**ĐẠI HỌC ĐÀ NẴNG**

**TRƯỜNG ĐẠI HỌC BÁCH KHOA**

**KHOA CÔNG NGHỆ THÔNG TIN**



**PBL4: DỰ ÁN HỆ ĐIỀU HÀNH & MẠNG MÁY TÍNH**

**ĐỀ TÀI: TÌM HIỂU HỆ ĐIỀU HÀNH LINUX VÀ XÂY DỰNG ỨNG DỤNG PHÁT HIỆN XÂM NHẬP**

**GIẢNG VIÊN HƯỚNG DẪN:**

**T.S Nguyễn Thế Xuân Ly**

**SINH VIÊN THỰC HIỆN:**

**1. Võ Hoàng Bảo Nhóm: 20.15B MSSV: 102200246**

**2. Bùi Hải Nam Nhóm: 20.15B MSSV: 102200273**

**3. Nguyễn Hoàng Quân Nhóm: 20.15B MSSV: 102200281**

**Đà Nẵng, 12/2022**

MỤC LỤC

[1](#_Toc123278597)

[MỤC LỤC 2](#_Toc123278598)

[DANH SÁCH HÌNH VẼ 4](#_Toc123278599)

[DANH SÁCH BẢNG BIỂU 5](#_Toc123278600)

[DANH SÁCH CÁC TỪ VIẾT TẮT 6](#_Toc123278601)

[MỞ ĐẦU 7](#_Toc123278602)

[CHƯƠNG 1. CƠ SỞ LÝ THUYẾT 8](#_Toc123278603)

[1.1. Giới thiệu về hệ điều hành Linux 8](#_Toc123278604)

[1.1.1. Giới thiệu chung 8](#_Toc123278605)

[1.1.2. Cấu trúc hệ điều hành Linux 8](#_Toc123278606)

[1.1.3. Ưu điểm của hệ điều hành Linux 9](#_Toc123278607)

[1.1.4. Nhược điểm của hệ điều hành Linux 10](#_Toc123278608)

[1.2. Tổng quan về IDS và IPS 11](#_Toc123278609)

[1.2.1. IDS 11](#_Toc123278610)

[1.2.2. Những kĩ thuật phát hiện xâm nhập 11](#_Toc123278611)

[1.2.3. Phân biệt mô hình IDS 11](#_Toc123278612)

[1.3. Hệ thống IDS Snort 12](#_Toc123278613)

[1.3.1. Giới thiệu 12](#_Toc123278614)

[1.3.2. Snort là gì? 12](#_Toc123278615)

[1.3.3. Các thành phần của Snort 12](#_Toc123278616)

[1.3.4. Các chế độ của Snort 16](#_Toc123278617)

[1.3.5. Chế độ NIDS của snort 16](#_Toc123278618)

[1.3.6. Cấu trúc luật của Snort 17](#_Toc123278619)

[1.4. Giới thiệu về IPtables 20](#_Toc123278620)

[1.4.1. Cơ chế xử lý package trong IPtables 21](#_Toc123278621)

[1.4.2. Một số khái niệm trong Iptables. 22](#_Toc123278622)

[1.4.3. Cài đặt Iptables 23](#_Toc123278623)

[1.5. Các giao thức liên quan và các cuộc tấn công tương ứng: 24](#_Toc123278624)

[1.5.1. Giao thức ICMP 24](#_Toc123278625)

[1.5.2. Giao thức TCP và tấn công Syn flood 25](#_Toc123278626)

[CHƯƠNG 2. PHÂN TÍCH THIẾT KẾ HỆ THỐNG 25](#_Toc123278627)

[2.1. Sơ đồ User-Case 25](#_Toc123278628)

[CHƯƠNG 3. TRIỂN KHAI VÀ ĐÁNH GIÁ KẾT QUẢ 27](#_Toc123278629)

[3.1. Tấn công Ping Of Death 27](#_Toc123278630)

[3.2. Tấn công Syn flood 28](#_Toc123278631)

[KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN 30](#_Toc123278632)

[TÀI LIỆU THAM KHẢO 31](#_Toc123278633)

[[1] Hà Quang, Giáo trình Hệ điều hành Unix-Linux 31](#_Toc123278634)

[[2] tryhackme.com, Snort, https://tryhackme.com/room/snort, 2/12/2022 31](#_Toc123278635)

[[3] Nguyễn Hưng, Iptables là gì và cách sử dụng iptables, https://vietnix.vn/iptables-la-gi/, 2/12/2022 31](#_Toc123278636)

DANH SÁCH HÌNH VẼ

[Hình 1: Cấu trúc hệ điều hành Linux 9](#_Toc123278637)

[Hình 2: Các thành phần của Snort 13](#_Toc123278638)

[Hình 3: Cấu trúc luật của Snort 17](#_Toc123278639)

[Hình 4: Ví dụ luật của Snort 17](#_Toc123278640)

[Hình 5: Sơ đồ Iptables/Netfilter 21](#_Toc123278641)

[Hình 6: Quá trình xử lý Nat Table 21](#_Toc123278642)

[Hình 7: Quá trình xử lý Filter Table 22](#_Toc123278643)

[Hình 8: Quá trình xử lý Mangle Table 22](#_Toc123278644)

[Hình 9: Sơ đồ User-Case 26](file:///C:\Users\HP\git\PBL4\PBL4-Mau-bao-cao.docx#_Toc123278645)

[Hình 10: Người tấn công gửi các ping có dung lượng cao 27](#_Toc123278646)

[Hình 11: Hệ thống phát hiện Ping Of Death 27](#_Toc123278647)

[Hình 12: Chi tiết cảnh báo Ping Of Dead 28](#_Toc123278648)

[Hình 13: Người tấn công gửi ngập cờ SYN 28](#_Toc123278649)

[Hình 14: Hệ thống cảnh báo SYN Attack 29](#_Toc123278650)

[Hình 15: Chi tiết cảnh báo SYN Attack 29](#_Toc123278651)

DANH SÁCH BẢNG BIỂU

[Bảng 1: Danh sách các từ viết tắt 6](#_Toc123278652)

[Bảng 2: Tham số chế độ NIDS của Snort 17](#_Toc123278653)

[Bảng 3: Tham số Rule Header 18](#_Toc123278654)

[Bảng 4: Tham số General Option 19](#_Toc123278655)

[Bảng 5: Tham số Payload Option 19](#_Toc123278656)

[Bảng 6: Tham số Non-Payload Option 20](#_Toc123278657)

[Bảng 7: Các tùy chọn iptables 24](#_Toc123278658)

[Bảng 8: Câu lệnh hồi phục và các tùy chọn 24](#_Toc123278659)

[Bảng 9: Câu lệnh hồi phục và các tùy chọn 24](#_Toc123278660)

DANH SÁCH CÁC TỪ VIẾT TẮT

|  |  |
| --- | --- |
| IP | Internet Protocol |
| ICMP | Internet Control Message Protocol |
| TCP | Transmission Control Protocol |
| IDS | Intrusion Detection System |
| NIDS | Network Intrusion Detection System |
| HIDS | Host-based Intrusion Detection System |
| IPS | Intrusion Prevention System |
| NIPS | Network Intrusion Prevention System |
| NBA | Network Behaviour Analysis |

Bảng 1: Danh sách các từ viết tắt

MỞ ĐẦU

Xã hội ngày càng phát triển, Internet trở thành một phần không thể thiếu đối với từng cá nhân, doanh nghiệp, các tổ chức, trường học. Internet đã trở thành công cụ, phương thức giúp cho các doanh nghiệp tiếp cận với khách hàng, cung cấp dịch vụ, quản lý dữ liệu của tổ chức một cách hiệu quả và nhanh chóng. Cùng với sự phát triển theo chiều hướng tốt, các cuộc tấn công và xâm nhập mạng của những kẻ xấu cũng phát triển theo và trở nên cực kỳ phổ biến. Vì thế, nhu cầu có được một hệ thống hỗ trợ giám sát và bảo vệ hệ thống mạng trực quan nhằm giúp cho công việc quản trị mạng được tập trung và đạt hiệu quả cao là rất cần thiết.

Trong bối cảnh công ty đang xây dựng một máy chủ truyền dữ liệu tin cậy, chúng em chọn nghiên cứu và thực hiện đề tài: “Tìm hiểu hệ điều hành Linux và xây dựng hệ thống phát hiện xâm nhập” để hỗ trợ cho việc thu thập thông tin mạng và phát hiện thâm nhập bằng snort, tập trung trên các giao thức ICMP, TCP.

Chúng em xin chân thành cảm ơn giảng viên ThS. Nguyễn Thế Xuân Ly đã giúp đỡ chúng em trong quá trình hoàn thiện đồ án này.

# CHƯƠNG 1. CƠ SỞ LÝ THUYẾT

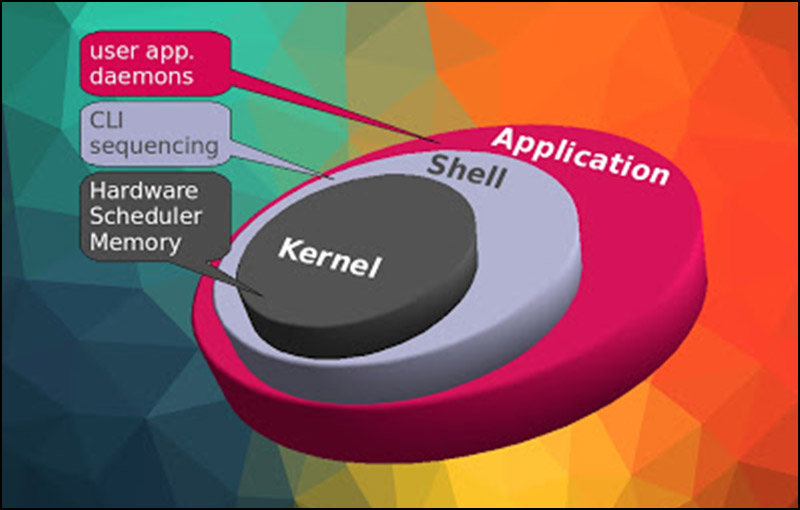
## Giới thiệu về hệ điều hành Linux

### Giới thiệu chung

Linux là tên gọi của một hệ điều hành máy tính và cũng là tên hạt nhân của hệ điều hành. Nó có lẽ là một ví dụ nổi tiếng nhất của phần mềm tự do và của việc phát triển mã nguồn mở. Linux được phát triển từ năm 1991 dựa trên hệ điều hành Unix và bằng viết bằng ngôn ngữ C.

### Cấu trúc hệ điều hành Linux

* Kernel: Hay được gọi là phần Nhân vì đây là phần quan trọng nhất trong máy tính bởi chứa đựng các module hay các thư viện để quản lý, giao tiếp giữa phần cứng máy tính và các ứng dụng.
* Shell: Shell là phần có chức năng thực thi các lệnh (command) từ người dùng hoặc từ các ứng dụng yêu cầu, chuyển đến cho Kernel xử lý. Shell chính là cầu nối để kết nối Kernel và Application, phiên dịch các lệnh từ Application gửi đến Kernel để thực thi.
* Application: Đây là phần quen thuộc với chúng ta nhất, phần để người dùng cài đặt ứng dụng, chạy ứng dụng để người dùng có thể phục vụ cho nhu cầu của mình.



Hình 1: Cấu trúc hệ điều hành Linux

### Ưu điểm của hệ điều hành Linux

*Bản quyền và chi phí hợp lý*

Với bản chất là một mã nguồn mở, Linux được phát triển miễn phí cho người sử dụng. Người dùng có thể nhìn thấy hầu như mọi dòng code trong Linux. Trong đó Windows bản quyền và bộ Office bản quyền sẽ phải mất khoảng vài triệu đồng để sở hữu tương tự.

Sử dụng Linux, bạn sẽ được sử dụng miễn phí tất cả các tính năng kèm bộ ứng dụng cho người dùng văn phòng miễn phí. Học tập, xem phim, nghe nhạc, chat,… mọi hoạt động của bạn không phải lo lắng gì về vấn đề bản quyền nữa.

*Hỗ trợ tốt cho lập trinh viên – quản trị mạng*

Hệ thống Linux hoạt động ổn định, hiệu quả cao. Do đó, Linux được hỗ trợ tốt cho nhiều công việc, đặc biệt là đối với những công việc đòi hỏi tính ổn định hệ thống cao như quản trị mạng và lập trình viên.

*Sự hỗ trợ kỹ thuật tích cực*

Linux cung cấp dịch vụ hỗ trợ kỹ thuật 24/7 trong cả năm đối với mọi dịch vụ. Có thể kể đến như Red Hat, Novell và Canonical.

*Linh hoạt*

Nếu hiểu sâu về Linux, bạn hoàn toàn có thể dễ dàng chỉnh sửa theo ý mình. Linux đúng là một môi trường lý tưởng cho các lập trình viên cũng như các nhà phát triển vì Linux mang lại sự tương thích với rất nhiều môi trường khác nhau.

*Giao diện phong phú*

Với hệ điều hành Linux, giao diện hoàn toàn tách rời với hệ thống. Do đó, bạn có thể đổi môi trường giao diện mà không cần lo lắng xem phải cài lại chương trình hay không. Bạn có thể tha hồ chọn giao diện như GNOME, KDE hay gần đây hơn là Unity.

*Độ an toàn và khả năng bảo mật, hạn chế sự tấn công của các mã độc và virut cao*

Nếu như trên Windows, những con virut, mã độc,… luôn là thứ khiến bạn phải chật vật đối mặt với Linux sẽ mang lại cho bạn cảm giác an toàn. Điều tuyệt vời là tất cả bọn chúng đều không thể hoạt động được trên nền tảng này. Công việc của bạn chỉ là xóa khi thấy bọn chúng trong USB hay ổ cứng di động (ổ cứng SSD).

*Hoạt động tốt ngay cả trên những máy có cầu hình yếu*

Thông thường, một phiên bản mới được nâng cấp sẽ kèm theo yêu cầu phần cứng cũng phải nâng lên theo. Vậy trong trường hợp cấu hình máy tính của bạn không đủ đáp ứng yêu cầu của phiên bản mới thì sao?

Linux sẽ giúp bạn tránh khỏi những lo lắng đó. Với sự nâng cấp, hỗ trợ thường xuyên từ cộng đồng lập trình, Linux tự tin hoạt động mượt mà cực kỳ ổn định ngay ca trên các máy tính có cấu hình thấp.

### Nhược điểm của hệ điều hành Linux

Bên cạnh những ưu điểm trên, hệ điều hành Linux vẫn còn một số điểm hạn chế:

* Số lượng ứng dụng hỗ trợ trên Linux chưa phong phú, còn khá ít sự lựa chọn cho người dùng.
* Một số nhà sản xuất không phát triển Driver hỗ trợ nền tảng Linux.
* Bạn sẽ mất một khoảng thời gian để “thích nghi” nếu chuyển từ hệ điều hành Window sang Linux vì thực sự khá khó để làm quen với Linux.
* Tùy vào tính chất, nhu cầu sử dụng mà bạn cần tìm một hệ điều hành thích hợp với mình.

## Tổng quan về IDS và IPS

### IDS

Hệ thống phát hiện xâm nhập (Intrusion Detection System - IDS) là một giải pháp giám sát thụ động nhằm phát hiện các hoạt động/mô hình độc hại, sự cố bất thường và các vi phạm chính sách. Nó chịu trách nhiệm cho việc tạo cảnh báo cho từng sự kiện đáng ngờ

### Những kĩ thuật phát hiện xâm nhập

* Phát hiện sự bất thường (Anomaly-Based):

Phát hiện giao thức bất thường (Protocol Anomaly Detection). Giao thức bất thường nghĩa là những trường hợp vi phạm các định dạng, các tiêu chuẩn các hành vi đã được quy định thành chuẩn Internet từ trước đó.

Vd: Kích thước gói tin icmp tối đa là 65,535 bytes, attacker cố tình gửi một gói tin có kích thước lớn hơn tiêu chuẩn đó để làm tràn bộ đệm (ping of death).

Phát hiện thâm nhập dựa trên sự thống kê bất thường (Statistical Anomaly Based ID). Kĩ thuật này nhấn mạnh việc đo đếm các hoạt động bình thường trên mạng. Ví dụ số tiến trình hoạt động quá mức trên CPU, số lượng gói tin gửi đi quá mức,…

* Phát hiện sự lạm dụng/ dấu hiệu (Misuse/Signature-Based)

Phát hiện sự lạm dụng là phát hiện những kẻ xâm nhập đang cố gắng đột nhập vào hệ thống mà sử dụng một số kỹ thuật đã biết. Nó liên quan đến việc mô tả đặc điểm các cách thức xâm nhập vào hệ thống đã được biết đến, mỗi cách thức này được mô tả như một mẫu. Hệ thống phát hiện sự lạm dụng chỉ thực hiện kiểm soát đối với các mẫu đã rõ ràng. Mẫu có thể là một xâu bit cố định (ví dụ như một virus đặc tả việc chèn xâu),… dùng để mô tả một tập hay một chuỗi các hành động đáng nghi ngờ.

* Phát hiện sự vi phạm chính sách (Policy-Based)

Kỹ thuật này so sánh các hoạt động được phát hiện với các chính sách bảo mật và cấu hình hệ thống. Mô hình này giúp phát hiện vi phạm chính sách.

### Phân biệt mô hình IDS

Có 2 mô hình IDS chính:

* Network Intrusion Detection System (NIDS): NIDS giám sát lưu lượng truy cập các khu vực khác nhau của mạng. Mục đích là để điều tra lưu lượng trên toàn bộ subnet. Nếu một dấu hiệu được nhận dạng, một thông báo được tạo ra.
* Host-based Intrusion Detection System (HIDS): HIDS giám sát lưu lượng truy cập một thiết bị đầu cuối duy nhất. Mục đích là để diều tra lưu lượng trên một thiết bị cụ thể. Nếu một dấu hiệu được nhận dạng, một thông báo được tạo ra.

## Hệ thống IDS Snort

### Giới thiệu

Cũng giống như virus, hầu hết các hoạt động tấn công hay xâm nhập đều có các dấu hiệu riêng. Các thông tin về các dấu hiệu này sẽ được sử dụng để tạo nên các luật cho Snort. Thông thường, các bẫy (honey pots) được tạo ra để tìm hiểu xem các kẻ tấn công làm gì cũng như các thông tin về công cụ và công nghệ chúng sử dụng. Và ngược lại, cũng có các cơ sở dữ liệu về các lỗ hổng bảo mật mà những kẻ tấn công muốn khai thác. Các dạng tấn công đã biết này được dùng như các dấu hiệu để phát hiện tấn công xâm nhập. Các dấu hiệu đó có thể xuất hiện trong phần header của các gói tin hoặc nằm trong phần nội dung của chúng. Hệ thống phát hiện của Snort hoạt động dựa trên các luật (rules) và các luật này lại được dựa trên các dấu hiệu nhận dạng tấn công. Các luật có thể được áp dụng cho tất cả các phần khác nhau của một gói tin dữ liệu.

Một luật có thể được sử dụng để tạo nên một thông điệp cảnh báo, log một thông điệp hay có thể bỏ qua một gói tin.

### Snort là gì?

Snort là một NIDS/NIPS mã nguồn mở hoạt, họat động dựa trên một tập luật linh họat, thông qua phân tích các protocol, tìm kiếm nội dung và các bộ tiền xử lý (preprocessor) để phát hiện ra hàng ngàn lọai sâu (worm), các kiểu tấn công, quét cổng và những hành động đáng ngờ khác trên mạng. Nó được phát triển và bảo trì bởi Martin Roesch, các nhà đóng góp mã nguồn mở và Cisco Talos team.

### Các thành phần của Snort

Snort được chia thành nhiều thành phần một cách logic. Những thành phần này làm việc cùng nhau để phát hiện các cuộc tấn công cụ thể và để tạo ra các định dạng cần thiết từ hệ thống phát hiện. Snort bao gồm các thành phần chính sau đây:

Bộ phận giải mã gói tin (Packet Decoder).

Bộ phận tiền xử lý (Preprocessors).

Bộ phận phát hiện (Detection Engine).

Bộ phận phát hiện (Detection Engine).

Bộ phận ghi nhận và thông báo (Loging and alerting system).

Bộ phận đầu ra (Output Modules).

Bất kì dữ liệu nào đến từ internet đều đi vào packet decoder. Trên đường đi của nó với các module đầu ra, nó có thể bị loại bỏ, ghi nhận hoặc có một cảnh báo được tạo ra.

Diagram

Description automatically generated

Hình 2: Các thành phần của Snort

* Bộ phận giải mã gói tin (Packet Decoder)

Bộ phận giải mã gói tin: các gói dữ liệu đi vào qua các cổng giao tiếp mạng, các cổng giao tiếp này có thể là: Ethernet, 802.11, FDDI, SLIP, PPP… Và được giải mã bởi packet decoder, trong đó xác định giao thức được sử dụng cho gói tin và dữ liệu phù hợp với hành vi được cho phép của phần giao thức của chúng. Packet Decoder có thể tạo ra các cảnh báo riêng của mình dựa trên tiêu đề của giao thức, các gói tin quá dài, bất thường hoặc không chính xác tùy chọn TCP, UDP, ICMP và IP được thiết lập trong các tiêu đề, và các hành vi khác. Có thể kích hoạt hoặc vô hiệu hóa các cảnh báo dài dòng cho tất cả các trường trong tập tin snort.conf. Sau khi dữ liệu được giải mã đúng, chúng sẽ được gửi đến bộ phận preprocessor.

* Bộ phận tiền xử lý (reprocessors)

Bộ phận tiền xử lý là một thành phần rất quan trọng đối với bất kỳ một hệ thống IDS nào để có thể chuẩn bị gói dữ liệu đưa vào cho module phát hiện phân tích. Ba nhiệm vụ chính của các module loại này là:

Kết hợp lại các gói tin: Khi một lượng dữ liệu lớn được gửi đi, thông tin sẽ không đóng gói toàn bộ vào một gói tin mà phải thực hiện việc phân mảnh, chia gói tin ban đầu thành nhiều gói tin rồi mới gửi đi. Khi Snort nhận được các gói tin này nó phải thực hiện việc ghép nối lại để có được dữ liệu nguyên dạng ban đầu, từ đó mới thực hiện được các công việc xử lý tiếp. Như ta đã biết khi một phiên làm việc của hệ thống diễn ra, sẽ có rất nhiều gói tin được trao đổi trong phiên đó. Một gói tin riêng lẻ sẽ không có trạng thái và nếu công việc phát hiện xâm nhập chỉ dựa hoàn toàn vào gói tin đó sẽ không đem lại hiệu quả cao. Module tiền xử lý phân thành các luồng riêng giúp Snort có thể hiểu được các phiên làm việc khác nhau từ đó giúp đạt được hiệu quả cao hơn trong việc phát hiện xâm nhập.

Giải mã và chuẩn hóa giao thức (decode/normalize): công việc phát hiện xâm nhập dựa trên dấu hiệu nhận dạng nhiều khi bị thất bại khi kiểm tra các giao thức có dữ liệu có thể được thể hiện dưới nhiều dạng khác nhau. Nếu Snort chỉ thực hiện đơn thuần việc so sánh dữ liệu với dấu hiệu nhận dạng sẽ xảy ra tình trạng bỏ sót các hành vi xâm nhập. Do vậy, một số module tiền xử lý của Snort phải có nhiệm vụ giải mã và chỉnh sửa, sắp xếp lại các thông tin đầu vào này để thông tin khi đưa đến module phát hiện có thể phát hiện được mà không bỏ sót. Hiện nay Snort đã hỗ trợ việc giải mã và chuẩn hóa cho các giao thức: telnet, http, rpc, arp.

Phát hiện các xâm nhập bất thường (nonrule/anormal): các plugin tiền xử lí dạng này thường dùng để đối phó với các xâm nhập không thể hoặc rất khó phát hiện được bằng các luật thông thường hoặc các dấu hiệu bất thường trong giao thức. Các module tiền xử lý dạng này có thể thực hiện việc phát hiện xâm nhập theo bất cứ cách nào mà ta nghĩ ra từ đó tăng cường thêm tính năng cho Snort. Ví dụ: một plugin tiền xử lý có nhiệm vụ thống kê thông lượng mạng tại thời điểm bình thường để rồi khi có thông lượng mạng bất thường xảy ra nó có thể tính toán, phát hiện và đưa ra cảnh báo (phát hiện xâm nhập theo mô hình thống kê). Phiên bản hiện tại của Snort có đi kèm hai plugin giúp phát hiện các xâm nhập bất thường đó là portscan và backoffice. Portcan dùng để đưa ra cảnh báo khi kẻ tấn công thực hiện việc quét các cổng của hệ thống để tìm lỗ hổng. Backoffice dùng để đưa ra cảnh báo khi hệ thống đã bị nhiễm troạn backoffice và kẻ tấn công từ xa kết nối tới backoffice thực hiện các lệnh từ xa.

* Bộ phận phát hiện (Detection Engine)

Bộ phận phát hiện: Đây là module quan trọng nhất của Snort. Nó chịu trách nhiệm phát hiện các dấu hiệu xâm nhập. Module phát hiện sử dụng các luật được định nghĩa trước để so sánh với dữ liệu thu thập được từ đó xác định xem có xâm nhập xảy ra hay không. Rồi tiếp theo mới có thể thực hiện một số công việc như ghi log, tạo thông báo và kết xuất thông tin.

Một vấn đề rất quan trọng trong module phát hiện là vấn đề thời gian xử lý các gói tin: một IDS thường nhận được rất nhiều gói tin và bản thân nó cũng có rất nhiều các luật xử lý. Có thể mất những khoảng thời gian khác nhau cho việc xử lý các gói tin khác nhau. Và khi thông lượng mạng quá lớn có thể xảy ra việc bỏ sót hoặc không phản hôi được cùng lúc. Khả năng xử lý của module phát hiện dựa trên một số yếu tố như: số lượng các luật, tốc độ của hệ thống đang chạy Snort, tải trên mạng. Một số thử nghiệm cho biết, phiên bản hiện tại của Snort khi được tối ưu hóa chạy trên hệ thống có nhiều bộ vi xử lý và cấu hình máy tính tương đối mạnh thì có thể hoạt động tốt trên cả các mạng cỡ Giga.

Một module phát hiện cũng có khả năng tách các phần của gói tin ra và áp dụng các luật lên từng phần nào của gói tin đó. Các phần đó có thể là:

IP header

Header ở tầng giao vận: TCP, UDP

Header ở tầng ứng dụng: DNS header, HTTP header, FTP header,…

Packet payload (Phần tải của gói tin bạn cũng có thể áp dụng các luật lên các phần dữ liệu được truyền đi của gói tin).

Một vấn đề nữa trong module phát hiện đó là việc xử lý thế nào khi một gói tin bị phát hiện bởi nhiều luật? Do các luật trong Snort cũng được đánh thứ tự ưu tiên, nên một gói tin khi bị phát hiện bởi nhiều luật khác nhau, cảnh báo được đưa ra sẽ là cảnh báo ứng với luật có mức ưu tiên lớn nhất.

Phiên bản 1.x: Việc xử lý gói tin còn hạn chế trong trường hợp các dấu hiệu trong gói tin đó phù hợp với dấu hiệu trong nhiều rule. Khi đó nếu có rule nào được áp dụng trước thì các rule còn lại sẽ bị bỏ qua mặc dù các rule có độ ưu tiên khác nhau. Như vậy sẽ nảy sinh trường hợp các rule có độ ưu tiên cao hơn bị bỏ qua.

Phiên bản 2.x: Nhược điểm trên của phiên bản 1.x được khắc phúc hoàn toàn nhờ vào cơ chế kiểm tra trên toàn bộ rule. Sau đó lấy ra rule có độ ưu tiên cao nhất để tạo thông báo.

Tốc độ của phiên bản 2.x nhanh hơn rất nhiều so với phiên bản 1.x nhờ phiên bản 2.x được biên dịch lại.

* Bộ phận ghi nhận và thông báo (Loging and alerting system)

Bộ phận ghi nhận và thông báo: khi bộ phận detection engine phát hiện ra các dấu hiệu tấn công thì nó sẽ thông báo cho bộ phận Logging and Alerting System. Các ghi nhận, thông báo có thể được lưu dưới dạng văn bản hoặc một số định dạng khác. Mặc định thì chúng được lưu tại thư mục /var/log/snort.

* Bộ phận đầu ra (Output Modules)

Bộ phận đầu ra: module này có thể thực hiện các thao tác khác nhau tùy theo việc bạn muốn lưu kết quả xuất ra như thế nào. Tùy theo việc cấu hình hệ thống mà nó có thể thực hiện các công việc như là:

Ghi log file

Ghi syslog: syslog và một chuẩn lưu trữ các file log được sử dụng rất nhiều trên các hệ thống Unix, Linux.

Ghi các cảnh báo vào cơ sở dữ liệu.

Tạo file log dưới dạng xml: ghi log file dạng xml rất thuận tiện cho việc trao đổi và chia sẻ dữ liệu.

Cấu hình lại Router, Firewall.

Gửi các cảnh báo được gói trong gói tin sử dụng giao thức SNMP. Các gói tin dạng SNMP này sẽ được gửi tới một SNMP server từ đó giúp cho việc quản lý các cảnh báo về hệ thống IDS một cách tập trung và thuận tiện hơn.

Gửi các thông điêp SMB (Server Message Block) tới các máy tính Windows. Nếu không hài lòng với các cách xuất thông tin như trên, ta có thể viết các module thể hiện thông tin riêng tùy theo mục đích sử dụng.

### Các chế độ của Snort

* Sniffer Mode: Đọc các gói tin IP và hiển thị chúng trong console ứng dụng.
* Packet Logger Mode: Ghi lại tất cả các gói IP (vào và ra) mà truy cập mạng.
* NIDS và NIPS: Ghi/bỏ các gói được coi là độc hại theo các luật do người dùng xác định. Ở đồ án này chúng ta sẽ sử dụng chế độ NIDS của Snort.

### Chế độ NIDS của snort

Trong chế độ NIDS của snort có các tham số sau

|  |  |
| --- | --- |
| -c | Xác định file cấu hình |
| -T | Kiểm thử file cấu hình |
| -N | Vô hiệu hóa ghi chép |
| -D | Chế độ nền |
| -A | Chế độ cảnh báo  full: Chế độ cảnh báo đầy đủ, cung cấp tất cả các thông tin có thể về cảnh báo. Đây cũng là chế độ mặc định.  fast: Chế độ cảnh báo nhanh cung cấp thông báo, thời gian, địa chỉ IP nguồn và đích cùng với số hiệu cổng  console: Hiển thị các cảnh báo nhanh trên màn hình console  cmg: Kiểu CMG, chi tiết header cơ bản với tải trọng ở dạng hex và dạng văn bản  none: Vô hiệu hóa cảnh báo |

Bảng 2: Tham số chế độ NIDS của Snort

### Cấu trúc luật của Snort

Hình 3: Cấu trúc luật của Snort

Table

Description automatically generatedVí dụ luật dưới đây dùng để hiển thị thông báo mỗi khi ghi nhận một gói tin ICMP

Hình 4: Ví dụ luật của Snort

Rules được chia thành 2 thành phần chính là rules header và rules options.

Phần Header chứa thông tin về hành động mà luật đó sẽ thực hiện khi phát hiện ra có xâm nhập nằm trong gói tin và nó cũng chứa các tiêu chuẩn để áp dụng luật với gói tin đó. Phần Header là bắt buộc phải có khi thiết kế luật.

Phần Option chứa một thông điệp cảnh báo và các thông tin về các phần của gói tin dùng để tạo nên cảnh báo. Phần Option chứa các tiêu chuẩn phụ thêm để đối sánh luật với gói tin. Một luật có thể phát hiện được một hay nhiều hoạt động thăm dò hay tấn công. Các luật thông minh có khả năng áp dụng cho nhiều dấu hiệu xâm nhập.Phần Option là tự chọn khi thiết kế luật.

Ở phần Header ta có:

|  |  |
| --- | --- |
| Action | Có một vài hành động cho một luật. Thường dùng nhất là những hành động sau:   * alert: Tạo một thông báo và ghi lại gói tin. * log: Ghi lại gói tin. * drop: Chặn và ghi lại gói tin. * reject: Chặn, ghi lại gói tin và hủy tiến trình làm việc của gói tin. |
| Protocol | Tham số Protocol xác định loại giao thức được lọc cho các luật. |
| Direction | Cho biết luồng lưu lượng sẽ được lọc bởi Snort. Phía bên trái của luật hiển thị nguồn và phía bên phải hiển thị đích.   * **->** Luồng nguồn tới đích. * **<>** Luồng 2 chiều |

Bảng 3: Tham số Rule Header

Có 3 loại Rule Option chính ở snort:

* **General Rule Options -**Tùy chọn cơ bản của luật.

|  |  |
| --- | --- |
| Msg | Trường thông báo là dấu nhắc cơ bản và mã định danh nhanh của luật. Khi luật được kích hoạt, thông báo đã gửi sẽ xuất hiện trong console hoặc log. Thông thường, phần tin nhắn là một dòng tóm tắt sự kiện. |
| Sid | Snort rule IDs (SID) đi kèm với phạm vi được xác định trước và mỗi luật phải có SID ở định dạng phù hợp. Có ba phạm vi khác nhau cho SID được hiển thị bên dưới.   * **<100:** Luật riêng. * **100-999,999:** Luật cài đặt sẵn. * **>=1,000,000:** Luật do người dùng tạo.   Tóm lại, các luật chúng ta sẽ tạo nên có sid lớn hơn 100.000.000. Một điểm quan trọng khác là; Các SID không được trùng nhau và mỗi id phải là duy nhất. |
| Reference | Mỗi luật có thể có thêm thông tin hoặc tài liệu tham khảo để giải thích mục đích của quy tắc hoặc mẫu mối đe dọa. Đó có thể là id Lỗ hổng bảo mật và Tiếp xúc Phổ biến (CVE) hoặc thông tin bên ngoài. Có tài liệu tham khảo cho các luật sẽ luôn giúp các nhà phân tích trong quá trình điều tra cảnh báo và sự cố. |
| Rev | Các luật của Snort có thể được sửa đổi và cập nhật cho các vấn đề về hiệu suất và hiệu quả. Tùy chọn Rev giúp các nhà phân tích có thông tin sửa đổi của từng quy tắc. Do đó, sẽ dễ hiểu các cải tiến luật. Mỗi luật có số vòng quay duy nhất và không có tính năng tự động sao lưu trên lịch sử luật. Các nhà phân tích nên tự giữ lịch sử luật. Tùy chọn Rev chỉ là một chỉ số về số lần luật đó đã sửa đổi. |

Bảng 4: Tham số General Option

* **Payload Rule Options -** Cáctùy chọn giúp điều tra dữ liệu tải trọng. Các tùy chọn này rất hữu ích để phát hiện các mô hình tải trọng cụ thể.

|  |  |
| --- | --- |
| Content | Dữ liệu tải trọng. Nó khớp dữ liệu tải trọng cụ thể theo ASCII, HEX hoặc cả hai. Có thể sử dụng tùy chọn này nhiều lần trong một luật. Tuy nhiên, bạn càng tạo ra nhiều tính năng khớp mẫu cụ thể, thì càng mất nhiều thời gian để điều tra một gói. |
| Nocase | Vô hiệu hóa độ nhạy trường hợp. Được sử dụng để tăng cường tìm kiếm nội dung. |
| Fast\_pattern | Ưu tiên tìm kiếm nội dung để tăng tốc hoạt động tìm kiếm tải trọng. Theo mặc định, Snort sử dụng nội dung lớn nhất và đánh giá nội dung đó theo các luật. Tùy chọn “fast\_pattern” giúp bạn chọn gói tin ban đầu phù hợp với giá trị cụ thể để điều tra thêm. Tùy chọn này luôn hoạt động không phân biệt chữ hoa chữ thường và có thể được sử dụng một lần cho mỗi luật. Lưu ý rằng tùy chọn này là bắt buộc khi sử dụng nhiều tùy chọn "content". |

Bảng 5: Tham số Payload Option

* **Non-Payload Rule Options -** Các tùy chọn tập trung vào các dữ liệu không tải trọng. Các tùy chọn này sẽ giúp tạo các mô hình cụ thể xác định các vấn đề về mạng.

|  |  |
| --- | --- |
| ID | Lọc trường id IP |
| Flags | Lọc các cờ TCP.   * F - FIN * S - SYN * R - RST * P - PSH * A - ACK * U - URG |
| Dsize | Lọc kích thước tải trọng gói.   * dsize:min<>max; * dsize:>100 * dsize:<100 |
| Sameip | Lọc địa chỉ IP nguồn và đích có sự trùng lặp. |

Bảng 6: Tham số Non-Payload Option

## Giới thiệu về IPtables

Iptables là một tường lửa ứng dụng lọc gói dữ liệu rất mạnh, miễn phí và có sẵn trên Linux. Netfilter/Iptables gồm 2 phần là Netfilter ở trong nhân Linux và 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.

Diagram

Description automatically generated

Hình 5: Sơ đồ Iptables/Netfilter

### Cơ chế xử lý package trong IPtables

Tất cả gói đi qua một chuỗi được xây dựng trong các bảng (hàng đợi) để xử lý.

Iptables được chia làm 3 bảng:

* *NAT table*

NAT table: thực thi chức năng NAT (Network Address Translation), bao gồm ba loại built-in chains sau đây:

+ Pre-routing chain: NAT từ ngoài vào trong nội bộ. Quá trình NAT sẽ thực hiện trước khi thực thi cơ chế routing. Điều này thuận lợi cho việc đổi địa chỉ đích để địa chỉ tương thích với bảng định tuyến của firewall, khi cấu hình ta có thể dùng khóa DNAT để mô tả kỹ thuật này.

+ Post-routing chain: NAT từ trong ra ngoài. Quá trình NAT sẽ thực hiện sau khi thực hiện cơ chế định tuyến. Quá trình này nhằm thay đổi địa chỉ nguồn của gói tin. Kỹ thuật này được gọi là NAT one-to-one hoặc many-to-one, được gọi là Source NAT hay SNAT.

+ OUTPUT: Trong loại này firewall thực hiện quá trình NAT.

Diagram

Description automatically generated

Hình 6: Quá trình xử lý Nat Table

* *Filter table*

Filter table: chịu trách nhiệm thiết lập bộ lọc packet (packet filtering), có ba loại built-in chains:

- Forward chain: Cho phép packet nguồn chuyển qua firewall.

- Input chain: Cho phép những gói tin đi vào từ firewall.

- Output chain: Cho phép những gói tin đi ra từ firewall.

Diagram

Description automatically generated

Hình 7: Quá trình xử lý Filter Table

* Mangle table

Mangle table: Chịu trách nhiệm biến đổi quality of service bits trong TCP header.

Diagram

Description automatically generated

Hình 8: Quá trình xử lý Mangle Table

### Một số khái niệm trong Iptables.

**- Chain:** là một chuỗi bao gồm nhiều Rule, mặc định các bảng có sẵn các chain như INPUT, OUTPUT …

- **Rule:** là các luật đưa ra các điều kiện so sánh, tập hợp nhiều rule tạo thành Chain.

- **Jump:** là cơ chế chuyển một packet đến một target nào đó để xử lý thêm một số thao tác khác.

- **Target:** là quá trình xử lý khi các gói tin đủ điều kiện. Các target được xây dựng sẵn trong iptables như:

+ ACCEPT: chấp nhận chuyển gói tin đi qua.

+ DROP: chặn những gói tin.

+ REJECT: chặn những gói tin và báo lại cho người gửi.

+ DNAT: thay đổi địa chỉ đích của gói tin.

+ SNAT: thay đổi địa chỉ nguồn của gói tin.

+ MASQUERADING: là một kỹ thuật đặc biệt của SNAT.

### Cài đặt Iptables

* Các câu lệnh trợ giúp

#iptables -h

#iptables -m match -h

#iptables -j TARGET -h

#man iptables

1.4.4.2 Các tùy chọn

|  |  |
| --- | --- |
| Tùy chọn | Giải thích |
| -A (--append) | Ghi thêm một rule vào chain |
| -D (--delete) | Xóa rule |
| -E (--rename-chain) | Đổi tên chain |
| -F (--flush) | Xóa tất cả các rule trong một chain |
| -I (--insert) | Chèn thêm rule vào chain tại vị trí được xác định |
| -L (--list) | Xem các rule trong chain |
| -N (--new-chain) | Tạo một chain mới |
| -P (--policy) | Tạo rule mặc định cho chain |
| -R (--replace) | Thay thế một rule bằng một rule khác tại vị trí xác định |
| -V (--version) | Xem phiên bản của Iptables |
| -X (--delete-chain) | Xóa chain |
| -Z (--zero) | Xóa bộ đếm trong chain |
| -t | Chỉ định bảng cho iptables bao gồm: filter, nat, mangle tables |
| -j | Nhảy đến một target |
| -p | Mô tả các giao thức bao gồm: icmp, tcp, udp và all |
| -s | Chỉ định địa chỉ nguồn |
| -d | Chỉ định địa chỉ đích |
| -i | Chỉ định địa chỉ interface cho các gói tin đi ra |
| -m multiport | Chỉ định nhiều dãy cổng |
| -m state –state | Kiểm tra trạng thái:  ESTABLISHED: đã thiết lập connection  NEW: bắt đầu thiết lập connection  RELATED: thiết lập connection thứ 2 |

Bảng 7: Các tùy chọn iptables

* Các câu lệnh phục hồi

-iptables-restore: lệnh phục hồi iptables có hai tùy chọn:

|  |  |
| --- | --- |
| Tùy chọn | Giải thích |
| -c (--counters) | Phục hồi bộ đếm gói và byte cho các rule |
| -n (--noflush) | Tắt chức năng xóa (-flush) trong tables trước khi phục hồi |

Bảng 8: Câu lệnh hồi phục và các tùy chọn

-iptables-save: lệnh ghi lại Iptables có hai tùy chọn:

|  |  |
| --- | --- |
| Tùy chọn | Giải thích |
| -c (--counters) | Hiện bộ đếm gói và byte cho các rule |
| -t (--table) | Chỉ hiện bảng được chỉ định |

Bảng 9: Câu lệnh hồi phục và các tùy chọn

## Các giao thức liên quan và các cuộc tấn công tương ứng:

### Giao thức ICMP

Internet Control Message Protocol (ICMP) là một giao thức ở tầng Network được các thiết bị mạng sử dụng để chẩn đoán các sự cố truyền thông mạng. ICMP chủ yếu được sử dụng để xác định liệu dữ liệu có đến đích dự kiến thành công hay không và thời gian thực hiện việc đó. Thông thường, giao thức ICMP được sử dụng trên các thiết bị mạng, chẳng hạn như bộ định tuyến. ICMP rất quan trọng, giao thức này được dùng để báo cáo lỗi và kiểm tra, nhưng nó cũng có thể được sử dụng trong các cuộc tấn công mạng.

Ví dụ như cuộc tấn công Ping of death, thông thường các gói tin ping rất nhỏ, nhưng gói tin IP4 lại có dung lượng có thể chứa rất lớn, có thể vượt quá dung lượng đối đa cho phép của 1 gói tin, việc gửi đến các ping có dung lượng lớn khiến cho hệ thống không thể hoạt động bình thường thậm chí là ngưng hoạt động vì bị chiếm tài nguyên

### Giao thức TCP và tấn công Syn flood

Thông thường, TCP kết nối bằng cách bắt tay 3 bước để kết nối

* Đầu tiên Client gửi một gói tin SYN tới Server để khởi tạo kết nối
* Server sẽ trả lời gói tin SYN đó bằng một gói tin SYN/ACK, nhằm xác nhận kết nối
* Cuối cùng, Client gửi một gói tin ACK để xác nhận việc đã nhận gói tin từ Server. Sau khi kết thúc quá trình này, kết nối TCP sẽ được mở và bắt đầu trao đổi dữ liệu

Nhằm lợi dụng việc sau khi nhận gói tin SYN, Server phải trả lời bằng gói tin SYN/ACK và đợi chờ việc xác nhận từ Client. Việc tấn công diễn ra như thế này

* Người tấn công sẽ gửi một lượng lớn các gói tin SYN cho Server
* Server phải trả lời hết tất cả các gói tin và dùng mỗi cổng riêng biệt để đợi chờ trả lời từ Client
* Vì chờ đợi gói tin ACK từ Client (thứ sẽ không bao giờ đến), việc sử dụng tài nguyên cổng cứ thế diễn ra cho đến khi các cổng hiện có được sử dụng hết, lúc đó hệ thống sẽ không thể làm việc bình thường được.

# CHƯƠNG 2. PHÂN TÍCH THIẾT KẾ HỆ THỐNG

## Sơ đồ User-Case

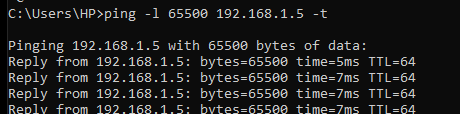
Hệ thống chỉ có 1 người dùng là Người giám sát, có các tương tác với hệ thống như: Giám sát các lưu lượng mạng, theo dõi các cảnh báo phát hiện thâm nhập, chỉnh sửa luật của Snort

Diagram

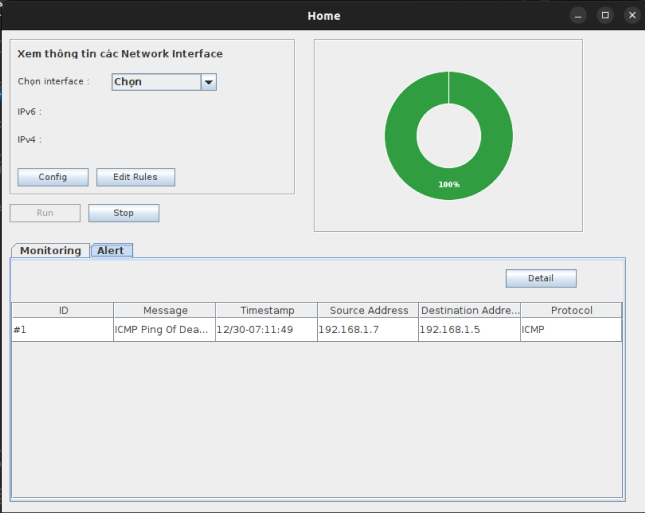
Description automatically generated

Hình 9: Sơ đồ User-Case

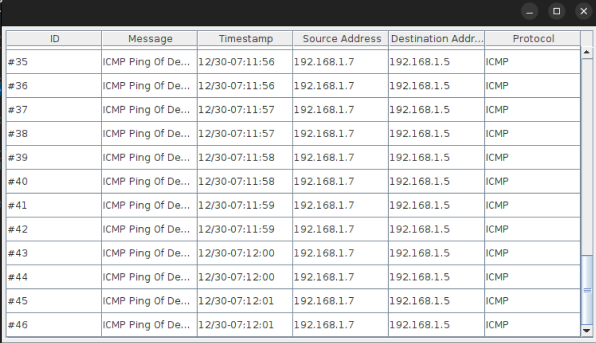
# CHƯƠNG 3. TRIỂN KHAI VÀ ĐÁNH GIÁ KẾT QUẢ

1. 

## Tấn công Ping Of Death

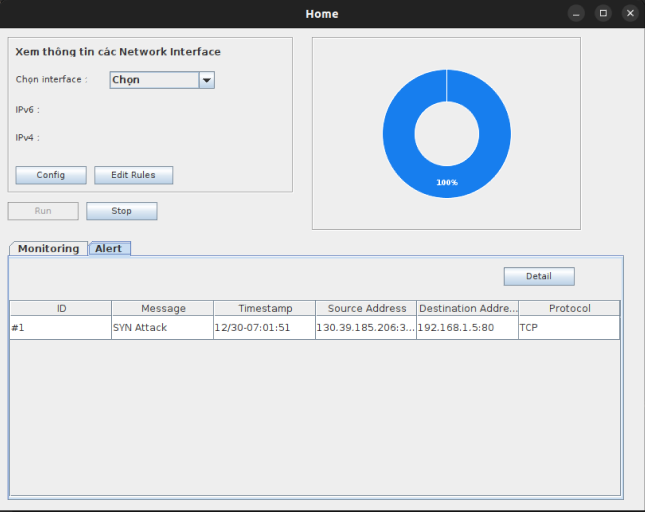
Hình 10: Người tấn công gửi các ping có dung lượng cao

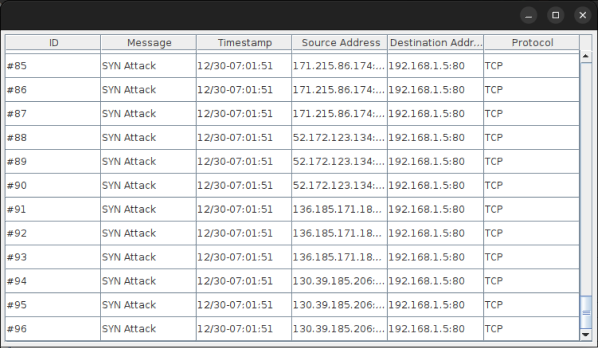
Hình 11: Hệ thống phát hiện Ping Of Death

Hình 12: Chi tiết cảnh báo Ping Of Dead

## Tấn công Syn flood

Hình 13: Người tấn công gửi ngập cờ SYN

Hình 14: Hệ thống cảnh báo SYN Attack

Hình 15: Chi tiết cảnh báo SYN Attack

# KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

**1. Kết luận**

Thông qua nghiên cứu và triển khai hệ thống IDS SNORT, chúng em đã rút ra được các kết luận về ưu và nhược điểm của kết quả mà bọn em đã hoàn thành:

Ưu điểm:

* Thành thục sử dụng Linux, Snort và iptables
* Kết quả cho ra chương trình Snort có giao diện dễ tiếp cận, thân thiện với người dùng.

Nhược điểm

* Các cuộc tấn công tìm hiểu được quá ít, chưa đầy đủ bảo mật cho một hệ thống có thể hoạt động tốt

**2. Hướng phát triển**

Thông qua bài PBL lần này, chúng em đã có một vài hướng phát triển cho đồ án trở nên hoàn thiện hơn:

* Trao dồi thêm kiến thức về các giao thức khác
* Tìm hiểu thêm về các hình thức tấn công khác, về công cụ tấn công cũng như biện pháp chống lại nó.

TÀI LIỆU THAM KHẢO

[1] Hà Quang, Giáo trình Hệ điều hành Unix-Linux

[2] tryhackme.com, Snort, https://tryhackme.com/room/snort, 2/12/2022

[3] Nguyễn Hưng, Iptables là gì và cách sử dụng iptables, https://vietnix.vn/iptables-la-gi/, 2/12/2022