Bởi vì mỗi bot trong một mạng botnet tạo ra các yêu cầu mạng dường như hợp pháp, lưu lượng truy cập không bị giả mạo và có thể xuất hiện trong bản gốc.

Các cuộc tấn công lớp ứng dụng đòi hỏi một chiến lược thích ứng bao gồm khả năng giới hạn lưu lượng dựa trên các bộ quy tắc cụ thể, có thể dao động thường xuyên.

Các công cụ như WAF được cấu hình đúng có thể giảm thiểu lưu lượng truy cập không có thật được chuyển đến máy chủ gốc, làm giảm đáng kể tác động của nỗ lực DDoS.

Với các cuộc tấn công khác như tràn SYN hoặc các cuộc tấn công phản chiếu như khuếch đại NTP, các chiến lược có thể được sử dụng để giảm lưu lượng khá hiệu quả với điều kiện mạng có băng thông đủ để nhận chúng hay không. Thật không may, hầu hết các mạng không thể nhận được một cuộc tấn công khuếch đại 300Gbps và thậm chí còn không có mạng nào có thể định tuyến và phục vụ đúng khối lượng lớp ứng dụng yêu cầu khi có một cuộc tấn công diễn ra.

Với hệ thống được cài đặt phòng thủ:

Khi xây dựng một hệ thống ghi dấu (log) tập trung, các ứng dụng chạy trên nền UNIX và nền tảng mã nguồn mở đều hỗ trợ sẵn syslog. Nhưng các máy chủ Windows cũng như các hệ thống mua ở ngoài mặc định sẽ không có syslog. Điều này làm cho người cài đặt phải thêm các tiện ích syslog cho Windows hoặc xây dựng riêng một quy định về log của các ứng dụng.

#### b) Giảm thiểu tấn công

Với các khó khăn như trên, việc ngăn chặn triệt để là điều dường như không thể đạt đến được. Nhưng có thể giảm thiểu bằng nhiều cách.

Thông thường, các máy chủ đơn giản, rẻ tiền sẽ cài đặt một công cụ kiểm tra xem lưu lượng đang truy cập có thực sự không phải đến từ một con bot, **CAPTCHA**, thường thì các yêu cầu sẽ được viết bằng Javascript xử lý trước khi gửi về server, điều này làm giảm nhẹ tính nghiêm trọng của cuộc tấn công.

Một giải pháp chống DDoS điển hình có 2 điểm cơ bản: phát hiện và đánh chặn. Một số cách cài đặt hiện đại tách hẳn phần phát hiện ra khỏi hệ thống phòng thủ và thực hiện nó trong môi trường offline bằng cách sử dụng thông tin từ việc thu thập.

Cụ thể, thông tin từ hệ thống đánh chặn cũng như các nguồn khác như **web server**, **proxy** hay **firewall** sẽ được đưa vào hệ thống phân tích để chạy offline, rồi kết quả phân tích này sẽ được cập nhật ngược trở lại cho hệ thống đánh chặn. Với cách làm này, giải pháp này có thể đáp ứng được lượng tải rất lớn vì hệ thống không phải tốn quá nhiều resource để phân tích on-the-fly một packet hay request như các giải pháp khác.

Nói một cách khác, ta sẽ tập trung hơn cho việc trả lời câu hỏi: “***Ai đang gửi hàng tá request trong 10 phút vừa rồi?***” thay vì phân tích cặn kẽ từng packet.

Quy trình thực hiện cơ bản:

wc kern.log | awk '{print $1}'

Output của dòng lệnh trên là số request hệ thống đã nhận từ lúc cài đặt đến nay, ví dụ số đầu tiên in ra sẽ là 1997, để lệnh chạy trong 1 phút, ta nhận được số 2997.

Từ 2 output trong khoảng đầu và cuối, ta ước tính được số request trong một phút là 1000. Chạy lệnh sau:

tail 'kern.log' -1000 | awk '{print $1}' | sort | uniq -c | sort -rn

Lệnh **tail** sẽ lấy từ kern.log 1000 dòng “cuối cùng trong file” (là con số ta trừ ở trên), truyền qua cho awk để chỉ lọc ra phần IP, gom chúng lại bằng lệnh sort rồi uniq -c sẽ đếm số lần xuất hiện của từng IP (các dòng giống nhau sẽ cộng lại) , chuyền qua cho sort -n để sắp từ nhỏ đến lớn (lớn nhất cuối cùng) và sau cùng là tail -1 là sẽ lấy 01 dòng cuối trong sort -n ở trên (tức là cái IP truy cập nhiều nhất trong 1 phút qua).

Thật là dễ dàng, ta có một output ví dụ như sau:

300 42.197.112.151

300 là số lần gửi request chạm server, dãy IP phía sau là IP nguồn của request. Với một người bình thường, 300 request sẽ trở nên bất bình thường, và server của ta sẽ chặn được kịp thời.

### 3.4.2 Thiết lập luật cho iptables

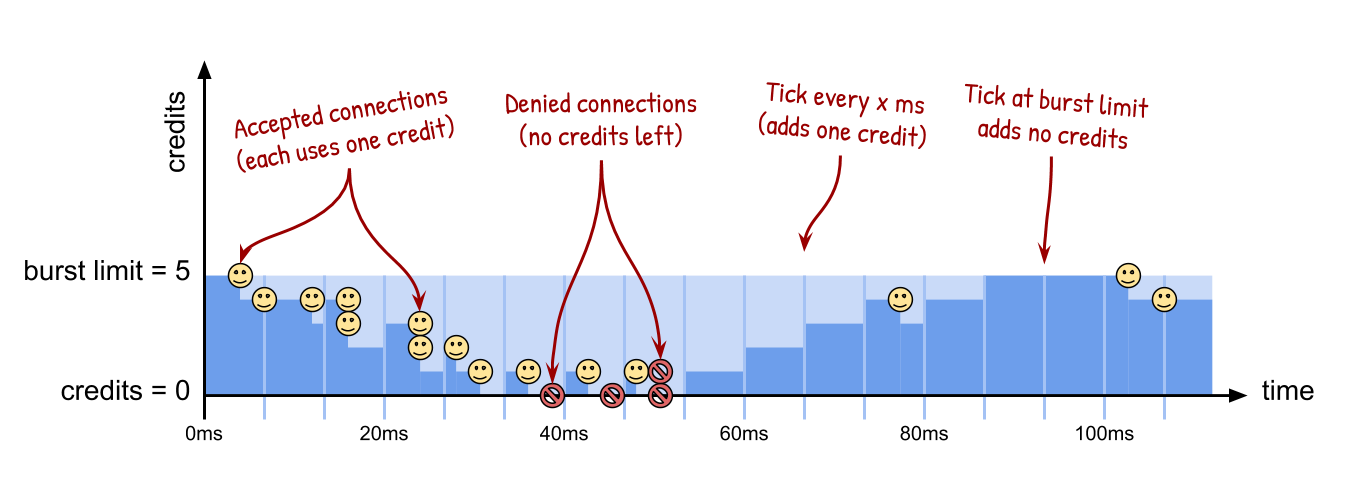
Về phía iptables, ta sẽ tiến hành cài đặt đơn giản các luật, cho iptables phân tích packet on-the-fly, nghĩa là tất cả packet sẽ được quét tại thời gian thực packet đi vào hệ thống.

Với mấu chốt là cổng 80 và 443 cho dịch vụ HTTP, ta sẽ thiết đặt luật cho iptables để nghe trên cổng đó.

Việc thiết lập luật sẽ sử dụng hashlimit của iptables. Hashlimit sử dụng các hàm băm để thể hiện kết quả của tỉ lệ giới hạn cho một nhóm các kết nối trong cùng một luật iptables. Việc phân nhóm có thể được thực hiện trên mỗi nhóm máy chủ hoặc trên mỗi cổng. Haslimit sẽ cho phép người dùng diễn đạt **n** gói mỗi chu kì thời gian.

Thiết lập các hashlimit cho luật, các hashlimit cơ bản để xác định lưu lượng vào có phải là một cuộc tấn công DDoS hay không.

* Chỉ cho phép 1 packet trên 1 giây, số lượng packet có thể lớn hơn, nhưng nếu để 1 thì chắc chắn sẽ chặn được rất nhiều.
* Thiết lập giới hạn burst ở mức 1000. Sử dụng logic “dùng hoặc mất” cho sự bùng nổ lưu lượng.



Hình 12 Giải thích về cách brust-limit hoạt động

Ví dụ về câu lệnh sẽ được sử dụng:

--limit 50/sec --limit-burst 20

Giả sử server sẽ nhận được 1 packet mỗi mili giây trong 1 giây, thì sẽ có 70 packet được nhận vào tối đa trong 1 giây. 20 packet đầu sẽ được chấp nhận và reset credit lại về 0. Cứ mỗi 20ms (1/50s) thì credit sẽ được làm đầy trở lại và packet được chấp nhận, và có thêm 50 packet nữa.

* Thiết lập thời gian tồn tại cho các giá trị băm, thường là 30s.
* Nhóm các địa chỉ nguồn theo độ dài tiền tố, từ đấy các mạng con cũng phải chịu quản lý của hashlimit, tránh việc kiểm tra thiếu gói tin.