

ĐẠI HỌC BÁCH KHOA HÀ NỘI
TRƯỜNG CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG



BÀI TẬP LỚN MÔN HỌC
NHẬP MÔN AN TOÀN THÔNG TIN
CHỦ ĐỀ: PHÁT HIỆN XÂM NHẬP MẠNG

Nhóm sinh viên thực hiện:

- | | |
|-----------------------------|-----------------------|
| 1. Mai Văn Đăng | MSSV: 20225699 |
| 2. Trương Ngọc Mai | MSSV: 20225879 |
| 3. Nguyễn Thanh Tân | MSSV: 20225923 |
| 4. Nguyễn Khắc Quang | MSSV: 20225760 |

Giảng viên hướng dẫn: PSG.TS. Nguyễn Linh Giang

Hà Nội, tháng 6 năm 2024

MỤC LỤC

I. Thực trạng	3
II. Cơ sở lý thuyết	4
1. An ninh mạng và các mối đe dọa xâm nhập mạng	4
2. Phát hiện xâm nhập mạng	4
3. Các thành phần của hệ thống phát hiện xâm nhập mạng:.....	6
4. Các kỹ thuật và công nghệ hỗ trợ phát hiện xâm nhập mạng.....	7
III. Phần mềm SNORT trong phát hiện xâm nhập mạng.....	8
1. Giới thiệu về snort.....	8
2. Kiến trúc của Snort.....	8
2.1 Module giải mã gói tin.....	9
2.2 Module tiền xử lý.....	10
2.3 Module phát hiện	11
2.4 Môđun log và cảnh báo	12
2.5 Môđun kết xuất thông tin.....	12
3. Cấu trúc luật của Snort.....	12
3.1 Cấu trúc của phần Header.....	13
3.2 Phần Option	18
4. Kiểm tra và bổ sung các quy tắc trong Snort	24
4.1. Phát hiện dựa trên quy tắc.....	26
4.2.Các quy tắc mới trong Snort 3:	26
4.3. Tùy chọn các quy tắc.....	30
5. Chạy demo phần mềm SNORT	33
5.1. Cài đặt hệ thống	33
6. Đánh giá phần mềm SNORT	39
IV. Ứng dụng.....	40
V. Những thách thức và cơ hội	40
VI. Kết luận và hướng phát triển.....	42
VII. Tài liệu tham khảo.....	43

I. Thực trạng

Chuyển đổi số đang là một trong những xu hướng quan trọng trên toàn cầu, với tác động mạnh mẽ đến nhiều lĩnh vực từ kinh tế đến xã hội. Sự phát triển của những công nghệ mạnh mẽ như AI, điện toán đám mây (Cloud), Big Data, ... đã góp phần tạo nên 1 xã hội phát triển khi mà gần như mọi tác vụ đều có thể giải quyết ở trên không gian mạng (mua bán trên sàn thương mại điện tử, học tập, làm việc, ...) thậm chí chỉ cần đến 1 nút chạm trên màn hình điện thoại.

Chính vì vậy việc đảm bảo an toàn thông tin là yếu tố then chốt, bởi dữ liệu đã trở thành 1 tài sản rất quý giá của mỗi cá nhân hay tổ chức tập đoàn, trong bối cảnh nguy cơ mất an toàn thông tin luôn thường trực bởi kẻ gian, chúng đánh cắp thông tin người dùng, xâm nhập trái phép vào những tổ chức, thậm chí để lộ bí mật quốc gia gây thất thoát rất lớn về tinh thần và vật chất.

Do đó vai trò của an ninh mạng là rất cấp thiết để góp phần đảm bảo không gian số lành mạnh, cá nhân người dùng hay tổ chức tiếp cận thoải mái hơn, phòng tránh và ngăn chặn những nguy cơ tiềm ẩn có thể ảnh hưởng xấu đến toàn xã hội, bởi hậu quả chúng đem lại rất khôn lường.

Các cuộc tấn công trong quá khứ:

- **Tổng kết năm 2023** : báo cáo tổng kết an ninh mạng Việt Nam năm 2023 do Công ty an ninh mạng NCS công bố ngày 12-12, 13.900 vụ tấn công mạng vào các tổ chức tại Việt Nam, trung bình mỗi tháng xảy ra 1.160 vụ. 3 điểm yếu bị tấn công nhiều nhất tại Việt Nam năm 2023 lần lượt là con người (32,6%), lỗ hổng nền tảng (27,4%) và lỗ hổng website (25,3%).
- **Tấn công Twitter 2020** : Vụ tấn công xảy ra ngày 15-7-2020, khi tin tặc xâm chiếm các tài khoản của nhiều cá nhân và tổ chức danh tiếng để đăng tải những bài viết nhằm lừa đảo người dùng gửi cho chúng tiền ảo bitcoin. Khoảng 130 tài khoản là mục tiêu của tin tặc trong cuộc tấn công, trong đó có rất nhiều tài khoản của các nhân vật nổi tiếng và có tầm ảnh hưởng. Nạn nhân của vụ tấn công này là những người cả tin, đã gửi đi 1.000USD vào ví bitcoin của các tin tặc với hy vọng nhận về số tiền gấp đôi theo lời hứa hẹn. Ước tính tin tặc đã chiếm đoạt hơn 120.000USD bằng hình thức này.

Phát hiện xâm nhập mạng được coi là một trong những giải pháp hiệu quả nhất để ngăn chặn các cuộc tấn công mạng. Tuy nhiên, đây là một lĩnh vực rất chuyên sâu và cần có kiến thức chuyên môn về an ninh mạng và các phương pháp phát hiện xâm nhập mạng. Do đó, việc tìm hiểu về phát hiện xâm nhập mạng sẽ giúp chúng ta có được kiến thức nền tảng và hiểu rõ hơn về cách thức

hoạt động của các công cụ phát hiện xâm nhập mạng và cách áp dụng chúng để bảo vệ hệ thống mạng của mình.

II. Cơ sở lý thuyết

1. An ninh mạng và các mối đe dọa xâm nhập mạng

An ninh mạng là một lĩnh vực quan trọng của bảo mật thông tin, đảm bảo rằng các dữ liệu, thiết bị và hệ thống mạng được bảo vệ khỏi các mối đe dọa và tấn công từ bên ngoài. An ninh mạng bao gồm các biện pháp bảo vệ chống lại các cuộc tấn công, giám sát các hoạt động mạng để phát hiện các mối đe dọa và cung cấp các biện pháp phòng ngừa và phản ứng khi các cuộc tấn công xảy ra. Các mối đe dọa xâm nhập mạng ngày càng trở nên phức tạp và đa dạng, bao gồm:

- **Malware:** Phần mềm độc hại được thiết kế để gây hại cho hệ thống mạng. Các loại malware phổ biến bao gồm virus, worm, trojan, ransomware và spyware.
- **Các cuộc tấn công từ mạng bên ngoài:** Các cuộc tấn công từ mạng bên ngoài bao gồm các cuộc tấn công từ hackers, botnets và các tổ chức tội phạm mạng.
- **Các cuộc tấn công từ bên trong:** Các cuộc tấn công từ bên trong bao gồm các nhân viên nội bộ hoặc các cá nhân được cho phép truy cập vào hệ thống mạng bị phá hoại.
- **Social engineering:** Kỹ thuật lừa đảo nhằm lấy thông tin cá nhân hoặc thông tin quan trọng của các tổ chức hoặc cá nhân.
- **Các lỗ hổng bảo mật:** Các lỗ hổng bảo mật trong phần mềm hoặc các thiết bị mạng có thể bị khai thác để tấn công hệ thống mạng.

Để đối phó với các mối đe dọa xâm nhập mạng này, các tổ chức cần triển khai các biện pháp an ninh mạng phù hợp, bao gồm cả các giải pháp phát hiện xâm nhập mạng để giúp phát hiện và ngăn chặn các cuộc tấn công trước khi chúng gây ra thiệt hại cho hệ thống mạng.

2. Phát hiện xâm nhập mạng

Phát hiện xâm nhập mạng trong an ninh mạng để giám sát và phát hiện các hành vi xâm nhập vào hệ thống mạng, đặc biệt là những hành động không hợp pháp hoặc bất thường.

Hệ thống phát hiện xâm nhập mạng (IDS):

- IDS là một hệ thống có nhiệm vụ giám sát các luồng dữ liệu (lưu lượng) đang lưu thông trên mạng.

- IDS có khả năng phát hiện những hành động khả nghi, những xâm nhập trái phép cũng như khai thác bất hợp pháp nguồn tài nguyên của hệ thống.
- IDS có thể phân biệt được những cuộc tấn công xuất phát từ bên ngoài hay từ chính bên trong hệ thống.
- Hệ thống IDS có khả năng chặn các cuộc tấn công thì nó được gọi là IPS (Intrusion Prevention System).

Mục đích của phát hiện xâm nhập mạng là để bảo vệ hệ thống mạng khỏi các cuộc tấn công và các mối đe dọa bảo mật khác. Nó giúp ngăn chặn các cuộc tấn công trên hệ thống mạng, giảm thiểu thiệt hại và nguy cơ bị mất dữ liệu.

Các phương pháp:

1.1.IDS phát hiện dựa trên chữ ký (Signature-Based IDS):

- Đặc điểm: sử dụng các mẫu tấn công đã biết được gọi là “chữ ký” và khớp chúng với nội dung gói mạng để phát hiện các cuộc tấn công.
- Ưu điểm: Dễ triển khai, hiệu quả với các tấn công đã biết.
- Nhược điểm: Không phát hiện được các tấn công mới, dễ bị đánh lừa bởi các kỹ thuật né tránh.

1.2.IDS phát hiện dựa trên hành vi (Behavior-Based IDS):

- Đặc điểm: Phân tích hành vi của người dùng và quy trình để phát hiện các hành vi bất thường.
- Ưu điểm: Phát hiện được các tấn công mới, linh hoạt và thích nghi.
- Nhược điểm: Tốn nhiều tài nguyên, có thể gây ra các cảnh báo giả.

1.3.IDS phát hiện dựa trên dữ liệu bất thường (Anomaly-Based IDS):

- Đặc điểm: sử dụng thuật toán học máy để chuẩn bị mô hình IDS học hỏi từ mô hình hoạt động thường xuyên của lưu lượng mạng. Nếu lưu lượng truy cập lệch so với hành vi bình thường thì cảnh báo sẽ được tạo.
- Ưu điểm: đặc biệt hiệu quả trong việc xác định các cuộc tấn công zero-day và các mối đe dọa dai dẳng nâng cao có thể không bị phát hiện bởi các biện pháp bảo mật khác.
- Nhược điểm: Độ chính xác không cao, khó xác định nguyên nhân gây ra các hoạt động bất thường.

1.4.IDS phát hiện dựa trên kết hợp các phương pháp trên (Hybrid IDS):

- Đặc điểm: Kết hợp các phương pháp phát hiện để tăng cường hiệu suất và độ chính xác.
- Ưu điểm: Tăng cường hiệu suất phát hiện, giảm số lượng cảnh báo giả.
- Nhược điểm: Tăng độ phức tạp, yêu cầu tài nguyên cao.

3. Các thành phần của hệ thống phát hiện xâm nhập mạng:

Các thành phần chính của hệ thống phát hiện xâm nhập mạng (IDS - Intrusion Detection System) bao gồm:

1. Cảm biến (Sensors): Là thành phần thu thập dữ liệu từ mạng hoặc hệ thống để phát hiện các hoạt động không mong muốn hoặc bất thường. Cảm biến có thể là các thiết bị vật lý hoặc phần mềm được triển khai trên các nút mạng.
2. Bộ phân tích (Analysis Engine): Là trung tâm xử lý dữ liệu thu thập từ các cảm biến. Nó sẽ phân tích dữ liệu này để xác định các mẫu tấn công và hành vi không mong muốn.
3. Cơ sở dữ liệu chữ ký (Signature Database): Là nơi lưu trữ các chữ ký của các mẫu tấn công đã biết trước. Các chữ ký này được sử dụng để so sánh với dữ liệu thu thập từ mạng.
4. Mô hình hành vi (Behavior Model): Là một phần của bộ phân tích, nó xây dựng và duy trì các mô hình về hành vi bình thường của người dùng và hệ thống. Dựa trên các mô hình này, nó có thể phát hiện các hành vi không bình thường.
5. Giao diện người dùng (User Interface): Là giao diện dùng để hiển thị thông tin và cảnh báo cho người quản trị hệ thống. Giao diện này cung cấp các công cụ để quản lý và phản ứng đối với các cảnh báo.
6. Hệ thống cảnh báo (Alerting System): Là thành phần thông báo cho người quản trị về các sự kiện hoặc hành vi không mong muốn được phát hiện bởi hệ thống. Cảnh báo có thể được gửi qua email, tin nhắn văn bản, hoặc thông qua các phương tiện khác.
7. Cơ sở dữ liệu (Database): Là nơi lưu trữ dữ liệu thu thập từ mạng và các sự kiện được ghi lại bởi hệ thống phát hiện xâm nhập. Cơ sở dữ liệu này có thể được sử dụng cho mục đích phân tích và báo cáo sau này.

Các thành phần này cùng hoạt động cùng nhau để giúp hệ thống phát hiện xâm nhập mạng nhận biết và đáp ứng đối với các mối đe dọa và tấn công từ bên ngoài.

4. Các kỹ thuật và công nghệ hỗ trợ phát hiện xâm nhập mạng

Có nhiều kỹ thuật và công nghệ hỗ trợ phát hiện xâm nhập mạng (IDS) để giúp nhận biết và phản ứng đối với các hoạt động không mong muốn và tấn công từ bên ngoài. Dưới đây là một số trong số chúng:

1. Mã hóa dữ liệu (Encryption): Sử dụng mã hóa để bảo vệ dữ liệu khi truyền qua mạng, giúp ngăn chặn các cuộc tấn công giữa chừng (Man-in-the-Middle) và lấy thông tin trực tiếp từ luồng dữ liệu.
2. Firewalls (Tường lửa): Tường lửa mạng giúp kiểm soát lưu lượng mạng vào và ra khỏi mạng, ngăn chặn các cuộc tấn công từ bên ngoài và hạn chế truy cập không ủy quyền.
3. Proxy Servers: Máy chủ proxy có thể giúp lọc và kiểm soát lưu lượng mạng, cung cấp một lớp bảo vệ bổ sung trước khi lưu lượng mạng tiếp cận các hệ thống nội bộ.
4. Network Intrusion Detection Systems (NIDS): Hệ thống phát hiện xâm nhập mạng cung cấp giám sát liên tục cho mạng và phát hiện các hoạt động không mong muốn hoặc bất thường.
5. Host Intrusion Detection Systems (HIDS): Tương tự như NIDS, HIDS theo dõi và phát hiện các hoạt động không mong muốn trên các hệ thống cụ thể.
6. Thiết bị phát hiện xâm nhập (Intrusion Detection Appliances): Thiết bị phát hiện xâm nhập cung cấp một giải pháp tích hợp cho việc phát hiện xâm nhập mạng, thường được triển khai tại các điểm trung tâm trong mạng.
7. Công nghệ Mô hình Học máy (Machine Learning Models): Sử dụng các mô hình học máy để phát hiện các mẫu và hành vi không mong muốn dựa trên dữ liệu lịch sử và các thông tin thu thập từ mạng.
8. Phân tích Giao thức (Protocol Analysis): Kiểm tra và phân tích các gói tin mạng để phát hiện các mẫu tấn công hoặc hành vi không bình thường dựa trên giao thức mạng.
9. Cơ sở dữ liệu chữ ký (Signature Databases): Là cơ sở dữ liệu chứa các chữ ký của các mẫu tấn công đã biết, được sử dụng để so sánh và phát hiện các mẫu tương tự trong luồng dữ liệu mạng.
10. Công nghệ Xử lý sự kiện thời gian thực (Real-Time Event Processing): Theo dõi và xử lý các sự kiện mạng ngay lập tức, giúp phát hiện các hành vi đáng ngờ ngay khi chúng xảy ra.

Kết hợp các kỹ thuật và công nghệ này có thể tạo ra một hệ thống phát hiện xâm nhập mạng mạnh mẽ và hiệu quả trong việc bảo vệ hệ thống và dữ liệu khỏi các mối đe dọa mạng.

III. Phần mềm SNORT trong phát hiện xâm nhập mạng

1. Giới thiệu về snort

Snort là một NIDS được Martin Roesch phát triển dưới mô hình mã nguồn mở. Tuy Snort miễn phí nhưng nó lại có rất nhiều tính năng tuyệt vời mà không phải sản phẩm thương mại nào cũng có thể có được. Với kiến trúc thiết kế theo kiểu module, người dùng có thể tự tăng cường tính năng cho hệ thống Snort của mình bằng việc cài đặt hay viết thêm mới các module. Cơ sở dữ liệu luật của Snort đã lên tới 2930 luật và được cập nhật thường xuyên bởi một cộng đồng người sử dụng.

Snort có thể chạy trên nhiều hệ thống nền như Windows, Linux, OpenBSD, FreeBSD, NetBSD, Solaris, HP-UX, AIX, IRIX, MacOS.

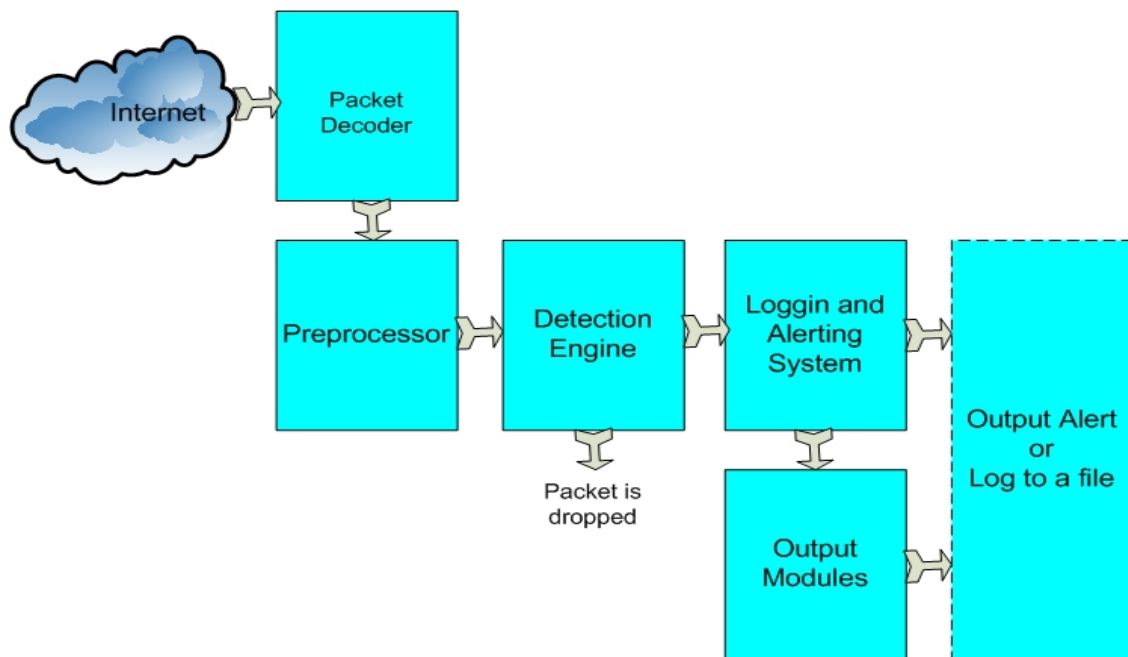
Bên cạnh việc có thể hoạt động như một ứng dụng thu bắt gói tin thông thường, Snort còn có thể được cấu hình để chạy như một NIDS. Snort hỗ trợ khả năng hoạt động trên các giao thức sau: Ethernet, 802.11, Token Ring, FDDI, Cisco HDLC, SLIP, PPP, và PF của OpenBSD.

2. Kiến trúc của Snort

❖ *Snort có 5 thành phần chính như sau:*

- Bộ giải mã gói tin - Packet Decoder
- Các bộ tiền xử lý - PreProcessers
- Máy phát hiện - Detection Engine
- Hệ thống cảnh báo và ghi dấu - Logging and Alerting System
- Môđun xuất - Output Modules

Sơ đồ sau biểu diễn quan hệ giữa các thành phần của Snort. Tại đó các gói dữ liệu giao tiếp từ mạng Internet vào trong hệ thống được đi qua Packet decoder. Tại mỗi thành phần các gói tin được xử lý rồi truyền kết quả cho thành phần kế tiếp trong hệ thống. Output modul sẽ loại bỏ các gói tin, ghi log hay sinh ra cảnh báo.

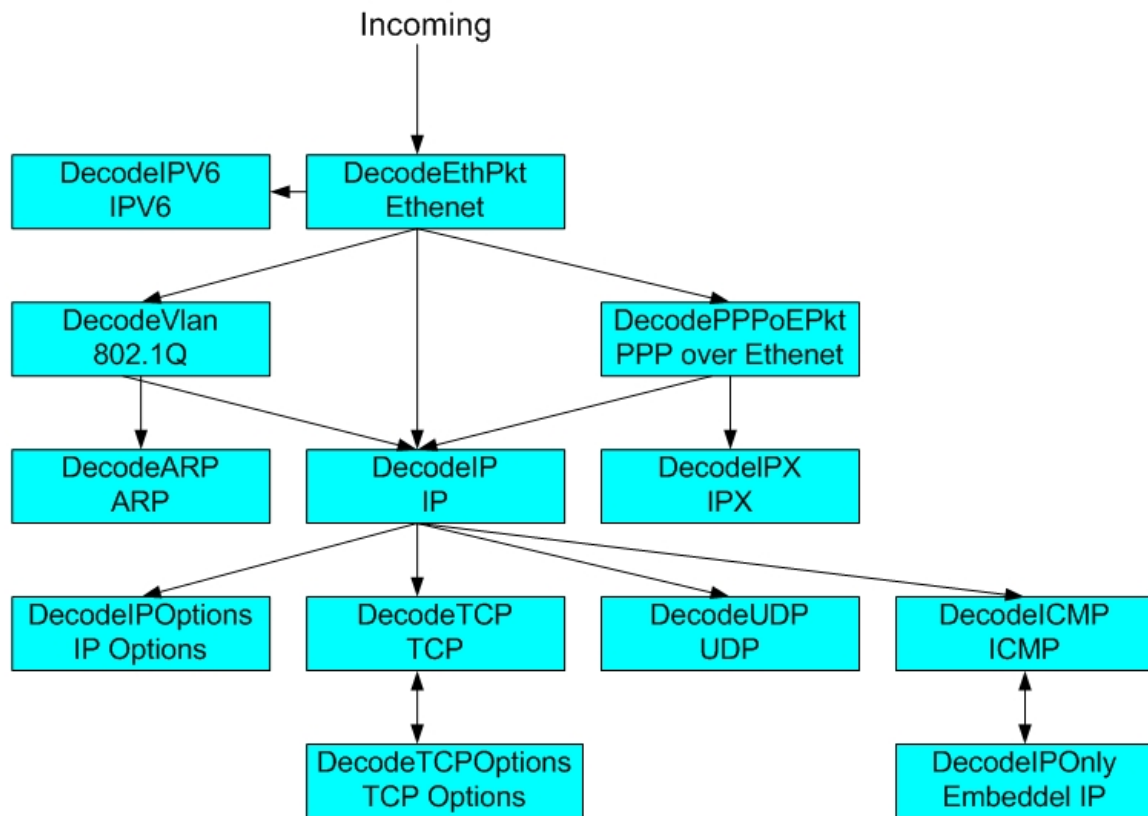


❖ *Snort có 3 chế độ hoạt động như sau:*

- Chế độ Sniffer: Trong chế độ này, Snort thụ động nắm bắt lưu lượng mạng và ghi lại để phân tích. Mặc dù nó không chủ động thực hiện bất kỳ hành động nào, nhưng nó cung cấp thông tin chi tiết có giá trị về các vi phạm bảo mật tiềm ẩn.
- Chế độ ghi nhật ký gói: Tại đây, Snort chụp và ghi nhật ký các gói, nhưng cũng có khả năng thả hoặc sửa đổi các gói dựa trên các quy tắc được xác định. Chế độ này cho phép các tổ chức chủ động ứng phó với các mối đe dọa được phát hiện.
- Chế độ Hệ thống phát hiện và ngăn chặn xâm nhập mạng (NIDPS): Chế độ này kết hợp các chức năng của chế độ Sniffer và Packet Logger. Snort phân tích lưu lượng mạng, chủ động chặn các gói tin độc hại và tạo cảnh báo để điều tra thêm.

2.1 Module giải mã gói tin

Snort chỉ sử dụng thư viện pcap để bắt mọi gói tin trên mạng lưu thông qua hệ thống.



Một gói tin sau khi được giải mã sẽ đưa tiếp vào module tiền xử lý.

2.2 Module tiền xử lý

Module này rất quan trọng đối với bất kỳ hệ thống nào để có thể chuẩn bị gói dữ liệu đưa vào cho Module phát hiện phân tích.

Gồm có 3 nhiệm vụ chính:

- Kết hợp lại các gói tin: Khi một dữ liệu lớn được gửi đi, thông tin sẽ không đóng gói toàn bộ vào một gói tin mà thực hiện phân mảnh, chia thành nhiều gói tin rồi mới gửi đi. Khi Snort nhận được các gói tin này, nó phải thực hiện kết nối lại để có gói tin ban đầu. Module tiền xử lý giúp Snort có thể hiểu được các phiên làm việc khác nhau.
- Giải mã và chuẩn hóa giao thức (decode/normalize): công việc phát hiện xâm nhập dựa trên dấu hiệu nhận dạng nhiều khi thất bại khi kiểm tra các giao thức có dữ liệu có thể được biểu diễn dưới nhiều dạng khác nhau. Ví dụ: một Web server có thể nhận nhiều dạng URL. Nếu Snort chỉ thực hiện đơn thuần việc so sánh dữ liệu với dấu hiệu nhận dạng sẽ xảy ra tình trạng bỏ sót hành vi xâm nhập. Do vậy, 1 số Module tiền xử lý của Snort phải có nhiệm vụ giải mã và chỉnh sửa, sắp xếp lại các thông tin đầu vào.

- Phát hiện các xâm nhập bất thường (nonrule/anormal): các plugin dạng này thường để xử lý với các xâm nhập không thể hoặc rất khó phát hiện bằng các luật thông thường. Phiên bản hiện tại của Snort có đi kèm 2 plugin giúp phát hiện xâm nhập bất thường đó là portscan và bo (backoffice). Portscan dùng để đưa ra cảnh báo khi kẻ tấn công thực hiện quét cổng để tìm lỗ hổng. Bo dùng để đưa ra cảnh báo khi hệ thống nhiễm trojan backoffice.

2.3 Module 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.

Một vấn đề quan trọng đối với module phát hiện và vấn đề thời gian xử lý gói tin: một IDS thường nhận rất nhiều gói tin và bản thân nó cũng có rất nhiều luật xử lý. Khi lưu lượng mạng quá lớn có thể xảy ra việc bỏ sót hoặc không phản hồi đúng lúc. Khả năng xử lý của module phát hiện phụ thuộc vào nhiều yếu tố: số lượng các luật, tốc độ hệ thống, băng thông mạng. Một module phát hiện có khả năng tách các phần của gói tin ra và áp dụng luật lên từng phần của gói tin:

- IP header
- Header ở tầng transport: TCP, UDP
- Header ở tầng application: DNS, HTTP, FTP ...
- Phần tải của gói tin

Do các luật trong Snort được đánh số thứ tự ưu tiên nên 1 gói tin khi bị phát hiện bởi nhiều luật khác nhau, cảnh báo được đưa ra theo luật có mức ưu tiên cao nhất.

Do các luật trong Snort được đánh số thứ tự ưu tiên nên một gói tin khi bị phát hiện bởi nhiều luật khác nhau, cảnh báo được đưa ra theo luật có mức ưu tiên cao nhất. Thứ tự là: **pass -> drop -> sdrop -> reject -> alert -> log**

- Alert: Tạo một cảnh báo bằng cách sử dụng phương pháp cảnh báo đã chọn, sau đó ghi lại log.
- Log: ghi lại gói tin.
- Pass: Cho phép bỏ qua gói tin này.

- Sdrop: Chặn gói tin và không ghi log.
- Drop: Chặn gói tin này và ghi lại log.
- Reject: Chặn gói tin, ghi lại log và sau đó gửi thiết lập lại TCP nếu giao thức là TCP hoặc thông báo không thể truy cập, cổng ICMP nếu giao thức là UDP

2.4 Môđun log và cảnh báo

Tùy thuộc vào việc môđun phát hiện có nhận dạng được xâm nhập hay không mà gói tin có thể bị ghi log hoặc đưa ra cảnh báo. Các file log là các file text dữ liệu trong đó có thể được ghi dưới nhiều định dạng khác nhau chẳng hạn tcpdump.

2.5 Môđun kết xuất thông tin

Môđun 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 cảnh báo vào cơ sở dữ liệu.
- Tạo file log dạng xml: việc 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 và hệ thống IDS một cách tập trung và thuận tiện hơn.
- Gửi các thông điệp 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 môđun kết xuất thông tin riêng tùy theo mục đích sử dụng

3. Cấu trúc luật của Snort

Tìm hiểu một ví dụ:

alert tcp 192.168.0.0/22 23 -> any any (content:"confidential"; msg: "Detected confidential").

Snort đi kèm với một tính năng mà chúng ta có thể sử dụng để phân loại các quy tắc, các quy tắc này được tùy chỉnh để phản ánh nhu cầu của mạng nói chung.

action	protocol	Source address	Source port	Direction	Destination address	Destination port	Rule option
Alert	Tcp	any	any	->	Ip address	23	Msg
Log	Udp						Logto
Pass	Icmp						Ipooption
Drop							Seq
Reject							Itype
sdrop							Icode
							Id

Figure: Bảng hiển thị cách viết các quy tắc của Snort

Ta thấy cấu trúc có dạng sau:

Rule Header	Rule Option
-------------	-------------

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 **tiêu chuẩn** để áp dụng luật với gói tin đó.

Phần Option: chứa 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 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.

3.1 Cấu trúc của phần Header

Action	Protocol	Address	Port	Direction	Address	Port
--------	----------	---------	------	-----------	---------	------

- Action: là phần quy định loại hành động nào được thực thi.
- Protocol: giao thức cụ thể.
- Address: địa chỉ nguồn và địa chỉ đích.
- Port: xác định các cổng nguồn, cổng đích của một gói tin.
- Direction: phần này sẽ chỉ ra địa chỉ nguồn và địa chỉ đích.

3.1.1 Action

Thông thường các hành động tạo ra một cảnh báo hoặc log thông điệp hay kích hoạt một luật khác. Action chỉ ra hành động nào được thực hiện khi mà các điều kiện của luật được thỏa mãn. Một hành động được thực hiện khi và chỉ khi tất cả các điều kiện đều phù hợp. Có 5 hành động đã được định nghĩa nhưng ta có thể tạo ra các hành động riêng tùy thuộc vào yêu cầu của mình.

Đối với các phiên bản trước của Snort thì khi nhiều luật là phù hợp với một gói tin nào đó thì chỉ một luật được áp dụng. Sau khi áp dụng luật đầu tiên thì các luật tiếp theo sẽ không áp dụng cho gói tin ấy nữa. Nhưng đối với các phiên bản sau của Snort thì tất cả các luật sẽ được áp dụng gói tin đó.

Có 5 luật được định nghĩa:

- **Pass:** Hành động này hướng dẫn Snort bỏ qua gói tin này. Hành động này đóng vai trò quan trọng trong việc tăng cường tốc độ hoạt động của Snort khi mà ta không muốn áp dụng các kiểm tra trên các gói tin nhất định. Ví dụ ta sử dụng các bẫy (đặt trên một máy nào đó) để nhử các hacker tấn công vào thì ta phải cho tất cả các gói tin đi đến được máy đó. Hoặc là dùng một máy quét để kiểm tra độ an toàn mạng của mình thì ta phải bỏ qua tất cả các gói tin đến từ máy kiểm tra đó.
- **Log:** Hành động này dùng để log gói tin. Có thể log vào file hay vào cơ sở dữ liệu tùy thuộc vào nhu cầu của mình.
- **Alert:** Gửi một thông điệp cảnh báo khi dấu hiệu xâm nhập được phát hiện. Có nhiều cách để gửi thông điệp như gửi ra file hoặc ra một Console. Tất nhiên là sau khi gửi thông điệp cảnh báo thì gói tin sẽ được log lại.
- **Activate:** tạo ra cảnh báo và kích hoạt thêm các luật khác để kiểm tra thêm điều kiện của gói tin.
- **Dynamic:** đây là luật được gọi bởi các luật khác có hành động là Activate.

Các hành động do người dùng định nghĩa: một hành động mới được định nghĩa theo cấu trúc sau:

ruletype action_name{ action definication}

Với ruletype là từ khóa. Hành động được định nghĩa chính xác trong dấu ngoặc nhọn: có thể là một hàm viết bằng ngôn ngữ C. VD:

```
ruletype smb_db_alert {
type alert
output alert_smb: workstation.list
output database: log, mysql, user=test password=test dbname=snort host =
localhost
}
```

Đây là hành động có tên là *smb_db_alert* dùng để gửi thông điệp cảnh báo dưới dạng cửa sổ pop-up SMB tới các máy có tên trong danh sách liệt kê trong file workstation.list và tới cơ sở dữ liệu MySQL tên là snort.

3.1.2 Protocol

Là phần thứ hai của một luật có chức năng chỉ ra loại gói tin mà luật sẽ được áp dụng. Protocols qui định việc áp dụng luật cho các packet chỉ thuộc một giao thức cụ thể nào đó. Hiện tại Snort hiểu được các protocol sau:

- IP
- ICMP
- TCP
- UDP

Nếu là IP thì Snort sẽ kiểm tra header của lớp liên kết để xác định loại gói tin. Nếu bất kì giao thức nào khác được sử dụng thì Snort sử dụng header IP để xác định loại protocol. Protocol chỉ đóng vai trò trong việc chỉ rõ tiêu chuẩn trong phần header của luật. Phần option của luật có thể có các điều kiện không liên quan gì đến protocol.

3.1.3 Address

Là phần địa chỉ nguồn và địa chỉ đích. Các địa chỉ có thể là một máy đơn, nhiều máy hoặc của một mạng nào đó. Trong hai phần địa chỉ trên thì một sẽ là địa chỉ nguồn, một sẽ là địa chỉ đích và địa chỉ nào thuộc loại nào sẽ do phần Direction “->” qui định. Có hai phần địa chỉ trong một luật của Snort. Các địa chỉ này được dùng để kiểm tra nguồn sinh ra và đích đến của gói tin. Địa chỉ có thể là địa chỉ của một IP đơn hoặc là địa chỉ của một mạng. Ta có thể dùng từ any để áp dụng luật cho tất cả các địa chỉ.

Chú ý: nếu là một host thì có dạng: IP-address/32. VD: 192.168.0.1/32

Địa chỉ được viết ngay theo sau một dấu gạch chéo và số bit trong subnet mask. Ví dụ như địa chỉ 192.168.2.0/24 thể hiện mạng lớp C 192.168.2.0 với 24 bit của subnet mask. Subnet mask 24 bit chính là 255.255.255.0. Ta biết rằng:

- Nếu subnet mask là 24 bit thì đó là mạng lớp C
- Nếu subnet mask là 16 bit thì đó là mạng lớp B

- Nếu subnet mask là 8bit thì đó là mạng lớp A
- Nếu subnet mask là 32 bit thì đó là địa chỉ IP đơn.

Trong hai địa chỉ của một luật Snort thì có một địa chỉ là địa chỉ nguồn và địa chỉ còn lại là địa chỉ đích. Việc xác định đâu là địa chỉ nguồn, đâu là địa chỉ đích thì phụ thuộc vào phần hướng (direction).

Ví dụ như luật:

alert tcp any any -> 192.168.1.10/32 80 (msg: "TTL=100"; ttl: 100;)

Luật trên sẽ tạo ra một cảnh báo đối với tất cả các gói tin từ bất kì nguồn nào có TTL = 100 đi đến web server 192.168.1.10 tại cổng 80.

Ngăn chặn địa chỉ hay loại trừ địa chỉ:

Snort cung cấp phương pháp để loại trừ địa chỉ IP bằng cách sử dụng dấu "!". Dấu phủ định này đứng trước địa chỉ sẽ chỉ cho Snort không kiểm tra các gói tin đến từ hay đi tới địa chỉ đó. Ví dụ, luật sau sẽ áp dụng cho tất cả các gói tin ngoại trừ các gói có nguồn xuất phát từ mạng lớp C 192.168.2.0.

VD:

alert icmp ![192.168.0.0/22] any -> any any (msg: "Ping with TTL=100"; ttl: 100;)

Ta có thể định rõ ra danh sách các địa chỉ trong một luật của Snort. Ví dụ nếu bạn muốn áp dụng luật cho tất cả các gói tin trừ các gói xuất phát từ hai mạng lớp C 192.168.2.0 và 192.168.8.0 thì luật được viết như sau:

alert icmp![192.168.2.0/24, 192.168.8.0/24] any -> any any (msg: "Ping with TTL=100"; ttl: 100;)

Hai dấu [] chỉ cần dùng khi có dấu ! đứng trước.

3.1.4 Port

Xác định các cổng nguồn và đích của một gói tin mà trên đó luật được áp dụng. Số hiệu cổng dùng để áp dụng luật cho các gói tin đến từ hoặc đi đến một cổng hay một phạm vi cổng cụ thể nào đó. Ví dụ ta có thể sử dụng số cổng nguồn là 23 để áp dụng luật cho tất cả các gói tin đến từ một server Telnet. Từ any cũng được dùng để đại diện cho tất cả các cổng. Chú ý là số hiệu cổng chỉ có ý nghĩa

trong các giao thức TCP và UDP thôi. Nếu protocol của luật là IP hay ICMP thì số hiệu cổng không đóng vai trò gì cả. Ví dụ:

alert tcp 192.168.2.0/24 23 -> any any (content: “confidential”; msg: “Detected confidential”);

Số hiệu cổng chỉ hữu dụng khi ta muốn áp dụng một luật chỉ cho một loại gói tin dữ liệu cụ thể nào đó. Ví dụ như là một luật để chống hack cho web thì ta chỉ cần sử dụng cổng 80 để phát hiện tấn công.

Dãy cổng hay vi phạm cổng:

Ta có thể áp dụng luật cho dãy các cổng thay vì chỉ cho một cổng nào đó. Cổng bắt đầu và cổng kết thúc phân cách nhau bởi dấu hai chấm “:”. Ví dụ:

alert udp any 1024:2048 -> any any (msg: “UDP ports”);

Ta cũng có thể dùng cổng theo kiểu cận trên và cận dưới, tức là chỉ sử dụng cổng bắt đầu hoặc cổng kết thúc. Ví dụ như là “1024:” hoặc là “:2048”. Dấu phủ định cũng được áp dụng trong việc sử dụng cổng. Ví dụ sau sẽ log tất cả các gói tin ngoại trừ các gói tin xuất phát từ cổng 53.

log udp any !53 -> any any log udp

Sau đây là một số cổng thông dụng hay là các cổng của các dịch vụ thông dụng nhất:

- 20 FTP data
- 21 FTP
- 22 SSH
- 23 Telnet
- 24 SMTP
- 53 DNS Server
- 80 HTTP
- 110 POP3
- 161 SNMP
- 443 HTTPS
- 3360 MySQL

3.1.5 Direction

Chỉ ra đâu là nguồn đâu là đích, có thể là -> hay <- hoặc <>. Trường hợp <> là khi ta muốn kiểm tra cả Client và Server.

Ví dụ:

“alert icmp any any -> any any (msg: “Ping with TTL=100”;ttl: 100;)”

Phần đứng trước dấu mở ngoặc là phần Header của luật, phần còn lại là phần Option. Chi tiết của phần Header như sau:

- Hành động của luật ở đây là “alert”: một cảnh báo sẽ được tạo ra nếu như các điều kiện của gói tin là phù hợp với luật(gói tin luôn được log lại mỗi khi cảnh báo được tạo ra).
- Protocol của luật, ở đây là ICMP tức là luật chỉ áp dụng cho các gói tin thuộc loại ICMP. Bởi vậy, nếu như một gói tin không thuộc loại ICMP thì phần còn lại của luật sẽ không cần đối chiếu.
- Địa chỉ nguồn ở đây là “any”: tức là luật sẽ áp dụng cho tất cả các gói tin đến từ mọi nguồn còn cổng thì cũng là “any” vì đối với loại gói tin ICMP thì cổng không có ý nghĩa. Số hiệu cổng chỉ có ý nghĩa với các gói tin thuộc loại TCP hoặc UDP thôi.

Còn phần Option trong dấu đóng ngoặc chỉ ra một cảnh báo chứa dòng “Ping with TTL=100” sẽ được tạo khi tìm thấy điều kiện TTL=100. TTL là Time To Live là một trường trong Header IP.

Để sử dụng 1 dãy các port thì ta phân biệt bởi dấu “:”. VD:

alert udp any 1024:8080 -> any any (msg: “UDP port”;) ”

3.2 Phần Option

Phần Rule Option nằm ngay sau phần Rule Header và được bao bọc trong dấu ngoặc đơn. Nếu có nhiều option thì các option sẽ được phân cách với nhau bằng dấu chấm phẩy “,”. Nếu nhiều option được sử dụng thì các option này phải đồng thời được thỏa mãn tức là theo logic các option này liên kết với nhau bằng AND.

Mọi option được định nghĩa bằng các từ khoá. Một số các option còn chứa các tham số. Nói chung một option gồm 2 phần: một từ khoá và một tham số, hai phần này phân cách nhau bằng dấu hai chấm. Ví dụ đã dùng:

msg: “Detected confidented”;

msg là từ khoá còn “Detected confidented” là tham số.

Chi tiết một số các option của luật Snort.

3.2.1 Từ khóa ack:

Trong header TCP có chứa trường Acknowledgement Number với độ dài 32 bit. Trường này chỉ ra số thứ tự tiếp theo gói tin TCP của bên gửi đang được chờ để nhận. Trường này chỉ có ý nghĩa khi mà cờ ACK được thiết lập. Các công cụ như Nmap sử dụng đặc điểm này để ping một máy. Ví dụ nó có thể gửi gói tin TCP tới cổng 80 với cờ ACK được bật và số thứ tự là 0. Bởi vậy bên nhận thấy gói tin không hợp lệ sẽ gửi lại gói tin RST. Và khi nhận được gói RST này, Nmap sẽ biết được IP này đang tồn tại hay không.

Để kiểm tra loại ping TCP này thì ta có thể dùng luật sau:

Alert tcp any any -> 192.168.0.0/22 any (flags: A; ack: 0; msg: “TCP ping detected”)

3.2.2 Từ khóa classtype

Các luật có thể được phân loại và gán cho 1 số chỉ độ ưu tiên nào đó để nhóm và phân biệt chúng với nhau. Để hiểu rõ hơn về classtype thì ta cần hiểu được file classification.config. Mỗi dòng trong file này đều có cấu trúc như sau:

Config classification: name, description, priority

Trong đó:

- **Name:** tên dùng để phân loại, tên này sẽ được dùng với từ khóa classtype trong các luật Snort
- **Description:** mô tả về loại lớp này.

- **Priority:** là 1 số chỉ độ ưu tiên mặc định của lớp này. Độ ưu tiên này có thể được điều chỉnh trong từ khóa priority của phần Option trong Snort

VD: Config classification: DoS, Denied of Service Attack, 2

Và luật :

alert udp any any -> 192.168.1.0/24 6838 (msg:"DoS"; content: "server"; classtype: DoS;)

alert udp any any -> 192.168.1.0/24 6838 (msg:"DoS"; content: "server"; classtype: DoS; priority: 1;)

Trong câu lệnh thứ 2 thì ta đã ghi đè lên giá trị priority mặc định của lớp đã định nghĩa.

3.2.3 Từ khóa content

Một đặc tính quan trọng của Snort là có khả năng tìm 1 mẫu dữ liệu bên trong một gói tin. Mẫu này có thể dưới dạng chuỗi ASCII hoặc là một chuỗi nhị phân dưới dạng các kí tự hệ 16. Giống như virus, các tấn công cũng có các dấu hiệu nhận dạng và từ khoá content này dùng để tìm các dấu hiệu đó bên trong gói tin. Ví dụ:

VD: **alert tcp 192.168.0.0/22 any -> ![192.168.0.0/22] any (content: "GET"; msg:"GET match");**

Luật trên tìm mẫu "GET" trong phần dữ liệu của tất cả các gói tin TCP có nguồn đi từ mạng 192.168.1.0/24 và đi đến các địa chỉ không thuộc mạng đó. Từ "GET" này rất hay được dùng trong các tấn công HTTP. Một luật khác cũng thực hiện đúng nhiệm vụ giống như lệnh trên nhưng mẫu dữ liệu lại dưới dạng hệ 16 là:

alert tcp 192.168.1.0/24 any -> ![192.168.1.0/24] any (content: "|47 45 54|"; msg: "GET match");

Để ý rằng số 47 ở hệ 16 chính là bằng kí tự ASCII: G và tương tự 45 là E và 54 là T. Ta có thể dùng cả hai dạng trên trong cùng một luật nhưng nhớ là phải để

dạng thập lục phân giữa cặp kí tự ||. Tuy nhiên khi sử dụng từ khoá content ta cần nhớ rằng:

- Đối sánh nội dung sẽ phải xử lý tính toán rất lớn và ta phải hết sức cân nhắc khi sử dụng nhiều luật có đối sánh nội dung.
- Ta có thể sử dụng nhiều từ khoá content trong cùng một luật để tìm nhiều dấu hiệu trong cùng một gói tin.
- Đối sánh nội dung là công việc rất nhạy cảm.

Có 3 từ khoá khác hay được dùng cùng với từ khoá content dùng để bổ sung thêm các điều kiện để tìm kiếm là:

- offset: dùng để xác định vị trí bắt đầu tìm kiếm (chuỗi chứa trong từ khoá content) là offset tính từ đầu phần dữ liệu của gói tin. Ví dụ sau sẽ tìm chuỗi “HTTP” bắt đầu từ vị trí cách đầu đoạn dữ liệu của gói tin là 4 byte:

alert tcp 192.168.1.0/24 any -> any any (content: “HTTP”; offset: 4; msg: “HTTP matched”);

- dept: dùng để xác định vị trí mà từ đó Snort sẽ dừng việc tìm kiếm. Từ khoá này cũng thường được dùng chung với từ khoá offset vừa nêu trên. Ví dụ:

alert tcp 192.168.1.0/24 any -> any any (content: “HTTP”; offset: 4; dept: 40; msg: “HTTP matched”);

Từ khoá này sẽ giúp cho việc tiêu tốn thời gian tìm kiếm khi mà đoạn dữ liệu trong gói tin là khá lớn.

- content-list: được sử dụng cùng với một file. Tên file (được chỉ ra trong phần tham số của từ khoá này) là một file text chứa danh sách các chuỗi cần tìm trong phần dữ liệu của gói tin. Mỗi chuỗi nằm trên một dòng riêng biệt. Ví dụ như file test có dạng như sau:

“test”

“Snort”

“NIDS”

và ta có luật sau:

alert tcp 192.168.1.0/24 any -> any any (content-list: “test”;msg: “This is my Test”);)

Ta cũng có thể dùng kí tự phủ định ! trước tên file để cảnh báo đối với các gói tin không tìm thấy một chuỗi nào trong file đó.

3.2.4 Từ khóa dsize

Dùng để đối sánh theo chiều dài của phần dữ liệu. Rất nhiều tấn công sử dụng lỗi tràn bộ đệm bằng cách gửi các gói tin có kích thước rất lớn. Sử dụng từ khóa này, ta có thể so sánh độ lớn của phần dữ liệu của gói tin với một số nào đó.

VD:

alert ip any any -> 192.168.0.0/22 any (dsize > 5000; msg: “Goi tin co kích thước lớn”);)

3.2.5 Từ khóa Flags

Từ khóa này dùng để phát hiện xem những bit cờ flag nào được bật trong phần TCP header của gói tin. Mỗi cờ có thể được sử dụng như 1 tham số trong từ khóa flags.

Flag	Kí hiệu tham số dùng trong luật của Snort
FIN – Finish Flag	F
SYN – Sync Flag	S
RST – Reset Flag	R
PSH – Push Flag	P

Flag	Kí hiệu tham số dùng trong luật của Snort
ACK – Acknowledge Flag	A
URG – Urgent Flag	U
Reversed Bit 1	1
Reversed Bit 2	2
No Flag set	0

VD: luật sau đây sẽ phát hiện một hành động quét dùng gói tin SYN-FIN:

Alert tcp any any -> 192.168.0.0/22 any (flags: SF; msg: “SYN-FIN flag detected”);

3.2.6 Từ khóa fragbits

Phần IP header của gói tin chứa 3 bit dùng để chống phân mảnh và tổng hợp các gói tin IP. Các bit đó là:

- Reversed bit (RB) dùng để dành cho tương lai
- Don't Fragment Bit (DF): nếu bit này được thiết lập tức là gói tin không bị phân mảnh
- More Fragments Bit (MF): nếu được thiết lập thì các phần khác của gói tin vẫn đang trên đường đi mà chưa tới đích. Nếu bit này không được thiết lập thì đây là phần cuối cùng của gói tin.

VD: luật sau sẽ phát hiện xem bit DF trong gói tin ICMP có được bật hay không: **alert icmp any any -> 192.168.0.0/22 any (fragbits: D; msg: “Don't Fragment bit set”**

4. Kiểm tra và bổ sung các quy tắc trong Snort

Các quy tắc Snort có thể được đặt trực tiếp trong (các) tệp cấu hình *Lua* của một người thông qua mô-đun ips, nhưng phần lớn chúng sẽ nằm trong các tệp *.rules* riêng biệt được *"include"*. Ví dụ: có tệp *Malware.rules* trong cùng thư mục với tệp cấu hình *Lua* của chúng ta. Chúng ta có thể *"include"* tệp quy tắc đó như sau:

```
ips = { include = 'malware.rules' }

ips=
{
    rules=[[
        include/path/to/rulesfile1.rules
        include/path/to/rulesfile2.rules
        .....
    ]]
}
```

```
$ snort -c $my_path/lua/snort.lua -R malware.rules -r bad.pcap
```

Tạo cảnh báo:

Snort cung cấp một vài tùy chọn "chế độ cảnh báo" khác nhau có thể được đặt trên dòng lệnh để điều chỉnh cách hiển thị cảnh báo. Các chế độ này bao gồm cmg hiển thị cảnh báo cùng với kết xuất hex của (các) gói cảnh báo, cũng như một số alert_*chế độ khác nhau được hiển thị bên dưới:

```
$ snort --help-modules | grep alert
alert_csv (logger): output event in csv format
alert_fast (logger): output event with brief text format
alert_full (logger): output event with full packet dump
alert_json (logger): output event in json format
alert_syslog (logger): output event to syslog
alert_talos (logger): output event in Talos alert format
alert_unixsock (logger): output event over unix socket
alerts (basic): configure alerts
```

Kết quả của lệnh `snort --help-modules | grep alert` dưới dạng các module liên quan đến alert:

1. `alert_csv` (logger): Đây là module dùng để xuất sự kiện ra dưới định dạng CSV (Comma-Separated Values). Nó giúp bạn lưu trữ thông tin sự kiện vào một tệp tin CSV để dễ dàng xử lý hoặc phân tích sau này.
2. `alert_fast` (logger): Module này xuất sự kiện ra dưới định dạng văn bản ngắn gọn. Nó thường được sử dụng để hiển thị thông tin cơ bản về sự kiện mà không cần chi tiết về gói tin.
3. `alert_full` (logger): Module này xuất sự kiện ra kèm theo toàn bộ nội dung của gói tin. Điều này hữu ích khi bạn muốn xem chi tiết về sự kiện và dữ liệu gói tin liên quan.
4. `alert_json` (logger): Module này xuất sự kiện ra dưới định dạng JSON (JavaScript Object Notation). JSON là một định dạng dữ liệu phổ biến và dễ dàng để tích hợp với các ứng dụng khác.
5. `alert_syslog` (logger): Module này xuất sự kiện ra hệ thống syslog. Syslog là một cơ chế ghi log phổ biến trên hệ thống Unix và Linux.
6. `alert_talos` (logger): Module này xuất sự kiện ra theo định dạng Talos alert. Talos là một tổ chức nghiên cứu về bảo mật và định dạng này thường được sử dụng trong các hệ thống IDS/IPS.
7. `alert_unixsock` (logger): Module này xuất sự kiện qua socket Unix. Điều này cho phép bạn gửi thông tin sự kiện đến một tiến trình khác trên cùng máy chủ.
8. `alerts` (basic): Module này dùng để cấu hình các cảnh báo (alerts). Bạn có thể tùy chỉnh các cảnh báo dựa trên các luật (rules) của Snort.
Để bảo vệ mạng, điều quan trọng là phải đảm bảo rằng các quy tắc đang chặn các cuộc tấn công một cách thích hợp và DAQ kết xuất cho phép thực hiện điều đó. Việc chỉ định tùy chọn `-Q` để bật chế độ nội tuyến rồi đặt `--daq` thành kết xuất sẽ "kết xuất" lưu lượng truy cập lẽ ra đã được chuyển qua, mô phỏng một hoạt động nội tuyến thực sự. Theo mặc định, lưu lượng truy cập thu được sẽ được chuyển sang tệp có tên

`inline-out.pcap`:

```
$ snort3 -Q --daq dump -q -r get.pcap -R local.rules
```

Trong ví dụ trên, nếu tệp `local.rules` chứa quy tắc chặn kích hoạt một số lưu lượng truy cập trong tệp `get.pcap` thì tệp `inline-out.pcap` thu được sẽ chỉ chứa lưu lượng truy cập không bị chặn. Có thể sử dụng chức năng này để kiểm tra xem các quy tắc có đang ngăn chặn (các) gói tấn công thực tế đi qua hay không.

4.1. Phát hiện dựa trên quy tắc

Hệ thống phát hiện của Snort dựa trên các quy tắc. Các quy tắc này lần lượt dựa trên chữ ký của kẻ xâm nhập. Những chữ ký này có thể có mặt trong các phần tiêu đề của gói hoặc trong tải trọng. Quy tắc Snort có thể được sử dụng để kiểm tra các phần khác nhau của gói dữ liệu. Snort 1.x version có thể phân tích các tiêu đề lớp 3 và 4 nhưng không thể phân tích các giao thức lớp ứng dụng. Phiên bản Snort 2 bổ sung hỗ trợ các tiêu đề lớp ứng dụng. Các quy tắc được áp dụng một cách có trật tự cho tất cả các gói tùy thuộc vào loại của chúng.

Thứ tự các quy tắc dựa trên hành động :

- Alert rules: Đây là loại quy tắc phổ biến nhất trong Snort. Khi Snort phát hiện một sự kiện phù hợp với quy tắc cảnh báo, nó sẽ tạo ra một cảnh báo (alert) và ghi lại thông tin về sự kiện đó. Cảnh báo có thể được gửi đến hệ thống syslog, lưu vào tệp tin log, hoặc hiển thị trực tiếp trên màn hình.
- Pass rules: Quy tắc này cho phép Snort bỏ qua việc kiểm tra một số lưu lượng mạng cụ thể. Nếu một gói tin khớp với quy tắc “pass”, Snort sẽ không tạo ra cảnh báo và không ghi lại thông tin về gói tin đó.
- Log rules: Loại quy tắc này chỉ đơn giản là ghi lại thông tin về gói tin mà không tạo ra cảnh báo. Điều này hữu ích khi bạn muốn theo dõi lưu lượng mạng mà không muốn bị làm phiền bởi các cảnh báo.

Khi một gói được nhận bởi Snort, nó được kiểm tra theo thứ tự này. Mỗi gói phải trải qua tất cả các kiểm tra quy tắc Cảnh báo trước khi được phép vượt qua. Lược đồ này là an toàn nhất vì không có gói nào đi qua mà không được kiểm tra đối với tất cả các loại cảnh báo. Tuy nhiên, hầu hết các gói tin là lưu lượng truy cập bình thường và không hiển thị bất kỳ hoạt động xâm nhập nào. Kiểm tra tất cả các gói tin chống lại tất cả các quy tắc cảnh báo đòi hỏi rất nhiều sức mạnh xử lý. Snort cung cấp một cách để thay đổi thứ tự thử nghiệm này thành một thứ tự hiệu quả hơn, nhưng nguy hiểm hơn.

- Pass rules
- Alert rules
- Log rules

4.2. Các quy tắc mới trong Snort 3:

Quy tắc dịch vụ, quy tắc tệp và quy tắc nhận dạng tệp. Quy tắc dịch vụ và tệp cho phép tạo các quy tắc trông rõ ràng hơn, tương ứng với dịch vụ cụ thể và không xác định dịch vụ, trong khi quy tắc nhận dạng tệp sử dụng tùy chọn quy tắc file_meta mới để thực hiện nhận dạng loại tệp. Mỗi loại quy tắc mới này được tạo bằng cách sử dụng một tiêu đề quy tắc duy nhất

4.2.1 Quy tắc dịch vụ:

- là một loại quy tắc mới trong Snort 3 cho phép người viết quy tắc đối sánh lưu lượng truy cập của một dịch vụ cụ thể bằng cách sử dụng tiêu đề quy tắc chỉ bao gồm một hành động và tên của dịch vụ lớp ứng dụng.
- Sự khác biệt giữa các tiêu đề này và các tiêu đề "truyền thống" được mô tả ở đây là những tiêu đề này không yêu cầu khai báo địa chỉ mạng, cổng hoặc toán tử định hướng.
- Các quy tắc dịch vụ này cho phép người viết quy tắc nhắm mục tiêu đến một dịch vụ cụ thể bất kể địa chỉ IP hoặc cổng đang được sử dụng trong một luồng mạng nhất định.
- Loại quy tắc này đặc biệt hữu ích cho các dịch vụ như HTTP, nơi thường thấy các máy chủ web chạy trên các cổng TCP khác 80. Ví dụ: tiêu đề quy tắc sau đây yêu cầu Snort chỉ áp dụng quy tắc này cho lưu lượng truy cập mà Snort phát hiện là HTTP:

Quy tắc sau sẽ cảnh báo về việc khớp lưu lượng HTTP bất kể cổng hoặc địa chỉ IP được sử dụng trong kết nối:

```
alert http
(
  msg:"SERVER-WEBAPP This rule only looks at HTTP traffic";
  flow:to_server,established;
  http_uri;
  content:"/admin.php",fast_pattern,nocase;
  content:"cmd=";nocase;
  pcre:"/[?&]cmd=[^&]*?\x3b/i";
  sid:1;
)
```

4.2.2 Quy tắc tệp(file rules)

"Quy tắc tệp" mới của Snort cho phép người viết quy tắc tạo quy tắc để khớp với một tệp cụ thể bất kể giao thức, IP nguồn, IP đích, cổng và dịch vụ. Snort có thể xử lý các tệp được gửi bằng bất kỳ giao thức lớp ứng dụng nào sau đây:

- HTTP
- SMTP
- POP3
- IMAP
- SMB
- FTP

Các quy tắc này được tạo bằng tiêu đề quy tắc chỉ chứa một hành động theo sau là tệp từ khóa: *action file*

- Chỉ định bộ đệm file_data cho tất cả nội dung khớp cần khớp trong tệp
- Bỏ qua mọi tùy chọn *flow* và service khỏi quy tắc

Để thấy được ưu điểm của tiêu đề quy tắc như vậy, hãy xem xét hai quy tắc bên dưới tìm kiếm "secret_encryption_key" trong một gói:

- Quy tắc đầu tiên tìm kiếm chuỗi có trong các gói HTTP và IMAP được gửi đến máy khách.
- Quy tắc thứ hai tìm kiếm chuỗi có trong các gói SMTP được gửi đến một số máy chủ SMTP.

```

alert tcp $EXTERNAL_NET [80,143] -> $HOME_NET any
(
  msg:"MALWARE-OTHER Win.Ransomware.Agent payload download
attempt";
  flow:to_client,established;
  file_data; content:"secret_encryption_key",fast_pattern,nocase;
  service:http, imap;
  classtype:trojan-activity;
  sid:1;
)
alert tcp $EXTERNAL_NET any -> $SMTP_SERVERS 25
(
  msg:"MALWARE-OTHER Win.Ransomware.Agent payload download
attempt";
  flow:to_server,established;
  file_data; content:"secret_encryption_key",fast_pattern,nocase;
  service:smtp;
  classtype:trojan-activity;
  sid:2;
)

```

Tuy nhiên, cặp quy tắc này có thể được viết dưới dạng một quy tắc tệp cảnh báo duy nhất, quy tắc này sẽ yêu cầu Snort tìm kiếm "***secret_encryption_key***" trong bất kỳ tệp nào được phát hiện trên mạng, bất kể nguồn, đích hoặc dịch vụ.

```

alert file
(
  msg:"MALWARE-OTHER Win.Ransomware.Agent payload download
attempt";
  file_data;
  content:"secret_encryption_key",fast_pattern,nocase;
  classtype:trojan-activity;
)

```

```
sid:3;
```

```
)
```

4.2.3 Quy tắc nhận dạng tệp

- Các quy tắc này là các quy tắc cơ bản của Snort , nhưng thay vì cảnh báo và/hoặc chặn lưu lượng truy cập, chúng xác định các tệp dựa trên nội dung của tệp đó và sau đó xác định loại tệp có thể được sử dụng trong các quy tắc tiếp theo với các tùy chọn file_type.
- Quy tắc gồm 2 thành phần chính:
 - Một tiêu đề quy tắc chỉ bao gồm file_id, cho Snort biết rằng quy tắc theo sau là định nghĩa loại tệp
 - Tùy chọn quy tắc file_meta đặt tiêu đề dữ liệu tệp cho quy tắc nhận dạng tệp nhất định
 - Vì các quy tắc này được sử dụng để xác định một tệp cụ thể nên người viết quy tắc nên tìm kiếm bất kỳ và tất cả các tùy chọn tải trọng trong bộ đệm file_data.

```
file_id (  
  msg:"Windows/DOS executable file";  
  file_meta:type MSEXEC, id 21, category "Executables,Dynamic Analysis  
Capable,Local Malware Analysis Capable";  
  file_data;  
  content:"| 4D 5A |", depth 2, offset 0;  
  gid:4;  
  sid:16;  
  rev:1;  
)
```

Quy tắc này kiểm tra xem gói tin có chứa chuỗi “| 4D 5A |” hay không. Nếu chuỗi này xuất hiện trong dữ liệu của gói tin, Snort sẽ tạo ra cảnh báo với thông điệp “Potential Windows/DOS executable file”. Chuỗi “| 4D 5A |” đại diện cho mã ký tự HEX của tệp thực thi Windows (được biết đến là tệp EXE). Mã ký tự này tương ứng với đầu của tệp thực thi, nên việc kiểm tra chuỗi này giúp Snort phát hiện các tệp thực thi trong lưu lượng mạng

Mục nhập file_id cũng có thể xác định phiên bản loại tệp cụ thể, được đặt thông qua đối số phiên bản

```
file_id (  
  msg:"Windows/DOS executable file";  
  file_meta:type MSEXEC, id 21, category "Executables,Dynamic Analysis  
Capable,Local Malware Analysis Capable";  
  file_data;  
  content:"| 4D 5A |", depth 2, offset 0;  
  gid:4;  
  sid:16;  
  rev:1;  
)
```

```
msg:"PDF file";
file_meta:type PDF, id 282, category "PDF files,Dynamic Analysis
Capable,Local Malware Analysis Capable", version "1.0";
file_data;
content:"| 25 50 44 46 2D 31 2E 30 |", depth 8, offset 0;
gid:4;
sid:158;
rev:1;
)
```

Kích hoạt nhận dạng tập tin

Việc sử dụng các quy tắc nhận dạng tệp yêu cầu các nội dung file_id và file_policy được bật trong cấu hình Snort của một người. May mắn thay, cả hai đều được bật theo mặc định trong tệp snort.lua tiêu chuẩn:

```
file_id = { rules_file = 'file_magic.rules' }
file_policy = { }
```

4.3. Tùy chọn các quy tắc

Là các phần mở rộng của quy tắc, cho phép bạn tùy chỉnh cách Snort xử lý lưu lượng mạng.

Có bốn loại tùy chọn quy tắc chính:

- general cung cấp ngữ cảnh bổ sung cho một quy tắc nhất định
- chỉ dành riêng cho tải trọng
- non-payload đặt tiêu chí cụ thể không tải trọng
- post-detection đặt hành payload đặt tiêu động để thực hiện trên một gói nhất định sau khi quy tắc đã "kích hoạt"

4.3.1 . General

Cung cấp thông tin về một quy tắc:

1. msg (Message): Tùy chọn này cho phép bạn đặt thông điệp cảnh báo (alert message) mà Snort sẽ hiển thị khi phát hiện một sự kiện phù hợp với quy tắc.

Ví dụ: msg:"Potential SSH Attack"

2. reference: Tùy chọn này được sử dụng để cung cấp ngữ cảnh bổ sung cho quy tắc bằng cách liên kết chúng với các hệ thống xác định tấn công hoặc các tài liệu tham khảo liên quan. Điều này giúp các nhà phân tích hiểu rõ hơn về ý nghĩa của cảnh báo.

Ví dụ: reference:cve,2019-1234

3. gid (Group ID): Số ID nhóm (group ID) xác định thành phần cụ thể của Snort tạo ra một sự kiện cụ thể.
4. sid (Signature ID): Số ID chữ ký (signature ID) xác định số duy nhất được gán cho một quy tắc Snort cụ thể.
5. rev (Revision): Số phiên bản quy tắc (rule revision) xác định phiên bản cụ thể của quy tắc.
6. classtype: Tùy chọn này gán một phân loại cho quy tắc để chỉ định loại tấn công liên quan đến sự kiện.
7. priority: Tùy chọn này đặt mức độ nghiêm trọng cho việc ưu tiên sự kiện phù hợp.
8. metadata: Tùy chọn này thêm thông tin bổ sung và tùy ý vào quy tắc dưới dạng cặp tên-giá trị.
9. service: Tùy chọn này đặt danh sách các dịch vụ được liên kết với một quy tắc cụ thể.
10. rem (Remark): Tùy chọn này được sử dụng để truyền đạt một bình luận tùy ý trong phần thân của quy tắc.
11. file_meta: Tùy chọn này được sử dụng để đặt thông tin metadata về tệp cho một quy tắc xác định tệp.

4.3.2. Payload

Các tùy chọn quy tắc liên quan đến dữ liệu (payload) cho phép kiểm tra các phần khác của gói tin ngoài phần dữ liệu TCP và UDP. Dưới đây là một số tùy chọn quy tắc liên quan đến dữ liệu:

1. content: Tùy chọn này được sử dụng để thực hiện kiểm tra chuỗi cơ bản và/hoặc kiểm tra mẫu hexa trong gói tin. Nếu nội dung được chỉ định xuất hiện trong gói tin, Snort sẽ kích hoạt cảnh báo.
2. fast_pattern: Đây là một tùy chọn của content cho biết Snort sẽ sử dụng kết quả khớp này để xác định xem việc xử lý quy tắc tiếp theo có nên tiếp tục với lưu lượng hay không. Nó cải thiện hiệu suất bằng cách cho phép khớp sớm.
3. nocase: Tùy chọn này cho biết Snort sẽ bỏ qua việc phân biệt chữ hoa/thường khi tìm kiếm mẫu cụ thể.
4. offset: Tùy chọn này chỉ định vị trí bắt đầu tìm kiếm mẫu liên quan đến đầu gói tin hoặc bộ đệm.
5. depth: Tùy chọn này chỉ định khoảng cách tìm kiếm mẫu liên quan đến đầu gói tin hoặc bộ đệm.
6. distance: Tùy chọn này chỉ định vị trí bắt đầu tìm kiếm mẫu liên quan đến khớp nội dung trước đó.

7. within: Tùy chọn này chỉ định khoảng cách tìm kiếm mẫu liên quan đến khớp nội dung trước đó.
8. http_* buffers: Các tùy chọn này là khai báo bộ đệm “sticky” để đặt con trỏ phát hiện ở đầu các phần HTTP khác nhau.

4.3.3. Non-payload

Các tùy chọn quy tắc không liên quan đến dữ liệu (non-payload) trong Snort cho phép kiểm tra các phần khác của gói tin ngoài phần dữ liệu TCP và UDP, cũng như theo dõi trạng thái gói tin để đánh giá trông tương lai. Dưới đây là một số tùy chọn các quy tắc:

1. fragoffset: Tùy chọn này kiểm tra giá trị offset của các phần tử fragment trong tiêu đề IP.
2. ttl: Tùy chọn này kiểm tra giá trị TTL (Time To Live) trong tiêu đề IP.
3. tos: Tùy chọn này kiểm tra giá trị TOS (Type of Service) trong tiêu đề IP.
4. id: Tùy chọn này kiểm tra giá trị ID trong tiêu đề IP.
5. ipopts: Tùy chọn này kiểm tra sự hiện diện của các tùy chọn IP cụ thể trong tiêu đề IP.
6. fragbits: Tùy chọn này kiểm tra các bit fragmentation hoặc các bit dự trữ trong tiêu đề IP.
7. ip_proto: Tùy chọn này kiểm tra giá trị trường protocol trong tiêu đề IP.
8. flags: Tùy chọn này kiểm tra các bit cờ trong tiêu đề TCP.
9. flow: Tùy chọn này kiểm tra các thuộc tính phiên liên quan đến gói tin.
10. flowbits: Tùy chọn này được sử dụng để thiết lập và kiểm tra các cờ boolean tùy ý để theo dõi trạng thái trong phiên giao thức truyền tải.
11. file_type: Tùy chọn này được sử dụng để tạo ra quy tắc giới hạn cho một loại tệp cụ thể hoặc một phiên bản cụ thể của một loại tệp.
12. seq: Tùy chọn này kiểm tra giá trị số thứ tự trong tiêu đề TCP.
13. ack: Tùy chọn này kiểm tra giá trị số xác nhận trong tiêu đề TCP.
14. window: Tùy chọn này kiểm tra giá trị kích thước cửa sổ trong tiêu đề TCP.

4.3.4 . Post-detection

Các tùy chọn quy tắc sau phát hiện (post-detection rule options) là các kích hoạt cụ thể xảy ra sau khi một quy tắc đã “kích hoạt”.

1. detection_filter: yêu cầu nhiều lần truy cập quy tắc trước khi tạo "sự kiện". Người viết quy tắc sử dụng tùy chọn này để xác định tốc độ (số mỗi giây)

phải vượt quá bởi máy chủ lưu trữ nguồn hoặc đích trước khi quy tắc có thể tạo sự kiện.

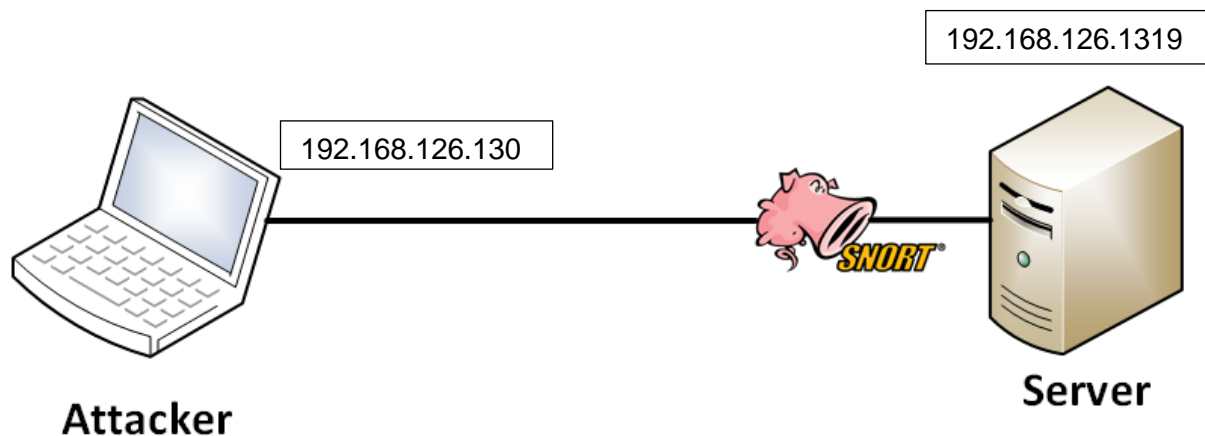
2. replace: Tùy chọn này được sử dụng để khớp và sau đó ghi đè dữ liệu payload.
3. tag: Tùy chọn này được sử dụng để ghi log các gói tin bổ sung sau một sự kiện quy tắc.

5. Chạy demo phần mềm SNORT

Tổng quát yêu cầu bài toán: Sử dụng công cụ nào đó để phát hiện và báo cáo những máy tính bên ngoài kết nối và sử dụng internet hoặc tấn công vào hệ thống mạng. IDS và IPS là công cụ cần thiết cho việc phát hiện xâm nhập và ngăn chặn. Nhóm em sẽ sử dụng và demo bằng Snort - một công cụ phổ biến và dễ sử dụng.

Do bị hạn chế về điều kiện phần cứng thật. Nhóm sẽ sử dụng phần mềm tạo máy ảo VMware để tạo ra một máy ảo Ubuntu. Và dùng máy thật đóng vai là một máy client thực hiện lệnh ping hay truy cập trang web, công việc của máy Server là ghi lại những cảnh báo đó.

Sử dụng mạng mặc định cho máy ở chế độ NAT.



Mô hình thực nghiệm

Hệ thống demo bao gồm:

Vmware Workstation Pro 14, ubuntu, Snort 2.9.11.1 + Rule cho Snort, WinPcap 4.1.3.

5.1. Cài đặt hệ thống

5.1.1. Cài đặt máy ảo

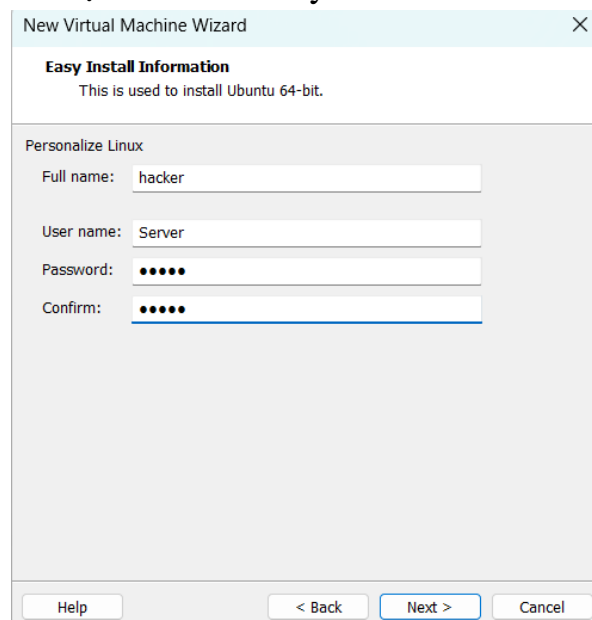
Bước 1: Khởi động Vmware Workstation.

Bước 2: Ctrl + N để mở trình tạo máy ảo mới -> Custom (advanced) -> Next.

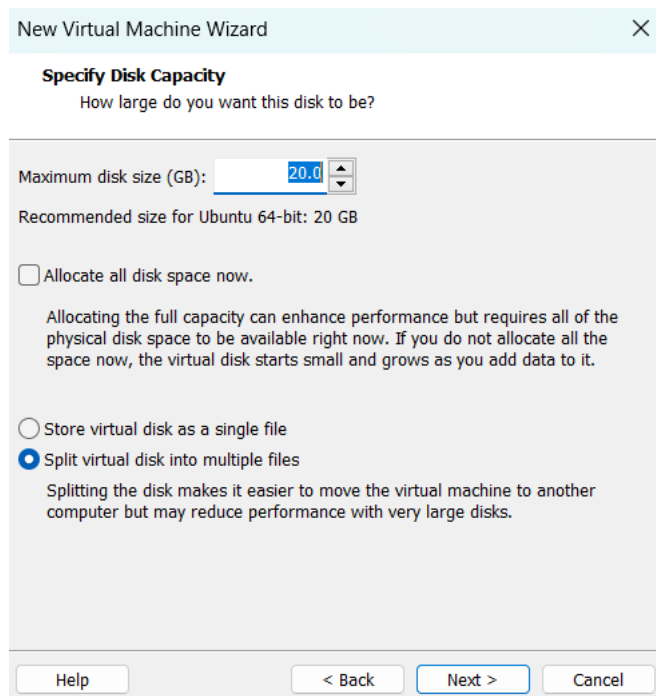


Bước 3: Installer disc_image file (iso) -> Browse đến nơi lưu file iso hệ điều hành -> Next.

Bước 5: Đặt tên và mật khẩu cho máy ảo.



Bước 6: Phân vùng dung lượng ổ cứng cho máy ảo (để 20 GB là được) -> Next



Bước 7: Finish.

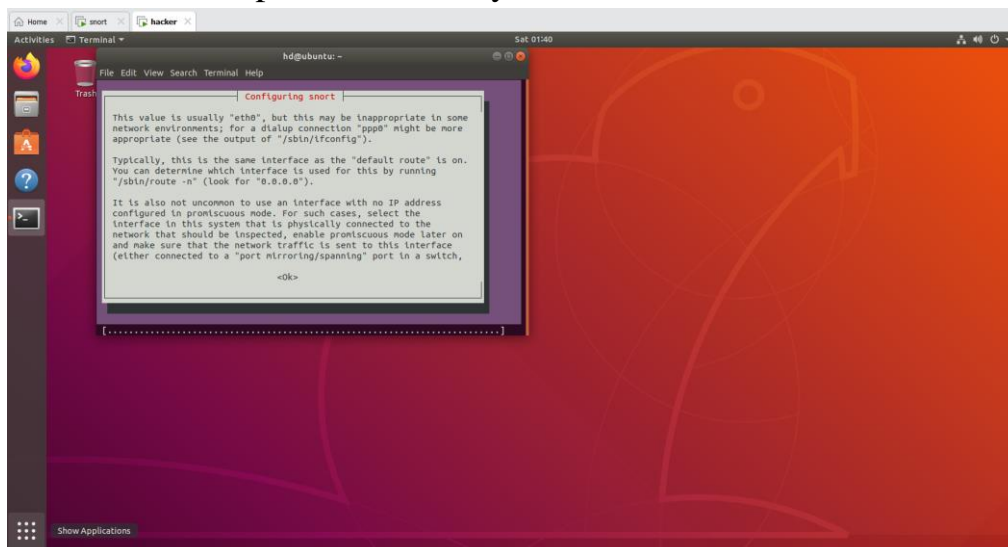
Bước 8: Bước vào trình cài đặt ubuntu, ta ta thiết lập các thông số cho phù hợp với máy tính.

5.1.2. Cài đặt SNORT trên Ubuntu

- Cài đặt SNORT

```
sudo apt update
```

```
sudo apt install snort -y
```

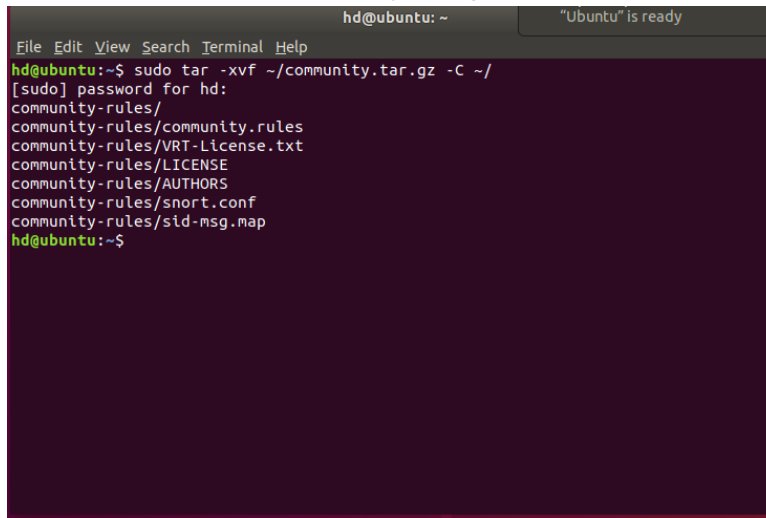


Tại đây ubuntu yêu cầu chúng ta cài một số các thông số cho cấu hình thì chúng ta cứ để tự động không thay đổi gì.

- Cài thêm các rules của Snort:

```
wget https://www.snort.org/rules/community -O ~/community.tar.gz
```

```
sudo tar -xvf ~/community.tar.gz -C ~/
```

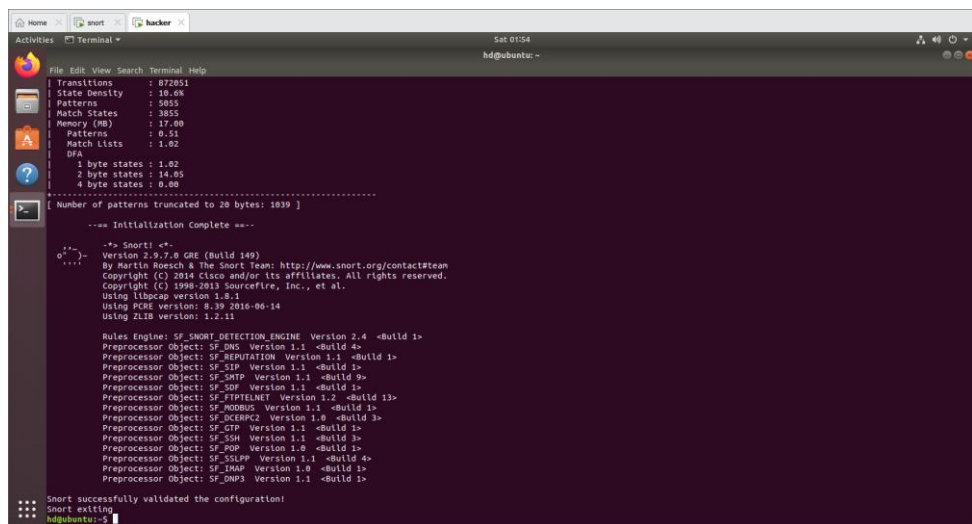


```
hd@ubuntu: ~  
File Edit View Search Terminal Help  
hd@ubuntu:~$ sudo tar -xvf ~/community.tar.gz -C ~/  
[sudo] password for hd:  
community-rules/  
community-rules/community.rules  
community-rules/VRT-License.txt  
community-rules/LICENSE  
community-rules/AUTHORS  
community-rules/snort.conf  
community-rules/sid-msg.map  
hd@ubuntu:~$
```

```
sudo cp ~/community-rules/* /etc/snort/rules
```

- Chạy thử SNORT:

```
sudo snort -T -c /etc/snort/snort.conf
```



```
hd@ubuntu:~$ sudo snort -T -c /etc/snort/snort.conf  
-----  
[ Number of patterns truncated to 20 bytes: 1039 ]  
-----  
--- Initialization Complete ---  
-----  
o***- -> Snort! <+>  
o***- Version 2.9.7.0 GRE (Build 149)  
o***- By Martin Roesch & The Snort Team: http://www.snort.org/contactteam  
o***- Copyright (C) 2014 Cisco and/or its affiliates. All rights reserved.  
o***- Copyright (C) 1998-2013 Sourcefire, Inc., et al.  
o***- Using libpcap version 1.8.1  
o***- Using PCRE version 8.39 2016-06-14  
o***- Using ZLIB version 1.2.11  
-----  
Rules Engine: SF_SNORT DETECTION ENGINE Version 2.4 <Build 1>  
Preprocessor Object: SF_DNS Version 1.1 <Build 4>  
Preprocessor Object: SF_REPUTATION Version 1.1 <Build 1>  
Preprocessor Object: SF_SIP Version 1.1 <Build 1>  
Preprocessor Object: SF_SMP Version 1.1 <Build 9>  
Preprocessor Object: SF_SDP Version 1.1 <Build 1>  
Preprocessor Object: SF_FTPTELNET Version 1.2 <Build 13>  
Preprocessor Object: SF_MODBUS Version 1.1 <Build 1>  
Preprocessor Object: SF_DCEMRP2 Version 1.0 <Build 1>  
Preprocessor Object: SF_GTP Version 1.1 <Build 1>  
Preprocessor Object: SF_SSH Version 1.1 <Build 3>  
Preprocessor Object: SF_POP Version 1.0 <Build 1>  
Preprocessor Object: SF_SSLPP Version 1.1 <Build 4>  
Preprocessor Object: SF_IMAP Version 1.0 <Build 1>  
Preprocessor Object: SF_DNP3 Version 1.1 <Build 1>  
-----  
... Snort successfully validated the configuration!  
... Snort exiting  
hd@ubuntu:~$
```

Lúc này việc cài đặt và cấu hình SNORT đã thành công.

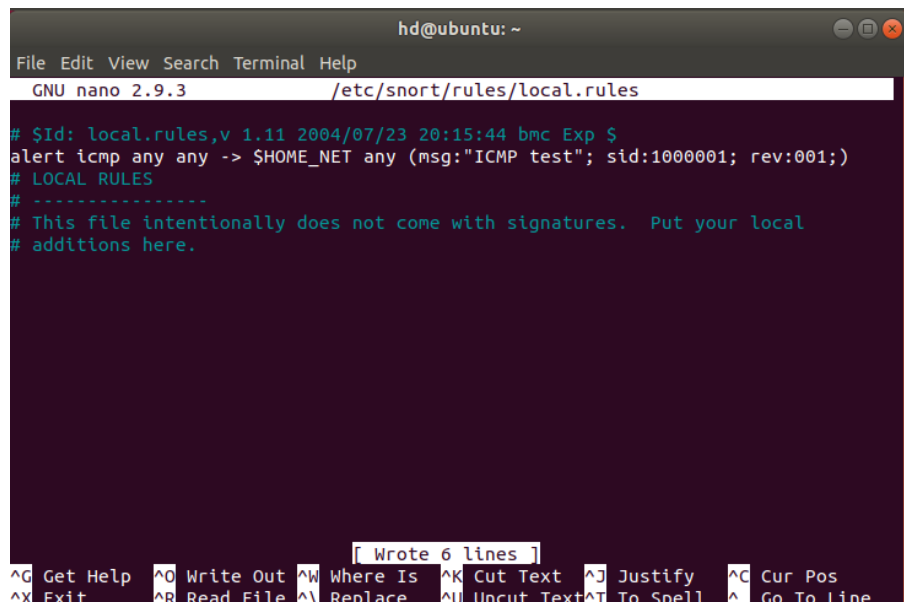
5.1.3 Phát hiện tấn công

- Thêm rules phát hiện tấn công bằng lệnh ping:

```
sudo nano /etc/snort/rules/local.rules
```

- Thêm dòng sau vào local.rules:

```
alert icmp any any -> $HOME_NET any (msg:"ICMP test";  
sid:1000001; rev:001;)
```

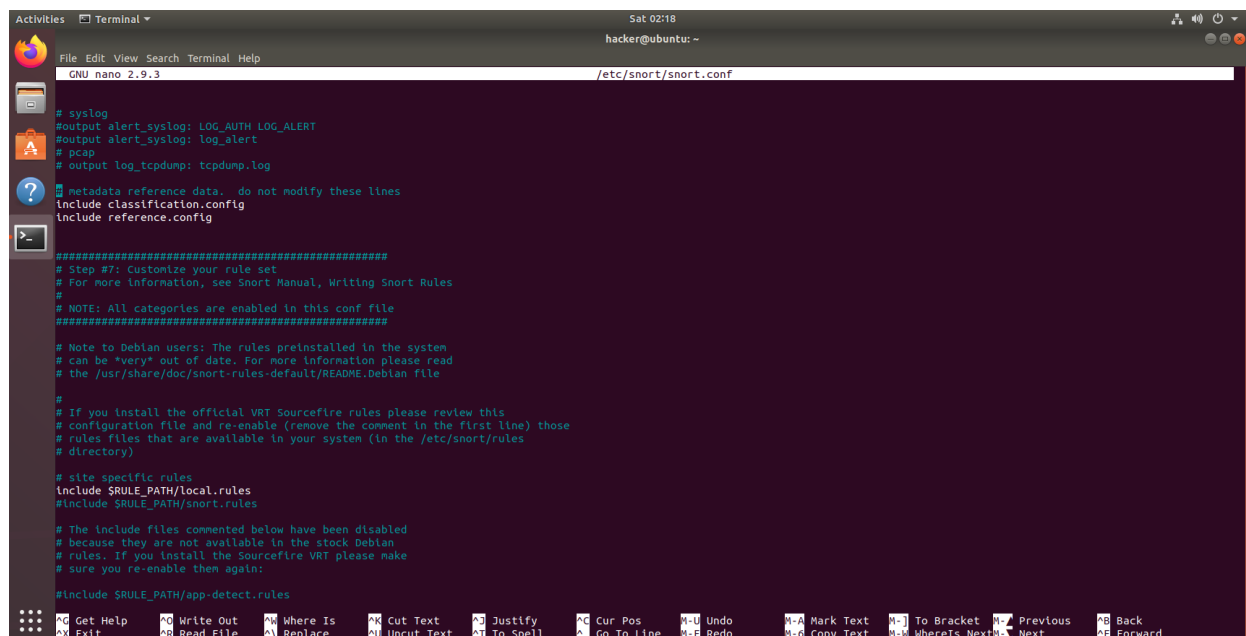


```
hd@ubuntu: ~
File Edit View Search Terminal Help
GNU nano 2.9.3 /etc/snort/rules/local.rules

# $Id: local.rules,v 1.11 2004/07/23 20:15:44 bmc Exp $
alert icmp any any -> $HOME_NET any (msg:"ICMP test"; sid:1000001; rev:001;)
# LOCAL RULES
# -----
# This file intentionally does not come with signatures. Put your local
# additions here.

[Wrote 6 lines]
^G Get Help ^O Write Out ^W Where Is ^K Cut Text ^J Justify ^C Cur Pos
^X Exit ^R Read File ^\ Replace ^U Uncut Text ^T To Spell ^_ Go To Line
```

- Vào file snort.conf bổ sung thêm luật vừa tạo ở local.rules
sudo nano /etc/snort/snort.conf



```
Activities Terminal Sat 02:18
hacker@ubuntu: ~
File Edit View Search Terminal Help
GNU nano 2.9.3 /etc/snort/snort.conf

# syslog
#output alert_syslog: LOG_AUTH LOG_ALERT
#output alert_syslog: log_alert
# pcap
# output log_tcpdump: tcpdump.log

# metadata reference data. do not modify these lines
include classification.config
include reference.config

#####
# Step #7: Customize your rule set
# For more information, see Snort Manual, Writing Snort Rules
#
# NOTE: All categories are enabled in this conf file
#####

# Note to Debian users: The rules preinstalled in the system
# can be "very" out of date. For more information please read
# the /usr/share/doc/snort-rules-default/README.Debian file

#
# If you install the official VRT Sourcefire rules please review this
# configuration file and re-enable (remove the comment in the first line) those
# rules files that are available in your system (in the /etc/snort/rules
# directory)

# site specific rules
include $RULE_PATH/local.rules
include $RULE_PATH/snort.rules

# The include files commented below have been disabled
# because they are not available in the stock Debian
# rules. If you install the Sourcefire VRT please make
# sure you re-enable them again:

#include $RULE_PATH/app-detect.rules

^G Get Help ^O Write Out ^W Where Is ^K Cut Text ^J Justify ^C Cur Pos
^X Exit ^R Read File ^\ Replace ^U Uncut Text ^T To Spell ^_ Go To Line ^Z Undo
^Y Mark Text ^M To Bracket ^N Previous ^B Back
^G Copy Text ^H WhereIs Next ^V Next ^F Forward
```

- Thực thi SNORT:

sudo snort -A console -i ens33 -u snort -g snort -c /etc/snort/snort.conf

```
hd@ubuntu: ~  
File Edit View Search Terminal Help  
'''' By Martin Roesch & The Snort Team: http://www.snort.org/contact#team  
Copyright (C) 2014 Cisco and/or its affiliates. All rights reserved.  
Copyright (C) 1998-2013 Sourcefire, Inc., et al.  
Using libpcap version 1.8.1  
Using PCRE version: 8.39 2016-06-14  
Using ZLIB version: 1.2.11  
  
Rules Engine: SF_SNORT_DETECTION_ENGINE Version 2.4 <Build 1>  
Preprocessor Object: SF_DNS Version 1.1 <Build 4>  
Preprocessor Object: SF_REPUTATION Version 1.1 <Build 1>  
Preprocessor Object: SF_SIP Version 1.1 <Build 1>  
Preprocessor Object: SF_SMTP Version 1.1 <Build 9>  
Preprocessor Object: SF_SDF Version 1.1 <Build 1>  
Preprocessor Object: SF_FTPTELNET Version 1.2 <Build 13>  
Preprocessor Object: SF_MODBUS Version 1.1 <Build 1>  
Preprocessor Object: SF_DCERPC2 Version 1.0 <Build 3>  
Preprocessor Object: SF_GTP Version 1.1 <Build 1>  
Preprocessor Object: SF_SSH Version 1.1 <Build 3>  
Preprocessor Object: SF_POP Version 1.0 <Build 1>  
Preprocessor Object: SF_SSLPP Version 1.1 <Build 4>  
Preprocessor Object: SF_IMAP Version 1.0 <Build 1>  
Preprocessor Object: SF_DNP3 Version 1.1 <Build 1>  
Commencing packet processing (pid=3170)
```

- Trên một máy khác có thể tại window. Dùng lệnh ping tới địa chỉ của máy nạn nhân.

Để biết địa chỉ máy nạn nhân ta dùng ifconfig

```
hacker@ubuntu:~$ ifconfig  
ens33: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500  
    inet 192.168.126.129 netmask 255.255.255.0 broadcast 192.168.126.255  
    inet6 fe80::bf18:63d9:64cd:7b5a prefixlen 64 scopeid 0x20<link>  
    ether 00:0c:29:18:d1:d9 txqueuelen 1000 (Ethernet)  
    RX packets 530 bytes 521572 (521.5 KB)  
    RX errors 0 dropped 0 overruns 0 frame 0  
    TX packets 308 bytes 40171 (40.1 KB)  
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0  
  
lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536  
    inet 127.0.0.1 netmask 255.0.0.0  
    inet6 ::1 prefixlen 128 scopeid 0x10<host>  
    loop txqueuelen 1000 (Local Loopback)  
    RX packets 163 bytes 13705 (13.7 KB)  
    RX errors 0 dropped 0 overruns 0 frame 0  
    TX packets 163 bytes 13705 (13.7 KB)  
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

Tại window dùng lệnh ping đến địa chỉ đó
ping 192.168.126.129

```
C:\Users\Admin>ping 192.168.126.129  
  
Pinging 192.168.126.129 with 32 bytes of data:  
Reply from 192.168.126.129: bytes=32 time<1ms TTL=64  
Reply from 192.168.126.129: bytes=32 time<1ms TTL=64  
Reply from 192.168.126.129: bytes=32 time=2ms TTL=64  
Reply from 192.168.126.129: bytes=32 time=1ms TTL=64  
  
Ping statistics for 192.168.126.129:  
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),  
    Approximate round trip times in milli-seconds:  
        Minimum = 0ms, Maximum = 2ms, Average = 0ms
```

Trên màn hình hiển thị máy nạn nhân lúc này sẽ là các thông báo hiển thị tấn công

```
Activities Terminal
File Edit View Search Terminal Help
pcap DAQ configured to passive.
Acquiring network traffic from "ens33".
Reload thread started...
Reload thread started, thread 0x7f56eb3c700 (2258)
Decoding Ethernet
Set gid to 127
Set uid to 122

==== Initialization Complete ====

-> Snort! <-
Version 2.9.7.0 GRE (Build 149)
By Martin Roesch & The Snort Team: http://www.snort.org/contact#team
Copyright (C) 2014 Cisco and/or its affiliates. All rights reserved.
Copyright (C) 1998-2013 Sourcefire, Inc., et al.
Using libpcap version 1.8.1
Using PCRE version: 8.39 2016-06-14
Using ZLIB version: 1.2.11

Rules Engine: SF_SNORT_DETECTION_ENGINE Version 2.4 <Build 1>
Preprocessor Object: SF_SMP Version 1.1 <Build 9>
Preprocessor Object: SF_DNS Version 1.1 <Build 4>
Preprocessor Object: SF_SDF Version 1.1 <Build 1>
Preprocessor Object: SF_MQDBUS Version 1.1 <Build 1>
Preprocessor Object: SF_SSH Version 1.1 <Build 3>
Preprocessor Object: SF_DNP3 Version 1.1 <Build 1>
Preprocessor Object: SF_IMAP Version 1.0 <Build 1>
Preprocessor Object: SF_SIP Version 1.1 <Build 1>
Preprocessor Object: SF_GTP Version 1.1 <Build 1>
Preprocessor Object: SF_REPUTATION Version 1.1 <Build 1>
Preprocessor Object: SF_DCEP2 Version 1.0 <Build 3>
Preprocessor Object: SF_FTPTELNET Version 1.2 <Build 13>
Preprocessor Object: SF_SSLPP Version 1.1 <Build 4>
Preprocessor Object: SF_POP Version 1.0 <Build 1>

Commencing packet processing (pid=2253)
06/15-02:24:49.290635 [**] [1:100000001:1] ICMP test [**] [Priority: 0] [ICMP] 192.168.126.1 -> 192.168.126.129
06/15-02:24:49.290683 [**] [1:100000001:1] ICMP test [**] [Priority: 0] [ICMP] 192.168.126.129 -> 192.168.126.1
06/15-02:24:50.290686 [**] [1:100000001:1] ICMP test [**] [Priority: 0] [ICMP] 192.168.126.1 -> 192.168.126.129
06/15-02:24:51.310641 [**] [1:100000001:1] ICMP test [**] [Priority: 0] [ICMP] 192.168.126.129 -> 192.168.126.1
06/15-02:24:51.310688 [**] [1:100000001:1] ICMP test [**] [Priority: 0] [ICMP] 192.168.126.129 -> 192.168.126.1
06/15-02:24:52.327081 [**] [1:100000001:1] ICMP test [**] [Priority: 0] [ICMP] 192.168.126.1 -> 192.168.126.129
06/15-02:24:52.327921 [**] [1:100000001:1] ICMP test [**] [Priority: 0] [ICMP] 192.168.126.129 -> 192.168.126.1
```

Để thuận tiện cho việc theo dõi các kết quả phát hiện được từ phần mềm snort ta sẽ quan lý dữ liệu trên sql sever:

```
Activities Terminal
File Edit View Search Terminal Help
06/15-02:24:49.290635 [**] [1:100000001:1] ICMP test [**] [Priority: 0] [ICMP] 192.168.126.1 -> 192.168.126.129
06/15-02:24:49.290683 [**] [1:100000001:1] ICMP test [**] [Priority: 0] [ICMP] 192.168.126.129 -> 192.168.126.1
06/15-02:24:50.290686 [**] [1:100000001:1] ICMP test [**] [Priority: 0] [ICMP] 192.168.126.1 -> 192.168.126.129
06/15-02:24:51.310641 [**] [1:100000001:1] ICMP test [**] [Priority: 0] [ICMP] 192.168.126.129 -> 192.168.126.1
06/15-02:24:51.310688 [**] [1:100000001:1] ICMP test [**] [Priority: 0] [ICMP] 192.168.126.129 -> 192.168.126.1
06/15-02:24:52.327081 [**] [1:100000001:1] ICMP test [**] [Priority: 0] [ICMP] 192.168.126.1 -> 192.168.126.129
06/15-02:24:52.327921 [**] [1:100000001:1] ICMP test [**] [Priority: 0] [ICMP] 192.168.126.129 -> 192.168.126.1
```

6. Đánh giá phần mềm SNORT

Ưu điểm:

- Các giải pháp an ninh mạng hiện nay tương đối đa dạng.
- Tính năng an ninh mạng tốt
- Dễ quản trị
- Kiến trúc an ninh không quá phức tạp.
- Dễ cài đặt, mở rộng.
- Chi phí đầu tư không cao.

Nhược điểm:

- Có thể xảy ra trường hợp báo động giả, tức là không có dấu hiệu bất thường mà IDS vẫn báo (False Positive).
- Không thể phân tích được các lưu lượng mã hóa như SSH, IPsec, SSL, v.v...

- NIDS đòi hỏi phải luôn được cập nhật các dấu hiệu tấn công mới nhất để thực sự hoạt động hiệu quả.
- Không thể cho biết việc mạng bị tấn công có thành công hay không, để người quản trị tiến hành bảo trì hệ thống bên cạnh đó sẽ bị hạn chế nếu như có hình thức tấn công mới mà chưa kịp bổ sung vào trong bộ luật của snort.
- Việc phát hiện tấn công theo tuần tự dẫn đến việc sẽ mất nhiều thời gian để phát hiện xâm nhập.
- Một trong những hạn chế là giới hạn băng thông. Những bộ thu thập dữ liệu phải thu thập tất cả lưu lượng mạng, sắp xếp lại và phân tích chúng. Khi tốc độ mạng tăng lên thì khả năng của bộ thu thập thông tin cũng vậy. Một giải pháp là đảm bảo cho mạng được thiết kế chính xác.

IV. Ứng dụng

- Giám sát an ninh mạng: Snort đóng một vai trò quan trọng trong việc giám sát lưu lượng mạng, cung cấp khả năng hiển thị thời gian thực về các mối đe dọa tiềm ẩn. Bằng cách liên tục phân tích các gói, nó có thể phát hiện và phản hồi các hoạt động đáng ngờ, chẳng hạn như quét cổng, lan truyền phần mềm độc hại và các nỗ lực truy cập trái phép.
- Phát hiện và ngăn chặn xâm nhập: Snort vượt trội trong việc xác định và giảm thiểu các mô hình và hành vi tấn công đã biết. Nó có thể phát hiện một loạt các mối đe dọa, bao gồm SQL injection, cross-site scripting (XSS), tấn công từ chối dịch vụ phân tán (DDoS) và hơn thế nữa. Bằng cách chủ động chặn lưu lượng truy cập độc hại, Snort giúp ngăn chặn sự xâm nhập thành công.
- Phân tích và nghiên cứu phần mềm độc hại: Khả năng kiểm tra tải trọng gói tin của Snort làm cho nó trở thành một công cụ có giá trị để phân tích và nghiên cứu phần mềm độc hại. Các nhà nghiên cứu bảo mật có thể tận dụng Snort để nắm bắt và phân tích các mẫu phần mềm độc hại, hỗ trợ phát triển các biện pháp đối phó và thông tin về mối đe dọa.
- Thiết kế mạng an toàn: Bằng cách tích hợp Snort vào kiến trúc mạng, các tổ chức có thể củng cố vị thế bảo mật tổng thể của họ. Snort hoạt động như một lớp phòng thủ bổ sung, bổ sung cho Tường lửa, phần mềm chống vi-rút và các biện pháp bảo mật khác

V. Những thách thức và cơ hội

Phát hiện xâm nhập mạng có những thách thức về tính hiệu quả và tính khả thi

mà cần được đổi mới và giải quyết để tăng cường khả năng phát hiện và ngăn chặn các cuộc tấn công mạng. Sau đây là một số thách thức chính của phát hiện xâm nhập mạng:

- Số lượng lớn các sự kiện mạng: Một mạng lớn có thể tạo ra hàng ngàn sự kiện mạng mỗi ngày, và phải đối mặt với các cuộc tấn công ngày càng phức tạp. Điều này gây ra khó khăn trong việc phát hiện các cuộc tấn công mạng trong một lượng lớn các sự kiện mạng.
- Khả năng phân tích và đánh giá dữ liệu: Dữ liệu phát sinh từ các thiết bị mạng và các ứng dụng mạng có thể khó đọc và phân tích. Việc đánh giá dữ liệu sai có thể dẫn đến kết quả phát hiện sai hoặc mất bỏ các cuộc tấn công mạng.
- Các cuộc tấn công mới: Tình hình an ninh mạng thay đổi liên tục, các cuộc tấn công mới được phát minh và triển khai, đòi hỏi các công cụ phát hiện xâm nhập mạng phải được cập nhật thường xuyên và có khả năng phát hiện các cuộc tấn công mới.
- Kỹ năng chuyên môn: Phát hiện xâm nhập mạng là một lĩnh vực rất phức tạp và đòi hỏi kỹ năng chuyên môn cao để xử lý các sự cố và phân tích các cuộc tấn công mạng.
- Sự đa dạng của môi trường mạng: Môi trường mạng của mỗi tổ chức khác nhau, với các hệ thống và ứng dụng mạng khác nhau. Điều này làm tăng độ phức tạp và khó khăn cho việc triển khai các giải pháp phát hiện xâm nhập mạng hiệu quả trên mọi môi trường mạng.

Vì vậy, để đối phó với những thách thức này, các tổ chức cần phải có một chiến lược an ninh mạng tốt và triển khai các giải pháp phát hiện xâm nhập mạng thích hợp. Các giải pháp này cần được cập nhật và nâng cao liên tục để đảm bảo tính hiệu quả.

Sự phát triển của trí tuệ nhân tạo (AI) và học máy (machine learning) đang mở ra những cơ hội mới để cải thiện tính hiệu quả và độ chính xác của phát hiện xâm nhập mạng. Một số ứng dụng của AI và machine learning trong phát hiện xâm nhập mạng bao gồm:

- Tự động hóa quá trình phát hiện: AI và machine learning có thể được sử dụng để tự động hóa việc phát hiện các hoạt động độc hại trên mạng mà không cần sự can thiệp của con người. Điều này giúp tiết kiệm thời gian và tăng tính hiệu quả của phát hiện xâm nhập mạng.
- Tối ưu hóa các thuật toán phát hiện: AI và machine learning có thể được sử dụng để tối ưu hóa các thuật toán phát hiện xâm nhập mạng hiện có. Việc này giúp cải thiện tính chính xác của phát hiện và giảm thiểu số lượng các báo động sai.

- Phát hiện các mối đe dọa mới: AI và machine learning có thể giúp phát hiện các mối đe dọa mới và chưa được biết đến trước đó. Điều này đặc biệt quan trọng trong bối cảnh các mối đe dọa mạng ngày càng phức tạp và đa dạng.

Tuy nhiên, cũng có một số thách thức với việc sử dụng AI và machine learning trong phát hiện xâm nhập mạng, bao gồm:

- Đòi hỏi dữ liệu lớn: AI và machine learning đòi hỏi dữ liệu lớn để huấn luyện và cải thiện hiệu quả. Việc thu thập và xử lý dữ liệu lớn có thể là một thách thức đối với các tổ chức nhỏ và vừa.
- Độ tin cậy của thuật toán: Mặc dù AI và machine learning có thể giúp cải thiện tính chính xác của phát hiện xâm nhập mạng, nhưng cũng có thể có những sai sót và báo động sai. Do đó, việc đảm bảo độ tin cậy của thuật toán là rất quan trọng.
- Khả năng thích nghi: Các tấn công mạng liên tục thay đổi và tiến hóa, do đó phát hiện xâm nhập mạng cũng phải thích nghi và cập nhật liên tục để có thể đối phó với các mối đe dọa mới.

VI. Kết luận và hướng phát triển

Trong bối cảnh môi trường mạng ngày nay, an ninh mạng trở thành một vấn đề cấp bách không chỉ đối với các tổ chức mà còn đối với cá nhân. Phát hiện xâm nhập mạng đóng vai trò quan trọng trong việc bảo vệ hệ thống và dữ liệu khỏi các mối đe dọa và tấn công từ bên ngoài. Qua đó, chúng ta có thể rút ra một số nhận định quan trọng như sau:

1. Đa dạng và phức tạp của mối đe dọa: Các mối đe dọa xâm nhập mạng ngày càng trở nên phức tạp và đa dạng, bao gồm malware, cuộc tấn công từ mạng bên ngoài và bên trong, kỹ thuật xã hội và lỗ hổng bảo mật. Đối với mỗi loại mối đe dọa, cần có các phương pháp và công nghệ phù hợp để phát hiện và ngăn chặn.
2. Sự quan trọng của công cụ và kỹ thuật: Các công cụ phát hiện xâm nhập mạng cung cấp giám sát liên tục và phát hiện các hành vi không mong muốn hoặc bất thường trong hệ thống mạng. Sự kết hợp của các kỹ thuật như phân tích dựa trên chữ ký, hành vi và dữ liệu bất thường giúp tăng cường khả năng phát hiện và giảm số lượng cảnh báo giả.
3. Thách thức và triển vọng: Mặc dù các công nghệ phát hiện xâm nhập mạng đã phát triển mạnh mẽ, nhưng vẫn còn nhiều thách thức cần vượt qua, bao gồm sự tiến triển của các kỹ thuật tấn công mới và sự phức tạp của môi trường mạng ngày nay. Tuy nhiên, có nhiều triển vọng trong việc sử dụng

các công nghệ tiên tiến như máy học và phân tích dữ liệu lớn để cải thiện hiệu suất của hệ thống phát hiện xâm nhập mạng.

Trong tương lai, việc tiếp tục nghiên cứu và phát triển các công nghệ phát hiện xâm nhập mạng sẽ đóng vai trò quan trọng trong việc đảm bảo an ninh mạng và bảo vệ dữ liệu của cả cá nhân và tổ chức.

VII. Tài liệu tham khảo

- [1] Pavithra P S and Durgadevi P, "An Approach for Network Based Intrusion Detection System using Snort", In: Satyasai Jagannath Nanda and Rajendra Prasad Yadav (eds), *Data Science and Intelligent Computing Techniques*, SCRS, India, 2023, pp. 481-491.
- [2] Papastergiou S, Mouratidis H, Kalogeraki E-M. 2020. Cyber security incident handling. Warning and response system for the european critical information infrastructures (CyberSANE). *Communications in Computer and Information Science* 1000:476–487 DOI 10.1007/978-3-030-20257-6_41.
- [3] Xiaojin Hong, Changzhen Hu, Zhigang Wang, Guoqiang Wang and Ying Wan, "VisSRA: Visualizing Snort Rules and Alerts," In *Proc. Of Fourth International Conference on Computational Intelligence and Communication Networks (CICN)*, pp.441-444, 2012.
- [4] Suman Rani and Vikram Singh, "SNORT: An Open Network Security Tool for Intrusion Detection in Campus Network Environment," *International Journal of Computer Technology and Electronics Engineering (IJCTEE)*, 2012.
- [5] Jinsheng Xu, Jinghua Zhang, TriveniGadipalli, Xiaohong Yuan and Huiming Yu, "Learning Snort Rule By Capturing Intrusions in Live Network Traffic Replay," *Proceedings of the 15th Colloquium for Information Systems Security Education (CISSE)*, pp.145-150, 2011.