BỘ CÔNG THƯƠNG

TRƯỜNG ĐẠI HỌC CÔNG THƯƠNG TP.HCM

KHOA CÔNG NGHỆ THÔNG TIN

_____o0o_____

TIỂU LUẬN HỌC PHẦN:

HỆ THỐNG TÌM KIẾM

NGĂN NGỪA VÀ PHÁT HIỆN XÂM NHẬP

TÊN ĐỀ TÀI: SURICATA IDS/IPS

Thành phố Hồ Chí Minh, tháng 04 năm 2024

BỘ CÔNG THƯƠNG

TRƯỜNG ĐẠI HỌC CÔNG THƯƠNG TP.HCM

KHOA CÔNG NGHỆ THÔNG TIN

_____

TÊN ĐỀ TÀI: SURICATA IDS/IPS

Thành viên:

1. Nguyễn Ngọc Hiếu_2033210461_(NT)
2. Nguyễn Ngọc Lan Anh_2033210445
3. Lê Thành Kỹ Nguyên_2033210488
4. Nguyễn Thị Nhã Trân_2033216586
5. Phạm Hoàng Bảo_2033216354
6. Lại Thành Trung_2033216592
7. Phan Thị Xuân Yên_2033216613
8. Lý Tiến Đạt_2033216386
9. Lê Quốc Anh_2033216341
10. Nguyễn Đăng Khoa_2033216452
11. Huỳnh Thanh Tâm_2033210951

Giảng viên hướng dẫn:

Bùi Duy Cương

Thành phố Hồ Chí Minh, tháng 04 năm 2024

LỜI CAM ĐOAN

Chúng em xin cam đoan đề tài tiểu luận: **"SURICATA"** do cả nhóm nghiên cứu và thực hiện.

Chúng em đã kiểm tra dữ liệu theo quy định hiện hành.

Kết quả bài làm của đề tài **"SURICATA"** là trung thực và không sao chép từ bất kỳ bài tập của nhóm khác.

Các tài liệu được sử dụng trong tiểu luận có nguồn gốc, xuất xứ rõ ràng.

(Ký và ghi rõ họ tên)

LỜI CẢM ƠN

Để hoàn thành bài báo cáo này, chúng em xin chân thành cảm ơn thầy Bùi Duy Cương đã tận tình hướng dẫn cũng như nhận xét, góp ý trong suốt quá trình học tập. Với lòng biết ơn sâu sắc nhất, chúng em xin phép gửi đến thầy. Cảm ơn thầy đã truyền đạt vốn kiến thức và những thông tin quý giá cần có cho chúng em trong suốt quá trình học tập tại trường. Nhờ sự hướng dẫn, chỉ bảo tận tình của thầy đã giúp chúng em hoàn thiện đề tài một cách tốt nhất.

Chúng em xin gửi lời cảm ơn chân thành đến thầy, người đã giúp đỡ và hướng dẫn trong quá trình hoàn thành bài báo cáo này. Vì kiến thức và thời gian hạn hẹp nên bài báo cáo không tránh khỏi những sai sót.

Chúng em mong nhận được sự đóng góp ý kiến từ thầy để giúp hoàn thiện kiến thức và bổ sung thêm thông tin cần thiết cho báo cáo của nhóm em.

Chúng em xin chân thành cảm ơn!!

MỤC LỤC

MỞ ĐẦU

1. Lý do chọn đề tài

Trong bối cảnh hiện nay, an ninh mạng trở thành mối quan tâm hàng đầu khi mà các cuộc tấn công mạng ngày càng gia tăng về cả số lượng và mức độ phức tạp. Sự phát triển nhanh chóng của công nghệ thông tin không chỉ mang lại nhiều cơ hội mà còn đặt ra nhiều thách thức về bảo mật. Điều này đòi hỏi các tổ chức và doanh nghiệp phải có những giải pháp bảo mật hiệu quả để bảo vệ dữ liệu và hệ thống của mình.

Suricata là một công cụ phát hiện xâm nhập và phân tích lưu lượng mạng nguồn mở, được thiết kế để đối phó với những mối đe dọa an ninh mạng hiện đại. Việc nghiên cứu và ứng dụng Suricata không chỉ giúp nâng cao khả năng phát hiện và phản ứng nhanh chóng với các cuộc tấn công mà còn cung cấp các kỹ năng cần thiết trong việc quản lý và bảo mật hệ thống mạng.

Bên cạnh đó, việc tìm hiểu về Suricata cũng góp phần nâng cao kiến thức về an ninh mạng, một lĩnh vực ngày càng quan trọng trong thời đại số hóa. Đặc biệt, trong bối cảnh Việt Nam đang nỗ lực phát triển hạ tầng công nghệ thông tin và đảm bảo an toàn thông tin, việc nghiên cứu và ứng dụng các công cụ bảo mật như Suricata sẽ đóng góp vào quá trình hiện đại hóa và bảo vệ an ninh quốc gia.

Vì vậy, chọn Suricata làm đề tài tiểu luận không chỉ nhằm mục đích nghiên cứu học thuật mà còn hướng tới việc ứng dụng thực tiễn, góp phần giải quyết các vấn đề bảo mật mạng, nâng cao chất lượng và hiệu quả bảo vệ hệ thống thông tin trong bối cảnh công nghệ ngày càng phát triển.

CHƯƠNG 1 : GIỚI THIỆU VỀ SURICATA

1.1. Giới thiệu

Suricata là một hệ thống phát hiện xâm nhập dựa trên mã nguồn mở. Nó được phát triển bởi Open Information Security Foundation (OISF). Công cụ này được phát triển không nhằm cạnh tranh hay thay thế các công cụ hiện có, nhưng nó sẽ mang lại những ý tưởng và công nghệ mới trong lĩnh vực an ninh mạng.

Công cụ này phát hiện và ngăn chặn xâm nhập dựa trên luật để theo dõi lưu lượng mạng và cung cấp cảnh báo đến người quản trị hệ thống khi có sự kiện đáng ngờ xảy ra. Nó được thiết kế để tương thích với các thành phần an ninh mạng hiện có. Bản phát hành đầu tiên chạy trên nền tảng linux 2.6 có hỗ trợ nội tuyến (inline)và cấu hình giám sát lưu lượng thụ động có khả năng xử lý lưu lượng lên đến gigabit.

Ngoài ra Suricata còn là công cụ IDS/IPS miễn phí trong khi nó vẫn cung cấp những lựa chọn khả năng mở rộng cho các kiến trúc an ninh mạng phức tạp nhất.Là một công cụ đa luồng, Suricata cung cấp tăng tốc độ và hiệu quả trongviệc phân tích lưu lượng mạng. Ngoài việc tăng hiệu quả phần cứng (với phần cứng và card mạng giới hạn), công cụ này được xây dựng để tận dụng khả năng xử lý cao được cung cấp bởi chip CPU đa lõi mới nhất.

1.2. Lịch sử ra đời

Dự án Suricata được khởi đầu vào năm 2009 bởi một nhóm các chuyên gia về an ninh mạng nhằm tạo ra một công cụ mạnh mẽ và linh hoạt để phát hiện và ngăn chặn các cuộc tấn công mạng. Mục tiêu của Suricata là cung cấp một giải pháp mã nguồn mở, đa nền tảng và hiệu quả cho việc phát hiện các hành vi xâm nhập không mong muốn trong mạng máy tính.

Tính đến thời điểm hiện tại, Suricata vẫn tiếp tục phát triển và được sử dụng rộng rãi trong cộng đồng an ninh mạng. Nó đã trở thành một trong những công cụ

quan trọng trong công cuộc bảo vệ các hệ thống mạng khỏi các mối đe dọa từ các cuộc tấn công mạng ngày càng phức tạp và nguy hiểm.

1.3. Nhu cầu ứng dụng

Trong bối cảnh các hình thức tấn công mạng ngày càng tinh vi và phổ biến, việc bảo đảm an ninh mạng cho các hệ thống thông tin của doanh nghiệp trở nên vô cùng cấp thiết. Một trong những giải pháp hiệu quả để bảo vệ hệ thống mạng là sử dụng công cụ phát hiện và ngăn chặn xâm nhập hiện đại như Suricata.

Suricata, với các công nghệ tiên tiến và chức năng đa dạng, giúp doanh nghiệp phát hiện và ngăn chặn các cuộc tấn công từ hacker, bảo vệ hệ thống khỏi những mối đe dọa nguy hiểm. Nghiên cứu và ứng dụng Suricata không chỉ giúp nâng cao khả năng phòng vệ của hệ thống mạng mà còn góp phần phát triển các kỹ năng cần thiết trong quản lý và bảo mật hệ thống.

CHƯƠNG 2 : CÔNG NGHỆ, KIẾN TRÚC CỦA SURCATA VÀ CƠ CHẾ HOẠT ĐỘNG CỦA SURICATA

2.1. Công nghệ

2.1.1. Hệ thống phát hiện xâm nhập mạng IDS

IDS (Intrusion Detection System) là hệ thống giám sát lưu thông mạng (có thể là phần cứng hoặc phần mềm), có khả năng nhận biết những hoạt động khả nghi hay những hành động xâm nhập trái phép trên hệ thống mạng trong tiến trình tấn công, cung cấp thông tin nhận biết và đưa ra cảnh báo cho hệ thống, nhà quản trị. IDS có thể phân biệt được các cuộc tấn công từ nội bộ hay tấn công từ bên ngoài.

IDS phát hiện dựa trên các dấu hiệu đặc biệt về nguy cơ đã biết hay dựa trên so sánh lưu thông mạng hiện tại với baseline (thông số chuẩn của hệ thống có thể chấp nhận được) để tìm ra các dấu hiệu bất thường.

Một số lợi ích:

- Ảnh hưởng mạng ít: Vì nó không thay đổi hoặc chặn lưu lượng truyền tải, nó gây tác động tối thiểu đến hiệu suất mạng.
- Tầm nhìn rộng hơn: Nó có thể bắt được toàn bộ lưu lượng trên giao diện được giám sát, cung cấp một cái nhìn tổng quan về hoạt động mạng.
- Xem xét tuân thủ: Trong một số quy định hoặc chính sách bảo mật, việc tác động trực tiếp đến lưu lượng mạng có thể bị hạn chế. Chế độ bị động có thể là một lựa chọn tốt trong các kịch bản như vậy.

2.1.2. Hệ thống phát hiện xâm nhập mạng IPS:

IPS (Intrusion Prevention System) là một hệ thống có thể phát hiện và ngăn chặn sự xâm nhập từ bên ngoài vào các hệ thống máy tỉnh.

IPS là một phương pháp tiếp cận an ninh mạng bằng cách ưu tiên sử dụng các công nghệ tiên tiến để phát hiện và ngăn chặn các nỗ lực xâm nhập vào hệ thống máy

tính. IPS kiểm tra các luồng lưu lượng ra vào một hệ thống máy tính hoặc mạng máy tính nhằm mục đích vi phạm an ninh. Nếu phát hiện mối đe dọa thì nó sẽ có những hành động bảo vệ như ngăn chặn gói tin hoặc ngắt toàn bộ kết nối. IPS kiểm tra, ghi chép lại một cách chi tiết các hành động đang đăng nhập vào hệ thống và gửi cảnh báo cho hệ thống hoặc quản trị mạng. Các IPS khác nhau có phương thức khác nhau trong việc kiểm tra các luồng dữ liệu để phát hiện các mối đe dọa, xâm nhập.

Hệ thống ngăn chặn xâm nhập mạng IPS có thể được cấu hình để thực hiện các hành động như:

- Chặn lưu lượng độc hại: Suricata có thể loại bỏ các gói tin được xác định là mối đe dọa, ngăn chúng không đến đích dự kiến.
- Giới hạn tốc độ: Nó có thể giới hạn tốc độ lưu lượng từ nguồn cụ thể để ngăn chặn các cuộc tấn công từ chối dịch vụ.
- Thiết lập lại kết nối: Suricata có thể thiết lập lại các kết nối liên quan đến hoạt động đáng ngờ.

Hệ thống ngăn chặn xâm nhập mạng IPS mang lại một phương pháp bảo mật tích cực hơn, ngăn chặn tiềm năng tấn công trước khi gây hại và phản ứng nhanh hơn đối với các mối đe dọa so với chỉ dựa vào cảnh báo được tạo ra trong Hệ thống ngăn chặn xâm nhập mạng IDS.

Một thách thức đối với Hệ thống ngăn chặn xâm nhập mạng IPS là khả năng phát sinh các cảnh báo giả và ảnh hưởng đến hoạt động bình thường của mạng. Khi Suricata thực hiện các hành động như chặn lưu lượng hoặc thiết lập lại kết nối, có thể xảy ra các tác động không mong muốn đến các ứng dụng và dịch vụ. Do đó, việc cấu hình chính xác và kiểm tra kỹ lưỡng trước khi triển khai chế độ chủ động là rất quan trọng để tránh các tác động phụ không mong muốn.

2.1.3. Công nghệ giám sát an ninh mạng NSM:

Giám sát an ninh mạng (Network Security Mornitoring) là việc thu thập các thông tin trên các thành phần của hệ thống, phân tích các thông tin, dấu hiệu nhằm đánh giá và đưa ra các cảnh báo cho người quản trị hệ thống. Thành phần chính của hệ thống NSM gồm: bộ thu thập dữ liệu, bộ giải mã, bộ phân tích, bộ hành động, giao diện quản lý.

Mục tiêu chính của NSM:

- Phát hiện hoạt động mạng độc hại (Xâm nhập, quét lỗ hổng, lây nhiễm phần mềm độc hại, tấn công DoS).
- Xác đin các mối đe dọa tiềm ẩn (Lỗ hổng bảo mật, cấu hình sai, vectơ tấn công tìm ẩn).
- Ngăn chặn các cuộc tấn công mạng.
- Điều tra các sự cố an ninh mạng.

Lợi ích của việc sử dụng NSM:

- Tăng cường bảo mật mạng: NSM có thể giúp phát hiện và ngăn chặn các cuộc tấn công mạng trước khi chúng có thể gây ra thiệt hại.
- Giảm thiểu rủi ro: NSM có thể giúp giảm thiểu rủi ro vi phạm dữ liệu và các mối đe dọa an ninh mạng khác.
- Cải thiện khả năng hiển thị mạng: NSM có thể cung cấp cho tổ chức cái nhìn tổng quan về hoạt động mạng và giúp xác định các hành vi đáng ngờ.
- Tuân thủ quy định: NSM cso thể giúp tổ chức tuân thủ các yêu cầu quy định liên quan đến bảo mật mạng.

Cần lưu ý những điều sau đây khi triển khai NSM:

- Xác định rõ ràng nhu cầu và mục tiêu của tổ chức.
- Chọn hệ thống NSM phù hợp với nhu cầu và ngân sách của tổ chức.
- Triển khai hệ thống NSM một cách chính xác và cấu hình đúng cách.

- Đảm bảo bảo trì và cập nhật hệ thống NSM thường xuyên.
- Được đào tạo về cách sử dụng hệ thống NSM.

2.1.4. Sử dụng PCAP log lại thông tin của lưu lượng dữ liệu mạng

PCAP (Packet Capture Analysis) là một tính năng của Suricata cho phép người dùng thu thập và phân tích lưu lượng truy cập mạng trực tiếp từ giao diện mạng. Nó cung cấp một cách linh hoạt để thu thập dữ liệu mạng cho mục đích phân tích, gỡ lỗi và chỉ có sẵn trong Suricata Pro và Enterprise. PCAP thực hiện các chức năng lọc gói dữ liệu theo những luật của người dùng khi chúng được truyền tới ứng dụng, truyền những gói dữ liệu thô tới mạng, thu thập thông tin thống kê lưu lượng mạng. Đối với các hệ thống thuộc họ Unix ta có thư viện libpcap, còn đối với Window ta có thư viện được port từ libpcap là winpcap.

Cách thức hoạt động:

- Khi được kích hoạt, PCAP sẽ bắt đầu thu thập lưu lượng truy cập mạng từ giao diện mạng được chỉ định.
- Lưu lượng truy cập được thu thập sẽ được lưu trữ trong tệp PCAP, có thể được phân tích bằng các công cụ phân tích mạng khác nhau, chẳng hạn như Wireshark.
- PCAP cũng có thể được sử dụng để cung cấp dữ liệu đầu vào cho các quy tắc Suricata, cho phép Suricata phân tích lưu lượng truy cập mạng được thu thập theo thời gian thực.

Ưu điểm	**Nhược điểm**
- Cung cấp cách thức linh hoạt để thu thập dữ liệu mạng. - Cho phép thu thập dữ liệu từ các nguồn không được hỗ trợ bởi các phương thức thu thập dữ liệu	- Có thể tạo ra lượng dữ liệu lớn, cần được lưu trữ và xử lý cẩn thận. - Có thể ảnh hưởng đến hiệu suất mạng nếu không được cấu hình

khác của Suricata. chính xác.

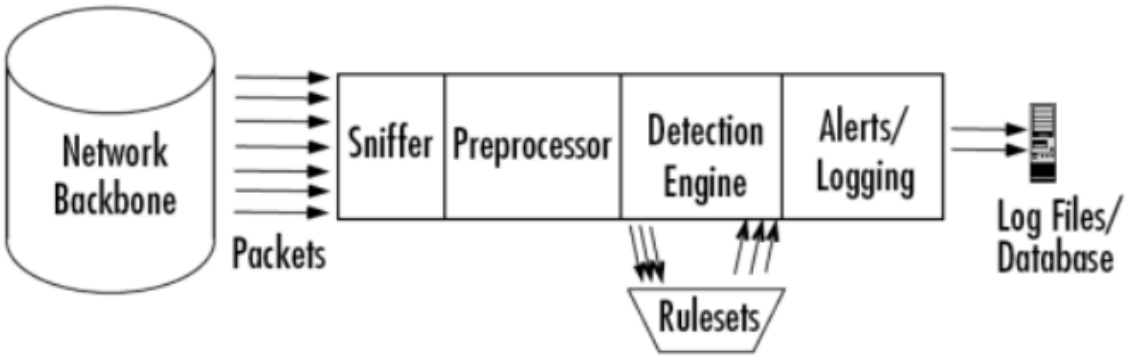
- Hỗ trợ phân tích lưu lượng truy cập mạng theo thời gian thực.

2.2. Kiến trúc

Suricata được phát triển dựa trên snort nên nó vẫn giữ nguyên kiến trúc bên trong của snort. Kiến trúc của nó có nhiều thành phần, với mỗi thành phần có một chức năng riêng.

Các thành phần chính là:

- Modul giải mã gói tin.
- Modul tiền xử lý (preprocessors).
- Modul phát hiện.
- Modul bản ghi và cảnh báo (logging and alerting system).
- Modul kết xuất thông tin.



Khi suricata hoạt động nó sẽ thực hiện lắng nghe và thu bắt tất cả các gói tin nào di chuyển qua nó. Các gói tin sau khi bị bắt được đưa và modul giải mã gói tin. Tiếp theo gói tin sẽ được đưa vào modul tiền xử lý, rồi đưa vào modul phát hiện. Tại đây, tùy theo việc có phát hiện được xâm nhập hay không mà gói tin có thể được lưu thông tiếp hay được đưa vào modul bản ghi và cảnh báo để xử lý. Khi các cảnh báo

được xác định modul kết xuất thông tin sẽ thực hiện việc đưa cảnh báo ra theo đúng định dạng mong muốn. Sau đây ta sẽ đi sâu vào nghiên cứu chi tiết hơn.

2.2.1. Module giải mã gói dữ liệu

Suricata sử dụng thư viện PCap để bắt mọi gói tin trên mạng lưu thông qua hệ thống. Mỗi gói tin sau khi được giải mã sẽ được đưa tiếp vào modul tiền xử lý.

2.2.2. Module tiền xử lý

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

Kết hợp lại các gói tin:

- Khi một lượng dữ liệu lớn được gửi đi, thông tin sẽ bị chia nhỏ thành nhiều gói tin. Khi suricata nhận được các gói tin này thì nó phải thực hiện ghép lại thành hình dạng ban đầu, từ đó mới thực hiện các công việc xử lý tiếp. Như ta đã biết khi một phiên làm việc diễn ra, sẽ có rất nhiều gói tin được trao đổi trong phiên đó. Một gói tin riêng rẽ sẽ không có trạng thái và nếu công việc phát hiện xâm nhập chỉ dựa vào gói tin đó sẽ không đem lại hiệu quả cao. Modul tiền xử lý giúp suricata 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 thực hiện dưới nhiều hình thức khác nhau. *Ví dụ*: một web server có thể chấp nhận nhiều dạng URL như URL viết dưới dạng mã hexa/unicode, URL chấp nhận cả dấu / hay \ hoặc nhiều ký tự này liên tiếp cùng lúc. Chẳng hạn ta có dấu hiệu nhận dạng :

"scripts/iiaadmin", kẻ tấn công có thể vượt qua bằng cách tùy biến các yêu cầu gửi đến web server như sau:

"scripts//iisadmin"

"scripts/examples/../iisadmin"

"scripts\iisadmin"

"scripts/.\iisadmin"

Hoặc thực hiện mã hóa các chuỗi này dưới dạng khác. Nếu suricata chi thực hiện đơn thuần việc so sánh dữ liệu nhận dạng sẽ xảy ra tình trạng bỏ sót các hành vi xâm nhập. Do vậy, một số modul tiền xử lý phải có nhiệm vụ giải mã và chỉnh sửa, sắp xếp lại các thông tin đầu vào này để thông tin khi đưa đến modul phát hiện cơ thể phát hiện được mà không bỏ sót.

Phát hiện các xâm nhập bất thường (nonrule/anormal): thường dùng để đối phó với các xâm nhập không thể hoặc rất khó phát hiện được bằng luật thông thường hoặc các dấu hiệu bất thường trong giao thức. Các modul tiền xử lý dạng này có thể phát hiện xâm nhập theo bất cứ cách nào mà ta nghĩ ra từ đó tăng thêm tính năng cho suricata. Ví dụ: một plugin tiền xử lý có nhiêm vụ thống kê thông lượng mạng tại thời điểm bình thường để rồi khi có thông lượng bất thường xảy ra nó có thể tính toán, phát hiện và đưa ra cảnh báo.

2.2.3. Module phát hiện:

Đây là modul quan trọng nhất. Nó chịu trách nhiệm phát hiện các dấu hiệu xâm nhập. Modul phát hiện sử dụng các luật được định nghĩa sẵn để so sánh với dữ liệu thu thập được từ đó xác định có xâm nhập xảy ra hay không. Rồi tiếp theo mới có thể thực hiện công việc ghi log, tạo báo cáo, kết xuất thông tin.

Một vấn đề quan trọng trong modul phát hiện là vấn đề thời gian xử lý các gói tin: IDS thường nhận được rất nhiều gói tin và bản thân nó cũng có rất nhiều luật xử lý. Vì vậy có thể mất những khoảng thời gian khác nhau cho việc xử lý các gói tin khác nhau. Và khi thông lượng qua mạng quá lớn có thể xảy ra việc bỏ sót hoặc không phản hồi đúng lúc. Khả năng xử lý của modul phát hiện dựa trên yếu tố như: số lượng các luật, tốc độ của hệ thống mạng.

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

- IP header.
- Header của tầng vận chuyển: TCP, UDP.
- Header của tầng ứng dụng: DNS header, HTTP header, …
- Phần tải của gói tin (bạn cũng có thể áp dụng các luật lên các phần dữ liệu được truyền đi trong gói tin).

Một vấn đề trong modul phát hiện là việc xử lý thế nào khi một gói tin bị phát hiện đã được đánh thứ tự ưu tiên nên một gói tin khi bị phát hiện bởi nhiều luật khác nhau, cảnh báo được đưa ra ứng với luật có mức ưu tiên cao nhất.

2.2.4. Module bản ghi và cảnh báo:

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

- Ghi log file.
- Ghi syslog: syslog là 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 này 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 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 thuật tiện.
- Gửi thông điệp SMB (server message block) tới các máy tính windows.

Ta cũng có thể tự viết module kết xuất thông tin riêng tùy thoe mục đích sử dụng.

2.3. Cơ chế hoạt động

2.3.1. Tiếp nhận gói tin (Packet Capture)

Quá trình này bắt đầu khi Suricata nhận được gói tin từ các giao diện mạng. Suricata cần phải bắt các gói tin mạng một cách hiệu quả để phân tích.

Suricata bắt các gói tin từ các giao diện mạng và đặt chúng vào một bộ đệm nội bộ. Bộ đệm này giúp đảm bảo rằng các gói tin không bị mất trong quá trình xử lý.

Gói tin sau đó được lưu tạm thời và chuẩn bị cho các bước xử lý tiếp theo như giải mã và phân tích.

Công nghệ và thư viện :

- Libpcap: Một thư viện phổ biến được sử dụng để bắt các gói tin trên hệ điều hành Unix/Linux. Nó cung cấp API để bắt gói tin và là một trong những nền tảng chính được Suricata sử dụng.
- PF_RING: Một thư viện và nhân mở rộng cho Linux cung cấp hiệu suất cao hơn trong việc bắt gói tin. PF_RING giúp tăng tốc độ bắt gói tin bằng cách sử dụng các kỹ thuật tối ưu hóa phần cứng và phần mềm.
- AF_PACKET: Giao diện cung cấp khả năng bắt gói tin hiệu quả cho các ứng dụng bảo mật mạng trên Linux.
- DNA (Direct NIC Access): Một tính năng của PF_RING giúp cải thiện tốc độ xử lý gói tin bằng cách truy cập trực tiếp vào card mạng (NIC).

2.3.2. Phân Loại và Giải Mã (Packet Decoding)

Sau khi bắt được các gói tin, Suricata phải giải mã chúng để trích xuất thông tin từ các tầng giao thức khác nhau.

Các bước giải mã:

- Ethernet Decoding:
  - Quy trình: Giải mã tiêu đề Ethernet để xác định loại giao thức (IP, ARP, VLAN, v.v.).
  - Thông tin trích xuất: Địa chỉ MAC nguồn và đích, loại giao thức (EtherType).
- IP Decoding:
  - IPv4 và IPv6: Suricata phải xử lý cả hai phiên bản của giao thức IP.
  - Quy trình: Giải mã tiêu đề IP để lấy thông tin về địa chỉ IP nguồn và đích, các trường như TTL, phiên bản IP, độ dài gói tin.
  - Thông tin trích xuất: Địa chỉ IP nguồn và đích, phiên bản IP, TTL, tổng độ dài, ID gói tin, các tùy chọn IP.
- TCP/UDP Decoding:
  - TCP:
    - Giải mã tiêu đề TCP để lấy thông tin về cổng nguồn và đích, số thứ tự gói tin, các cờ điều khiển (SYN, ACK, FIN, RST).
    - Kiểm tra các cờ điều khiển để xác định trạng thái của kết nối (kết nối mới, kết thúc kết nối, reset, v.v.).
  - UDP:
    - Giải mã tiêu đề UDP để lấy thông tin về cổng nguồn và đích, độ dài gói tin, checksum.
    - Thông tin trích xuất: Cổng nguồn và đích, số thứ tự, các cờ TCP (SYN, ACK, v.v.), độ dài gói tin.
- Application Layer Decoding:

- Quy trình: Giải mã các giao thức tầng ứng dụng như HTTP, DNS, FTP, SMTP.
- HTTP: Trích xuất thông tin về phương thức HTTP (GET, POST), URL, header HTTP.
- DNS: Trích xuất thông tin về tên miền truy vấn, loại truy vấn, phản hồi DNS.
- FTP: Trích xuất thông tin về lệnh FTP, phản hồi.
- SMTP: Trích xuất thông tin về email gửi, nhận, tiêu đề email.
- Thông tin trích xuất: Dữ liệu cụ thể của ứng dụng như URL, tên miền, lệnh ứng dụng.

2.3.3. Xử lý Luồng (Flow Management)

Flow Engine của Suricata theo dõi và quản lý trạng thái của các kết nối mạng :

- Luồng (Flow): Một luồng dữ liệu được xác định bởi bộ tứ (IP nguồn, cổng nguồn, IP đích, cổng đích).
- Trạng thái Luồng:
  - Suricata lưu trữ thông tin về mỗi luồng như số lượng gói tin, byte đã truyền, thời gian bắt đầu và kết thúc của luồng.
  - Trạng thái của kết nối TCP (SYN, SYN-ACK, ACK, FIN, RST) cũng được theo dõi để xác định các giai đoạn của kết nối.
- Phân tích Luồng:
  - Tấn công DoS/DDoS: Phát hiện các luồng dữ liệu lớn bất thường hoặc kéo dài bất thường.
  - Theo dõi chỉ số: Suricata theo dõi các chỉ số như tốc độ truyền, số lượng kết nối, để phát hiện hành vi bất thường.
  - Flow Timeout: Suricata sử dụng các giá trị timeout để xác định khi nào một luồng kết thúc nếu không có hoạt động trong một khoảng thời gian xác định.

2.3.4. Phát hiện (Detection)

Detection Engine sử dụng các quy tắc (rule set) để phân tích nội dung của gói tin và các thuộc tính luồng để phát hiện các mối đe dọa.

Các Loại Quy Tắc:

- Quy tắc Chữ ký (Signature-Based Rules):
  - So khớp mẫu: So sánh nội dung gói tin với các mẫu tấn công đã biết (chữ ký).
  - Ví dụ: Phát hiện các chuỗi ký tự đặc biệt trong HTTP request có thể là dấu hiệu của SQL injection.
- Quy tắc Hành vi (Behavior-Based Rules):
  - Phân tích hành vi: Phát hiện các hành vi bất thường dựa trên phân tích hành vi lưu lượng mạng.
  - Ví dụ: Một máy tính nội bộ gửi một lượng lớn email bất thường có thể là dấu hiệu của một cuộc tấn công spam hoặc malware.
- Quy tắc Chính sách (Policy-Based Rules):
  - Chính sách bảo mật: Áp dụng các chính sách bảo mật để phát hiện hành vi trái phép.
  - Ví dụ: Phát hiện các kết nối từ các địa chỉ IP bị cấm theo chính sách bảo mật của tổ chức.

Cơ chế Phát hiện:

- Pattern Matching:
  - Thuật toán Aho-Corasick: Sử dụng để tìm kiếm các chuỗi ký tự trong các gói tin một cách hiệu quả.
  - Thuật toán Hyperscan: Được sử dụng cho so khớp mẫu nhanh hơn và hiệu quả hơn trên các hệ thống có nhiều lõi CPU.
- Protocol Anomaly Detection:
  - Phát hiện bất thường giao thức: Xác định các hành vi không chuẩn của giao thức, như sử dụng các trường không hợp lệ hoặc không đúng định dạng.

- Ví dụ: HTTP request chứa các phương thức không hợp lệ hoặc tiêu đề không hợp lệ.

- Flow Analysis:
  - Phân tích hành vi tổng thể của các luồng dữ liệu: Phát hiện bất thường dựa trên các mẫu hành vi như số lượng kết nối, thời gian tồn tại của kết nối.
  - Ví dụ: Phát hiện các kết nối ngắn hạn nhưng tần suất cao, dấu hiệu của tấn công DDoS.

2.3.5. Ghi Nhận và Cảnh Báo (Logging and Alerting)

Kết quả phát hiện được gửi đến các module đầu ra để ghi log, tạo báo cáo và gửi cảnh báo.

Ghi nhận thông tin: Suricata ghi lại các thông tin chi tiết về sự kiện như thời gian xảy ra, địa chỉ IP nguồn và đích, cổng nguồn và đích, nội dung gói tin, loại mối đe dọa phát hiện được.

Tạo báo cáo: Các báo cáo chi tiết có thể được tạo ra để phân tích các mối đe dọa và đánh giá hiệu quả của các quy tắc bảo mật.

Các Tùy chọn Đầu ra:

- Log Files:
  - Ghi log: Ghi lại các sự kiện phát hiện vào các tệp nhật ký để lưu trữ và phân tích sau này.
  - Định dạng log: Có thể cấu hình để ghi lại thông tin chi tiết như thời gian, địa chỉ IP, cổng, loại mối đe dọa.
- JSON Output:
  - Định dạng JSON: Cung cấp dữ liệu sự kiện dưới dạng JSON, dễ dàng tích hợp với các hệ thống phân tích và quản lý bảo mật.
  - Ví dụ: Log dữ liệu HTTP, DNS, các sự kiện TLS.
- EVE JSON:

    - Định dạng EVE JSON: Một định dạng nâng cao của Suricata, cung cấp các bản ghi chi tiết về các sự kiện, hỗ trợ nhiều loại sự kiện khác nhau.
    - Thông tin bao gồm: HTTP requests/responses, DNS queries/responses, thông tin TLS (handshake, certificates).
- Syslog:
    - Giao thức Syslog: Gửi các sự kiện phát hiện đến hệ thống quản lý log thông qua giao thức Syslog.
    - Tích hợp SIEM: Hệ thống SIEM có thể phân tích và quản lý log từ Suricata, cung cấp cái nhìn toàn diện về an ninh mạng.
- Cảnh Báo Thời gian Thực:
    - Gửi cảnh báo: Gửi cảnh báo ngay lập tức qua email, webhook, hoặc các giao thức thông báo khác.
    - Ví dụ: Cảnh báo qua email khi phát hiện tấn công, thông báo qua webhook đến hệ thống quản lý sự kiện.

CHƯƠNG 3 : LUẬT TRONG SURICATA

3.1. Rule Format

Chữ ký đóng một vai trò rất quan trọng trong Suricata. Trong hầu hết các dịp mọi người đang sử dụng các quy tắc hiện có.

Tài liệu Rules Suricata này giải thích tất cả về chữ ký, làm thế nào để đọc, điều chỉnh và tạo ra chúng.

Một quy tắc/chữ ký bao gồm:

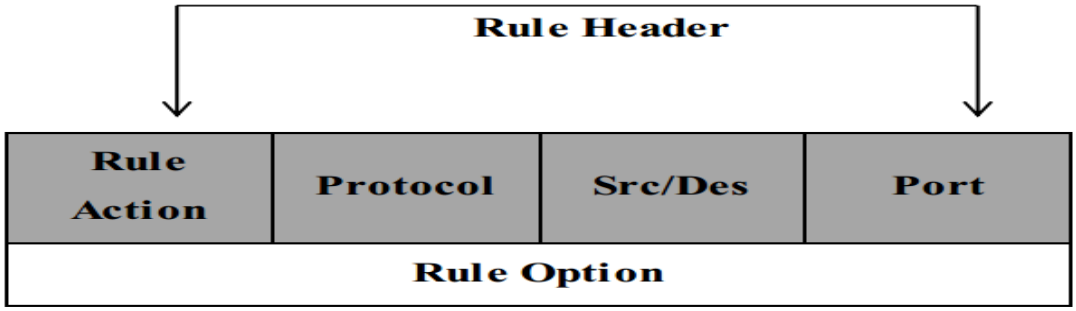
- Các Action , xác định những gì sẽ xảy ra khi các chữ ký.
- Các Header , xác định giao thức, địa chỉ IP, cổng và chỉ đạo của các quy tắc.
- Các rules option, xác định các chi tiết cụ thể của quy tắc.

Ví dụ về quy tắc như sau :

*Drop tcp $HOME_NET any -> $EXTERNAK_NET any (msg:"ET TROJAN Likely Bot Nick in IRC (USA +..)"; flow:established,to_server; flowbits:isset,is_proto_irc; content:"NICK "; pcre:"/NICK .*USA.*[0-9]{3,}/i: reference:url,doc.emergingthreat.net/2008124; classtype:trojan-activity; sid:2008124; rev:2;)*

Trong ví dụ này, Drop là Action, tcp là Header và phần trong ngoặc là Rules Option.

3.2. Rule header



Rule Header là thành phần quan trọng trong các quy tắc của Suricata, xác định các tiêu chí cơ bản mà gói tin phải đáp ứng để được xử lý theo quy tắc đó. Rule Header luôn là phần đầu tiên trong mỗi quy tắc và bao gồm các thành phần sau:

3.2.1. Rule Action:

Mục đầu tiên trong một luật đó chính là phần rule action, rule action sẽ nói cho Suricata biết phải làm gì khi thấy các gói tin phù hợp với các luật đã được quy định sẵn. Có 4 hành động mặc định trong Suricata đó là: pass (cho qua), drop (chặn gói tin), reject, alert (cảnh báo):

- **Pass**: nếu signature được so sánh trùng khớp và chỉ ra là pass thì Suricata sẽ thực hiện dừng quét gói tin và bỏ qua tất cả các luật phía sau đối với gói tin này.
- **Drop**: nếu chương trình tìm thấy một signature hợp lệ và nó chỉ ra là drop thì gói tin đó sẽ bị hủy bỏ và dừng truyền ngay lập tức, khi đó gói tin không thể đến được nơi nhận.
- **Reject**: là hành động bỏ qua gói tin, bỏ qua ở cả bên nhận và bên gửi. Suricata sẽ tạo ra một cảnh báo với gói tin này.
- **Alert**: nếu signature được so sánh là hợp lệ và có chứa một alert thì gói tin đó sẽ được xử lý giống như với một gói tin không hợp lệ. Suricata sẽ tạo ra một cảnh báo.

3.2.2. Protocol

Trường tiếp theo trong luật đó là protocol. Các giao thức mà Suricata hiện đang phân tích các hành vi bất thường đó là TLS, SSH, SMTP (tải thư điện tử qua mạng internet), IMAP (đặt sự kiểm soát email trên mail server), MSN, SMB (chia sẻ file), TCP, UDP, ICMP và IP, DNS, HTTP, HTTPS.

3.2.3. IPAddress

Mục tiếp theo của phần header đó là địa chỉ IP. Các địa chỉ này dùng để kiểm tra nơi đi và nơi đến của một gói tin. Địa chỉ ip đó có thể là địa chỉ của một máy đơn

hoặc cũng có thể là địa chỉ của một lớp mạng. Từ khóa "any" được sử dụng để định nghĩa một địa chỉ bất kỳ.

Một địa chỉ ip sẽ được viết dưới dạng ip_address/netmask. Điều này có nghĩa là nếu netmask là /24 thì lớp mạng đó là lớp mạng C, /16 là lớp mạng B hoặc /32 là chỉ một máy đơn. Ví dụ: địa chỉ 192.168.1.0/24 có nghĩa là một dải máy có địa chỉ IP từ 192.168.1.1-192.168.1.255.

Trong hai địa chỉ IP trong một luật Suricata thì sẽ có một địa chỉ IP nguồn và một địa chỉ IP đích. Việc xác định đâu là địa chỉ nguồn, đâu là địa chỉ đích phụ thuộc vào "→".

Ngoài ra toán tử phủ định có thể được áp dụng cho việc định địa chỉ IP. Có nghĩa là khi sử dụng toán tử này thì Suricata sẽ bỏ qua việc kiểm tra địa chỉ của gói tin đó. Toán tử đó là "!". Ngoài ra ta có thể định nghĩa một danh sách các địa chỉ IP bằng cách viết liên tiếp chúng cách nhau bởi một dấu ",".

Ví dụ:

Alert TCP any any → ![192.168.1.0/24, 172.16.0.0/16] 80 (msg: "Access").

3.2.4. Port

Port có thể được định nghĩa bằng nhiều cách. Với từ khóa "any" giống như địa chỉ IP để chỉ có thể sử dụng bất kỳ port nào. Gán một port cố định, ví dụ như gán kiểm tra ở port 80 http hoặc port 22 ssh. Ngoài ra ta cũng có thể sử dụng toán tử phủ định để bỏ qua một port nào đó hoặc liệt kê một dải các port.

Ví dụ:

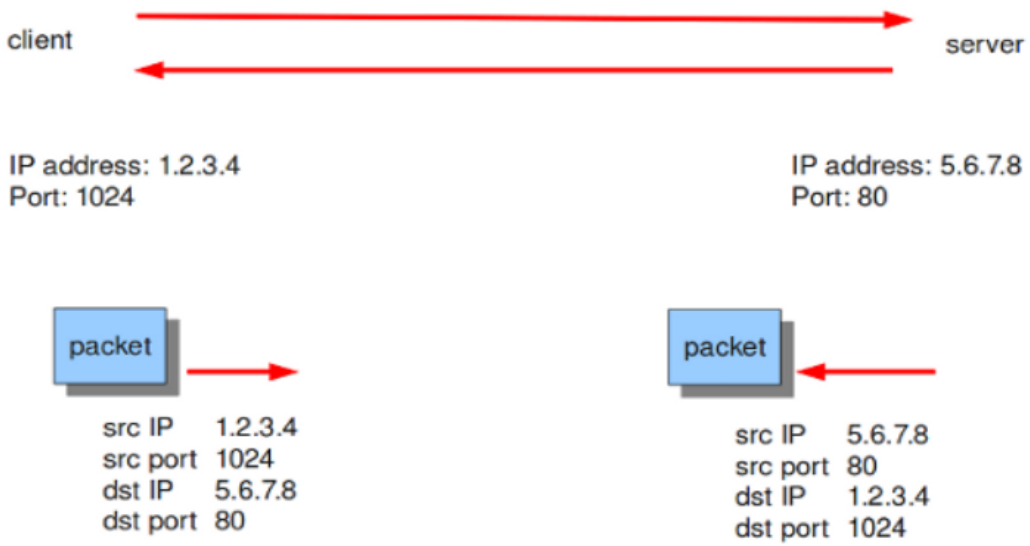
Alert UDP any any → 192.168.1.0/24 1:1024 - port bất kỳ tới dãy port từ 1 - 1024.

Alert UDP any any → 192.168.1.0/24 :6000 - port bất kỳ tới dãy port nhỏ hơn 6000.

Alert UDP any any → 192.168.1.0/24 !6000:6010 - port bất kỳ tới bất kỳ port nào, bỏ qua dãy port từ 6000 – 6010.

3.2.5. Điều hướng

Toán tử hướng "→" chỉ ra đâu là hướng nguồn, đâu là hướng đích. Phần địa chỉ IP và port ở phía bên trái của toán tử được coi như là địa chỉ nguồn và port nguồn, phần bên phải được coi như địa chỉ đích và port đích. Ngoài ra còn có toán tử "<>" Suricata sẽ xem cặp địa chỉ/port nguồn và đích là như nhau. Nghĩa là nó sẽ ghi/phân tích ở cả hai phía của cuộc hội thoại.



Hình Tổng quan về suricata.2: Có một máy khách có địa chỉ IP 1.2.3.4 và cổng 1024 và máy chủ có địa chỉ IP 5.6.7.8, nghe trên cổng 80 (thường là HTTP)

Ví dụ:

Alert tcp 1.2.3.4 1024 -> 5.6.7.8 80

3.3. Rule Option

Rule Options chính là trung tâm của việc phát hiện xâm nhập. Nội dung chứa các dấu hiệu để xác định một cuộc xâm nhập. Nó nằm ngay sau phần Rule Header và được bọc

bởi dấu ngoặc đơn "()". Tất cả các rule options sẽ được phân cách nhau bởi dấu chấm phẩy ";", phần đối số sẽ được tách ra bởi dấy hai chấm ":".

Có 4 loại rule options chính bao gồm:

- ***General***: Tùy chọn này cung cấp thông tin về luật đó nhưng không có bất cứ ảnh hưởng nào trong quá trình phát hiện.
- ***Payload***: Tùy chọn liên quan đến phần tải trong một gói tin.
- ***Non-payload***: Bao gồm các tùy chọn không liên quan đến phần tải của gói tin (header).
- ***Post-detection***: Các tùy chọn này sẽ gây ra những quy tắc cụ thể sau khi một luật đã được kích hoạt.

Các thành phần khác trong Rule :

3.3.1. General

Msg (Message): được dùng để cho biết thêm thông tin về từng signature và các cảnh báo. Phần đầu tiên sẽ cho biết tên tập tin của signature và phần này quy ước là phải viết bằng chữ in hoa. Định dạng của msg như sau:

*Msg:".....";

Sid (signature id): cho ta biết định danh riêng của mỗi signature. Định danh này được bắt đầu với số. Định dạng của sid như sau:

Sid:123;

Rev (revision): mỗi sid thường đi kèm với một rev. Rev đại diện cho các phiên bản của signature. Mỗi khi signature được sửa đổi thì số rev sẽ được tăng lên bởi người tạo ra. Định dạng của rev như sau:

Rev:123;

Reference: cung cấp cho ta địa chỉ đến được những nơi chứa các thông tin đầy đủ về signature. Các tham chiếu có thể xuất hiện nhiều lần trong một signature. Ví dụ về một tham chiếu như sau:

Refrrence: url, www.info.nl

Classtype: cung cấp thông tin về việc phân loại các lớp quy tắc và cảnh báo. Mỗi lớp bao gồm một tên ngắn gọn, một tên đầy đủ và mức độ ưu tiên.

Config classification: web- application-attack, Web Application Attack, 1 config classification: not-suspicious,Not Suspicious Traffic,3

Signature	Classification.config	Alert
web-attack	web-attack, Web Application Attack, priority:1	Web Application Attack
not-suspicious	not-suspicious, Not Suspicious Traffic, priority:3	Not Suspicious Traffic

Priority: chỉ ra mức độ ưu tiên của mỗi signature. Các giá trị ưu tiên dao động từ 1 đến 255, nhưng thường sử dụng các giá trị từ 1 -> 4. Mức ưu tiên cao nhất là 1. Những signature có mức ưu tiên cao hơn sẽ được kiểm tra trước. Định dạng như sau:

Priority:1;

Metadata: Suricata sẽ bỏ qua những gì viết sau metadata. Định dạng như sau:

metadata:.....;

3.3.2. Payload

Content: thể hiện nội dung chúng ta cần viết trong signature, nội dung này được đặt giữa 2 dấu nháy kép. Nội dung là các byte dữ liệu, có 256 giá trị khác nhau (0-255). Chúng có thể là các ký tự thường, ký tự hoa, các ký tự đặc biệt, hay là các mã hexa tương ứng với các ký tự và các mã hexa này phải được đặt giữa 2 dấu gạch dọc. Định dạng của một nội dung như sau:

*Content:"....."; *

Nocase: được dùng để chỉnh sửa nội dung thành các chữ thường, không tạo ra sự khác biệt giữa chữ hoa và chữ thường. Nocase cần được đặt sau nội dung cần chỉnh sửa.

Ví dụ:

content: "abC"; nocase;

Depth: sau từ khóa depth là một số, chỉ ra bao nhiêu byte từ đầu một payload cần được kiểm tra. Depth cần được đặt sau một nội dung.

Ví dụ: Ta có một payload : abCdefghij. Ta thực hiện kiểm tra 3 byte đầu của payload.

content: "abC"; deptth:3;

Offset: chỉ độ lệch byte trong tải trọng sẽ được kiểm tra

Ví dụ: độ lệch là 3 thì sẽ kiểm tra từ byte thứ 4 trong tải trọng.

content: "def"; offset:3;

Ví dụ:

Alert TCP 192.168.1.0/24 any -> any any (content: \"HTTP"; offset: 4; *depth: 40; msg: "HTTP matched";)*

Distance: xác định khoảng cách giữa các nội dung cần kiểm tra trong payload. Khoảng cách này có thể là một số âm.

Ví dụ:

content: “abC”; content: “efg”; distance:1;

Within: được dùng cùng với distance, để chỉ độ rộng của các byte cần kiểm tra sau một nội dung với khoảng cách cho trước đó.

Ví dụ:

content:"GET"; depth:3 content:"download"; distance:10 \within:9;

Luật có nghĩa là tìm “GET” trong 3 byte đầu tiên của trường dữ liệu, di chuyển thêm 10 byte bắt đ ầu từ “GET” và tìm khớp “download”. Tuy nhiên, “download” phải xuất hiện trong 9 byte tiếp theo.

Dsize: được dùng để tìm một payload có độ dài bất kỳ.

dsize:min<>max;

Rpc (Remote Procedure Call): là một ứng dụng cho phép một chương trình máy tính thực hiện một thủ tục nào đó trên một máy tính khác, thường được sử dụng cho quá trình liên lạc. Định dạng của rpc như sau:

rpc:<application number>,[<version number>|],[<procedure number>]|*]>;*

Replace: được dùng để thay đổi nội dung của payload, điều chỉnh lưu lượng mạng. Việc sửa đổi nội dung của payload chỉ có thể được thực hiện đối với gói dữ liệu

cá nhân. Sau khi thực hiện thay đổi nội dung xong thì Suricata sẽ thực hiện tính toán lại trường checksum.

3.3.3. Non – Payload

- **IP:**

Ttl: Được sử dụng để kiểm tra về thời gian sống, tồn tại tên mạng của một địa chỉ IP cụ thể trong phần đầu của mỗi gói tin. Giá trị time-to-live (thời gian sống), xác định thời gian tối đa mà mỗi gói tin có thể được lưu thông trên hệ thống mạng. Nếu giá trị này về 0 thì gói tin sẽ bị hủy bỏ. Thời gian sống được xác định dựa trên số hop, khi đi qua mỗi hop/router thì thời gian sống sẽ bị trừ đi 1. Cơ chế này nhằm hạn chế việc gói tin lưu thông trên mạng vô thời hạn. Định dạng của một ttl như sau:

ttl:<number>;

Ipopts: Chúng ta có thể xem và tùy chỉnh các tùy chọn cho việc thiết lập các địa chỉ IP. Việc thiết lập các tùy chọn cần được thực hiện khi bắt đầu một quy tắc. Một số tùy chọn có thể sử dụng:

IP-option	Description
rr	Record Route
eol	End of List
nop	No Op
ts	Time Stamp
sec	IP Security
esec	IP Extended Security
lsrr	Loose Source Routing
ssrr	Strict Source Routing
satid	Stream Identifier
any	any IP options are set

Một số tùy chọn của ipopts

Định dạng của một ipopts như sau:

ipopts: <name>;

Sameip: Mỗi gói tin sẽ có một địa chỉ IP nguồn và đích. Chúng ta có thể sử dụng sameip để kiểm tra xem địa chỉ IP nguồn và đích có trùng nhau hay không. Định dạng của sameip như sau:

sameip;

Ip_proto: Được dùng để giúp ta lựa chọn giao thức. Ta có thể chọn theo tên hoặc số tương ứng với từng giao thức. Có một số giao thức phổ biến sau:

1	*ICMP*	*Internet Control Message*
6	*TCP*	*Transmission Control Protocol*
17	*UDP*	*User Datagram*
47	*GRE*	*General Routing Encapsulation*
50	*ESP*	*Encap Security Payload for IPv6*
51	*AH*	*Authentication Header for Ipv6*
58	*Ipv6-ICMP*	*ICMP for Ipv6*

Định dạng của ip_proto như sau:

ip_protp:<number/name>;

Id: Được sử dụng để định danh cho các phân mảnh của gói tin được truyền đi. Khi gói tin truyền đi sẽ được phân mảnh, và các mảnh của một gói tin sẽ có ID giống nhau. Việc này giúp ích cho việc ghép lại gói tin một cách dễ dàng. Định dạng như sau:

id:<number>;

Geoip: Cho phép xác định địa chỉ nguồn, đích để gói tin lưu thông trên mạng.

Frapbits: Được dùng để kiểm tra các phân mảnh của gói tin. Nó bao gồm các cơ chế sau:

M – More Fragments	
D – Do not Fragment	
R – Reserved Bit	
+	*match on the specifiend bít, plus any others*
***	*match if any of the specified bít are set*
!	*match if the specified bít are not set*

Định dạng của một Fragbits như sau:

fragbits:[+!]<[MDR]>;*

Fragoffset: Kiểm tra sự phù hợp trên các giá trị thập phân của từng mảnh gói tin trên trường offset. Nếu muốn kiểm tra phân mảnh đầu tiên của gói tin, chúng ta cần kết hợp fragoffset 0 với các tùy chọn fragment khác. Các tùy chọn fragment như sau:

<	*match if the value is smaller than the specified value*
<	*match if the value is greater than the specified value*
!	*match if the specified value is not present*

Định dạng của fragoffset:

fragoffset:[!|<|>]<number>;

TCP:

Sed: Là một số ngẫu nhiên được tạo ra ở cả bên nhận và bên gửi gói tin để kiểm tra số thứ tự của các gói tin đến và đi. Máy khách và máy chủ sẽ tự tạo ra một số seq riêng của mình. Khi một gói tin được truyền thì số seq này sẽ tăng lên 1. Seq giúp chúng ta theo dõi được những gì diễn ra khi một dòng dữ liệu được truyền đi.

Ack: Được sử dụng để kiểm tra xem gói tin đã được nhận bởi nơi nhận hay chưa trong giao thức kết nối TCP. Số thứ tự của ACK sẽ tăng lên tương ứng với số byte dữ liệu đã được nhận thành công.

Window: Được sử dụng để kiểm tra kích thước của cửa sổ TCP. Kích thước cửa sổ TCP là một cơ chế dùng để kiểm soát các dòng dữ liệu. Cửa sổ được thiết lập bởi người nhận, nó chỉ ra số lượng byte có thể nhận để tránh tình trạng bên nhận bị tràn dữ liệu. Giá trị kích thước của cửa sổ có thể chạy từ 2 đến 65.535 byte.

ICMP:

Itype: Cung cấp cho việc xác định các loại ICMP. Các thông điệp khác nhau sẽ được phân biệt bởi các tên khác nhau hay các giá trị khác nhau.

Định dạng của itype như sau:

itype:min<>max; *itype:[<|>]<number>;*

Type	Name	Reference
0	Echo Reply	[RFC792]
3	Destination Unreachable	[RFC792]
4	Source Quench	[RFC792]
5	Redirect	[RFC792]
6	Alternate Host Address	[JBP]
7	Unassigned	[JBP]
8	Echo	[RFC792]
9	Router Advertisement	[RFC1256]
10	Router Selection	[RFC1256]
11	Time Exceeded	[RFC792]
12	Parameter Problem	[RFC792]
13	Timestamp	[RFC792]
14	Timestamp Reply	[RFC792]
15	Information Request	[RFC792]
16	Information Reply	[RFC792]
17	Address Mask Request	[RFC950]
18	Address Mask Reply	[RFC950]
19	Reserved (for Security)	[Solo]
20-29	Reserved (for Robustness Experiment)	[ZSu]
30	Traceroute	[RFC1393]
31	Datagram Conversion Error	[RFC1475]
32	Mobile Host Redirect	[David Johnson]
37	Domain Name Request	[RFC1788]
38	Domain Name Reply	[RFC1788]
39	SKIP	[Markson]
40	Photuris	[RFC2521]

Bảng Type của ICMP Header

Iclode: Cho phép xác định mã của từng ICMP để làm rõ hơn cho từng gói tin ICMP. Định dạng của icode như sau:

icode:min<>max;

icode:[<|>]<number>;

Icmp_id: Mỗi gói tin ICMP có một giá trị ID khi chúng được gửi. Tại thời điểm đó, người nhận sẽ trả lại tin nhắn với cùng một giá trị ID để người gửi sẽ nhận ra và kết nối nó đúng với yêu cầu ICMP đã gửi trước đó. Định dạng của một icmp_id như sau:

icmp_id:<number>;

Icmp_sed: Được sử dụng để kiểm tra số thứ tự của ICMP. Định dạng của icmp_seq như sau:

icmp_seq:<number>;

FLOW:

Flowbits: Gồm 2 phần, phần đầu mô tả các hành động được thực hiện, phần thứ 2 là tên của flowbit. Các hành động của flowbit:

flowbits: set, name	Được dùng để thiết lập các điều kiện/ tên cho các flow.
flowbits:isset, name	Có thể được sử dụng trong các luật để đảm bảo răgf sẽ tạo ra một cảnh báo khi các luật là phù hợp và các điều kiện sẽ được thiết lập trong flow.
flowbits:toggle, name	Dùng để đảo ngược các thiết lập hiện tại.
flowbits:unset, name	Được dùng để bỏ các thiết lập về điều kiện trong luật.
flowbits:isnotset, name	Được sử dụng dể đảm bảo rằng sẽ tạo ra một cảnh báo khi các luật là phù hợp và các điều kiện sẽ không được thiết lập trong flow.

Flow: Có thể được sử dụng để kết nối các thư mục chứa các flow lại với nhau. Các flow có thể được đi từ hoặc đến từ Client/Server và các flow này có thể ở trạng thái được thiết lập hoặc không. Việc kết nối các flow có thể xảy ra các trường hợp sau:

to_client	established/ stateless
from_client	established/ stateless

to_server established/ stateless from_serverestablished/ stateless

HTTP:

Có các sửa đổi nội dung bổ sung có thể cung cấp các khả năng dành riêng cho giao thức ở lớp ứng dụng. Thông tin thêm có thể được tìm thấy tại *Payload Keywords* Các từ khóa này đảm bảo chữ ký chỉ kiểm tra các phần cụ thể của lưu lượng mạng. Chẳng hạn, để kiểm tra cụ thể về URI yêu cầu, cookie hoặc cơ quan phản hồi hoặc yêu cầu HTTP, v.v.

Tất cả các từ khóa HTTP là sửa đổi. Lưu ý sự khác biệt giữa sửa đổi nội dung và bộ đệm dính. Xem *từ khóa sửa đổi* để biết thêm thông tin.

content modifiers nhìn lại quy tắc, ví dụ:

```
alert http any any -> any any (content:"index.php"; http_uri; sid:1;)
```

sticky buffers được đặt đầu tiên và tất cả các từ khóa theo sau nó áp dụng cho bộ đệm đó, ví dụ:

```
alert http any any -> any any (http_response_line; content:"403 Forbidden"; sid:1;
```

Các từ khóa phản hồi

Keyword	Sticky or Modifier	Direction
http_stat_msg	Modifier	Response
http_stat_code	Modifier	Response
http_response_line	Sticky Buffer	Response
http_header	Modifier	Both
http_raw_header	Modifier	Both
http_cookie	Modifier	Both
http_server_body	Modifier	Response
http.server	Modifier	Response
http.location	Modifier	Response
file_data	Sticky Buffer	Response
http_content_type	Sticky Buffer	Both
http_content_len	Sticky Buffer	Both
http_start	Sticky Buffer	Both
http_protocol	Sticky Buffer	Both
http_header_names	Sticky Buffer	Both

Các từ khóa yêu cầu:

Keyword	Sticky or Modifier	Direction
http_uri	Modifier	Request
http_raw_uri	Modifier	Request
http_method	Modifier	Request
http_request_line	Sticky Buffer	Request
http_client_body	Modifier	Request
http_header	Modifier	Both
http_raw_header	Modifier	Both
http_cookie	Modifier	Both
http_user_agent	Modifier	Request
http_host	Modifier	Request
http_raw_host	Modifier	Request
http_accept	Sticky Buffer	Request
http_accept_lang	Sticky Buffer	Request
http_accept_enc	Sticky Buffer	Request
http_referer	Sticky Buffer	Request
http_connection	Sticky Buffer	Request
http_content_type	Sticky Buffer	Both
http_content_len	Sticky Buffer	Both
http_start	Sticky Buffer	Both
http_protocol	Sticky Buffer	Both
http_header_names	Sticky Buffer	Both

CHƯƠNG 4 : TÍCH HỢP TỐI ƯU HÓA VÀ HIỆU SUẤT:

Suricata có thể tích hợp với các hệ thống khác thông qua nhiều cách khác nhau:

- Giao diện dòng lệnh
- Giao diện API
- Giao diện Syslog
- Giao diện SNMP

4.1. Tích hợp và tương thích với hệ thống khác

4.1.1. Hệ thống quản lý thông tin bảo mật (SIEM):

Suricata phân tích lưu lượng mạng và phát hiện các sự kiện bảo mật như xâm nhập và tấn công. Khi phát hiện các sự kiện này, Suricata sẽ tạo ra các bản ghi (log) chi tiết về sự kiện, bao gồm thông tin về nguồn, đích, loại tấn công, và các dữ liệu liên quan khác. Các log này có thể được xuất ra dưới nhiều định dạng khác nhau như JSON, phù hợp với yêu cầu của các hệ thống SIEM. Suricata sau đó gửi các log này tới SIEM thông qua các phương thức như Syslog, file, hoặc giao thức mạng. Hệ thống SIEM tiếp nhận và phân tích dữ liệu từ Suricata, kết hợp với các nguồn dữ liệu khác để cung cấp cái nhìn toàn diện về tình hình an ninh mạng của tổ chức.

Ví dụ:

- Splunk: Sử dụng ứng dụng Suricata Universal Forwarder để chuyển tiếp cảnh báo đến Splunk.
- ELK Stack: Sử dụng Logstash để thu thập và chuyển đổi cảnh báo Suricata sang định dạng Elasticsearch.

Graylog: Sử dụng GELF để gửi cảnh báo Suricata đến Graylog.

4.1.2. Hệ thống quản lý sự cố bảo mật (SOAR):

Suricata giám sát lưu lượng mạng và phát hiện các sự kiện bảo mật như xâm nhập và tấn công. Khi phát hiện các sự kiện này, Suricata tạo ra các bản ghi (log) chi

tiết, chứa thông tin quan trọng về sự kiện như nguồn, đích, loại tấn công và các dữ liệu liên quan. Các log này được xuất ra dưới định dạng phù hợp và gửi tới hệ thống SOAR thông qua các phương thức như API, Syslog, hoặc file.

Hệ thống SOAR tiếp nhận các log từ Suricata và tự động hóa quá trình phân tích, quản lý và phản ứng với các sự cố bảo mật. SOAR sử dụng các playbooks (kịch bản tự động) để xử lý các sự cố dựa trên thông tin nhận được từ Suricata. Các playbooks này có thể bao gồm việc tạo cảnh báo, gửi thông báo, cô lập thiết bị bị ảnh hưởng, hoặc khởi động các biện pháp khắc phục.

SOAR có thể sử dụng thông tin trong cảnh báo Suricata để kích hoạt các hành động như:

- Chặn các địa chỉ IP nguy hiểm trên tường lửa.
- Cách ly các thiết bị bị nhiễm phần mềm độc hại.
- Gửi thông báo cho đội ngũ bảo mật.

Ví dụ:

- Demisto: Sử dụng tích hợp Suricata tích hợp sẵn trong Demisto.
- Palo Alto Networks Cortex XSOAR: Sử dụng playbook Suricata trong Cortex XSOAR.
- IBM Resilient: Sử dụng Resilient SOAR connector for Suricata.

4.1.3. Hệ thống phân tích lưu lượng mạng (NTA):

Suricata, với khả năng giám sát và phân tích lưu lượng mạng, đóng vai trò như một "điểm nhìn" chi tiết về các hoạt động trên mạng. Khi phát hiện các sự kiện bảo mật như xâm nhập hay tấn công, Suricata tạo ra các log chi tiết về sự kiện, bao gồm thông tin về nguồn, đích, loại tấn công và dữ liệu liên quan. Những thông tin này rất quan trọng để hiểu rõ về mối đe dọa mạng đang diễn ra.

Hệ thống phân tích lưu lượng mạng (NTA) tiếp nhận dữ liệu từ Suricata và tiến hành phân tích sâu hơn. Nó xác định các mẫu hoạt động không bình thường, phân tích

thông tin liên kết giữa các sự kiện, máy chủ, địa chỉ IP và ứng dụng mạng. Qua đó, NTA có khả năng phát hiện các mối đe dọa tiềm ẩn và các hành vi độc hại trên mạng.

NTA có thể sử dụng dữ liệu lưu lượng mạng Suricata để:

- Phát hiện các cuộc tấn công DoS.
- Phát hiện các botnet.
- Phát hiện các hoạt động bất thường trên mạng.

Ví dụ:

- Wireshark: Sử dụng Wireshark để phân tích trực tiếp dữ liệu lưu lượng mạng Suricata.
- nTop: Sử dụng nTop để phân tích dữ liệu lưu lượng mạng Suricata theo thời gian thực.
- FlowJo: Sử dụng FlowJo để phân tích dữ liệu lưu lượng mạng Suricata theo lịch sử.

4.1.4. Hệ thống tường lửa:

Suricata cung cấp khả năng phát hiện các sự kiện an ninh mạng bằng cách giám sát và phân tích lưu lượng mạng. Khi phát hiện các hoạt động đáng ngờ hoặc tiềm ẩn mối đe dọa, Suricata tạo ra các log chi tiết về sự kiện, bao gồm thông tin về nguồn, đích, loại tấn công và dữ liệu liên quan.

Hệ thống tường lửa, trong khi đó, hoạt động như một bức tường bảo vệ giữa mạng nội bộ và mạng bên ngoài. Nó kiểm soát lưu lượng mạng bằng cách áp dụng các quy tắc và chính sách an ninh để cho phép hoặc từ chối các kết nối và gói tin.

Khi tích hợp với Suricata, hệ thống tường lửa có thể sử dụng các thông tin từ Suricata để làm nền tảng cho quyết định lọc lưu lượng mạng. Bằng cách này, nó có thể áp dụng các biện pháp an ninh phản ứng nhanh chóng dựa trên các mối đe dọa được phát hiện bởi Suricata.

Ví dụ:

- iptables: Sử dụng iptables rules để ngăn chặn các địa chỉ IP được Suricata xác định là nguy hiểm.

- pf: Sử dụng pf rules để chặn các địa chỉ IP và cổng được Suricata xác định là nguy hiểm.

4.1.5. Kết hợp với Honeypots và Hệ thống giả mạo (Deception_Technology):

Suricata có thể được sử dụng để phát hiện các cuộc tấn công và hoạt động độc hại và sau đó chuyển hướng chúng đến các honeypots hoặc hệ thống giả mạo để tiếp tục quan sát và phân tích.

Kết hợp Suricata với honeypots và deception technology giúp tăng cường khả năng phát hiện và phản ứng đối với các cuộc tấn công, cũng như thu thập thông tin bổ sung về các mối đe dọa mạng.

4.1.6. Tương tác với Proxy Server và Load Balancers:

Suricata có thể tương tác với các Proxy Server như Squid hoặc các Load Balancers như HAProxy để phát hiện và ngăn chặn các hoạt động độc hại trước khi chúng đạt đến người dùng cuối.

Tương tác với Proxy Server và Load Balancers giúp Suricata kiểm soát và lọc lưu lượng mạng vào và ra khỏi mạng một cách hiệu quả.

4.2. Tối ưu hóa hiệu suất

Việc này có thể giúp nâng cao hiệu suất và khả năng bảo mật mạng của bạn. Dưới đây là một số phương pháp hiệu quả:

4.2.1. Cấu hình Suricata phù hợp:

- Chọn bộ quy tắc phù hợp:
  - Sử dụng bộ quy tắc được thiết kế cho nhu cầu cụ thể của bạn.
  - Loại bỏ các quy tắc không cần thiết để giảm tải CPU.
  - Cập nhật bộ quy tắc thường xuyên để bảo vệ khỏi các mối đe dọa mới.
- Tùy chỉnh quy tắc:
  - Sử dụng các tùy chọn nâng cao trong quy tắc để tối ưu hóa hiệu suất.

Ví dụ: sử dụng --rule-priority để ưu tiên các quy tắc quan trọng.

- Điều chỉnh cài đặt Suricata:
    - Tùy chỉnh các cài đặt như --nfqueue-buffer-size và --pcap-ring-buffer-size để tối ưu hóa việc sử dụng bộ nhớ.
    - Sử dụng --cpu-affinity để gán các luồng Suricata cho các CPU cụ thể.

4.2.2. Sử dụng phần cứng phù hợp:

- CPU: Sử dụng CPU có nhiều lõi và xung nhịp cao để xử lý lưu lượng mạng hiệu quả.
- Bộ nhớ: Sử dụng đủ bộ nhớ RAM để lưu trữ bộ quy tắc và dữ liệu lưu lượng mạng.
- Card mạng: Sử dụng card mạng tốc độ cao để tránh tắc nghẽn lưu lượng.

4.2.3. Giám sát và điều chỉnh:

- Theo dõi hiệu suất Suricata: Sử dụng các công cụ như --stats và --logtostderr để theo dõi hiệu suất CPU, bộ nhớ và lưu lượng mạng.
- Điều chỉnh cài đặt: Dựa trên dữ liệu giám sát, điều chỉnh cài đặt Suricata để tối ưu hóa hiệu suất.

4.2.4. Sử dụng các kỹ thuật nâng cao:

- Cài đặt Suricata trong chế độ multi-process: Chia nhỏ các tác vụ của Suricata thành nhiều tiến trình để tận dụng tối đa phần cứng.
- Sử dụng các công cụ tăng tốc phần cứng: Sử dụng các card mạng chuyên dụng hoặc các công cụ tăng tốc phần mềm để tăng hiệu suất xử lý lưu lượng mạng.

4.2.5. Cập nhật và duyệt chương trình thường xuyên:

- Cập Nhật Quy Tắc và Ruleset: Theo dõi và cập nhật các quy tắc và ruleset để đảm bảo rằng Suricata có thể nhận biết các mối đe dọa mới.
- Cập Nhật Phần Mềm: Duyt chương trình Suricata thường xuyên để áp dụng các bản vá và cập nhật mới nhất, cũng như tối ưu hóa hiệu suất.

4.2.6. Kiểm tra và đánh giá:

- Thực Hiện Kiểm Tra Hiệu Suất Định Kỳ: Thực hiện kiểm tra hiệu suất định kỳ để xác định các điểm yếu và cải thiện hiệu suất.
- Phân Tích Log và Cảnh Báo: Liên tục phân tích log và cảnh báo để đảm bảo rằng Suricata hoạt động hiệu quả và có thể phát hiện các mối đe dọa mạng.

4.3. So sánh Suricata và Snort

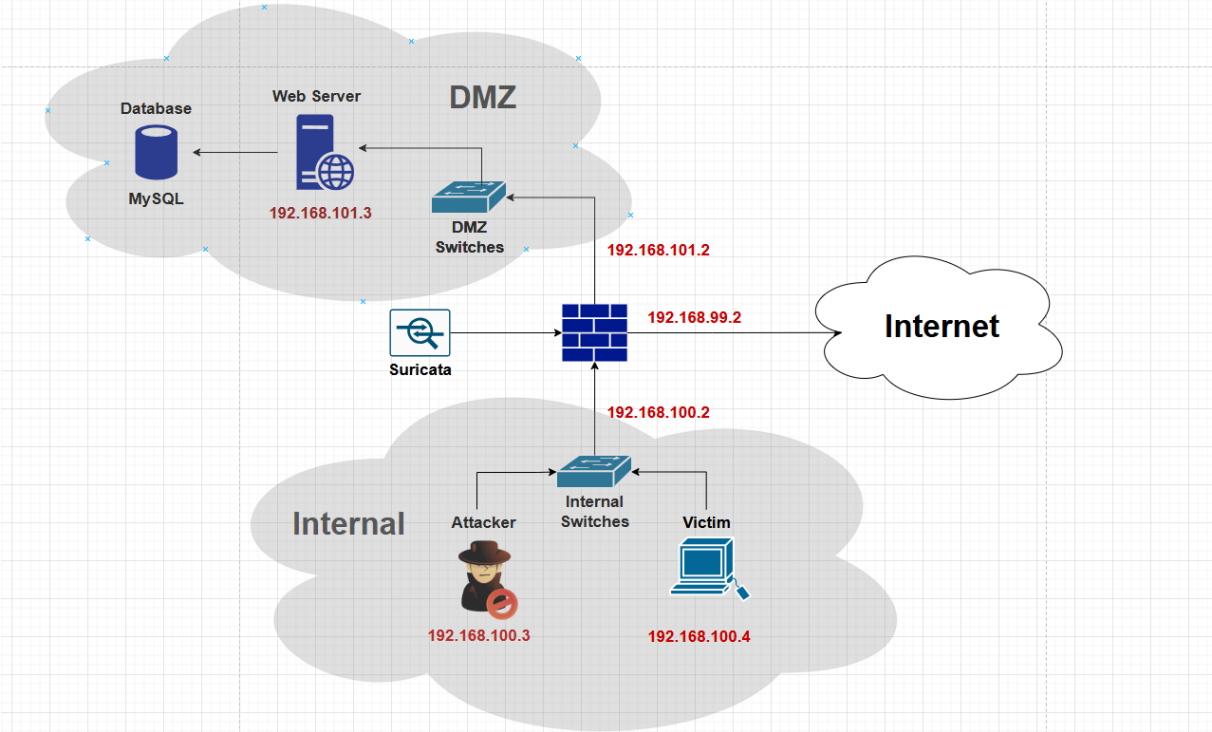
- Giống nhau:
  - Đều là NIDS mã nguồn mở và miễn phí.
  - Đều sử dụng kỹ thuật phát hiện dựa trên chữ ký từ các bộ quy tắc (VRT, SO, Emerging Threats rules).
  - Có thể xảy ra trường hợp báo động giả.
  - Không thể phân tích được các lưu lượng đã được mã hóa.
  - Đòi hỏi phải luôn được cập nhật các dấu hiệu tấn công mới nhất để thực sự hoạt động hiệu quả.
- Khác nhau:

Tiêu chí	**Snort**	**Suricata**
Luồng xử lý	Đơn luồng	Đa luồng
Sử dụng tài nguyên của hệ thống	Trung bình	Nhiều
Tỷ lệ bỏ qua gói tin khi lưu lượng mạng lớn	Cao	Thấp
Tính năng kiểm tra trạng thái	Căn cứ vào việc phát hiện các rules và ngưỡng để theo dõi số lần một rule được kích hoạt	Sử dụng session variable tạo bộ đếm

Tập luật	Sử dụng các luật từ VRT, Emerging Threat cũng như các tập luật được biết bởi cộng đồng	Sử dụng các luật từ VRT, Emerging Threat. Ngoài ra còn hỗ trợ các luật được viết bằng Lua script
Kết quả đầu ra	Có thể ghi kết quả đầu ra dưới dạng syslog, tcpdump, csv hoặc unified2	Cho phép ghi kết quả đầu ra dưới dạng EVE JSON và syslog. Ngoài ra còn hỗ trợ dung Lua script để lấy kết quả đầu ra

CHƯƠNG 5 : KIỂM THỬ HỆ THỐNG SURICATA

Mô hình / Mô tả mô hình



Máy	Hệ điều hành	Địa chỉ IP	Interface
Firewall	Pfsense	192.168.99.2	VMNet3
		192.168.100.2	VMNet4
		192.168.101.2	VMNet5
Attacker	Kali Linux	192.168.100.3	VMNet4
Web Server	Windows Server 2016	192.168.101.3	VMNet5
Victim	Windows 7	192.168.100.4	VMNet4

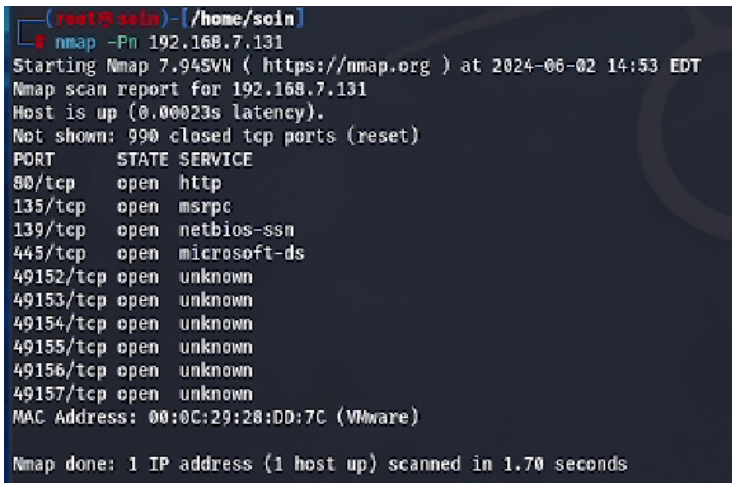
5.1. Erternal Blue (MS17 – 010)

5.1.1. Kịch bản :

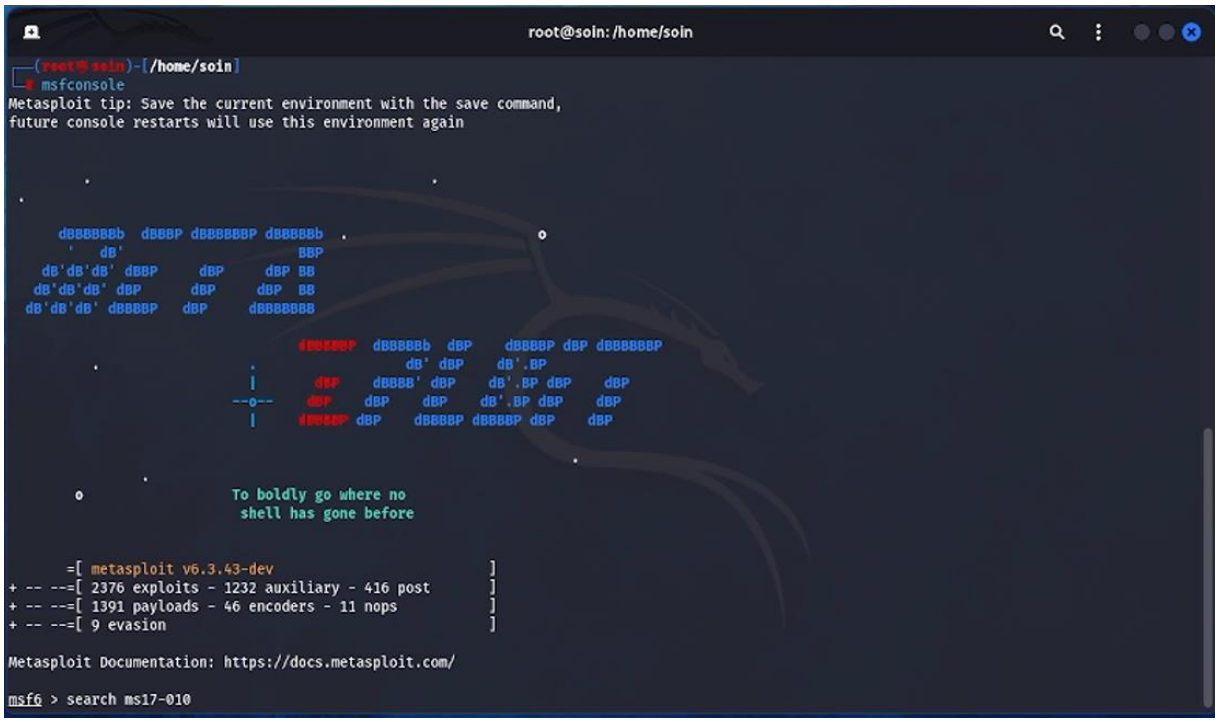
- Máy Ubuntu, Kali Linux và Windows 7 được kết nối trong cùng mạng LAN. Máy Ubuntu cài đặt suricata IDS để phân tích lưu lượng mạng, máy Windows 7 có lỗ hổng MS17-010, một lỗ hổng bảo mật nghiêm trọng có thể bị khai thác để thực hiện tấn công từ xa, kali linux đóng vai trò kẻ tấn công sử dụng Metasploit trên máy Kali Linux để thực hiện tấn công EternalBlue vào máy Windows 7.
- Công cụ tấn công : Metasploit
- Mục tiêu : chiếm quyền điều khiển máy Windows 7 dựa vào lỗ hổng dịch vụ Eternal Blue.

5.1.2. Thực hiện

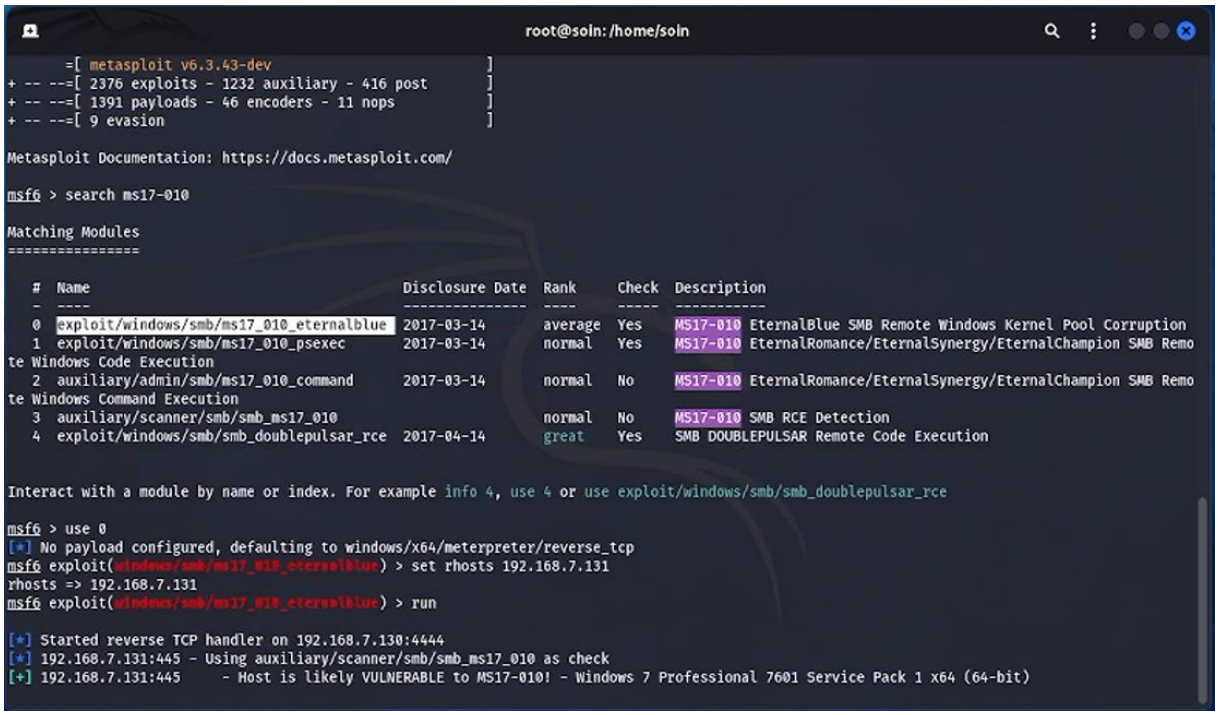
- Tóm tắt các bước thực hiện :
  - Quét Victim bằng nmap.
  - Sử dụng Metasploit để tấn công.
  - Tạo Payload và tiến hành khai thác.
  - Xâm nhập thành công vào Victim.
  - Bật Suricata và cấu hình rule.
  - Kiểm tra Suricata có đang hoạt động không.
  - Cấu hình rule phát hiện lỗ hổng MS17-010.
  - Thực hiện lại tấn công và xem kết quả.
- Bước 1 : Sử dụng Nmap để quét các cổng đang mở trên Victim



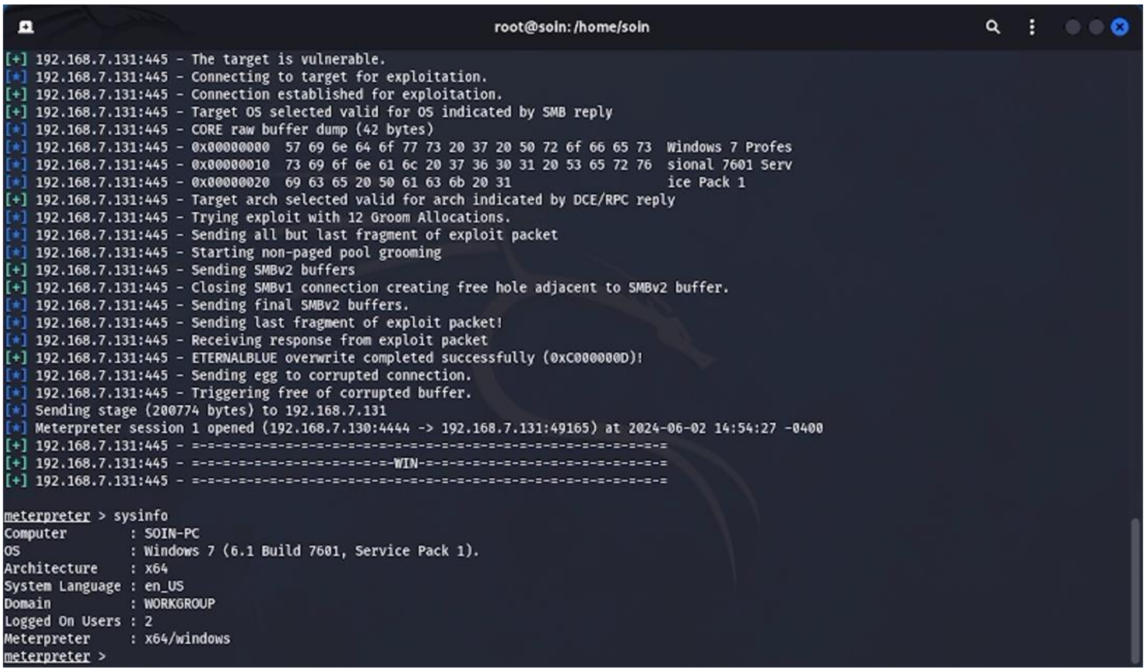
- Bước 2 : Sử dụng lệnh Msfconsole để khởi động Metasploit



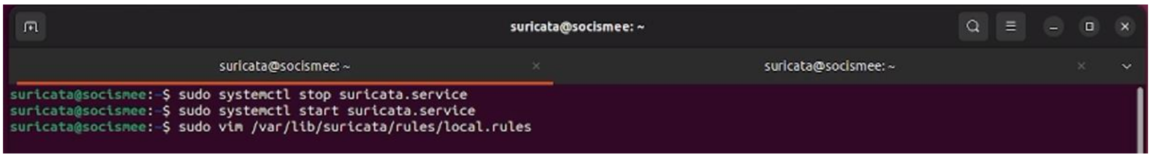
- Bước 3 : Tìm kiếm module phù hợp để sử dụng và đặt các options



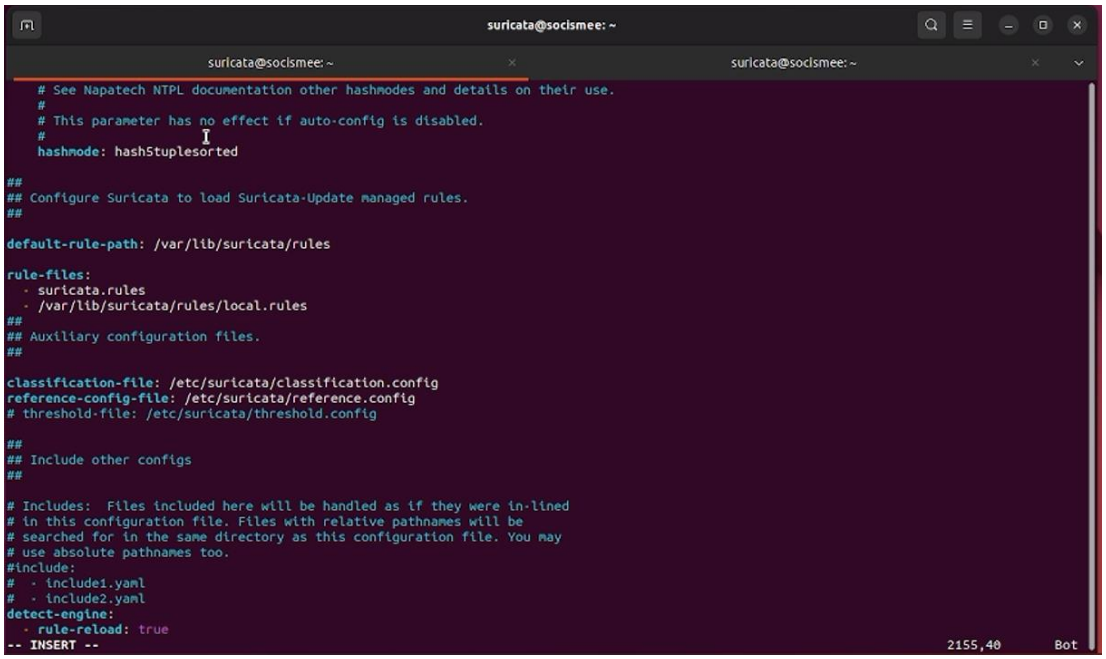
- Bước 4 : Tấn công thành công vào máy Vitcim



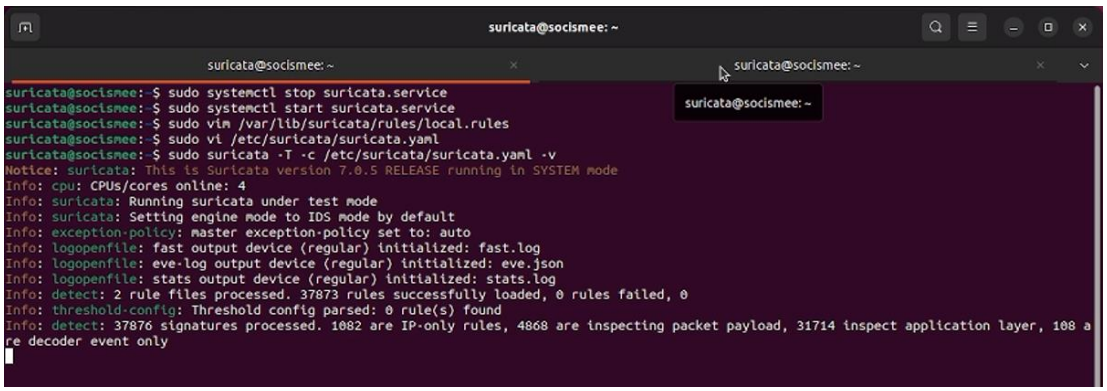
- Bước 5 : Bật Suricata và tạo 1 file Rule mới
  - Thêm Rule dùng để cảnh báo gói tin icmp trong đường mạng:
  - Alert icmp any any -> any any (msg: “ICPM Ping”; sid:1; rev:1;)



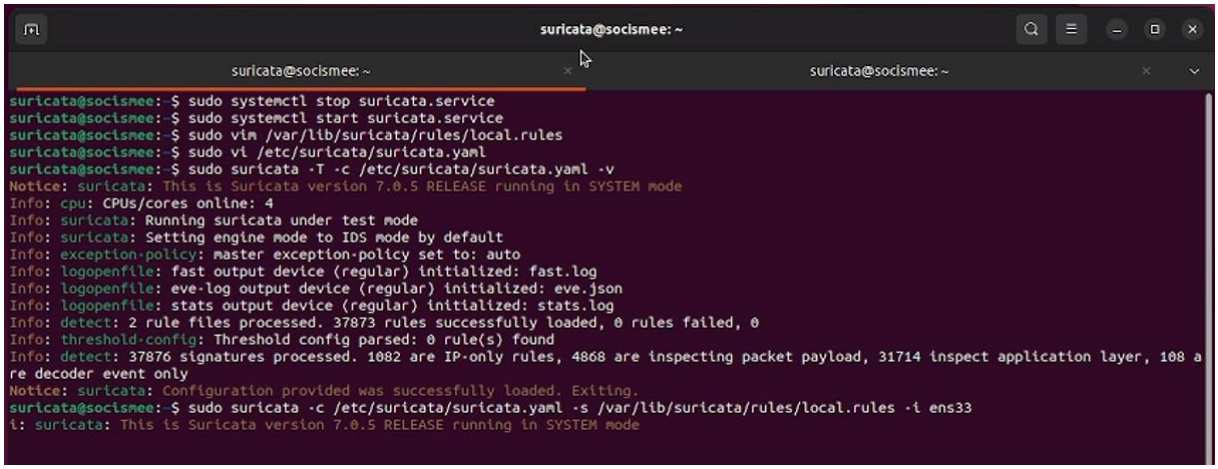
- Bước 6 : Vào file Suricata.yaml để thêm đường dẫn của file Rule vừa rồi



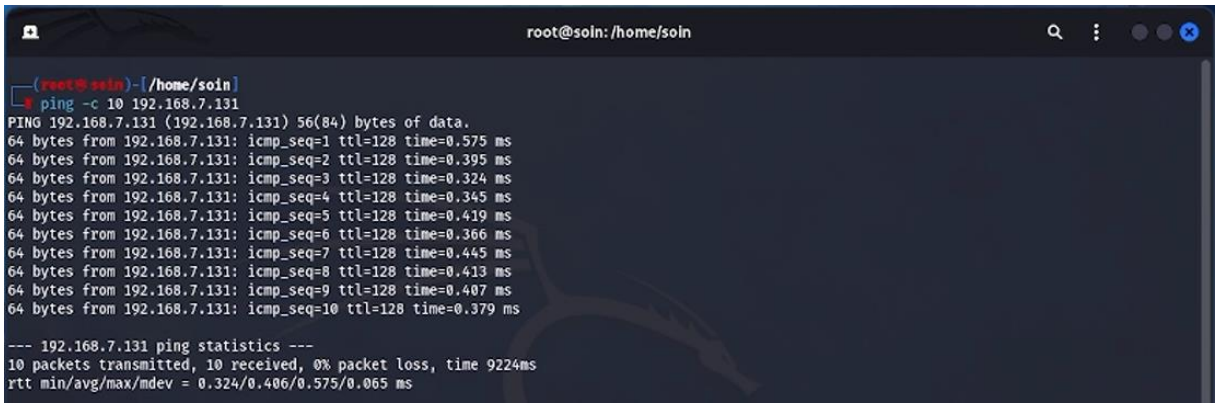
- Bước 7 : Cập nhật lại file cấu hình của Suricata



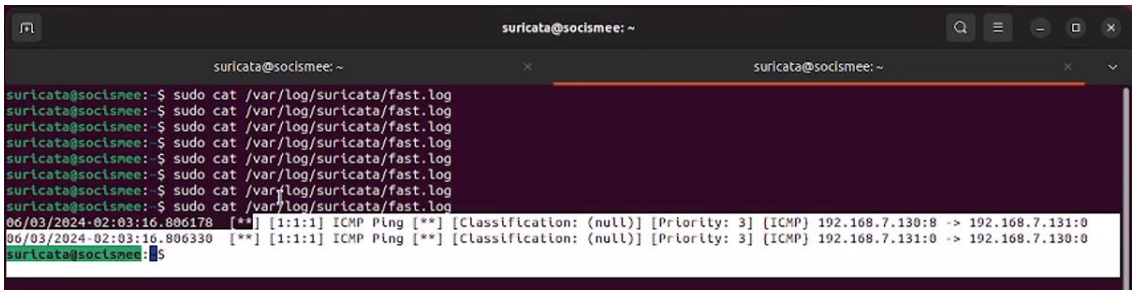
- Bước 8 : Tiến hành cho Suricata lắng nghe lưu lượng mạng trên Interfaces ens33



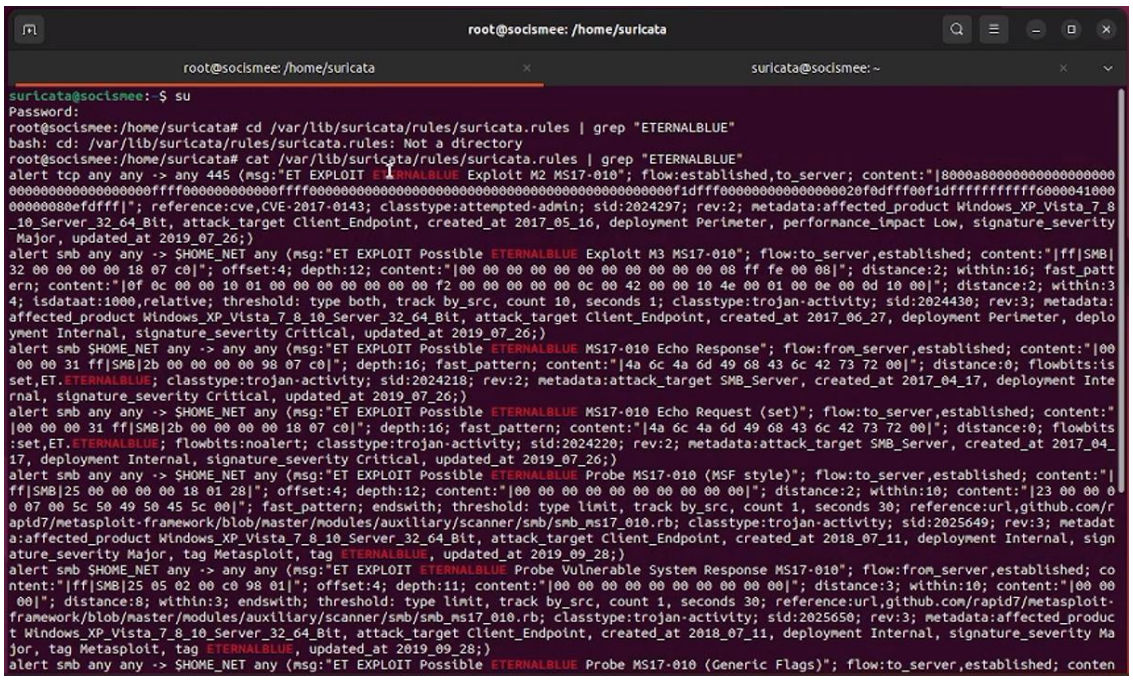
- Bước 9 : Ping lại từ máy Kali qua Victim để kiểm tra kết nối và thử rule



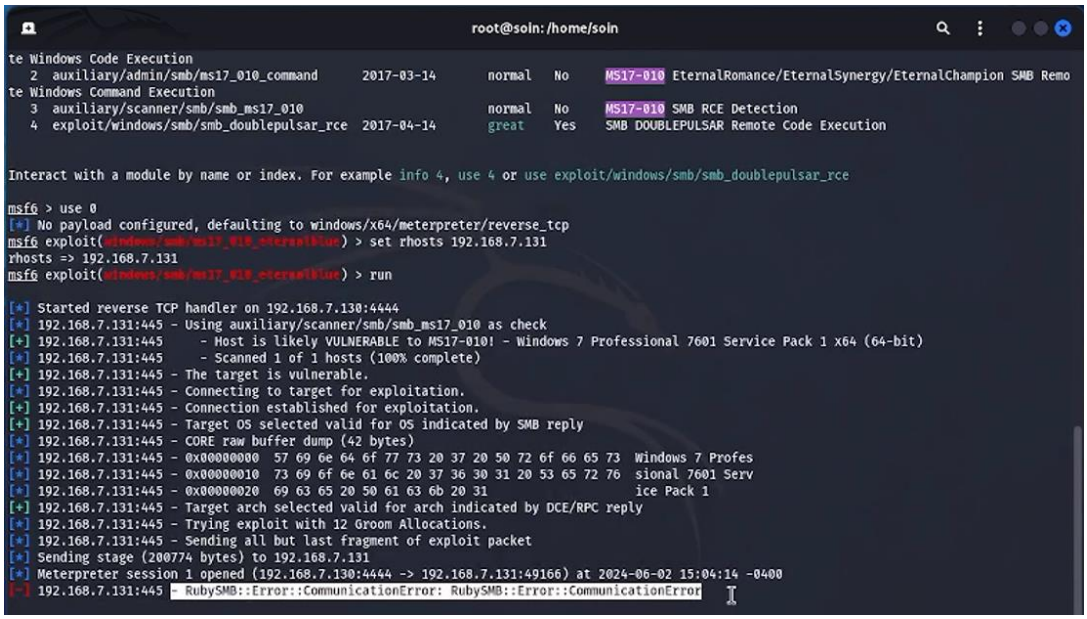
- Bước 10 : Xem log thấy Suricata đã hoạt động và phát ra cảnh báo



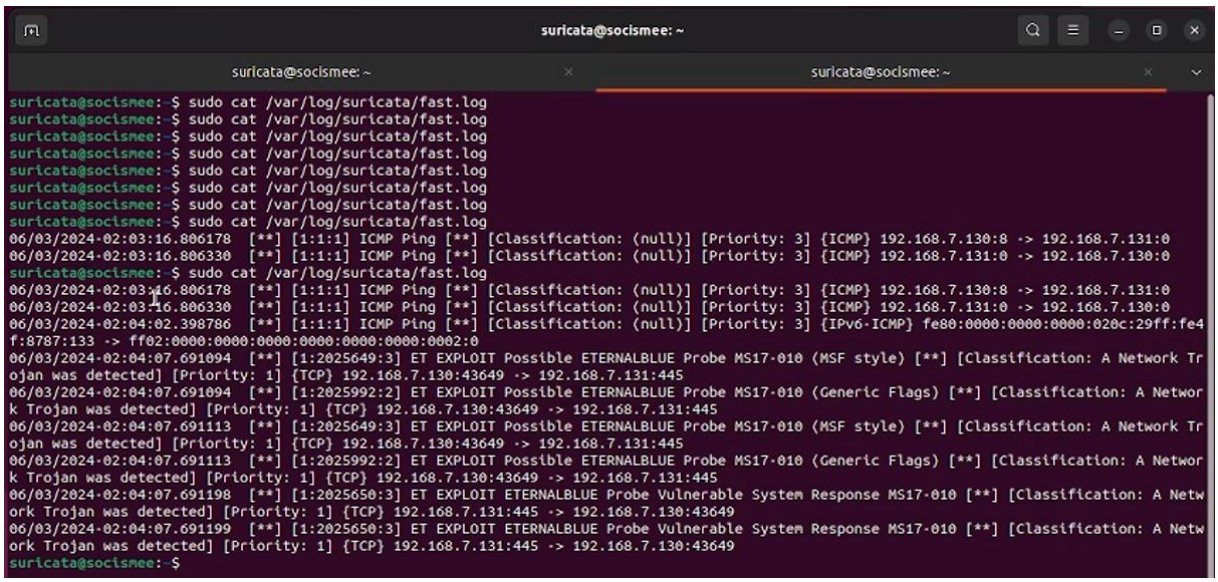
- Bước 11 : Thêm Rule để phát hiện cuộc tấn công vào lỗ hổng MS17-010



- Bước 12 : Thực hiện lại cuộc tấn công thấy cuộc tấn công đã bị từ chối



- Bước 13 : Xem log thấy Suricata đã phát hiện ra cuộc tấn công và đưa ra cảnh báo



5.1.3. Phân tích kết quả đạt được :

- Khi không sử dụng Suricata :
  - Cuộc tấn công diễn ra thành công, Attacker có thể xâm nhập vào hệ thống và thực hiện được rất nhiều hành động phá hoại do khi tấn công vòa bằng lỗ hổng MS17-010 thì Attacker có thể toàn quyền kiểm soát được máy tính nạn nhân.

- Khi sử dụng Suricata :
    - Cuộc tấn công diễn ra không thành công do Suricata đã chặn lưu lượng mạng đáng ngờ từ Attacker, đồng thời Suricata cũng đưa ra cảnh báo về cuộc tấn công giúp cho Admin có thể nhanh chóng phát hiện và khoanh vùng cuộc tấn công hoặc bổ sung thêm các biện pháp bảo mật khác. Nói chung khi cuộc tấn công bị phát hiện và ngăn chặn thì sẽ không có tổn thất nào cho phía hệ thống.

5.1.4. Hướng phát triển :

- Có thể sử dụng trường hợp Attacker đã chiếm dụng thành công Victim sau đó triển khai Suricata để kiểm thử xem Suricata có thể phản ứng với các cuộc tấn công từ bên trong nội bộ không.
- Phát triển hệ thống Suricata ngăn chặn các lưu lượng bất thường từ bước Nmap chứ không để phải tới khi Attacker tấn công mới phát hiện được.
- Cải thiện Rule để Suricata có thể phản ứng nhanh hơn đồng nghĩa với việc giảm tải bộ cảm biến nhằm mục đích phát hiện và ngăn chặn lỗi nhưng không ảnh hưởng quá nhiều đến lưu lượng mạng.
- Cập nhật các quy tắc mới bởi vì một cuộc tấn công chỉ thay đổi một vài chỉ số nhỏ nhưng vẫn có thể vượt qua bộ phận cảm biến để thực hiện cuộc tấn công.
- Cải thiện các chức năng ngăn chặn bằng cách cấu hình các hành động phản ứng tự động khi phát hiện lưu lượng đáng ngờ, chẳng hạn như ngắt kết nối hoặc cô lập hệ thống bị tấn công.

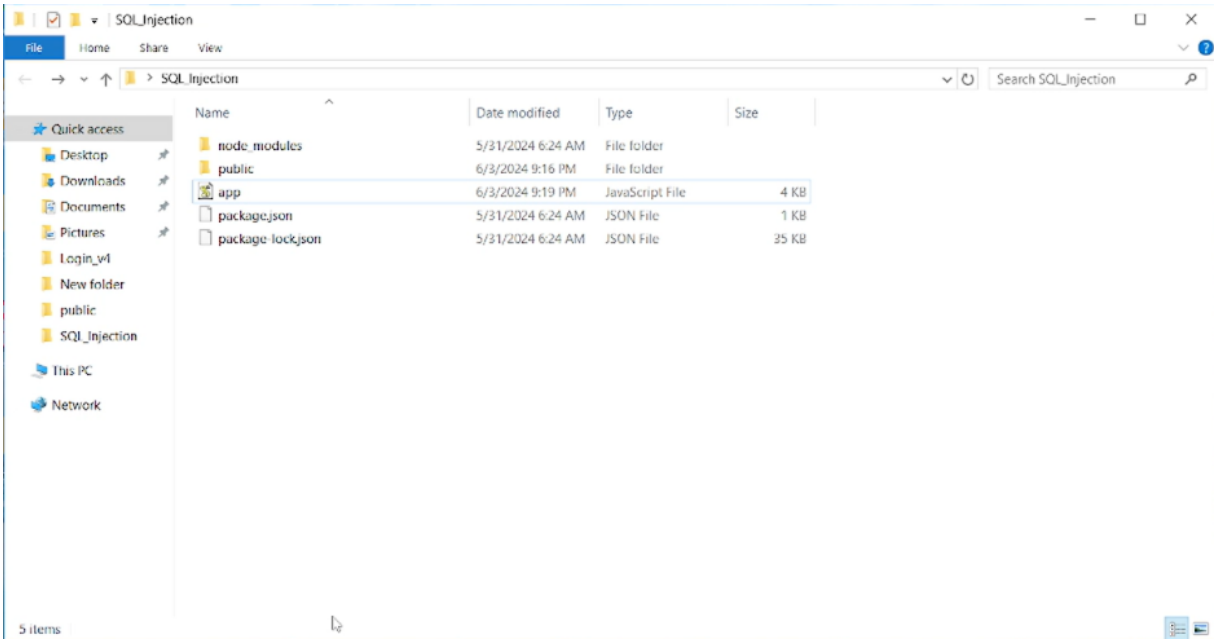
5.2. SQL Injection

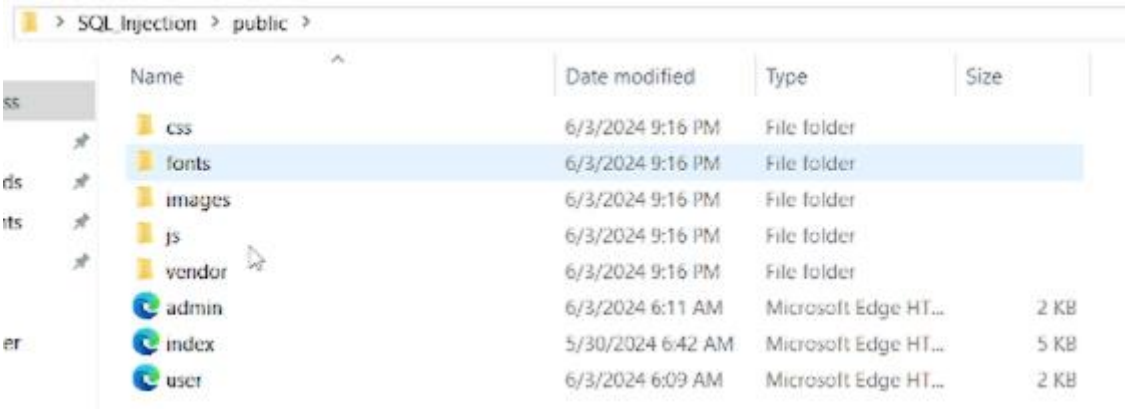
5.2.1. Kịch bản

- Giả lập một hệ thống mạng có Suricata IDS, triển khai một Website có lỗ hổng SQL Injection, một máy tính Attacker trong mạng LAN lợi dụng lỗ hổng này tấn công sang vùng DMZ - nơi đang triển khai Website để truy cập trái phép vào cơ sở dữ liệu. Suricata IDS được thiết lập để phát hiện ra các xâm nhập từ phía Attacker.
- Công cụ : Một trình duyệt bất kì
- Mục tiêu : Sử dụng chuỗi có ký tự đặc biệt như ‘ OR ‘1’=’1 nhằm tiến hành bypass qua hệ thống đăng nhập của Website để truy cập trái phép vào Website Admin.

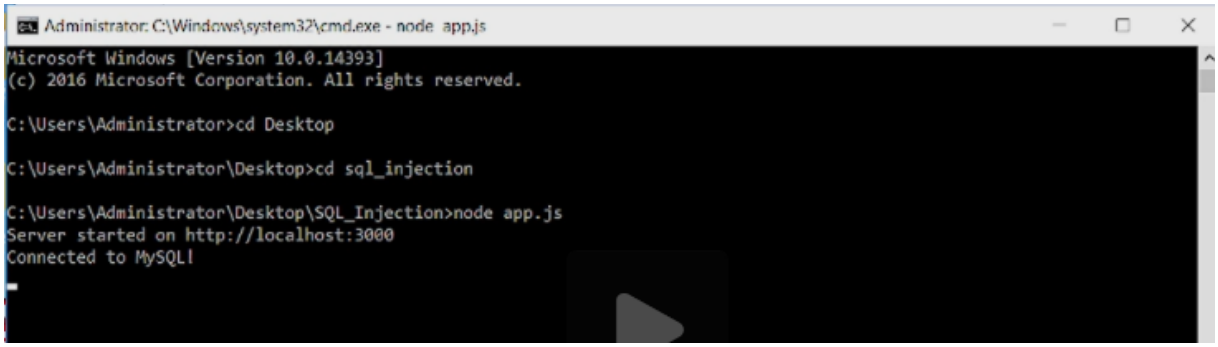
5.2.2. Thực hiện

- Tóm tắt các bước thực hiện :
  - Triển khai Website có lỗ hổng trên Windows Server.
  - Truy cập vào Website bằng IP nội bộ trên máy Attacker.
  - Thực hiện bypass sử dụng lỗ hổng SQL Injection.
  - Sử dụng Suricata trên Pfsense và cấu hình Rule phát hiện SQL Injection.
  - Thực hiện lại tấn công và xem cảnh báo.
- Bước 1 : Xem cấu trúc thư mục của Website trên Web Server

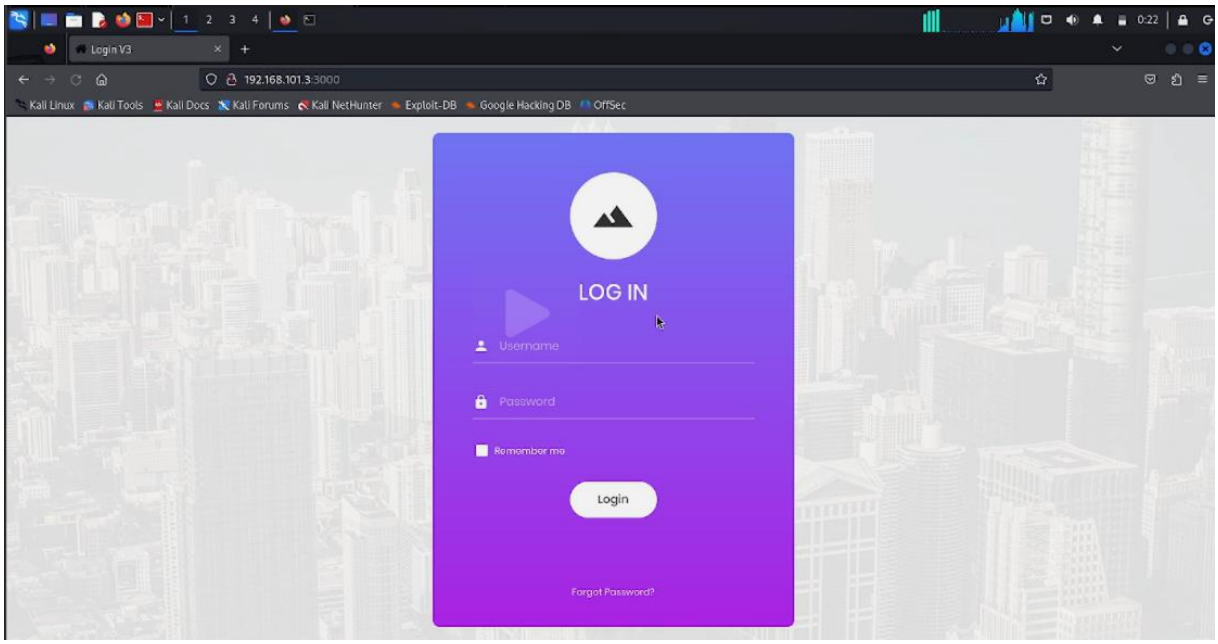




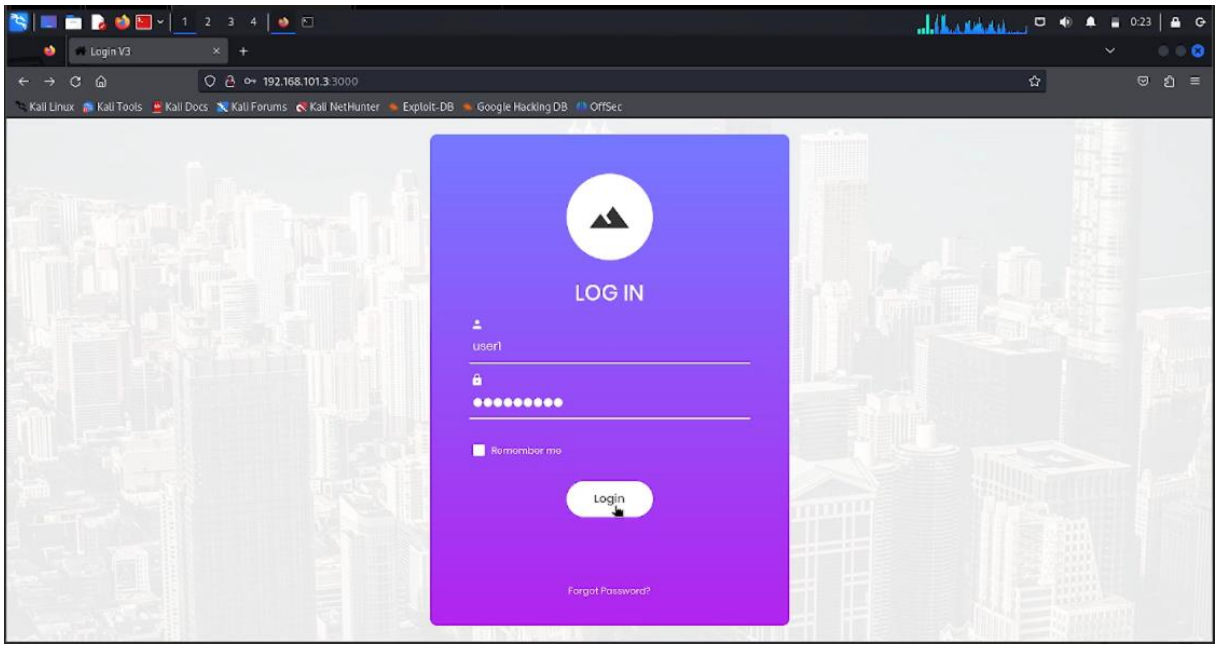
- Bước 2 : Dùng CMD truy cập vào thư mục chính của Website và dùng lệnh : node app.js để triển khai Website bằng NodeJS

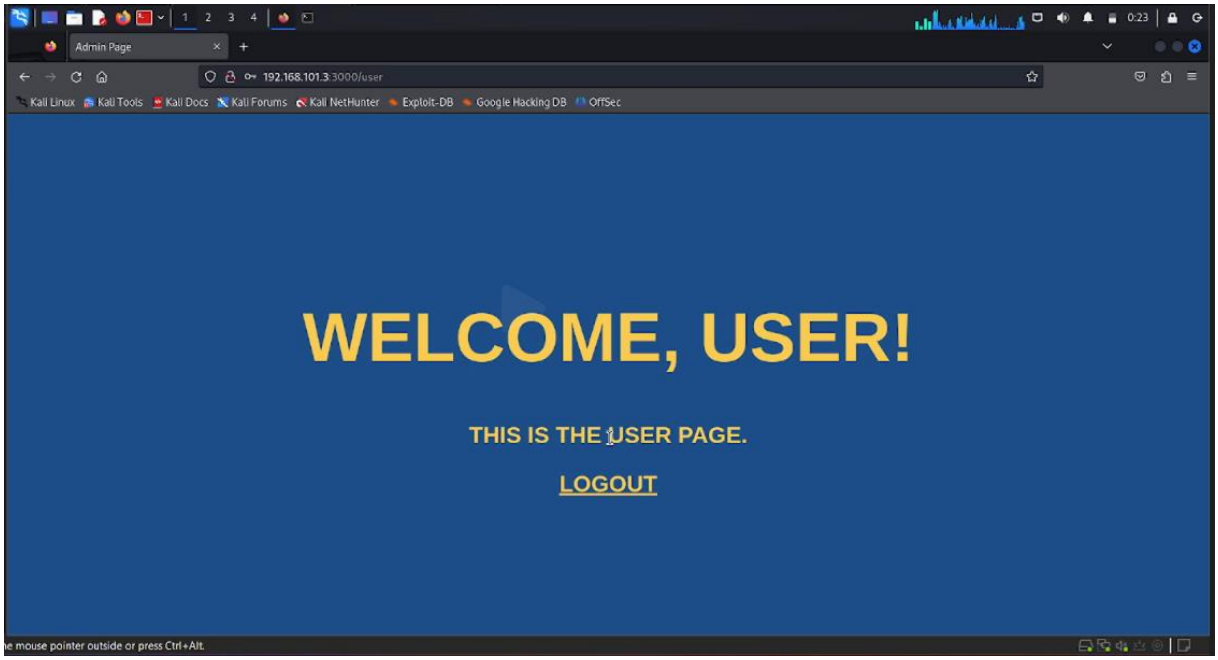


- Bước 3 : Truy cập vào Website trên máy Attacker

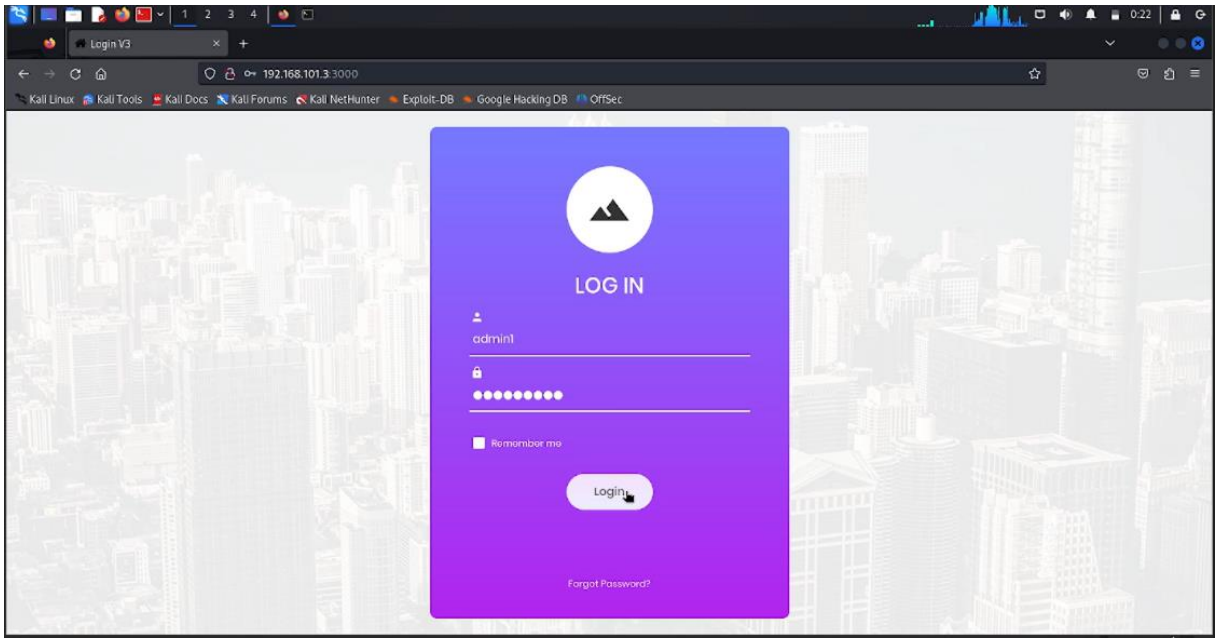


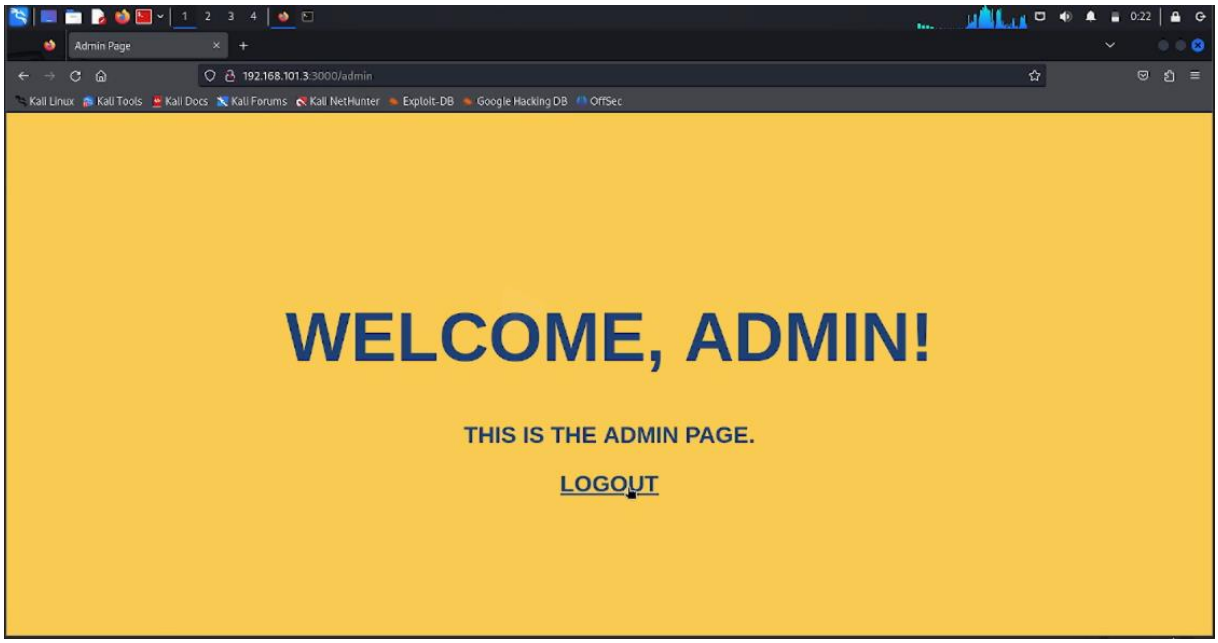
- Bước 4 : Đăng nhập bình thường vào tài khoản User bằng Username và Password hợp lệ



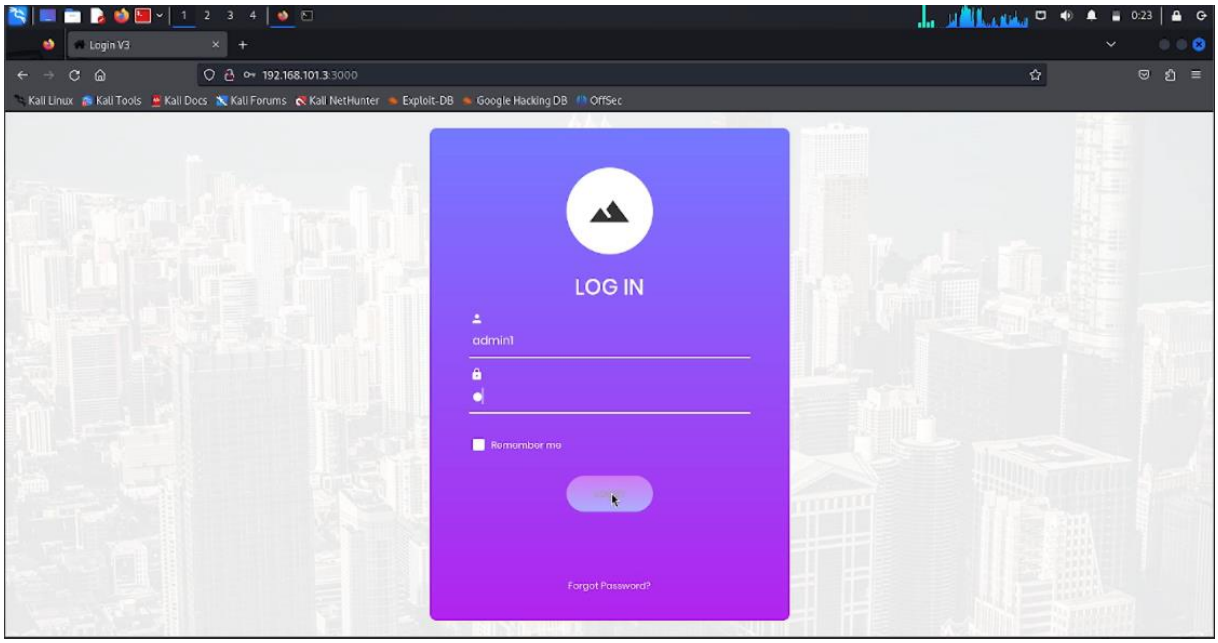


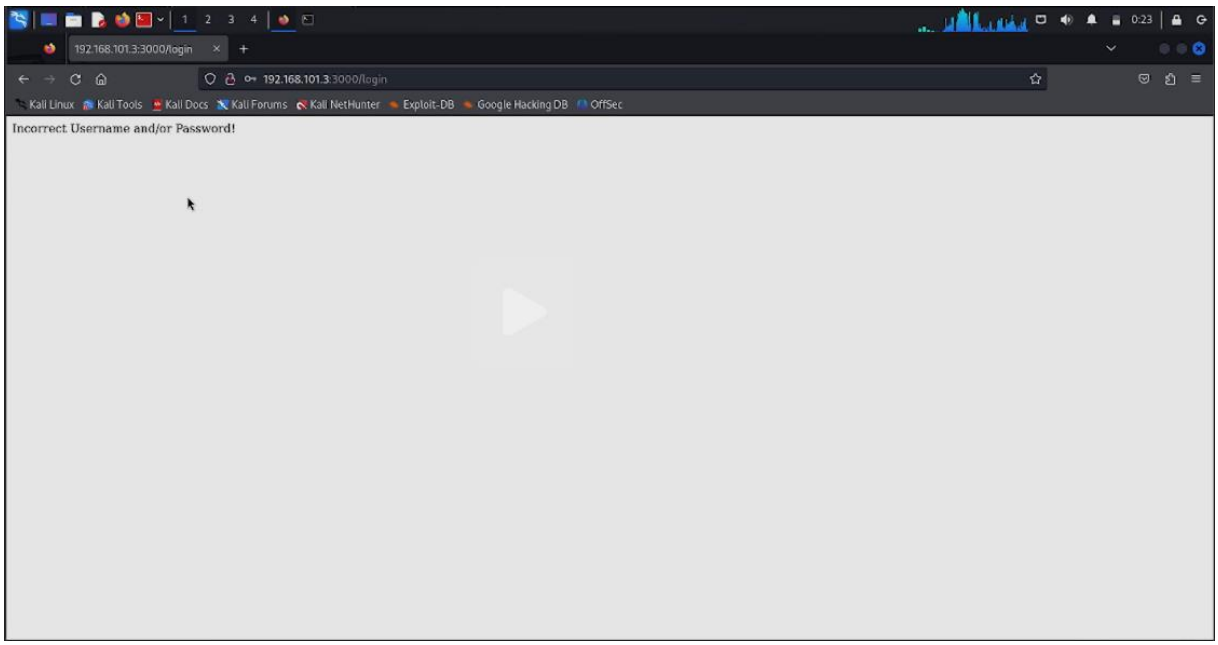
- Bước 5 : Đăng nhập bình thường vào tài khoản Admin bằng Username và Password hợp lệ



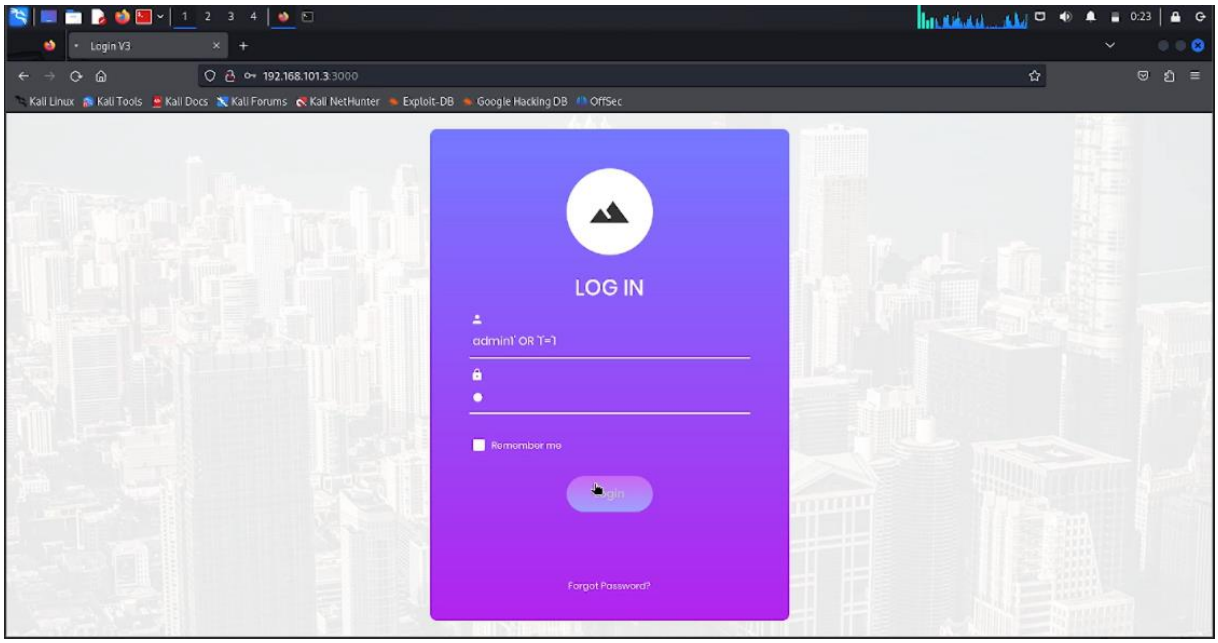


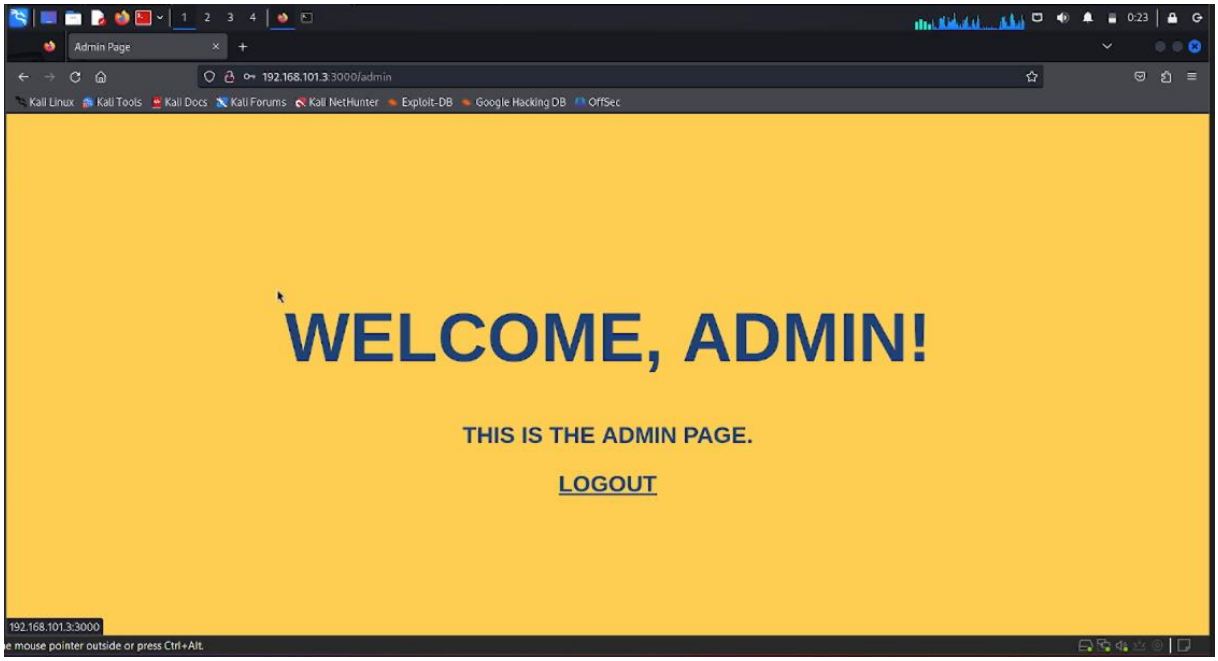
- Bước 6 : Thử đăng nhập vào tài khoản Admin nhưng sử dụng Password không hợp lệ (chỉ sử dụng 1 ký tự) ta thấy Website phản hồi là sai tên tài khoản hoặc mật khẩu



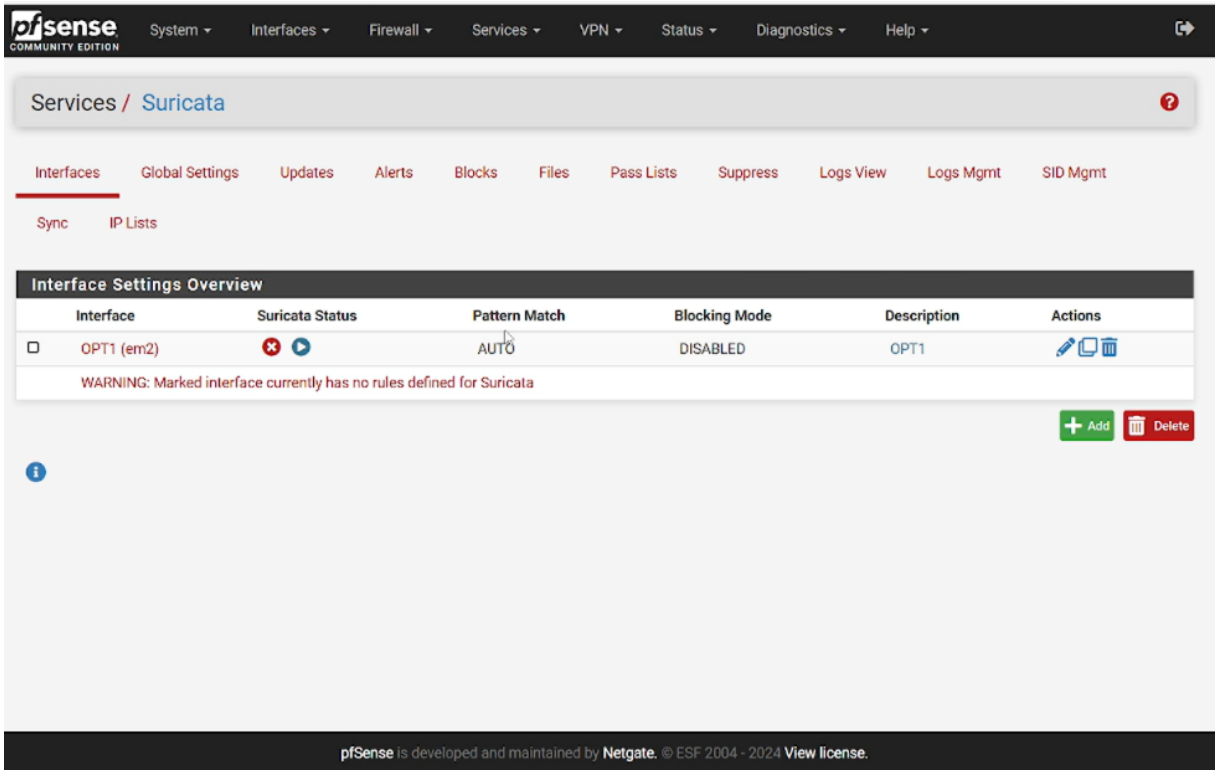


- Bước 7 : Thêm chuỗi ‘ OR ‘1’=’1 vào sau tên tài khoản Admin và nhập mật khẩu không hợp lệ (chỉ nhập 1 ký tự) ta thấy đã truy cập thành công vào trang Admin

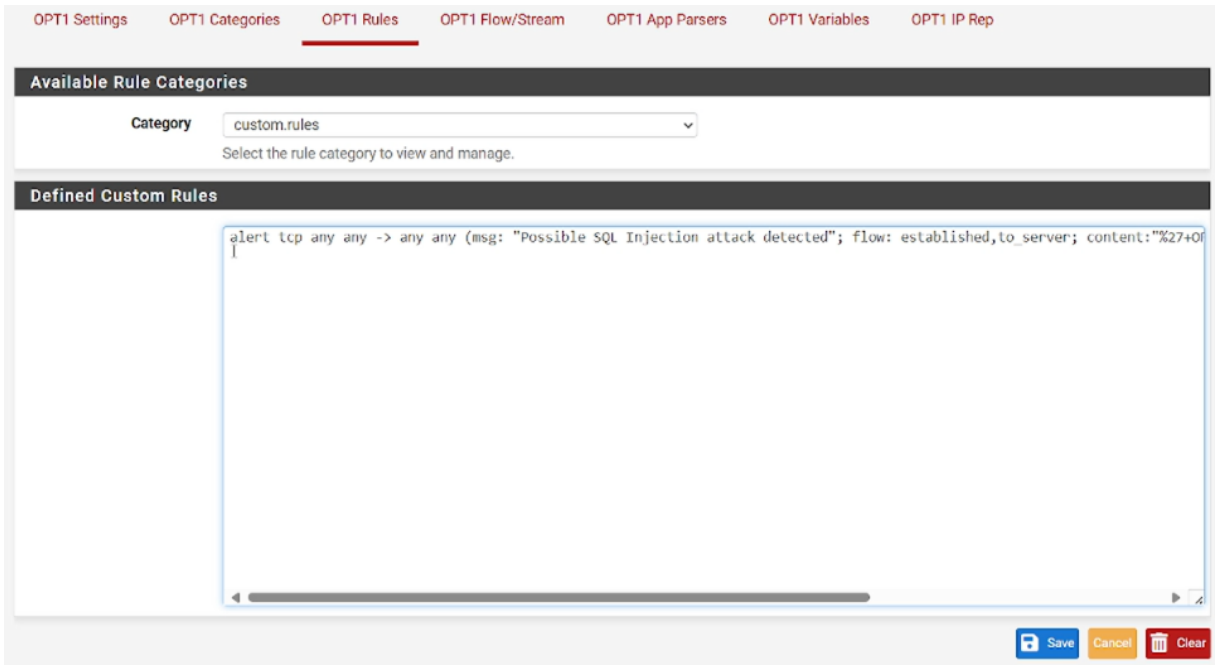




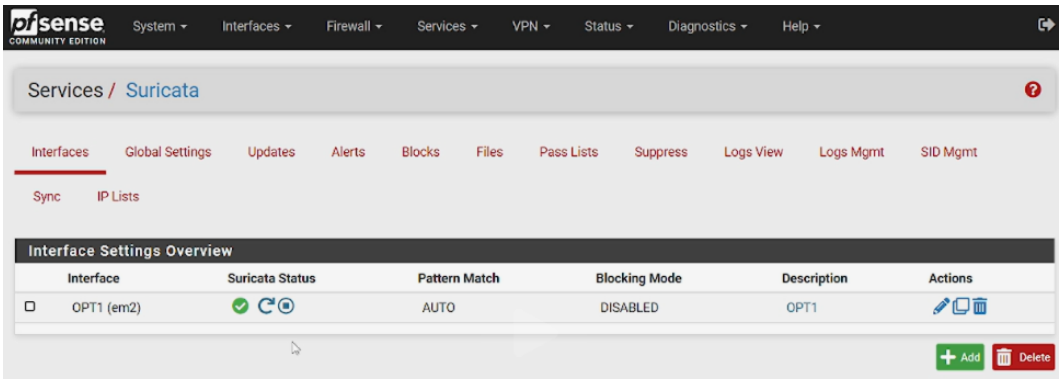
- Bước 8 : Truy cập vào trang cấu hình Pfsense và chọn dịch vụ Suricata sau đó tạo một Suricata Interfaces để giám sát Interfaces qua vùng DMZ



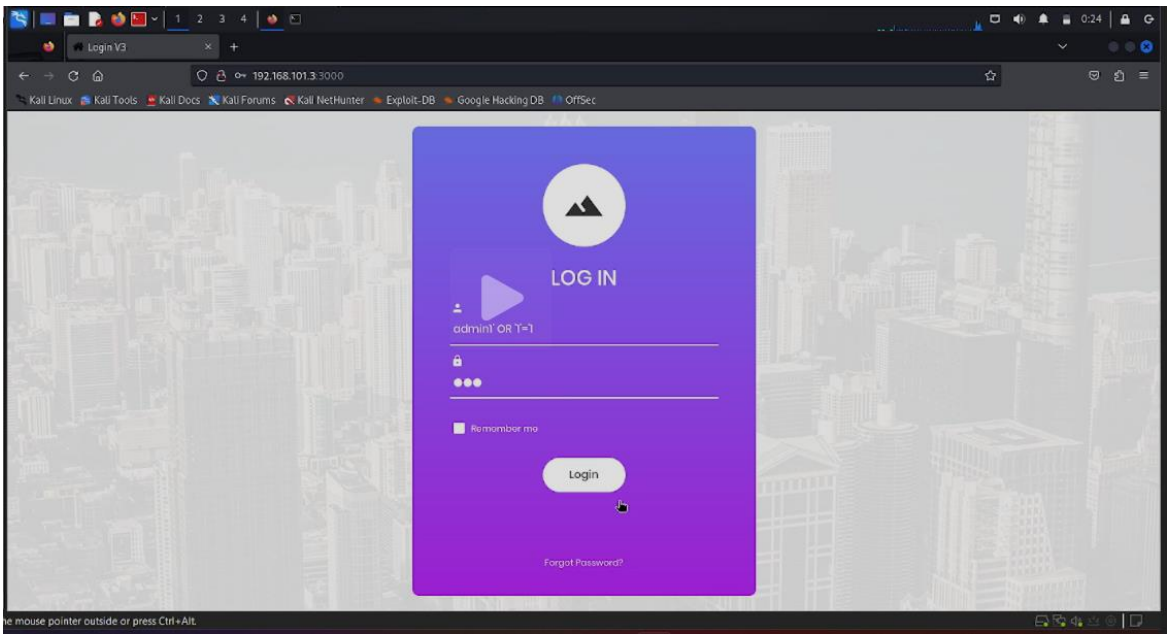
- Bước 9 : Cấu hình Rule để phát hiện SQL Injection

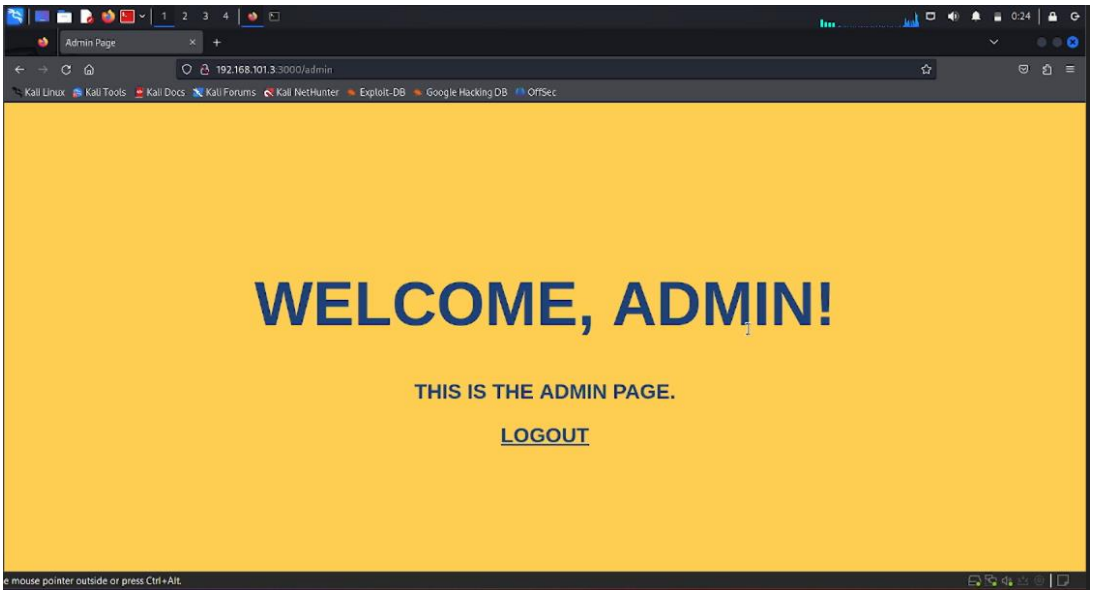


- Bước 10 : Khởi động Suricata để giám sát Interfaces tới vùng DMZ

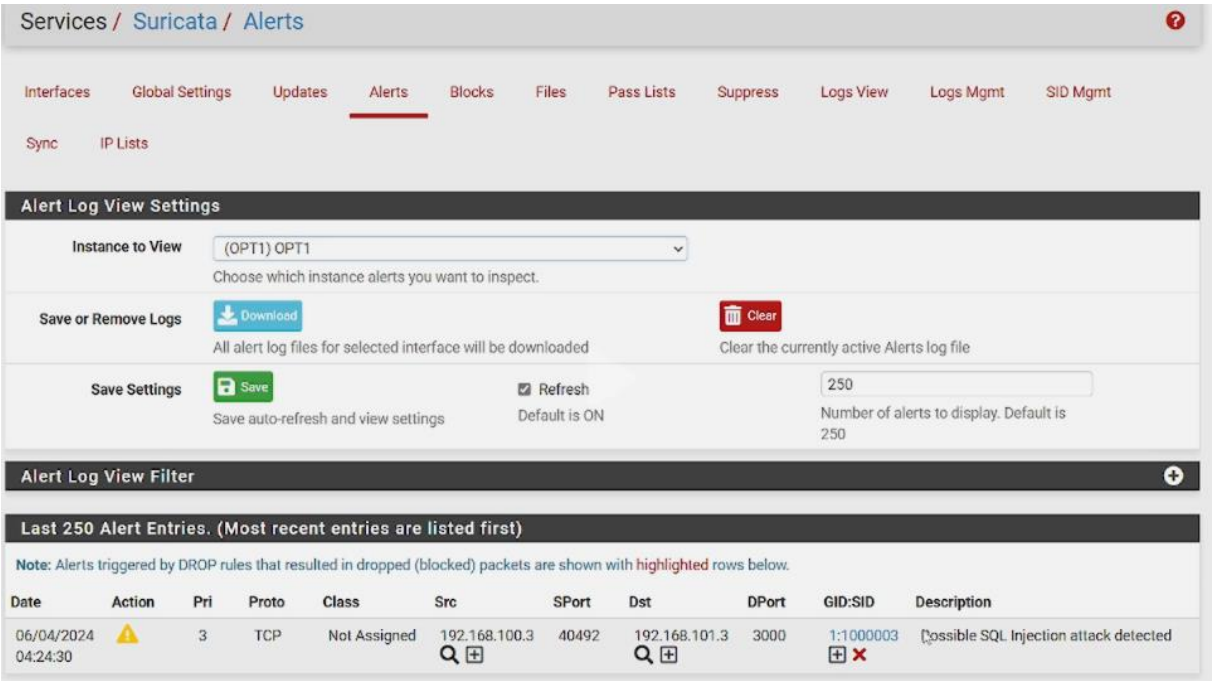


- Bước 11 : Thực hiện lại cuộc tấn công





- Bước 12 : Truy cập vào trang cấu hình Pfsense chọn Suricata và chuyển qua tab Alert để xem cảnh báo



5.2.3. *Phân tích kết quả*

- Khi không sử dụng Suricata :
  - Cuộc tấn công đã diễn ra thành công, Attacker có thể sử dụng lỗ hổng để toàn quyền chiếm dụng cơ sở dữ liệu, sau đó thực hiện các biện pháp phá hoại nhằm trục lợi cho bản thân, Admin có thể phát hiện nhưng khi đó thì mọi chuyện đã quá trễ.
- Khi sử dụng Suricata :
  - Cuộc tấn công vẫn diễn ra thành công, nhưng Admin có thể phát hiện và đưa ra các biện pháp để ngăn chặn cuộc tấn công kịp thời, chẳng hạn như chặn luôn địa chỉ IP hoặc dừng triển khai Website để khắc phục lỗ hổng. Nếu Admin không phát hiện sớm thì vẫn có thể xảy ra thiệt hạt nhất định nhưng không hoàn toàn nghiêm trọng.

5.2.4. *Hướng phát triển*

- Cải thiện Rule bằng cách cập nhật thông tin từ các nguồn uy tín để đảm bảo các Rule luôn mới nhất để phát hiện các mối đe dọa mới và các biến thể của chúng.

- Phát triển quy tắc phát hiện tùy chỉnh dựa trên lưu lượng mạng và các hành vi cụ thể của hệ thống và ứng dụng nhằm tăng cường độ chính xác và giảm thiểu cảnh báo giả (false positives), đồng thời tăng cường khả năng phát hiện các cuộc tấn công cụ thể nhắm vào hệ thống của tổ chức.
- Tích hợp Suricata với các hệ thống khác như SIEM để thu thập, phân tích và tương quan sự kiện bảo mật từ nhiều nguồn khác nhau.
- Cải thiện Rule có thể tự động phát hiện các mối đe dọa hoặc các hành vi bất thường của ứng dụng Web.
- Cập nhật thường xuyên Website để ngăn chặn các lỗ hổng không mong muốn.
- Để Attacker xâm nhập vào hệ thống Database nhằm cải thiện Suricata xem hệ thống này có thể phát hiện những xâm nhập sâu hơn không.

KẾT LUẬN

Qua đề tài này, chúng ta thấy được rằng Suricata là một công cụ phát hiện và ngăn chặn xâm nhập mạng mạnh mẽ, linh hoạt và dễ sử dụng. Với khả năng phân tích lưu lượng mạng và phát hiện các mối đe dọa, Suricata cung cấp một giải pháp hiệu quả cho việc bảo vệ hệ thống mạng khỏi các cuộc tấn công ngày càng tinh vi. Suricata không chỉ giúp các tổ chức, doanh nghiệp bảo vệ an toàn dữ liệu và hệ thống thông tin mà còn nâng cao hiệu quả quản lý an ninh mạng. Sự linh hoạt trong việc tích hợp với các hệ thống khác và khả năng tối ưu hóa hiệu suất giúp Suricata phù hợp với nhiều môi trường mạng khác nhau.

Việc nghiên cứu và phát triển Suricata đóng góp vào lĩnh vực an ninh mạng bằng cách cung cấp các công nghệ và phương pháp mới trong phát hiện và phản ứng với các mối đe dọa. Sự mở rộng và cải tiến liên tục của Suricata cũng thúc đẩy cộng đồng an ninh mạng cùng nhau chia sẻ và phát triển các công cụ bảo mật hiệu quả hơn.

Mặc dù Suricata mang lại nhiều lợi ích, nhưng vẫn còn một số hạn chế như việc không thể phân tích các lưu lượng mạng đã được mã hóa và yêu cầu phải cập nhật liên tục các dấu hiệu tấn công mới nhất để duy trì hiệu quả. Ngoài ra, Suricata cần sử dụng tài nguyên hệ thống nhiều, điều này có thể gây khó khăn cho các tổ chức có nguồn lực phần cứng hạn chế.

Trong tương lai, việc nghiên cứu Suricata có thể tập trung vào việc cải thiện khả năng xử lý các lưu lượng mạng mã hóa và tối ưu hóa việc sử dụng tài nguyên hệ thống. Hơn nữa, việc phát triển các giải pháp tích hợp với các công nghệ bảo mật mới và nâng cao khả năng tự động hóa trong phát hiện và phản ứng với các mối đe dọa sẽ là hướng đi quan trọng để Suricata ngày càng hoàn thiện và hiệu quả hơn. Việc tiếp tục nghiên cứu và ứng dụng Suricata không chỉ mang lại lợi ích cho an ninh mạng của các tổ chức và doanh nghiệp mà còn đóng góp vào sự phát triển bền vững của lĩnh vực an ninh thông tin trong bối cảnh công nghệ ngày càng phát triển nhanh chóng.

TÀI LIỆU THAM KHẢO

[1]. Source: https://suricata.io/features/

[2]. Payload Keywords:

https://docs.suricata.io/en/suricata-5.0.0/rules/payload-keywords.html

[3]. Từ khóa sửa đổi:

https://docs.suricata.io/en/suricata-5.0.0/rules/intro.html#rules-modifiers

[4]. Source: https://suricata.io/features/