**HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG**

**KHOA AN TOÀN THÔNG TIN**

****

**BÁO CÁO BÀI TẬP LỚN**

**HỌC PHẦN: AN TOÀN VÀ BẢO MẬT HỆ THỐNG THÔNG TIN**

**MÃ HỌC PHẦN: INT1303**

**ĐỀ TÀI: TÌM HIỂU VỀ HỆ THỐNG PHÁT HIỆN TẤN CÔNG, ĐỘT NHẬP SURICATA**

Các sinh viên thực hiện (trưởng nhóm xếp số 1):

Đinh Quang Hưng B22DCCN407

Nguyễn Như Duy B22DCCN153

Nguyễn Đăng Trường B22DCCN882

Hoàng Minh Tuấn B22DCCN753

Phạm Trung Kiên B22DCCN431

Tên nhóm: 05

Tên lớp: 10

Giảng viên hướng dẫn: ThS Quản Trọng Thế

**HÀ NỘI 2024**

**PHÂN CÔNG NHIỆM VỤ NHÓM THỰC HIỆN**

|  |  |  |  |
| --- | --- | --- | --- |
| **TT** | **Công việc / Nhiệm vụ** | **SV thực hiện** | **Thời hạn  hoàn thành** |
| 1 | Làm tài liệu chương I + tổng hợp các file thành file báo cáo hoàn chỉnh | Nguyễn Như Duy | 08/04/2025 |
| 2 | Làm tài liệu chương Chương II,III +làm silde chương I,II,III | Hoàng Minh Tuấn | 01/04/2025 |
| 3 | Chương IV + làm silde chương IV,V | Nguyễn Đăng Trường | 08/04/2025 |
| 4 | Chương V: Kịch bản 1+ Demo + Phân công + Đánh giá, góp ý + thuyết trình | Đinh Quang Hưng | 01/04/2025 |
| 5 | Chương V: Kịch bản 2, 3 + Demo | Phạm Trung Kiên | 01/04/2025 |

**NHÓM THỰC HIỆN TỰ ĐÁNH GIÁ**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **TT** | **SV thực hiện** | **Thái độ tham gia** | **Mức hoàn thành CV** | **Kỹ năng giao tiếp** | **Kỹ năng hợp tác** | **Kỹ năng lãnh đạo** |
| 1 | Nguyễn Như Duy | 5 | 4 | 5 | 4 | 4 |
| 2 | Hoàng Minh Tuấn | 5 | 4 | 5 | 4 | 4 |
| 3 | Nguyễn Đăng Trường | 5 | 4 | 5 | 4 | 4 |
| 4 | Đinh Quang Hưng | 5 | 4 | 5 | 4 | 4 |
| 5 | Phạm Trung Kiên | 5 | 4 | 5 | 4 | 4 |

***Ghi chú***:

* Thái độ tham gia: Đánh giá điểm thái độ tham gia công việc chung của nhóm (từ 0: không tham gia, đến 5: chủ động, tích cực).
* Mức hoàn thành CV: Đánh giá điểm mức độ hoàn thành công việc được giao (từ 0: không hoàn thành, đến 5: hoàn thành xuất sắc).
* Kỹ năng giao tiếp: Đánh giá điểm khả năng tương tác, giao tiếp trong nhóm (từ 0: không hoặc giao tiếp rất yếu, đến 5: giao tiếp xuất sắc).
* Kỹ năng hợp tác: Đánh giá điểm khả năng hợp tác, hỗ trợ lẫn nhau, giải quyết mâu thuẫn, xung đột
* Kỹ năng lãnh đạo: Đánh giá điểm khả năng lãnh đạo (từ 0: không có khả năng lãnh đạo, đến 5: có khả năng lãnh đạo tốt, tổ chức và điều phối công việc trong nhóm hiệu quả).

MỤC LỤC

[MỤC LỤC 4](#_Toc195561455)

[DANH MỤC CÁC HÌNH VẼ 7](#_Toc195561456)

[DANH MỤC CÁC BẢNG BIỂU 7](#_Toc195561457)

[DANH MỤC CÁC TỪ VIẾT TẮT 8](#_Toc195561458)

[MỞ ĐẦU 9](#_Toc195561459)

[CHƯƠNG 1. GIỚI THIỆU TỔNG QUAN VỀ HỆ THỐNG IDS/IPS VÀ HỆ THỐNG SURICATA 10](#_Toc195561460)

[1.1 Định nghĩa Suricata 10](#_Toc195561461)

[1.2 Giới thiệu tổng quan về IDS/IPS 10](#_Toc195561462)

[1.2.1 IDS/IPS là gì? 10](#_Toc195561463)

[1.2.2 Vai trò của IDS/IPS trong bảo mật mạng 10](#_Toc195561464)

[1.2.3 Sự khác biệt giữa IDS và IPS 10](#_Toc195561465)

[1.3 Đặc điểm nổi bật của Suricata 11](#_Toc195561466)

[1.3.1 Hỗ trợ phân tích lưu lượng mạng theo thời gian thực 11](#_Toc195561467)

[1.3.2 Phát hiện các mối đe dọa bảo mật dựa trên tập luật 12](#_Toc195561468)

[1.3.3 Giúp bảo vệ hệ thống trước các cuộc tấn công mạng 13](#_Toc195561469)

[1.4 Ứng dụng thực tế của Suricata 15](#_Toc195561470)

[1.4.1 Bảo vệ hệ thống mạng doanh nghiệp khỏi các cuộc tấn công 15](#_Toc195561471)

[1.4.2 Hỗ trợ phân tích forensic 15](#_Toc195561472)

[1.4.3 Tích hợp với ELK Stack (Elasticsearch, Logstash, Kibana) 16](#_Toc195561473)

[1.4.4 Triển khai trên Cloud (AWS, Azure, GCP) 17](#_Toc195561474)

[1.5 Lịch sử phát triển của Suricata 17](#_Toc195561475)

[1.5.1 Giai đoạn khởi đầu (2007 - 2009) 17](#_Toc195561476)

[1.5.2 Giai đoạn phát triển và cải tiến (2010 - 2015) 17](#_Toc195561477)

[1.5.3 Giai đoạn mở rộng và tối ưu hóa (2016 - nay) 18](#_Toc195561478)

[1.6 Lý do Suricata được ưa chuộng so với các công cụ khác 18](#_Toc195561479)

[1.6.1 Hiệu suất cao với kiến trúc đa luồng (Multi-threading) 18](#_Toc195561480)

[1.6.2 Phát hiện mối đe dọa mạnh mẽ hơn 19](#_Toc195561481)

[1.6.3 Khả năng làm việc linh hoạt với nhiều hệ thống 19](#_Toc195561482)

[CHƯƠNG 2. Tìm hiểu về Kiến Trúc của Suricata. 22](#_Toc195561483)

[2.1 Tổng quan về Suricata 22](#_Toc195561484)

[2.1.1 Tại sao nó quan trọng? 22](#_Toc195561485)

[2.1.2 Các chế độ hoạt động chính của Suricata: IDS, IPS, NSM 22](#_Toc195561486)

[2.2 Phân tích chi tiết Kiến trúc của Suricata 23](#_Toc195561487)

[2.2.1 Thu thập gói tin (Packet Acquisition) 23](#_Toc195561488)

[2.2.2 Giải mã gói tin (Packet Decoder) 26](#_Toc195561489)

[2.2.3 Bộ máy phát hiện (Detection Engine) 28](#_Toc195561490)

[2.2.4 Ghi log và cảnh báo (Logging & Alerting) 29](#_Toc195561491)

[2.2.5 Xử lý đầu ra (Output Processing) 31](#_Toc195561492)

[2.3 So sánh Suricata với Snort 31](#_Toc195561493)

[2.3.1 Phân tích chi tiết về kiến trúc, hiệu suất, tính năng và các trường hợp sử dụng khác nhau 31](#_Toc195561494)

[CHƯƠNG 3. Hướng Dẫn Cài đặt Suricata trên hệ điều hành Ubuntu. 33](#_Toc195561495)

[3.1 Các yêu cầu hệ thống 33](#_Toc195561496)

[3.2 Hướng dẫn chi tiết các bước cài đặt Suricata trên Ubuntu 34](#_Toc195561497)

[3.2.1 Cài đặt từ kho chính thức và PPA 34](#_Toc195561498)

[3.2.2 Các bước cấu hình cơ bản sau khi cài đặt 35](#_Toc195561499)

[3.3 Kiểm tra và xác minh cài đặt 40](#_Toc195561500)

[CHƯƠNG 4. TẠO LUẬT PHÁT HIỆN TẤN CÔNG TRONG SURICATA 42](#_Toc195561501)

[4.1 Giới thiệu 42](#_Toc195561502)

[4.2 Rule header 42](#_Toc195561503)

[4.2.1 Rule action 42](#_Toc195561504)

[4.2.2 Protocol 43](#_Toc195561505)

[4.2.3 . IP Address 43](#_Toc195561506)

[4.2.4 Port 43](#_Toc195561507)

[4.2.5 Điều hướng 44](#_Toc195561508)

[4.3 Rule Option 44](#_Toc195561509)

[4.3.1 General 45](#_Toc195561510)

[4.3.2 Payload 46](#_Toc195561511)

[4.3.3 Non-Payload 48](#_Toc195561512)

[4.4 Các nguồn luật phổ biến 54](#_Toc195561513)

[4.5 Ưu và nhược điểm của Suricata Rules 54](#_Toc195561514)

[4.6 Ví dụ 55](#_Toc195561515)

[CHƯƠNG 5. Kịch bản 57](#_Toc195561516)

[5.1 Chuẩn bị và cài đặt các công cụ cần thiết: 57](#_Toc195561517)

[5.1.1 Chuẩn bị 57](#_Toc195561518)

[5.1.2 Cài đặt các công cụ cần thiết: 57](#_Toc195561519)

[5.2 Tấn công UDP Flood 57](#_Toc195561520)

[5.2.1 Mục tiêu: 57](#_Toc195561521)

[5.2.2 Mô tả cách demo 58](#_Toc195561522)

[5.3 Tấn công SSH Brute Force 59](#_Toc195561523)

[5.3.1 Mục tiêu: 59](#_Toc195561524)

[5.3.2 Mô tả cách demo 60](#_Toc195561525)

[5.4 Tấn công SQL Injection 61](#_Toc195561526)

[5.4.1 Mục tiêu: 61](#_Toc195561527)

[5.4.2 Cấu hình môi trường 61](#_Toc195561528)

[5.4.3 Mô tả demo 63](#_Toc195561529)

[KẾT LUẬN 67](#_Toc195561530)

[TÀI LIỆU THAM KHẢO 67](#_Toc195561531)

DANH MỤC CÁC HÌNH VẼ

[Hình 1 - Set up để cài đặt suricata 35](#_Toc195393581)

[Hình 2 - Cài đặt suricata 35](#_Toc195393582)

[Hình 3 - Kiểm tra phiên bản Suricata 36](#_Toc195393583)

[Hình 4 - Kiểm tra trạng thái hoạt động của suricata 37](#_Toc195393584)

[Hình 5 - Dừng dịch vụ Suricata 37](#_Toc195393585)

[Hình 6 - Chỉnh sửa file cấu hình chính 38](#_Toc195393586)

[Hình 7 - Bật khởi động cùng hệ thống 40](#_Toc195393587)

[Hình 8 - Kiem tra log suricata 41](#_Toc195393588)

[Hình 9 - Bảng các tuỳ chọn của Reference 45](#_Toc195393589)

[Hình 10 - Thông tin phân loại lớp quy tắc 46](#_Toc195393590)

DANH MỤC CÁC BẢNG BIỂU

[Bảng 1. Sự khác biệt giữa IDS và IPS 11](#_Toc195393591)

DANH MỤC CÁC TỪ VIẾT TẮT

|  |  |  |
| --- | --- | --- |
| **Từ  viết tắt** | **Thuật ngữ tiếng Anh/Giải thích** | **Thuật ngữ tiếng Việt/Giải thích** |
| IDS | Intrusion Detection System | Hệ thống phát hiện xâm nhập |
| IPS | Intrusion Prevention System | Hệ thống ngăn chặn xâm nhập |
| Suricata | (Tên phần mềm) | Hệ thống IDS/IPS mã nguồn mở |
| DPI | Deep Packet Inspection | Kiểm tra gói tin sâu |
| HTTP | Hypertext Transfer Protocol | Giao thức truyền tải siêu văn bản |
| TLS | Transport Layer Security | Bảo mật tầng truyền tải |
| IP | Internet Protocol | Giao thức Internet |
| DNS | Domain Name System | Hệ thống phân giải tên miền |
| JSON | JavaScript Object Notation | Định dạng trao đổi dữ liệu dựa trên văn bản |
| PCAP | Packet Capture | Ghi và phân tích gói tin mạng |
| CPU | Central Processing Unit | Bộ xử lý trung tâm |
| GPU | Graphics Processing Unit | Bộ xử lý đồ họa |

MỞ ĐẦU

Trong bối cảnh an ninh mạng ngày càng trở thành mối quan tâm hàng đầu, các hệ thống giám sát và phát hiện xâm nhập đóng vai trò quan trọng trong việc bảo vệ hạ tầng công nghệ thông tin. Suricata là một hệ thống phát hiện và ngăn chặn xâm nhập (IDS/IPS) mã nguồn mở, có khả năng phân tích lưu lượng mạng tốc độ cao, phát hiện các mối đe dọa an ninh và cung cấp các biện pháp phòng vệ hiệu quả. Với khả năng mở rộng linh hoạt, Suricata đã trở thành một trong những công cụ phổ biến trong lĩnh vực bảo mật mạng.

Báo cáo bài tập lớn này tập trung nghiên cứu về Suricata, một hệ thống giám sát an ninh mạng mạnh mẽ, với các nội dung chính sau:

Chương 1: Nghiên cứu tổng quan về Suricata, bao gồm kiến trúc, nguyên lý hoạt động, các tính năng nổi bật và vai trò của Suricata trong bảo mật mạng.

Chương 2: Phân tích, cài đặt và cấu hình Suricata trên hệ thống thực tế, nhằm kiểm thử các tính năng giám sát lưu lượng mạng và phát hiện các mối đe dọa tiềm ẩn.

Chương 3: Thử nghiệm và đánh giá hiệu suất của Suricata trong môi trường thực tế, so sánh với các hệ thống IDS/IPS khác, đồng thời đưa ra nhận xét và đề xuất cải tiến.

Chương 4: Xây dựng các quy tắc tùy chỉnh trong Suricata để tăng cường khả năng phát hiện và phòng thủ trước các mối đe dọa mới.

Chương 5: Kết luận và đề xuất hướng phát triển trong tương lai nhằm nâng cao hiệu quả ứng dụng Suricata trong các hệ thống bảo mật mạng.

Báo cáo không chỉ giúp hiểu rõ hơn về Suricata mà còn cung cấp cái nhìn thực tiễn về cách triển khai và tối ưu hóa hệ thống này trong các môi trường mạng thực tế.

1. GIỚI THIỆU TỔNG QUAN VỀ HỆ THỐNG IDS/IPS VÀ HỆ THỐNG SURICATA
   1. Định nghĩa Suricata

Suricata là một công cụ giám sát an ninh mạng, hệ thống phát hiện và ngăn chặn xâm nhập (IDS/IPS) hiệu suất cao. Nó là mã nguồn mở và thuộc sở hữu của một tổ chức phi lợi nhuận do cộng đồng điều hành, Tổ chức An ninh Thông tin Mở (OISF). Suricata được phát triển bởi OISF.

Giới thiệu về Tổ chức An ninh Thông tin Mở ( Open Information Security Foundation)

* Tổ chức An ninh Thông tin Mở là một tổ chức phi lợi nhuận được thành lập để xây dựng cộng đồng và hỗ trợ các công nghệ bảo mật mã nguồn mở như Suricata, công cụ IDS/IPS đẳng cấp thế giới.
  1. Giới thiệu tổng quan về IDS/IPS
     1. IDS/IPS là gì?

Hệ thống Phát hiện Xâm nhập (IDS - Intrusion Detection System) và Hệ thống Ngăn chặn Xâm nhập (IPS - Intrusion Prevention System) là hai công nghệ quan trọng trong bảo mật mạng, giúp phát hiện và bảo vệ hệ thống khỏi các cuộc tấn công mạng.

IDS (Intrusion Detection System): Hệ thống giám sát lưu lượng mạng hoặc các sự kiện trên hệ thống để phát hiện các hoạt động bất thường hoặc có dấu hiệu tấn công. IDS chỉ cảnh báo mà không can thiệp trực tiếp vào lưu lượng mạng.

IPS (Intrusion Prevention System): Là một phiên bản nâng cao của IDS, không chỉ phát hiện mà còn ngăn chặn các mối đe dọa bằng cách chặn hoặc vô hiệu hóa lưu lượng mạng độc hại trước khi nó gây hại.

* + 1. Vai trò của IDS/IPS trong bảo mật mạng

IDS/IPS đóng vai trò quan trọng trong việc bảo vệ hệ thống mạng khỏi các mối đe dọa an ninh như:

* Phát hiện tấn công mạng: Xác định các hành vi đáng ngờ, chẳng hạn như quét cổng, tấn công từ chối dịch vụ (DDoS), khai thác lỗ hổng, v.v.
* Giảm thiểu rủi ro bảo mật: Giúp tổ chức kịp thời phát hiện và phản ứng với các sự cố bảo mật.
* Hỗ trợ tuân thủ bảo mật: Nhiều tổ chức phải triển khai IDS/IPS để tuân thủ các quy định như PCI-DSS, HIPAA.
* Bảo vệ dữ liệu quan trọng: Ngăn chặn các cuộc tấn công nhằm vào dữ liệu nhạy cảm, tránh rò rỉ hoặc mất mát thông tin.
* Giám sát và phân tích lưu lượng mạng: Giúp quản trị viên mạng theo dõi hoạt động bất thường và điều tra các sự cố bảo mật.
  + 1. Sự khác biệt giữa IDS và IPS

1. Sự khác biệt giữa IDS và IPS

|  |  |  |
| --- | --- | --- |
| Tiêu chí | IDS(Intrusion Detection System) | IPS (Intrusion Prevention System) |
| Chức năng chính | Phát hiện và cảnh báo về các mối đe dọa | Phát hiện và chủ động ngăn chặn các mối đe dọa |
| Tác động đến lưu lượng mạng | Không làm gián đoạn lưu lượng, chỉ giám sát và ghi nhận | Có thể chặn hoặc thay đổi lưu lượng để ngăn chặn tấn công |
| Vị trí triển khai | Thường được đặt sau tường lửa để giám sát lưu lượng mạng | Thường được đặt giữa tường lửa và hệ thống mạng để kiểm soát lưu lượng |
| Khả năng phản ứng | Chỉ gửi cảnh báo cho quản trị viên khi phát hiện mối đe dọa | Chủ động chặn lưu lượng độc hại ngay khi phát hiện |
| Mức độ can thiệp | Thụ động (chỉ giám sát) | Chủ động (có thể chặn hoặc sửa đổi gói tin) |

* 1. Đặc điểm nổi bật của Suricata

Suricata là một hệ thống phát hiện và ngăn chặn xâm nhập mạng (IDS/IPS) mã nguồn mở, được thiết kế để phân tích lưu lượng mạng theo thời gian thực và phát hiện các mối đe dọa bảo mật bằng cách sử dụng các tập luật (rule-based detection). Đây là một trong những công cụ bảo mật hàng đầu, hỗ trợ bảo vệ hệ thống khỏi các cuộc tấn công mạng nguy hiểm.

* + 1. Hỗ trợ phân tích lưu lượng mạng theo thời gian thực
       1. Giám sát và kiểm tra gói tin tốc độ cao

Suricata có thể xử lý lưu lượng mạng tốc độ cao nhờ vào kiến trúc đa luồng (multi-threading), giúp tận dụng tối đa tài nguyên phần cứng như CPU và RAM.

Hỗ trợ xử lý gói tin ở tốc độ 10Gbps hoặc cao hơn, phù hợp với các hệ thống doanh nghiệp và mạng lớn.

* + - 1. Hỗ trợ nhiều giao thức mạng

Suricata có thể phân tích và hiểu được nhiều loại giao thức, từ các giao thức cơ bản đến nâng cao:

Bao gồm các giao thức như TCP, UDP, ICMP, HTTP, HTTPS, DNS, FTP, SMB, SSH, và nhiều giao thức khác.

* Khả năng phân tích sâu gói tin (Deep Packet Inspection – DPI):
* Cho phép kiểm tra nội dung bên trong gói tin để phát hiện hành vi đáng ngờ, mã độc hoặc các khai thác lỗ hổng.
* Có thể nhận biết lưu lượng mã hóa hoặc bị giả mạo, giúp phát hiện những mối đe dọa tinh vi ẩn sâu trong lớp dữ liệu ứng dụng.
  + - 1. Kết hợp với các công cụ giám sát khác

Suricata hỗ trợ tích hợp linh hoạt với các công cụ giám sát và phân tích bảo mật khác, giúp quản trị viên dễ dàng vận hành và theo dõi:

* Tích hợp với ELK Stack (Elasticsearch, Logstash, Kibana):
* Thu thập và xử lý log theo thời gian thực.
* Hiển thị dữ liệu bảo mật thông qua các biểu đồ, bảng điều khiển (dashboard) sinh động và trực quan.

Kết hợp với Zeek (trước đây là Bro IDS):

* Suricata phát hiện dựa trên mẫu (signature-based), còn Zeek cung cấp phân tích hành vi mạng dựa trên ngữ cảnh và các mẫu bất thường (anomaly-based).
* Khi kết hợp, quản trị viên có góc nhìn toàn diện hơn về các sự kiện mạng và nguy cơ bảo mật tiềm ẩn.
  + 1. Phát hiện các mối đe dọa bảo mật dựa trên tập luật

Suricata sử dụng cơ chế phát hiện dựa trên tập luật, tương tự như Snort, nhưng có những cải tiến vượt trội.

* + - 1. Sử dụng tập luật mạnh mẽ

Một trong những điểm mạnh nổi bật của Suricata là khả năng hỗ trợ và sử dụng các tập luật phát hiện mối đe dọa bảo mật phong phú, giúp phát hiện và phản ứng với các kiểu tấn công đa dạng và liên tục cập nhật.

* Tương thích với tập luật của Snort
* Suricata hoàn toàn tương thích với định dạng luật của Snort – một hệ thống IDS nổi tiếng và lâu đời.
* Điều này cho phép Suricata kế thừa và khai thác kho luật lớn mạnh của Snort, mà không cần phải viết lại từ đầu.
* Nhờ đó, Suricata tiết kiệm thời gian cấu hình, nhưng vẫn đảm bảo hiệu quả cao trong việc phát hiện các loại tấn công phổ biến như: tấn công từ chối dịch vụ (DoS/DDoS), tấn công khai thác lỗ hổng, tấn công thông qua mã độc, lừa đảo, khai thác trình duyệt…
* Hỗ trợ nhiều nguồn luật bảo mật uy tín

Suricata có thể dễ dàng tích hợp với các nguồn luật được cập nhật thường xuyên từ cộng đồng và các nhà cung cấp bảo mật:

* Emerging Threats (ET):

Là một trong những nguồn luật phổ biến và miễn phí. Cung cấp các luật phát hiện mới nhất về phần mềm độc hại, botnet, và các hoạt động đáng ngờ.

* Proofpoint VRT (Vulnerability Research Team):

Cung cấp các luật chính xác, được cập nhật thường xuyên từ đội ngũ chuyên gia nghiên cứu bảo mật của Proofpoint. Phù hợp với các doanh nghiệp yêu cầu tính chính xác và độ tin cậy cao trong phát hiện.

* Custom Rules (Luật tùy chỉnh):

Quản trị viên mạng có thể viết luật thủ công để phù hợp với yêu cầu riêng của tổ chức, ví dụ: phát hiện truy cập vào một URL nội bộ bị cấm, giám sát lưu lượng liên quan đến một máy chủ cụ thể, điều này giúp hệ thống linh hoạt hơn, dễ thích nghi với môi trường thực tế và các mối nguy riêng biệt của từng tổ chức.

* + - 1. Phát hiện các kiểu tấn công phổ biến

Suricata có thể phát hiện các mối đe dọa bảo mật như:

* Tấn công từ chối dịch vụ (DDoS)
* Khai thác lỗ hổng (Exploit Attempt Detection)
* Tấn công brute-force (đoán mật khẩu)
* Tấn công SQL Injection, Cross-Site Scripting (XSS)
* Phát hiện phần mềm độc hại và botnet
  + - 1. Hỗ trợ Signature-Based và Anomaly-Based Detection

**Signature-Based Detection**: Sử dụng tập luật có sẵn để so khớp với các mẫu tấn công đã biết.

**Anomaly-Based Detection**: Phát hiện các hành vi bất thường dựa trên phân tích hành vi mạng.

* + 1. Giúp bảo vệ hệ thống trước các cuộc tấn công mạng

Suricata không chỉ phát hiện mà còn có khả năng ngăn chặn các cuộc tấn công, giúp bảo vệ hệ thống mạng một cách hiệu quả.

* + - 1. Chế độ hoạt động linh hoạt

Suricata có thể hoạt động ở các chế độ sau:

* IDS Mode (Intrusion Detection System): Giám sát và ghi nhận các cuộc tấn công mà không can thiệp vào lưu lượng mạng.
* IPS Mode (Intrusion Prevention System): Chủ động chặn lưu lượng mạng độc hại bằng cách chặn gói tin hoặc ngắt kết nối.
* NSM Mode (Network Security Monitoring): Thu thập thông tin chi tiết về lưu lượng mạng để phân tích chuyên sâu.
  + - 1. Hỗ trợ ngăn chặn theo thời gian thực

Suricata không chỉ dừng lại ở vai trò phát hiện mối đe dọa như một IDS (Intrusion Detection System), mà còn có thể hoạt động như một IPS (Intrusion Prevention System) với khả năng phản ứng ngay lập tức khi phát hiện hành vi bất thường:

* Chặn kết nối ngay lập tức
* Khi phát hiện một mối đe dọa, Suricata có thể tự động chặn hoặc vô hiệu hóa kết nối đang diễn ra:

+ Ngăn chặn lưu lượng độc hại từ nguồn IP nghi ngờ.

+ Dừng ngay truy cập đến các địa chỉ URL nguy hiểm.

+ Cắt kết nối đến các cổng dịch vụ bị khai thác.

* Việc này giảm thiểu thiệt hại và ngăn chặn tấn công lan rộng, đặc biệt quan trọng trong các cuộc tấn công như:

+ Phát tán ransomware

+ Tấn công brute-force

+ Tấn công vào các lỗ hổng zero-day

* Tự động cập nhật tập luật phát hiện
* Suricata hỗ trợ công cụ suricata-update để:

+ Cập nhật tập luật mới nhất từ các nguồn đáng tin cậy như Emerging Threats.

+ Đảm bảo hệ thống luôn được trang bị khả năng phát hiện các mối đe dọa mới xuất hiện trên toàn cầu.

* Quá trình cập nhật có thể được lên lịch hoặc tự động hóa, giúp duy trì khả năng phản ứng nhanh chóng và liên tục trước các mối nguy mới.
  + - 1. Bảo vệ đa lớp

Suricata có thể được triển khai trong nhiều lớp bảo vệ khác nhau của hệ thống mạng, góp phần xây dựng kiến trúc phòng thủ nhiều tầng (defense-in-depth):

* Kết hợp với hệ thống tường lửa (Firewall)
* Suricata có thể hoạt động song song hoặc tích hợp với các firewall như:

+ iptables (Linux) – hỗ trợ chặn kết nối dựa trên luật Suricata phát hiện.

+ pfSense – một hệ thống tường lửa mã nguồn mở phổ biến, có thể tích hợp trực tiếp với Suricata để nâng cao năng lực phát hiện và ngăn chặn.

* Việc này gia tăng khả năng bảo vệ, giúp lọc lưu lượng từ lớp thấp đến lớp cao, từ kết nối mạng đến nội dung trong gói tin.
* Triển khai linh hoạt: endpoint hoặc mạng trung tâm
* Trên endpoint (máy trạm, server):

+ Giúp giám sát lưu lượng đến/đi riêng lẻ của từng thiết bị.

+ Bảo vệ các hệ thống quan trọng hoặc đầu cuối dễ bị tấn công.

* Trong hệ thống mạng trung tâm (gateway, switch, router):

+ Kiểm soát toàn bộ lưu lượng đi qua mạng nội bộ.

+ Phù hợp để triển khai tại doanh nghiệp, trung tâm dữ liệu, tổ chức lớn.

* 1. Ứng dụng thực tế của Suricata

Suricata không chỉ là một công cụ IDS/IPS mạnh mẽ, mà còn có nhiều ứng dụng thực tế giúp bảo vệ hệ thống mạng doanh nghiệp khỏi các cuộc tấn công, hỗ trợ phân tích forensic, tích hợp với hệ thống giám sát log như ELK stack, và có thể triển khai trên nền tảng cloud như AWS, Azure, GCP.

* + 1. Bảo vệ hệ thống mạng doanh nghiệp khỏi các cuộc tấn công

Suricata giúp các doanh nghiệp ngăn chặn nhiều loại tấn công mạng phổ biến:

* + - 1. Chống tấn công DDoS (Distributed Denial of Service)

Suricata có thể phát hiện các dấu hiệu của tấn công DDoS như lưu lượng bất thường, lượng request tăng đột biến, hoặc các gói tin có dấu hiệu giả mạo.

Khi hoạt động ở chế độ IPS (Intrusion Prevention System), Suricata có thể chặn địa chỉ IP nguồn, vô hiệu hóa kết nối để ngăn chặn kẻ tấn công làm gián đoạn dịch vụ.

* + - 1. Phát hiện và ngăn chặn malware

Suricata có thể phát hiện phần mềm độc hại dựa trên chữ ký (signature-based detection) hoặc hành vi bất thường (anomaly detection).

Hỗ trợ quét lưu lượng HTTP, SMTP, FTP để kiểm tra tệp tin đáng ngờ.

Có thể tích hợp với Threat Intelligence Feeds (danh sách IP/tên miền độc hại) để tự động cập nhật thông tin về malware.

* + - 1. Ngăn chặn tấn công brute force

Suricata có thể giám sát các nỗ lực đăng nhập thất bại liên tiếp vào hệ thống SSH, FTP, RDP hoặc các dịch vụ web.

Khi phát hiện dấu hiệu tấn công brute force, Suricata có thể cảnh báo hoặc chặn IP nguồn, giúp bảo vệ hệ thống trước các cuộc tấn công dò mật khẩu.

* + 1. Hỗ trợ phân tích forensic
       1. Lưu trữ và phân tích gói tin để điều tra sự cố

Suricata có thể ghi lại toàn bộ lưu lượng mạng để phục vụ công tác forensic (điều tra bảo mật).

Hỗ trợ xuất dữ liệu dưới dạng PCAP hoặc JSON, giúp dễ dàng phân tích bằng Wireshark hoặc các công cụ giám sát khác.

* + - 1. Xác định nguồn gốc tấn công

Bằng cách ghi lại các sự kiện bảo mật và thông tin chi tiết về các kết nối mạng, Suricata giúp quản trị viên xác định:

* Ai là kẻ tấn công?
* Họ đang cố gắng tấn công vào đâu?
* Lưu lượng độc hại có đến từ botnet hoặc máy chủ C2 (Command & Control) không?
  + - 1. Phân tích lưu lượng mã hóa (Encrypted Traffic Analysis - ETA)

Mặc dù không thể giải mã HTTPS/TLS, nhưng Suricata có thể phân tích metadata của lưu lượng được mã hóa để phát hiện các dấu hiệu bất thường.

Ví dụ: Thời gian kết nối, số lượng gói tin, độ dài phiên giao dịch có thể giúp nhận diện mã độc đang giao tiếp với máy chủ điều khiển.

* + 1. Tích hợp với ELK Stack (Elasticsearch, Logstash, Kibana)

Suricata có thể tích hợp với ELK Stack để lưu trữ, phân tích và trực quan hóa dữ liệu bảo mật.

* + - 1. Elasticsearch: Lưu trữ log sự kiện

Suricata có thể gửi dữ liệu log theo thời gian thực đến Elasticsearch, giúp doanh nghiệp lưu trữ và truy vấn dữ liệu dễ dàng.

* + - 1. Logstash: Xử lý và chuẩn hóa dữ liệu

Logstash giúp biến đổi log của Suricata thành định dạng có thể phân tích được, lọc ra các thông tin quan trọng như:

* Địa chỉ IP tấn công
* Thời gian sự kiện
* Loại tấn công phát hiện được
  + - 1. Kibana: Trực quan hóa dữ liệu

Dữ liệu từ Suricata có thể được hiển thị bằng biểu đồ, dashboard, giúp quản trị viên dễ dàng theo dõi các cuộc tấn công theo thời gian thực.

Một số thông tin quan trọng có thể được hiển thị trên Kibana:

* Số lượng cuộc tấn công theo ngày/tuần/tháng
* Địa chỉ IP bị tấn công nhiều nhất
* Các loại tấn công phổ biến (DDoS, brute force, malware, v.v.)
  + 1. Triển khai trên Cloud (AWS, Azure, GCP)

Suricata có thể triển khai trên các nền tảng điện toán đám mây để bảo vệ hệ thống khỏi các mối đe dọa mạng.

* + - 1. AWS (Amazon Web Services)

Suricata có thể chạy trên EC2 Instance để giám sát lưu lượng giữa các máy chủ trong hệ thống.

Có thể tích hợp với AWS VPC Traffic Mirroring để kiểm tra lưu lượng trong môi trường cloud.

Kết hợp với AWS GuardDuty để tự động phát hiện các mối đe dọa bảo mật.

* + - 1. Azure (Microsoft Azure)

Suricata có thể được triển khai trong Azure Virtual Network (VNet) để giám sát lưu lượng nội bộ.

Tích hợp với Azure Sentinel để cung cấp thông tin về các cuộc tấn công đang diễn ra.

* + - 1. GCP (Google Cloud Platform)

Suricata có thể chạy trên Compute Engine để giám sát lưu lượng đến và đi từ các ứng dụng trên cloud.

Tích hợp với Google Chronicle để lưu trữ và phân tích dữ liệu bảo mật.

* 1. Lịch sử phát triển của Suricata

Suricata là một hệ thống **phát hiện và ngăn chặn xâm nhập (IDS/IPS)** mã nguồn mở, được phát triển nhằm cung cấp **hiệu suất cao, khả năng mở rộng, và hỗ trợ phân tích lưu lượng mạng nâng cao.**

* + 1. Giai đoạn khởi đầu (2007 - 2009)
* Suricata được phát triển bởi Open Information Security Foundation (OISF), một tổ chức phi lợi nhuận được thành lập vào năm 2007.
* Mục tiêu chính của OISF là tạo ra một hệ thống IDS/IPS thế hệ mới với hiệu suất cao hơn so với các giải pháp hiện có như Snort.
* Dự án Suricata được tài trợ bởi nhiều tổ chức, bao gồm Bộ An ninh Nội địa Hoa Kỳ (DHS) thông qua chương trình bảo mật mạng mở.
* Năm 2009, phiên bản Suricata 1.0 được phát hành, đánh dấu cột mốc quan trọng trong ngành an ninh mạng.
  + 1. Giai đoạn phát triển và cải tiến (2010 - 2015)
* 2010: Suricata 2.0 được ra mắt, mang lại nhiều cải tiến về hiệu suất và tính linh hoạt trong việc phát hiện tấn công.
* 2012:
* Suricata được bổ sung hỗ trợ đa luồng (multi-threading), giúp tận dụng tối đa tài nguyên phần cứng, vượt trội hơn Snort (chỉ chạy đơn luồng).
* Hỗ trợ PCAP offline mode, giúp phân tích lưu lượng đã được ghi lại trước đó.
* 2014 - 2015:
* Suricata 3.0 được phát hành với khả năng phát hiện tấn công dựa trên hành vi (behavior-based detection) thay vì chỉ sử dụng chữ ký tĩnh.
* Hỗ trợ HTTP, TLS, SSH, DNS logging, giúp giám sát chi tiết hơn các giao thức mạng phổ biến.
  + 1. Giai đoạn mở rộng và tối ưu hóa (2016 - nay)
* 2016:
* Suricata bổ sung khả năng tích hợp với Threat Intelligence Feeds, giúp phát hiện nhanh các IP và tên miền độc hại.
* Tích hợp với Elasticsearch, Logstash, Kibana (ELK Stack) để giám sát và phân tích lưu lượng mạng theo thời gian thực.
* 2018:
* Suricata 4.0 được phát hành với hiệu suất cao hơn, tối ưu bộ nhớ, cải thiện khả năng xử lý các cuộc tấn công DDoS.
* Hỗ trợ GPU Acceleration để xử lý lưu lượng lớn một cách nhanh chóng.
* 2020 - 2023:
* Suricata 6.x và 7.x ra đời với nhiều cải tiến về machine learning, giúp tự động phát hiện các cuộc tấn công nâng cao.
* Cải thiện khả năng phát hiện tấn công trên môi trường cloud như AWS, Azure, GCP.
* Hỗ trợ Encrypted Traffic Analysis (ETA), giúp phân tích lưu lượng được mã hóa để phát hiện dấu hiệu bất thường.
  1. Lý do Suricata được ưa chuộng so với các công cụ khác

Suricata là một trong những hệ thống IDS/IPS (Intrusion Detection and Prevention System) phổ biến nhất hiện nay. Nó được ưa chuộng hơn nhiều công cụ khác như Snort, Zeek (Bro), và Cisco Firepower nhờ vào các ưu điểm nổi bật sau:

* + 1. Hiệu suất cao với kiến trúc đa luồng (Multi-threading)
       1. Khả năng xử lý lưu lượng lớn
* Suricata hỗ trợ đa luồng (multi-threading), cho phép tận dụng tối đa tài nguyên CPU đa nhân để xử lý nhiều gói tin đồng thời.
* Các công cụ như Snort (phiên bản trước 3.0) chỉ chạy đơn luồng, khiến nó dễ bị quá tải khi phải phân tích lưu lượng lớn.
* Kết quả: Suricata có thể xử lý hàng gigabit lưu lượng mạng mỗi giây mà không ảnh hưởng đến hiệu suất.
  + - 1. Tối ưu hóa bộ nhớ và tài nguyên
* Suricata có cơ chế quản lý bộ nhớ thông minh, giúp giảm tải hệ thống khi phân tích gói tin.
* Hỗ trợ zero-copy packet capture, giúp giảm độ trễ khi xử lý dữ liệu từ card mạng.
  + 1. Phát hiện mối đe dọa mạnh mẽ hơn
       1. Hỗ trợ phân tích lưu lượng sâu (DPI - Deep Packet Inspection)

Suricata có thể giải mã và phân tích nội dung của các gói tin, hỗ trợ nhiều giao thức như:

* HTTP, TLS/SSL, FTP, DNS, SMTP, SSH…
* Cho phép phát hiện phần mềm độc hại, mã độc ẩn trong tệp tin đính kèm email, traffic mã hóa bất thường.
  + - 1. Hỗ trợ cả phát hiện theo chữ ký (Signature-based) và hành vi (Anomaly-based)
* Signature-based detection: Sử dụng các tập luật (rule sets) để xác định các mẫu tấn công đã biết.
* Anomaly-based detection: Phát hiện các hành vi bất thường dựa trên AI & Machine Learning.
* Khác biệt so với Snort: Snort chủ yếu dựa vào chữ ký, trong khi Suricata hỗ trợ cả hai phương pháp giúp phát hiện các mối đe dọa zero-day.
* Hỗ trợ tích hợp với Threat Intelligence Feeds:
* Suricata có thể kết nối với các nguồn Threat Intelligence để cập nhật IP, domain, hash của mã độc, botnet… theo thời gian thực.
* Các công cụ khác như Snort cần cấu hình thủ công, trong khi Suricata có các plugin tự động cập nhật Threat Feeds.
  + 1. Khả năng làm việc linh hoạt với nhiều hệ thống
       1. Tích hợp mạnh mẽ với hệ thống SIEM & ELK Stack

Suricata cung cấp khả năng tích hợp sâu với các nền tảng phân tích và giám sát sự kiện bảo mật (SIEM), đặc biệt là:

* + ELK Stack (Elasticsearch – Logstash – Kibana): Gửi log trực tiếp đến hệ thống ELK giúp hiển thị các sự kiện tấn công dưới dạng biểu đồ, bảng, dashboard trực quan.
  + Splunk, Graylog: Các nền tảng SIEM thương mại và mã nguồn mở cũng được hỗ trợ, giúp dễ dàng đồng bộ hóa dữ liệu log.

Việc tích hợp này giúp quản trị viên:

* + Theo dõi các hành vi mạng đáng ngờ theo thời gian thực.
  + Truy vết lại lịch sử các cuộc tấn công và phát hiện mối đe dọa tiềm ẩn.
  + Thiết lập cảnh báo và tự động hóa quy trình phản ứng.

So với các hệ thống tương tự như Snort và Zeek:

* + Suricata có khả năng tích hợp nhanh chóng hơn nhờ hỗ trợ định dạng log chuẩn (EVE JSON).
  + Snort và Zeek thường đòi hỏi các bước cấu hình phức tạp và cần thêm các thành phần trung gian để xử lý log.
    - 1. Triển khai linh hoạt trên nền tảng Cloud & On-Premise

Suricata có thể được triển khai trên cả môi trường truyền thống tại chỗ (On-Premise) và nền tảng đám mây (Cloud):

* + Hỗ trợ các nền tảng đám mây phổ biến như: Amazon Web Services (AWS), Microsoft Azure, Google Cloud Platform (GCP).
  + Tương thích với Kubernetes, giúp dễ dàng triển khai trong các mô hình microservices hoặc container hóa.

Ngoài ra, Suricata cũng hỗ trợ hình thức triển khai dựa trên container như:

* + Docker: Dễ dàng đóng gói, triển khai và quản lý các phiên bản Suricata linh hoạt.
  + Kubernetes: Hỗ trợ khả năng mở rộng tự động (scaling) và giám sát tập trung theo cụm.

Việc triển khai linh hoạt này mang lại nhiều lợi ích:

* + Phù hợp với hạ tầng hiện đại, tiết kiệm tài nguyên và tăng hiệu quả vận hành.
  + Dễ tích hợp vào các mô hình DevSecOps để kiểm tra an ninh ngay trong quá trình phát triển phần mềm.
    - 1. Khả năng hoạt động như một IDS hoặc IPS linh hoạt

Suricata có thể hoạt động linh hoạt ở cả hai chế độ:

* + IDS Mode (Intrusion Detection System):
* Giám sát lưu lượng mạng và phát hiện các hành vi tấn công.
* Gửi cảnh báo nhưng không thực hiện chặn, phù hợp với hệ thống giám sát thụ động.
  + IPS Mode (Intrusion Prevention System):
* Có thể chặn trực tiếp các gói tin độc hại theo thời gian thực.
* Phù hợp với các môi trường yêu cầu phòng thủ chủ động, tránh thiệt hại ngay từ sớm.

Suricata hỗ trợ các chế độ vận hành nâng cao giúp tối ưu hiệu suất và khả năng xử lý:

* + Inline mode với NFQUEUE, AF\_PACKET, hay hỗ trợ offloading với IPSec.
  + Giảm độ trễ, tăng tốc độ phân tích và khả năng phản ứng so với Snort.
    - 1. Cộng đồng mã nguồn mở mạnh mẽ & Hỗ trợ lâu dài

Suricata được phát triển và duy trì bởi Open Information Security Foundation (OISF) – một tổ chức phi lợi nhuận chuyên về an ninh mạng.

Có sự hậu thuẫn và tài trợ từ các tổ chức lớn trong lĩnh vực bảo mật, tạo nên:

* + Một cộng đồng phát triển năng động với hàng nghìn người đóng góp.
  + Nhiều bản cập nhật thường xuyên bao gồm vá lỗi, cải thiện hiệu năng, cập nhật các tập luật mới.

So với Snort:

* + Suricata có định hướng cộng đồng rõ ràng hơn và được xây dựng như một dự án mã nguồn mở thực sự.
  + Trong khi đó, Snort bị Cisco mua lại, hiện đang có xu hướng thương mại hóa mạnh hơn, ảnh hưởng đến tính mở và khả năng tiếp cận của cộng đồng.

Khả năng được hỗ trợ lâu dài và có nhiều tài liệu chính thức giúp Suricata trở thành lựa chọn đáng tin cậy cho các hệ thống doanh nghiệp quy mô vừa và lớn.

1. Tìm hiểu về Kiến Trúc của Suricata.
   1. Tổng quan về Suricata
      1. Tại sao nó quan trọng?

Suricata một công cụ mã nguồn mở mạnh mẽ, được thiết kế để cung cấp khả năng phát hiện và ngăn chặn xâm nhập mạng (IDS/IPS), cũng như giám sát an ninh mạng (NSM) với hiệu suất cao. Được phát triển bởi Tổ chức An ninh Thông tin Mở (OISF) và một cộng đồng tích cực, Suricata không chỉ là một giải pháp miễn phí mà còn rất linh hoạt và có khả năng mở rộng, đáp ứng nhu cầu bảo mật đa dạng của cả tổ chức tư nhân và công cộng.

Công cụ này được thiết kế để giám sát lưu lượng mạng trong thời gian thực, phân tích các gói tin đi qua mạng để phát hiện các dấu hiệu của hoạt động độc hại hoặc các vi phạm chính sách an ninh. Trong trường hợp phát hiện mối đe dọa, Suricata có thể ghi lại các sự kiện này để phân tích sâu hơn, gửi cảnh báo đến các nhà quản trị hệ thống, và thậm chí chủ động ngăn chặn các cuộc tấn công bằng cách chặn lưu lượng nguy hiểm.

Suricata hoạt động dựa trên một bộ quy tắc (ruleset) và ngôn ngữ chữ ký (signature language) mạnh mẽ, cho phép nó xác định các mẫu lưu lượng mạng liên quan đến các mối đe dọa đã biết. Khả năng này, kết hợp với kiến trúc đa luồng hiệu suất cao, giúp Suricata tận dụng tối đa sức mạnh của các hệ thống phần cứng hiện đại, đặc biệt là các bộ vi xử lý đa nhân, để xử lý lượng lớn dữ liệu mạng một cách hiệu quả.

Tầm quan trọng của Suricata ngày càng được khẳng định trong bối cảnh các mối đe dọa an ninh mạng ngày càng tinh vi và phức tạp. Với khả năng hoạt động đa dạng như IDS, IPS và NSM, Suricata cung cấp một giải pháp toàn diện để bảo vệ tài sản số của các tổ chức. Bản chất mã nguồn mở của nó cũng mang lại lợi ích về tính minh bạch, khả năng tùy chỉnh và sự hỗ trợ từ một cộng đồng lớn mạnh, đảm bảo công cụ này liên tục được cập nhật và cải tiến để đối phó với các thách thức an ninh mới nhất.

* + 1. Các chế độ hoạt động chính của Suricata: IDS, IPS, NSM

Suricata có khả năng hoạt động ở nhiều chế độ khác nhau, tùy thuộc vào yêu cầu cụ thể của môi trường mạng và mục tiêu bảo mật:

* **Hệ thống Phát hiện Xâm nhập (IDS - Intrusion Detection System):** Ở chế độ này, Suricata hoạt động như một người quan sát thụ động, chỉ giám sát lưu lượng mạng mà không thực hiện bất kỳ hành động can thiệp nào. Khi phát hiện một gói tin khớp với một quy tắc trong tập luật, Suricata sẽ ghi lại sự kiện này dưới dạng log và có thể gửi cảnh báo đến các hệ thống quản lý sự kiện bảo mật (SIEM) hoặc các nhà quản trị. Mục đích chính của chế độ IDS là cung cấp thông tin về các hoạt động đáng ngờ trên mạng để phân tích và điều tra sau này.
* **Hệ thống Ngăn chặn Xâm nhập (IPS - Intrusion Prevention System):** Khác với chế độ IDS, khi hoạt động ở chế độ IPS, Suricata có khả năng chủ động ngăn chặn các mối đe dọa. Khi phát hiện một gói tin được xác định là nguy hiểm dựa trên tập luật, Suricata có thể thực hiện các hành động như loại bỏ (drop) gói tin đó, ngăn không cho nó đến đích. Để hoạt động hiệu quả ở chế độ IPS, Suricata thường được triển khai cùng với các công cụ khác như iptables trên hệ thống Linux hoặc sử dụng chế độ Suricata Inline Mode để chặn lưu lượng độc hại trực tiếp.
* **Giám sát An ninh Mạng (NSM - Network Security Monitoring):** Chế độ NSM của Suricata tập trung vào việc thu thập và phân tích thông tin chi tiết về lưu lượng mạng, vượt ra ngoài việc chỉ phát hiện các mối đe dọa cụ thể. Ở chế độ này, Suricata có khả năng phân tích các giao thức mạng phổ biến như HTTP, DNS, TLS, SSH, và thậm chí trích xuất các tập tin được truyền qua mạng. Thông tin thu thập được sẽ được lưu trữ dưới dạng log, cung cấp dữ liệu quan trọng cho các hoạt động điều tra và phân tích sau này, giúp hiểu rõ hơn về các hoạt động trên mạng và xác định các mẫu hành vi bất thường.
* Các chế độ hoạt động khác nhau này cho phép Suricata được triển khai trong nhiều kịch bản mạng khác nhau, từ việc giám sát thụ động để thu thập thông tin đến việc chủ động ngăn chặn các cuộc tấn công và phân tích sâu rộng lưu lượng mạng. Sự linh hoạt này là một trong những yếu tố quan trọng làm cho Suricata trở thành một công cụ bảo mật mạng mạnh mẽ và được ưa chuộng.
  1. Phân tích chi tiết Kiến trúc của Suricata
     1. Thu thập gói tin (Packet Acquisition)
        1. Tổng quan về các thành phần và vai trò

Thành phần Thu thập gói tin (Packet Acquisition) đóng vai trò là cửa ngõ đầu tiên trong kiến trúc của Suricata, chịu trách nhiệm chính trong việc nhận dữ liệu thô từ card mạng để sau đó chuyển đến các giai đoạn phân tích tiếp theo 1. Để thực hiện nhiệm vụ này, Suricata sử dụng một loạt các thư viện khác nhau, cho phép nó tương tác với phần cứng mạng và hệ điều hành một cách hiệu quả.

Khả năng của Suricata trong việc hỗ trợ nhiều phương pháp thu thập gói tin mang lại sự linh hoạt cao, cho phép nó hoạt động trong nhiều môi trường mạng khác nhau và đáp ứng các yêu cầu hiệu suất khác nhau. Suricata có thể hoạt động ở cả chế độ trực tuyến (live traffic), nơi nó phân tích các gói tin ngay khi chúng đi qua mạng, và chế độ ngoại tuyến (offline pcap), nơi nó phân tích các tập tin chứa dữ liệu mạng đã được ghi lại trước đó.

Sự linh hoạt trong việc thu thập gói tin là nền tảng cho hoạt động của Suricata. Việc lựa chọn thư viện và phương pháp thu thập phù hợp có thể ảnh hưởng đáng kể đến hiệu suất tổng thể của hệ thống, đặc biệt là trong các môi trường mạng có lưu lượng truy cập lớn.

* + - 1. Phân tích sâu về các thư viện thu thập gói tin: AF\_PACKET, PF\_RING, DPDK và Netmap (hiệu suất, ưu nhược điểm)

Suricata hỗ trợ nhiều thư viện thu thập gói tin khác nhau, mỗi thư viện có những đặc điểm riêng về hiệu suất, ưu điểm và nhược điểm:

* **AF\_PACKET:** Đây là một giao diện socket đặc biệt của hệ điều hành Linux, được thiết kế để cung cấp quyền truy cập trực tiếp đến các thiết bị mạng ở tầng liên kết dữ liệu. So với thư viện libpcap truyền thống, AF\_PACKET có khả năng mang lại hiệu suất tốt hơn trong một số môi trường Linux nhất định 1. Một trong những ưu điểm nổi bật của AF\_PACKET là khả năng sử dụng chế độ fanout, cho phép cân bằng tải lưu lượng truy cập mạng trên nhiều socket và luồng xử lý khác nhau, giúp tận dụng tối đa sức mạnh của các hệ thống đa nhân. Ngoài ra, AF\_PACKET còn hỗ trợ chế độ IPS bằng cách cho phép Suricata sao chép các gói tin nhận được trên một giao diện mạng sang một giao diện khác, tạo thành một cầu nối mạng ở tầng 2. Tuy nhiên, AF\_PACKET là một giải pháp đặc thù cho Linux, do đó không thể sử dụng trên các hệ điều hành khác.
* **PF\_RING:** PF\_RING là một framework thu thập gói tin tốc độ cao, chủ yếu được sử dụng trong môi trường Linux, được thiết kế đặc biệt cho các ứng dụng giám sát và phân tích mạng đòi hỏi hiệu suất cao. Mục tiêu chính của PF\_RING là cải thiện đáng kể tốc độ thu thập gói tin và đồng thời giảm thiểu mức sử dụng tài nguyên CPU so với các phương pháp thu thập gói tin tiêu chuẩn. PF\_RING thường được ưu tiên sử dụng trong các môi trường mạng có lưu lượng truy cập cực lớn, nơi việc đảm bảo thu thập đầy đủ mọi gói tin mà không gây ra tình trạng nghẽn cổ chai về hiệu suất là vô cùng quan trọng.
* **DPDK (Data Plane Development Kit):** DPDK là một bộ thư viện và trình điều khiển mã nguồn mở, cung cấp một cách tiếp cận hiệu quả để xử lý gói tin bằng cách bỏ qua các giới hạn của kernel hệ điều hành truyền thống. Việc tích hợp DPDK vào Suricata đã cho phép công cụ này đạt được mức hiệu suất chưa từng có, đặc biệt là trong việc xử lý các gói tin ở tốc độ rất cao. DPDK được tích hợp vào Suricata như một phương pháp thu thập gói tin tùy chọn, cho phép Suricata giao tiếp trực tiếp với các card mạng tốc độ cao, giảm thiểu đáng kể chi phí xử lý của kernel. DPDK cũng hỗ trợ các chiến lược phân phối gói tin dựa trên hàm băm (hash-based packet distribution) để cân bằng tải lưu lượng một cách hiệu quả trên nhiều core CPU. Hơn nữa, DPDK còn cung cấp khả năng hardware offloading, cho phép chuyển một số tác vụ xử lý lặp đi lặp lại sang phần cứng, giúp giảm tải cho CPU và tăng cường hiệu suất tổng thể.
* **Netmap:** Netmap là một framework khác được thiết kế để cung cấp khả năng truy cập gói tin hiệu suất cao, có sẵn trên cả hệ điều hành FreeBSD và Linux. Trên FreeBSD 11 trở lên, Netmap được tích hợp sẵn và bật mặc định trong kernel. Đối với Linux, Netmap cần được cài đặt riêng từ nguồn. Suricata có thể sử dụng Netmap ở cả chế độ IDS và IPS. Trong chế độ IPS, Netmap hoạt động dựa trên ý tưởng tạo ra một cầu nối phần mềm ở tầng 2 giữa hai giao diện mạng. Suricata sẽ đọc các gói tin trên một giao diện và truyền chúng đi trên giao diện kia, và các gói tin bị chặn bởi chính sách IPS sẽ đơn giản là không được truyền đi. Khi sử dụng Netmap, người dùng có thể chỉ định các tùy chọn đặc biệt trong chuỗi tên giao diện để cấu hình thêm.

Để có cái nhìn tổng quan và so sánh rõ ràng hơn về các thư viện thu thập gói tin này, bảng sau đây tóm tắt các đặc điểm chính:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Tên thư viện** | **Hệ điều hành hỗ trợ chính** | **Ưu điểm** | **Nhược điểm** | **Trường hợp sử dụng phù hợp** |
| AF\_PACKET | Linux | Hiệu suất tốt hơn libpcap, cân bằng tải với fanout, hỗ trợ IPS | Chỉ dành cho Linux | Mạng Linux có lưu lượng trung bình đến cao, cần hiệu suất tốt và khả năng cân bằng tải. |
| PF\_RING | Linux | Thu thập gói tin tốc độ cao, giảm tải CPU | Chủ yếu cho Linux | Mạng Linux có lưu lượng cực cao, yêu cầu thu thập gói tin với tốc độ tối đa. |
| DPDK | Linux, FreeBSD, Windows | Hiệu suất vượt trội, bỏ qua kernel, hardware offloading, cân bằng tải | Phức tạp trong cấu hình, có thể yêu cầu phần cứng đặc biệt. | Mạng có lưu lượng cực cao, yêu cầu hiệu suất cao nhất và sẵn sàng đầu tư vào phần cứng phù hợp. |
| Netmap | FreeBSD, Linux | Hiệu suất cao, tích hợp tốt với FreeBSD, hỗ trợ IDS và IPS | Cần cài đặt riêng trên Linux, giao diện mạng có thể không khả dụng cho OS. | Mạng FreeBSD hoặc Linux có yêu cầu hiệu suất cao, đặc biệt trong các triển khai IPS. |

Sự đa dạng của các thư viện thu thập gói tin mà Suricata hỗ trợ cho thấy khả năng thích ứng của nó với nhiều hệ điều hành và yêu cầu hiệu suất khác nhau. Việc lựa chọn thư viện phù hợp thường phụ thuộc vào trường hợp sử dụng cụ thể, phần cứng cơ sở và mức hiệu suất mong muốn.

* + 1. Giải mã gói tin (Packet Decoder)
       1. Quy trình và chức năng của bộ giải mã

Sau khi các gói tin được thu thập, giai đoạn tiếp theo trong kiến trúc của Suricata là Giải mã gói tin (Packet Decoder). Thành phần này có vai trò quan trọng trong việc phân tích và hiểu cấu trúc của các gói tin đã thu thập. Quy trình giải mã bắt đầu bằng việc phân tích các gói tin theo mô hình OSI (Open Systems Interconnection), từ tầng vật lý đến tầng ứng dụng.

Bộ giải mã sẽ xác định và phân tách các lớp khác nhau của gói tin, chẳng hạn như Ethernet, IP (Internet Protocol), TCP (Transmission Control Protocol) hoặc UDP (User Datagram Protocol). Đối với mỗi lớp, bộ giải mã sẽ giải mã các tiêu đề và dữ liệu tương ứng, cho phép Suricata hiểu được thông tin chứa trong gói tin, bao gồm địa chỉ nguồn và đích, cổng, giao thức sử dụng và dữ liệu payload.

Chức năng chính của bộ giải mã là làm cho dữ liệu mạng thô trở nên có cấu trúc và dễ hiểu đối với các thành phần khác của Suricata, đặc biệt là Bộ máy phát hiện (Detection Engine). Bằng cách giải mã các giao thức mạng khác nhau, Suricata có thể kiểm tra sâu hơn vào nội dung của các gói tin để tìm kiếm các dấu hiệu của hoạt động độc hại hoặc bất thường.

* + - 1. Các giao thức mạng được Suricata hỗ trợ (mở rộng danh sách ngoài DNS, HTTP, TLS, SSH và SMB)

Ngoài các giao thức mạng phổ biến như DNS, HTTP, TLS, SSH và SMB đã được đề cập, Suricata còn có khả năng giải mã một lượng lớn các giao thức khác ở nhiều tầng khác nhau của mô hình OSI. Sự hỗ trợ rộng rãi này cho phép Suricata thực hiện kiểm tra sâu gói tin trên nhiều loại lưu lượng mạng khác nhau, tăng cường khả năng phát hiện các mối đe dọa đa dạng.

* **Tầng mạng:** Suricata hỗ trợ giải mã các giao thức như IPv4 và IPv6, là hai phiên bản chính của giao thức Internet. Nó cũng có khả năng xử lý các giao thức liên quan như ICMPv4 và ICMPv6, được sử dụng cho các thông báo lỗi và kiểm soát trong mạng IP. Ngoài ra, Suricata còn hỗ trợ giải mã các giao thức tunneling như GRE (Generic Routing Encapsulation), cũng như các giao thức đóng gói khác như IP-in-IP, IPv6-in-IPv4 và IPv4-in-IPv6 .
* **Tầng giao vận:** Ở tầng giao vận, Suricata có khả năng giải mã các giao thức chính là TCP (Transmission Control Protocol), được sử dụng cho các kết nối đáng tin cậy, hướng kết nối, và UDP (User Datagram Protocol), được sử dụng cho các truyền thông không kết nối, tốc độ cao. Giao thức SCTP (Stream Control Transmission Protocol), một giao thức giao vận đáng tin cậy khác, cũng được Suricata hỗ trợ .
* **Tầng liên kết dữ liệu:** Suricata có thể giải mã các giao thức ở tầng liên kết dữ liệu như Ethernet, giao thức phổ biến cho mạng cục bộ (LAN). Nó cũng hỗ trợ PPP (Point-to-Point Protocol) và PPPoE (PPP over Ethernet), thường được sử dụng cho các kết nối dial-up và DSL . Các giao thức khác được hỗ trợ bao gồm Raw IP (cho phép xử lý các gói tin IP thô), SLL (Linux Socket Layer), VLAN (Virtual Local Area Network) và QINQ (IEEE 802.1ad), được sử dụng để phân đoạn và gắn thẻ lưu lượng mạng. Suricata cũng hỗ trợ MPLS (Multiprotocol Label Switching), ERSPAN (Encapsulated Remote SPAN), VXLAN (Virtual Extensible LAN) và Geneve, là các công nghệ overlay mạng. Gần đây, Suricata còn được bổ sung khả năng giải mã cho giao thức 802.1BR E-tag.
* **Tầng ứng dụng:** Bên cạnh DNS, HTTP (bao gồm cả HTTP/2), TLS/SSL, SSH và SMB, Suricata còn hỗ trợ một loạt các giao thức ứng dụng khác. Chúng bao gồm FTP (File Transfer Protocol) để truyền tệp tin, SMTP (Simple Mail Transfer Protocol) để gửi email, DHCP (Dynamic Host Configuration Protocol) để cấp phát địa chỉ IP, TFTP (Trivial File Transfer Protocol), KRB5 (Kerberos), IKEv2 (Internet Key Exchange version 2) cho VPN, SIP (Session Initiation Protocol) cho VoIP, SNMP (Simple Network Management Protocol) để quản lý thiết bị mạng, RDP (Remote Desktop Protocol) cho kết nối máy tính từ xa, RFB (Remote Framebuffer) cho VNC, MQTT (Message Queuing Telemetry Transport) cho IoT, DCERPC (Distributed Computing Environment Remote Procedure Call), NFS (Network File System), NTP (Network Time Protocol), Modbus, ENIP/CIP (EtherNet/IP Common Industrial Protocol) và DNP3 (Distributed Network Protocol 3), thường được sử dụng trong các hệ thống công nghiệp.
* Việc hỗ trợ một danh sách dài các giao thức mạng cho phép Suricata thực hiện kiểm tra sâu gói tin trên nhiều loại hình truyền thông khác nhau, từ các giao thức cơ bản của Internet đến các giao thức chuyên dụng được sử dụng trong các ứng dụng và ngành công nghiệp cụ thể. Khả năng này giúp Suricata phát hiện các mối đe dọa tiềm ẩn ẩn chứa trong nhiều loại lưu lượng mạng khác nhau.
  + 1. Bộ máy phát hiện (Detection Engine)
       1. Cơ chế so sánh gói tin với tập luật

Bộ máy phát hiện (Detection Engine) là trái tim của Suricata, nơi các gói tin đã được giải mã sẽ được so sánh với một tập hợp các quy tắc (rules) để xác định xem có bất kỳ hoạt động nào đáng ngờ hoặc độc hại hay không. Các quy tắc này, thường được gọi là signatures, định nghĩa các mẫu cụ thể của lưu lượng mạng hoặc các hành vi mà Suricata nên tìm kiếm.

Cơ chế hoạt động của bộ máy phát hiện bao gồm việc lấy thông tin đã được giải mã từ mỗi gói tin và đối chiếu nó với các điều kiện và mẫu được định nghĩa trong tập luật. Các quy tắc có thể dựa trên nhiều yếu tố khác nhau, bao gồm địa chỉ IP nguồn và đích, cổng, giao thức, và đặc biệt là nội dung của gói tin (payload).

Suricata hỗ trợ hai phương pháp chính để phát hiện các mối đe dọa: phát hiện dựa trên chữ ký (rule-based signatures) và phân tích hành vi (behavioral analysis). Phát hiện dựa trên chữ ký là phương pháp truyền thống, tập trung vào việc tìm kiếm các mẫu đã biết của các cuộc tấn công hoặc phần mềm độc hại. Phân tích hành vi, mặt khác, cố gắng xác định các hoạt động bất thường hoặc đáng ngờ dựa trên cách lưu lượng mạng hoạt động, ngay cả khi không có chữ ký cụ thể nào khớp.

* + - 1. Nghiên cứu về các thuật toán Multi-pattern Matching để phát hiện xâm nhập hiệu quả

Để có thể so sánh các gói tin với hàng ngàn, thậm chí hàng chục ngàn quy tắc một cách hiệu quả, Suricata sử dụng các thuật toán Multi-pattern Matching (khớp nhiều mẫu). Các thuật toán này cho phép Suricata kiểm tra một gói tin duy nhất đối với nhiều quy tắc cùng một lúc, thay vì phải xử lý từng quy tắc một cách tuần tự, giúp tăng đáng kể hiệu suất và tốc độ xử lý.

Một trong những thuật toán multi-pattern matching quan trọng mà Suricata sử dụng là thuật toán Aho-Corasick (AC). Thuật toán này xây dựng một cấu trúc dữ liệu dạng cây (trie) từ tập hợp các mẫu (signatures) và sau đó sử dụng cấu trúc này để quét văn bản (trong trường hợp của Suricata là dữ liệu gói tin) một cách hiệu quả, tìm ra tất cả các vị trí mà bất kỳ mẫu nào xuất hiện.

Ngoài ra, Suricata còn có một cơ chế được gọi là Fast Pattern. Cơ chế này giúp tối ưu hóa quá trình so khớp bằng cách cho phép người viết quy tắc hoặc Suricata tự động xác định content (nội dung) quan trọng nhất trong một quy tắc để thực hiện việc so khớp đầu tiên. Nếu content này không khớp, Suricata có thể bỏ qua quy tắc đó cho gói tin hiện tại, giúp tiết kiệm tài nguyên xử lý. Suricata sẽ ưu tiên sử dụng content dài nhất và có độ đa dạng ký tự cao nhất làm fast pattern.

Việc sử dụng các thuật toán multi-pattern matching là yếu tố then chốt giúp Suricata có thể xử lý lưu lượng mạng lớn và tập luật phức tạp mà không làm giảm đáng kể hiệu suất. Điều này đặc biệt quan trọng trong các môi trường mạng tốc độ cao, nơi mà việc xử lý gói tin nhanh chóng là rất cần thiết.

* + - 1. Tìm hiểu về cách Suricata thực hiện phân tích hành vi (behavioral analysis)

Bên cạnh việc so khớp với các chữ ký đã biết, Suricata còn được trang bị khả năng phân tích hành vi (behavioral analysis) để phát hiện các mối đe dọa phức tạp hơn, có thể không khớp với bất kỳ chữ ký cụ thể nào. Phương pháp này tập trung vào việc xác định các hành vi bất thường hoặc đáng ngờ trong lưu lượng mạng so với các mẫu lưu lượng bình thường đã được thiết lập.

Suricata có thể phân tích lưu lượng giao thức ở tầng ứng dụng để tìm kiếm các dấu hiệu của các hành vi đáng ngờ. Ví dụ, nó có thể theo dõi các luồng mạng (flow tracking) để xác định các kết nối đến các địa chỉ IP hoặc cổng không phổ biến, hoặc phát hiện các chuỗi hành động bất thường có thể chỉ ra một cuộc tấn công đang diễn ra.

Một khía cạnh quan trọng của phân tích hành vi là khả năng thiết lập baseline cho lưu lượng mạng bình thường. Bằng cách giám sát lưu lượng trong một khoảng thời gian, Suricata có thể học được các mẫu hoạt động thông thường của mạng và sau đó phát hiện ra bất kỳ sự sai lệch đáng kể nào có thể là dấu hiệu của một mối đe dọa.

Phân tích hành vi đặc biệt hữu ích trong việc phát hiện các cuộc tấn công zero-day (các cuộc tấn công khai thác các lỗ hổng chưa được biết đến) hoặc các phần mềm độc hại tinh vi cố gắng ẩn mình bằng cách không khớp với các chữ ký truyền thống. Bằng cách tập trung vào cách thức hoạt động của lưu lượng mạng thay vì chỉ tìm kiếm các mẫu tĩnh, Suricata có thể cung cấp một lớp bảo vệ bổ sung chống lại các mối đe dọa mới và đang phát triển.

* + 1. Ghi log và cảnh báo (Logging & Alerting)
       1. Các định dạng log được hỗ trợ và cách cấu hình

Suricata cung cấp khả năng ghi lại các sự kiện và cảnh báo dưới nhiều định dạng khác nhau, cho phép người dùng linh hoạt lựa chọn định dạng phù hợp với nhu cầu phân tích và tích hợp với các hệ thống khác. Các định dạng log chính mà Suricata hỗ trợ bao gồm:

* **EVE JSON (Extensible Event Format JSON):** Đây là một định dạng log dựa trên JSON rất linh hoạt và chi tiết, được sử dụng rộng rãi trong Suricata. EVE JSON có thể chứa nhiều loại sự kiện khác nhau, bao gồm các cảnh báo (alerts), thông tin về các giao dịch HTTP, DNS, TLS, các luồng mạng (flows), thông tin về tệp tin và nhiều loại sự kiện khác, tất cả trong một định dạng có cấu trúc dễ dàng phân tích và xử lý bằng các công cụ khác.
* **Syslog:** Suricata cũng hỗ trợ ghi log theo định dạng Syslog, một giao thức tiêu chuẩn để gửi thông điệp log trong các hệ thống Unix-like. Định dạng này cho phép Suricata tích hợp dễ dàng với các hệ thống quản lý log tập trung.
* **Unified2:** Đây là một định dạng log nhị phân thường được sử dụng với Snort và các công cụ liên quan như Barnyard2. Suricata hỗ trợ định dạng này để đảm bảo khả năng tương thích với các hệ thống đã quen thuộc với Snort.
* **PCAP (Packet Capture):** Suricata có khả năng lưu trữ toàn bộ gói tin đã chụp vào các tập tin theo định dạng PCAP. Điều này đặc biệt hữu ích cho việc phân tích pháp y sau sự cố, cho phép các nhà phân tích xem xét chi tiết nội dung của các gói tin liên quan đến một sự kiện bảo mật. Suricata còn hỗ trợ conditional PCAP, cho phép chỉ lưu trữ các gói tin đáp ứng một số tiêu chí nhất định.
* **fast.log:** Đây là một định dạng log ngắn gọn, chứa thông tin cơ bản về các kết nối đã được ghi log, bao gồm thời gian, thông điệp cảnh báo, mức độ nghiêm trọng, giao thức và địa chỉ IP nguồn và đích. Tuy nhiên, định dạng này được coi là cũ và không còn được khuyến nghị cho các tác vụ phức tạp như phản hồi sự cố hoặc săn lùng mối đe dọa.

Việc cấu hình các tùy chọn ghi log được thực hiện thông qua file cấu hình chính của Suricata là suricata.yaml. Trong file này, người dùng có thể chỉ định định dạng log mong muốn, vị trí lưu trữ các tập tin log, cũng như các tùy chọn cấu hình khác liên quan đến việc ghi log, chẳng hạn như bật/tắt các loại sự kiện cụ thể trong log EVE JSON.

Sự đa dạng của các định dạng log mà Suricata hỗ trợ mang lại sự linh hoạt lớn cho người dùng trong việc lựa chọn cách thức quản lý và phân tích dữ liệu bảo mật, cũng như khả năng tích hợp với nhiều công cụ và hệ thống khác nhau.

* + - 1. Tích hợp với các hệ thống SIEM

Một trong những tính năng quan trọng của Suricata là khả năng tích hợp với các hệ thống quản lý thông tin và sự kiện bảo mật (SIEM - Security Information and Event Management) như Splunk, Elasticsearch và Kibana (thường được gọi là ELK stack). Việc tích hợp này cho phép các tổ chức tập trung dữ liệu bảo mật từ nhiều nguồn khác nhau, bao gồm cả Suricata, để thực hiện phân tích, tương quan và đưa ra các cảnh báo một cách hiệu quả hơn.

Định dạng JSON, đặc biệt là EVE JSON mà Suricata cung cấp, rất phù hợp cho việc tích hợp với các hệ thống SIEM. Các hệ thống SIEM có thể dễ dàng phân tích và xử lý dữ liệu có cấu trúc theo định dạng JSON, cho phép trích xuất thông tin quan trọng và tạo ra các dashboard trực quan để theo dõi tình trạng an ninh mạng.

Việc tích hợp Suricata với các hệ thống SIEM như Elasticsearch cho phép người dùng lập chỉ mục và tìm kiếm trong lượng lớn dữ liệu log một cách nhanh chóng và hiệu quả. Logstash có thể được sử dụng để thu thập, xử lý và chuẩn hóa các log từ Suricata trước khi chúng được gửi đến Elasticsearch. Kibana cung cấp một giao diện web mạnh mẽ để trực quan hóa dữ liệu, cho phép các nhà phân tích bảo mật tạo ra các biểu đồ, bảng và dashboard tùy chỉnh để theo dõi các xu hướng, phát hiện các hoạt động bất thường và điều tra các sự cố bảo mật.

Khả năng tích hợp liền mạch với các hệ thống SIEM là một lợi thế lớn của Suricata, giúp nó trở thành một thành phần quan trọng trong một kiến trúc an ninh mạng toàn diện, cho phép các tổ chức có được cái nhìn tổng quan về tình hình an ninh và phản ứng kịp thời với các mối đe dọa.

* + 1. Xử lý đầu ra (Output Processing)
       1. Các phương pháp xử lý và xuất dữ liệu

Suricata cung cấp nhiều phương pháp để xử lý và xuất dữ liệu sau khi đã thu thập và phân tích lưu lượng mạng. Các phương pháp này bao gồm việc ghi log vào các tập tin theo nhiều định dạng khác nhau (như đã đề cập ở trên), gửi log đến hệ thống theo giao thức Syslog, và tích hợp trực tiếp với các công cụ phân tích khác.

Suricata sử dụng các output module (mô-đun đầu ra) để định cấu hình cách dữ liệu được xuất. Người dùng có thể tùy chỉnh các mô-đun này trong file cấu hình suricata.yaml để chỉ định định dạng, vị trí lưu trữ và các tùy chọn khác cho từng loại đầu ra.

Để tối ưu hóa hiệu suất, Suricata còn cung cấp khả năng cấu hình buffering đầu ra. Việc sử dụng bộ đệm có thể giúp giảm tải cho hệ thống bằng cách trì hoãn việc ghi dữ liệu ra đĩa hoặc gửi đi, tuy nhiên, cần cân nhắc đến nguy cơ mất dữ liệu nếu hệ thống gặp sự cố trước khi bộ đệm được xả.

* + - 1. Tích hợp với các công cụ phân tích như ELK stack

Như đã đề cập ở phần trước, Suricata có khả năng tích hợp rất tốt với ELK stack (Elasticsearch, Logstash, Kibana) để phục vụ cho việc phân tích và trực quan hóa dữ liệu bảo mật.

* **Elasticsearch:** Được sử dụng như một công cụ lập chỉ mục và tìm kiếm mạnh mẽ, cho phép các nhà phân tích nhanh chóng tìm kiếm và lọc qua lượng lớn dữ liệu log do Suricata tạo ra.
* **Logstash:** Có thể được sử dụng như một bộ xử lý log trung gian, thu thập dữ liệu từ Suricata (thường là từ các tập tin log), sau đó chuẩn hóa và chuyển đổi dữ liệu này sang một định dạng phù hợp trước khi gửi đến Elasticsearch.
* **Kibana:** Cung cấp một giao diện người dùng web trực quan, cho phép các nhà phân tích tạo ra các dashboard, biểu đồ và bảng để theo dõi các chỉ số bảo mật quan trọng, khám phá các mẫu lưu lượng mạng và điều tra các sự cố bảo mật một cách hiệu quả.

Việc tích hợp với ELK stack cung cấp một giải pháp mạnh mẽ để quản lý và phân tích lượng lớn dữ liệu mà một hệ thống IDS/IPS như Suricata có thể tạo ra, giúp các tổ chức nâng cao khả năng giám sát và ứng phó với các mối đe dọa an ninh mạng.

* 1. So sánh Suricata với Snort
     1. Phân tích chi tiết về kiến trúc, hiệu suất, tính năng và các trường hợp sử dụng khác nhau

Suricata và Snort là hai hệ thống phát hiện xâm nhập mã nguồn mở phổ biến nhất hiện nay. Mặc dù có nhiều điểm tương đồng, chúng cũng có những khác biệt đáng kể về kiến trúc, hiệu suất, tính năng và các trường hợp sử dụng phù hợp.

* **Kiến trúc:** Một trong những khác biệt lớn nhất giữa Suricata và Snort là kiến trúc đa luồng (multi-threaded) của Suricata, cho phép nó tận dụng hiệu quả sức mạnh của các bộ vi xử lý đa nhân hiện đại. Trong khi đó, các phiên bản cũ của Snort (v2) chủ yếu sử dụng kiến trúc đơn luồng (single-threaded), điều này có thể trở thành một hạn chế về hiệu suất trong các môi trường mạng có lưu lượng truy cập lớn. Snort phiên bản 3 đã giới thiệu khả năng đa luồng, nhưng Suricata thường được đánh giá cao hơn về khả năng này. Kiến trúc đa luồng của Suricata mang lại lợi thế đáng kể về hiệu suất, đặc biệt khi xử lý lưu lượng mạng lớn trên các hệ thống có nhiều core CPU.
* **Hiệu suất:** Nhiều nghiên cứu đã chỉ ra rằng Suricata thường có hiệu suất tốt hơn Snort, đặc biệt là trong các tình huống xử lý lưu lượng truy cập cao. Suricata được thiết kế để có thể xử lý lưu lượng truy cập ở tốc độ gigabit một cách hiệu quả hơn so với Snort. Mặc dù Snort có thể có mức sử dụng CPU và bộ nhớ thấp hơn trong một số trường hợp nhất định, nhưng nó có xu hướng bị drop gói tin nhiều hơn khi tốc độ lưu lượng tăng cao. Do đó, Suricata thường được ưa chuộng hơn trong các môi trường mạng hiện đại, nơi băng thông ngày càng tăng.
* **Tính năng:** Suricata nổi bật với các tính năng giám sát an ninh mạng (NSM) mạnh mẽ, cho phép tạo ra dữ liệu phong phú ở định dạng JSON (EVE), bao gồm thông tin về các giao dịch HTTP, DNS, TLS và nhiều giao thức khác. Suricata cũng hỗ trợ conditional PCAP, cho phép người dùng cấu hình để chỉ ghi lại các gói tin đáp ứng các tiêu chí cụ thể, giúp tiết kiệm dung lượng lưu trữ. Một tính năng mạnh mẽ khác của Suricata là khả năng sử dụng Lua scripting để tạo ra các logic phát hiện tùy chỉnh, mang lại sự linh hoạt cao trong việc xây dựng các quy tắc phát hiện phức tạp. Snort, với lịch sử phát triển lâu đời hơn, có một tập luật rất lớn và được cộng đồng hỗ trợ rộng rãi, đồng thời có khả năng tương thích tốt với nhiều loại thiết bị và hệ điều hành. Điều quan trọng là Suricata có khả năng sử dụng hầu hết các quy tắc được viết cho Snort với một số điều chỉnh nhỏ, giúp người dùng dễ dàng chuyển đổi giữa hai hệ thống.
* **Trường hợp sử dụng:** Suricata thường là lựa chọn ưu tiên cho các mạng có băng thông cao và yêu cầu hiệu suất xử lý lớn. Nó đặc biệt phù hợp cho các tổ chức cần khả năng giám sát an ninh mạng sâu rộng và khả năng tùy chỉnh cao. Snort vẫn là một lựa chọn phù hợp cho các mạng nhỏ hơn hoặc các môi trường legacy, nơi yêu cầu về tài nguyên có thể là một yếu tố quan trọng hơn hiệu suất tuyệt đối.

Để tóm tắt những khác biệt chính, bảng so sánh sau đây cung cấp một cái nhìn tổng quan:

|  |  |  |
| --- | --- | --- |
| **Tiêu chí** | **Suricata** | **Snort** |
| Kiến trúc | Đa luồng (Multi-threaded) | Đơn luồng (Single-threaded) (v2), Đa luồng (Multi-threaded) (v3) |
| Hiệu suất | Thường tốt hơn, đặc biệt với lưu lượng lớn | Có thể thấp hơn trong lưu lượng lớn, có thể drop gói tin nhiều hơn |
| NSM | Mạnh mẽ, dữ liệu phong phú ở định dạng JSON (EVE) | Hạn chế hơn |
| Lua Scripting | Hỗ trợ cho logic phát hiện tùy chỉnh | Không hỗ trợ |
| Rule Compatibility | Tương thích tốt với luật Snort | Luật riêng, nhưng Suricata có thể sử dụng hầu hết. |
| Resource Usage | Có thể cao hơn Snort trong một số trường hợp | Có thể thấp hơn trong một số trường hợp |
| Cộng đồng | Tích cực, đang phát triển | Lớn mạnh, lâu đời |
| Trường hợp SD | Mạng băng thông cao, yêu cầu hiệu suất lớn, cần NSM và tùy chỉnh cao | Mạng nhỏ hơn, môi trường legacy, yêu cầu tài nguyên thấp có thể quan trọng hơn |

Tóm lại, sự lựa chọn giữa Suricata và Snort thường phụ thuộc vào nhu cầu và điều kiện cụ thể của từng môi trường mạng. Suricata thường được ưa chuộng vì hiệu suất và các tính năng nâng cao, trong khi Snort vẫn là một lựa chọn đáng tin cậy với cộng đồng lớn và lịch sử lâu đời.

1. Hướng Dẫn Cài đặt Suricata trên hệ điều hành Ubuntu.
   1. Các yêu cầu hệ thống

Để cài đặt và chạy Suricata một cách hiệu quả trên hệ điều hành Ubuntu, hệ thống của bạn cần đáp ứng một số yêu cầu tối thiểu sau:

* **Hệ điều hành:** Cần có một máy tính hoặc máy ảo đang chạy một phiên bản Ubuntu được hỗ trợ. Các phiên bản Ubuntu mới hơn thường được khuyến nghị để đảm bảo khả năng tương thích tốt nhất và các bản cập nhật bảo mật mới nhất.
* **Kết nối Internet:** Một kết nối internet ổn định là cần thiết để tải xuống các gói cài đặt Suricata và các bộ luật (ruleset).
* **Bộ nhớ RAM:** Đối với môi trường production (sản xuất), nên có ít nhất 4-8GB RAM để Suricata có thể xử lý lưu lượng mạng một cách hiệu quả. Trong môi trường thử nghiệm hoặc nhỏ, có thể sử dụng ít RAM hơn, nhưng hiệu suất có thể bị ảnh hưởng.
* **CPU:** Tương tự như RAM, đối với môi trường production, nên có ít nhất 2 CPU cores để Suricata có thể tận dụng khả năng đa luồng của mình. Số lượng core CPU càng nhiều thường sẽ giúp Suricata xử lý lưu lượng mạng lớn tốt hơn.
* **Dung lượng đĩa:** Cần có đủ dung lượng đĩa để cài đặt Suricata và lưu trữ các tập tin log. Dung lượng cần thiết sẽ phụ thuộc vào lượng lưu lượng mạng mà Suricata giám sát và cấu hình ghi log của bạn.

Việc đảm bảo hệ thống đáp ứng các yêu cầu này sẽ giúp quá trình cài đặt diễn ra suôn sẻ và Suricata có thể hoạt động với hiệu suất tốt nhất trong môi trường của bạn.

* 1. Hướng dẫn chi tiết các bước cài đặt Suricata trên Ubuntu
     1. Cài đặt từ kho chính thức và PPA

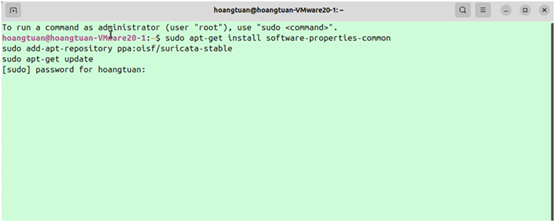
Có hai phương pháp chính để cài đặt Suricata trên Ubuntu: cài đặt từ kho phần mềm chính thức của Ubuntu hoặc cài đặt từ kho PPA (Personal Package Archive) của OISF (Open Information Security Foundation), tổ chức phát triển Suricata:

* **Cài đặt từ kho chính thức:** Phương pháp này đơn giản nhất và có thể được thực hiện bằng các lệnh sau :

sudo apt update  
sudo apt install suricata -y  
  
Tuy nhiên, phiên bản Suricata trong kho chính thức của Ubuntu có thể không phải là phiên bản mới nhất.

* **Cài đặt từ PPA của OISF (khuyến nghị):** Để cài đặt phiên bản ổn định mới nhất của Suricata, bạn nên sử dụng PPA của OISF. Các bước thực hiện như sau:

1. Cài đặt gói software-properties-common nếu nó chưa được cài đặt:  
    *sudo apt-get install software-properties-common*
2. Thêm kho PPA của OISF vào hệ thống:  
    *sudo add-apt-repository ppa:oisf/suricata-stable*
3. Cập nhật danh sách các gói phần mềm từ các kho:  
    *sudo apt update*



1. - Set up để cài đặt suricata

1. Cài đặt Suricata:  
    *sudo apt-get install suricata*



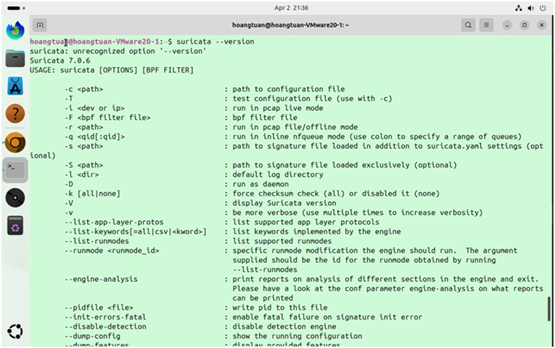
1. - Cài đặt suricata

Phương pháp cài đặt từ PPA được khuyến nghị vì nó thường cung cấp phiên bản Suricata mới nhất và ổn định nhất, bao gồm các tính năng mới và các bản vá bảo mật mới nhất.

* + 1. Các bước cấu hình cơ bản sau khi cài đặt

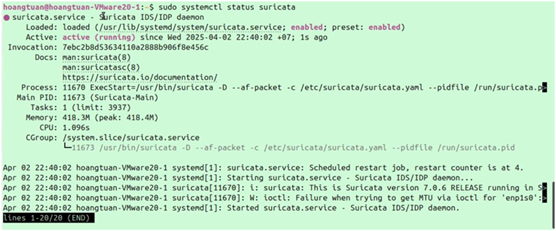
Sau khi cài đặt Suricata, bạn cần thực hiện một số bước cấu hình cơ bản để đảm bảo nó hoạt động đúng cách trong môi trường của bạn:

* **Kiểm tra phiên bản Suricata:** Bạn có thể kiểm tra phiên bản Suricata đã cài đặt bằng lệnh:  
   *suricata --version*  
    
  hoặc  
   *suricata -V*



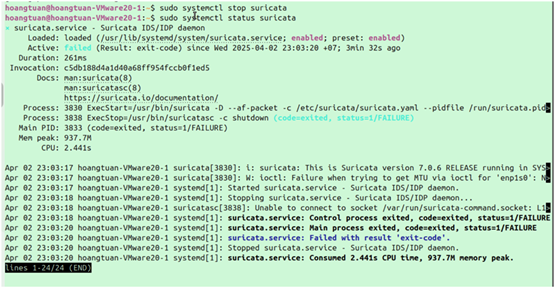
1. - Kiểm tra phiên bản Suricata

* **Kiểm tra trạng thái dịch vụ Suricata:** Để xem dịch vụ Suricata có đang chạy hay không, sử dụng lệnh:  
   *sudo systemctl status suricata*



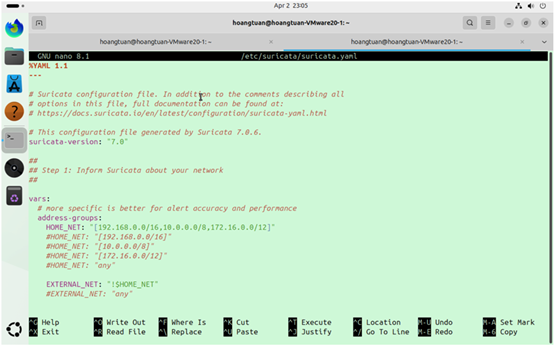
1. - Kiểm tra trạng thái hoạt động của suricata

* **Dừng dịch vụ Suricata (nếu cần cấu hình):** Nếu bạn cần chỉnh sửa file cấu hình, bạn có thể dừng dịch vụ Suricata bằng lệnh:  
   *sudo systemctl stop suricata*



1. - Dừng dịch vụ Suricata

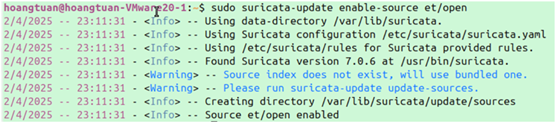
* **Chỉnh sửa file cấu hình chính:** File cấu hình chính của Suricata nằm tại /etc/suricata/suricata.yaml. Bạn có thể mở file này bằng một trình soạn thảo văn bản với quyền sudo:  
   *sudo nano /etc/suricata/suricata.yaml*

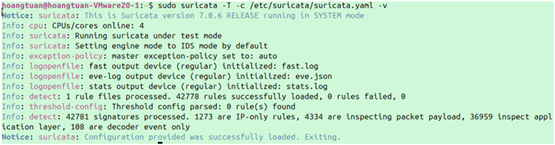


1. - Chỉnh sửa file cấu hình chính

Trong file này, bạn cần thực hiện một số cấu hình quan trọng:

* **Định cấu hình HOME\_NET:** Tìm đến biến HOME\_NET và thay đổi địa chỉ mạng mặc định thành địa chỉ mạng nội bộ của bạn. Ví dụ, nếu mạng nội bộ của bạn là 192.168.1.0/24, hãy thay đổi dòng tương ứng. Điều này giúp Suricata phân biệt lưu lượng mạng nội bộ và bên ngoài.
* **Định cấu hình giao diện mạng:** Tìm đến phần cấu hình giao diện mạng (thường nằm trong phần af-packet nếu bạn muốn sử dụng phương pháp này hoặc trong cấu hình mặc định). Chỉ định giao diện mạng mà Suricata sẽ giám sát. Ví dụ, nếu giao diện của bạn là eth0, hãy đảm bảo nó được cấu hình đúng. Bạn có thể cần thay đổi các tùy chọn khác trong phần này tùy thuộc vào phương pháp thu thập gói tin bạn chọn.
* **Kích hoạt Community ID:** Tìm đến phần cấu hình community-id trong phần outputs và đảm bảo nó được bật (true). Tính năng này thêm một trường community\_id vào log EVE, giúp tương quan các sự kiện với các công cụ khác như Zeek.
* **Cấu hình logging:** Xem xét các tùy chọn logging trong phần outputs để đảm bảo Suricata đang ghi log theo định dạng và vị trí bạn mong muốn.
* **Cập nhật bộ luật (ruleset):** Suricata sử dụng các bộ luật để phát hiện các mối đe dọa. Bạn cần tải xuống và cập nhật các bộ luật này thường xuyên. Công cụ suricata-update được cung cấp để giúp bạn thực hiện việc này:
* Để xem danh sách các nhà cung cấp luật mặc định, chạy lệnh:  
   *sudo suricata-update --list-providers*
* **Để** kích hoạt một nhà cung cấp luật, ví dụ như Emerging Threats Open, chạy lệnh:  
   *sudo suricata-update enable-source et/open*
* Để thực hiện cập nhật luật, chạy lệnh:  
   *sudo suricata-update*  
  hoặc  
   *sudo suricata-update -o /etc/suricata/rules*Lệnh này sẽ tải xuống các luật mới nhất và lưu chúng vào thư mục cấu hình của Suricata.
* **Kiểm tra cấu hình:** Sau khi thực hiện các thay đổi trong file cấu hình, bạn nên kiểm tra xem cấu hình có hợp lệ hay không bằng lệnh:  
   *sudo suricata -T -c /etc/suricata/suricata.yaml -v*  
  Nếu không có lỗi nào được báo cáo, cấu hình của bạn là hợp lệ.
* **Bắt đầu dịch vụ Suricata:** Sau khi kiểm tra cấu hình, bạn có thể khởi động dịch vụ Suricata bằng lệnh:  
   *sudo systemctl start suricata*  
  hoặc  
   *sudo systemctl start suricata.service*
* **Bật khởi động cùng hệ thống:** Để đảm bảo Suricata tự động khởi động khi hệ thống khởi động, sử dụng lệnh:  
   *sudo systemctl enable suricata*







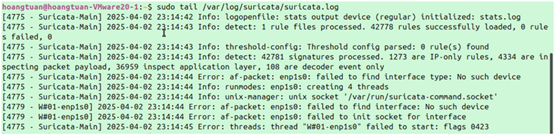
1. - Bật khởi động cùng hệ thống

Việc cấu hình Suricata đúng cách là rất quan trọng để nó có thể hoạt động hiệu quả trong môi trường mạng cụ thể của bạn và phát hiện các mối đe dọa một cách chính xác.

* 1. Kiểm tra và xác minh cài đặt

Sau khi cài đặt và cấu hình Suricata, bạn nên thực hiện một số bước kiểm tra để đảm bảo nó đang hoạt động đúng như mong đợi:

* **Kiểm tra log Suricata:** Xem file log chính của Suricata tại /var/log/suricata/suricata.log để kiểm tra xem có bất kỳ thông báo lỗi nào trong quá trình khởi động hay không. Bạn có thể sử dụng lệnh tail để xem các dòng cuối cùng của file log:  
   *sudo tail /var/log/suricata/suricata.log*



1. - Kiem tra log suricata

* **Kiểm tra file stats.log:** File stats.log chứa các thống kê về lưu lượng mạng mà Suricata đã xử lý. Bạn có thể theo dõi file này để đảm bảo Suricata đang nhận và xử lý các gói tin:  
   *sudo tail -f /var/log/suricata/stats.log*
* **Thực hiện kiểm tra cơ bản:** Một cách đơn giản để kiểm tra xem Suricata có phát hiện xâm nhập hay không là sử dụng một trang web được thiết kế để kích hoạt các quy tắc IDS. Bạn có thể sử dụng lệnh curl để truy cập trang web http://testmynids.org/uid/index.html:  
   *curl http://testmynids.org/uid/index.html*  
    
  Sau khi truy cập trang web này, hãy kiểm tra lại file log của Suricata (/var/log/suricata/fast.log hoặc file log EVE JSON tùy thuộc vào cấu hình của bạn) để xem có cảnh báo nào được tạo ra hay không. Thông thường, một quy tắc với SID là 2100498 sẽ được kích hoạt khi truy cập trang web này.

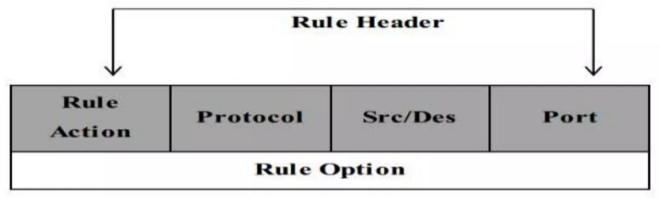
Thực hiện các bước kiểm tra này sẽ giúp bạn xác minh rằng Suricata đã được cài đặt và cấu hình thành công và đang hoạt động như một hệ thống phát hiện xâm nhập.

1. TẠO LUẬT PHÁT HIỆN TẤN CÔNG TRONG SURICATA  
   1. Giới thiệu

“Luật” (Rule) trong Suricata ta có thể hiểu một cách đơn giản nó giống như các quy tắc và luật lệ trong thế giới thực. Nghĩa là nó sẽ có phần mô tả một trạng thái và hành động gì sẽ xảy ra khi trạng thái đó đúng. Một trong những điểm đáng giá nhất của Suricata đó là khả năng cho phép người sử dụng có thể tự viết các luật của riêng mình hoặc tùy biến các luật có sẵn cho phù hợp với hệ thống mạng của mình. Ngoài một cơ sở dữ liệu lớn mà người sử dụng có thể download từ trang chủ của Suricata, người quản trị có thể tự phát triển các luật cho hệ thống của mình. Thay vì phải phụ thuộc vào nhà cung cấp, một cơ quan bên ngoài, hoặc phải cập nhật khi có một cuộc tấn công mới hay một phương pháp khai thác lỗ hổng mới được phát hiện. Người quản trị có thể viết riêng một luật dành cho hệ thống của mình khi nhìn thấy các lưu lượng mạng bất thường và so sánh với bộ luật được cộng đồng phát triển. Ưu điểm của việc tự viết các luật là có thể tùy biến và cập nhật một cách cực kỳ nhanh chóng khi hệ thống mạng có sự bất thường.

Để biết cách viết một luật từ các dữ liệu của hệ thống ta cần phải hiểu cấu trúc của luật trong Suricata như thế nào. Một luật trong Suricata được chia thành hai phần đó là phần rule header và rule options. Phần rule header bao gồm: rule action, protocol, địa chỉ ip nguồn, địa chỉ ip đích, subnetmask, port nguồn, port đích. Phần options bao gồm các thông điệp cảnh báo, thông tin các phần của gói tin sẽ được kiểm tra để xác định xem hành động nào sẽ được áp dụng.

* 1. Rule header

****

* + 1. Rule action

Phần Header sẽ chứa các thông tin xác định ai, ở đâu, cái gì của một gói tin, cũng như phải làm gì nếu tất cả các thuộc tính trong luật được hiện lên. 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.
  + 1. 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.

* + 1. . IP Address

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")

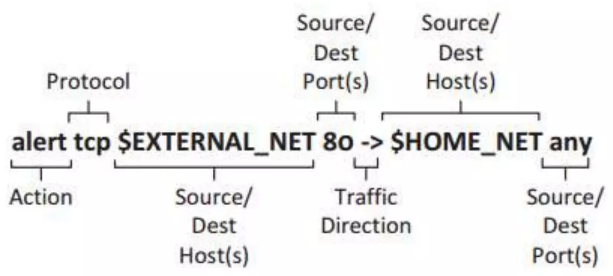
* + 1. Port

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.
  + 1. 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.



* Ví dụ:

Alert TCP !192.168.1.0/24 any <> 192.168.1.0/24 23

* 1. 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:
  + 1. General
* **Msg**

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**

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**

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**

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:

reference: url, [www.info.nl](http://www.info.nl)



1. - Bảng các tuỳ chọn của Reference

* **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.

Ví dụ:

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



1. - Thông tin phân loại lớp quy tắc

* **Priority**

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**

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

metadata:........;

* + 1. Payload
* **Content**

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**

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**

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"; depth:3;

* **Offset**

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**

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**

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 đầu từ “GET” và tìm khớp “download”. Tuy nhiên, “download” phải xuất hiện trong 9 byte tiếp theo.

* **Dsize**

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

dsize:min<>max;

* **Rpc**

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**

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.

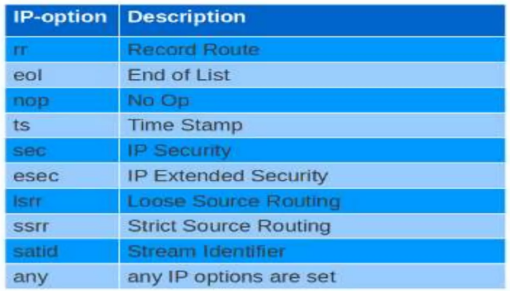
* + 1. Non-Payload
       1. IP
* **Ttl**

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 thì gói tin sẽ bị hủy bỏ. Thời gian sống được xác định dựa trên số họp, 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:



Định dạng của 1 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
2. 6 TCP Transmission Control Protocol
3. UDP User Datagram
4. GRE General Routing Encapsulation
5. ESP Encap Security Payload for IPv6
6. AH Authentication Header for Ipv6
7. IPv6-ICMP ICMP for Ipv6

Định dạng của ip\_proto như sau:

ip\_proto:<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.

* **Fragbits**

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 specified bits, plus any others

\* match if any of the specified bits are set

! match if the specified bits 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>;

* + - 1. 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.

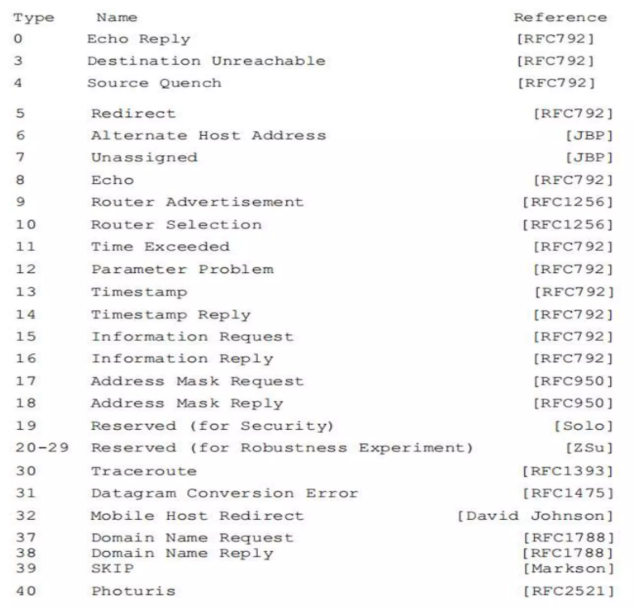
* + - 1. ICMP
* **Itype**

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>;



Bảng type của ICPM Header

* **Icode**

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\_seq**

Được sử dụng để kiểm tra số thứ tự của ICMP. Định dạng của icmp\_seq như sau:

icmp\_seq:<number>:

* + - 1. HTTP
* **http\_method**

Chỉ ra các phương thức được áp dụng với các request http. Các phương thức http: GET, POST, PUT, HEAD, DELETE, TRACE, OPTIONS, CONNECT và PATCH.

* **http\_uri và http\_raw\_uri**

Chỉ ra đường dẫn tới nơi chứa nội dung yêu cầu.

* **http\_header**

Chỉ ra phương thức sử dụng, địa chỉ cần truy cập tới và tình trạng kết nối.

* **http\_cookie.**
* **http\_user\_agent**

Là một phần của http\_header, chỉ ra thông tin về trình duyệt của người dùng.

* **http\_client\_body**

Chỉ ra các yêu cầu của máy trạm.

* **http\_stat\_code**

Chỉ ra mã trạng thái của server mà máy trạm yêu cầu kết nối tới.

* **http\_stat\_msg**

Các dòng tin thông báo về tình trạng máy chủ, hay tình trạng về việc đáp ứng các yêu cầu kết nối của máy trạm.

* **http\_server\_body**

Chỉ ra nội dung đáp trả các yêu cầu từ máy trạm của máy chủ.

* **File\_data**

Chỉ ra nội dung, đường dẫn tới file chứa dữ liệu được yêu cầu.

* + - 1. 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 rằng 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 sử dụng để bỏ các thiết lập về điều kiện trong luật.

flowbits: isnotset, name. Được sử dụng để 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\_server established/ stateless

* 1. Các nguồn luật phổ biến
* Emerging Threats (ET): Rule miễn phí và trả phí, cập nhật thường xuyên.
* Snort Rules: Tương thích với Suricata (vì Suricata hỗ trợ syntax của Snort).
* Luật tự viết (local.rules): Tuỳ chỉnh theo môi trường mạng nội bộ.
  1. Ưu và nhược điểm của Suricata Rules
* **Ưu điểm:**
* Dễ học: cú pháp đơn giản, dễ mở rộng.
* Mạnh mẽ: hỗ trợ nhiều giao thức và phân tích sâu (DPI).
* Mở rộng linh hoạt: thêm các luật tùy chỉnh hoặc lấy từ bên thứ ba.
* **Nhược điểm:**
* Đòi hỏi hiểu sâu về mạng để viết luật chính xác.
* Nếu viết luật không tối ưu dễ gây **false positives** hoặc **ảnh hưởng hiệu năng**.

* 1. Ví dụ
* **Phát hiện truy cập SSH bất thường (Brute-force SSH login)**

alert tcp any any -> any 22 (

    msg: "Possible SSH Brute-Force Attack";

    flow: to\_server, established;

    flags: S;

    threshold: type both, track by\_src, count 5, seconds 60;

    sid:1000001;

    rev:1;

)

* **Giải thích:**
* alert: hành động tạo cảnh báo.
* tcp: giao thức TCP.
* any any -> any 22: từ bất kỳ địa chỉ và cổng nào đến bất kỳ địa chỉ nào, cổng 22 (SSH).
* flow: to\_server, established: kiểm tra các kết nối tới máy chủ, phiên đã thiết lập.
* flags: S: bắt các gói TCP SYN (đánh dấu khởi tạo kết nối).
* threshold: nếu có ≥5 kết nối từ cùng một IP nguồn trong 60 giây thì báo động.
* sid và rev: ID và version của rule (dùng để quản lý rule).
* Phát hiện 1 IP gửi ≥5 gói SYN đến cổng 22 trong 60 giây.
* **Phát hiện tấn công quét cổng TCP SYN**

alert tcp any any -> any any (

    msg:"TCP SYN Port Scan Detected";

    flags:S;

    threshold: type both, track by\_src, count 20, seconds 10;

    sid:1000002;

    rev:1;

)

* + **Giải thích:**
* Phát hiện 1 IP gửi ≥20 gói SYN đến nhiều cổng khác nhau trong 10 giây.
* Phát hiện một IP gửi nhiều gói SYN tới các cổng khác nhau trong thời gian ngắn dấu hiệu quét cổng.

1. Kịch bản
   1. Chuẩn bị và cài đặt các công cụ cần thiết:
      1. Chuẩn bị

Một máy Ubuntu (Máy mục tiêu): chạy dịch vụ Web, cơ sở dữ liệu MySQL và Suricata để giám sát và phát hiện tấn công. Cấu hình địa chỉ 1 IP tĩnh cho máy (VD: 192.168.198.132).

Một máy Ubuntu (Máy tấn công): Thực hiện các cuộc tấn công nhằm vào máy mục tiêu. Cấu hình địa chỉ 1 IP tĩnh cho máy (VD: 192.168.198.131).

* + 1. Cài đặt các công cụ cần thiết:
* **Trên máy mục tiêu:**

sudo apt update && sudo apt install -y suricata apache2 php libapache2-mod-php mysql-server php-mysql

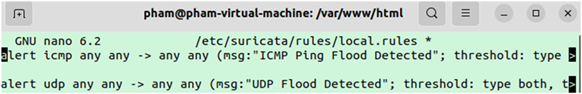
sudo apt install -y hydra

Giải thích:

* hydra: công cụ tấn công SSH
* suricata: Hệ thống phát hiện xâm nhập (IDS).
* Apache, PHP, MySQL: Dịch vụ web và cơ sở dữ liệu để mô phỏng tấn công SQL Injection.
* **Trên máy tấn công:**
* sudo apt update && sudo apt install -y hping3 sqlmap
* sudo apt íntall hydra -y
* hydra: Công cụ để thực hiện tấn công brute-force.
* hping3: Công cụ tạo tấn công DdoS và UDP Flood.
* sqlmap: Công cụ tự động hóa tấn công SQL Injection.
  1. Tấn công UDP Flood
     1. Mục tiêu:

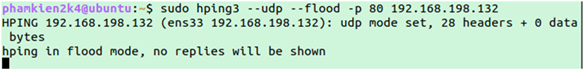
Mô phỏng một cuộc tấn công UDP Flood nhằm làm quá tải máy mục tiêu bằng cách gửi một lượng lớn gói tin UDP trong thời gian ngắn. Suricata sẽ được sử dụng để phát hiện và ghi lại log về hành vi tấn công này dựa trên ngưỡng lưu lượng được cấu hình trong luật.

* Luật sử dụng:

  
alert udp any any -> any any (msg:"UDP Flood Detected"; threshold: type both, track by\_dst, count 100, seconds 10; sid:1000002;)

* Phân tích luật:
* alert udp: Áp dụng cho giao thức UDP.
* any any -> any any: Phát hiện trên mọi nguồn và đích.
* msg:”UDP Flood Detected”: Hiển thị cảnh báo khi phát hiện tấn công.
* threshold: type both, track by\_dst, count 100, seconds 10: Phát hiện 100 gói UDP đến cùng một địa chỉ trong 10 giây.
* sid: 1000002: Mã số của quy tắc
  + 1. Mô tả cách demo

**Bước 1: Trên máy tấn công chạy lệnh:**

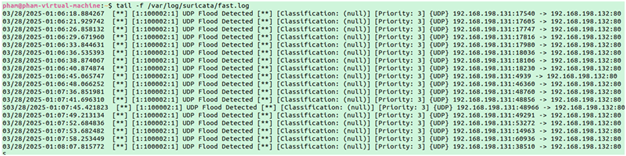
sudo hping3 --udp --flood -p 80 192.168.198.132

* Giải thích:
* sudo: Chạy lệnh với quyền quản trị cao nhất
* hping3: Công cụ gửi gói tin mạng tùy chỉnh
* --udp: Sử dụng giao thức UDP(User Datagram Protocol)
* --flood: Chế độ gửi gói tin liên tục với tốc độ cao
* -p 80: Nhắm tới cổng 80
* 192.168.198.132: Địa chỉ IP đích nhận gói tin

**Bước 2: Kiểm tra log trên máy mục tiêu:**

sudo tail -f /var/log/suricata/fast.log

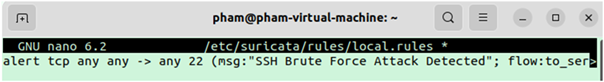
* Kết quả đạt được:
* 03/28/2025-01:08:07.815772 [\*\*] [1:100002:1] UDP Flood Detected [\*\*] [Classification: (null)] [Priority: 3] {UDP} 192.168.198.131:38510 -> 192.168.198.132:80
* Giải thích log:
* 03/28/2025-01:08:07.815772: Thời gian phát hiện.
* [1:1000002:1]: Mã sid:1000002 của luật.
* UDP Flood Detected: Thông báo từ luật.
* {UDP}: Giao thức được phát hiện.
* 192.168.198.131-> 192.168.198.132: Địa chỉ nguồn và đích cùng cổng ngẫu nhiên.



* 1. Tấn công SSH Brute Force
     1. Mục tiêu:

Mô phỏng một cuộc tấn công SSH Brute Force nhằm dò tìm mật khẩu đăng nhập vào máy mục tiêu. Suricata sẽ được sử dụng để phát hiện và ghi lại log về hành vi tấn công này dựa trên ngưỡng số lần đăng nhập thất bại trong một khoảng thời gian ngắn.

* Luật sử dụng:



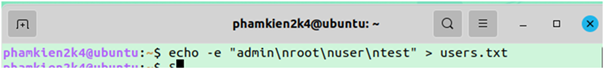
alert tcp any any -> any 22 (msg:"SSH Brute Force Attack Detected"; flow:to\_server,established; content:"SSH-"; depth:4; detection\_filter:track by\_src, count 5, seconds 10; sid:1000008;)

* Phân tích luật:
* alert tcp: Áp dụng cho giao thức TCP.
* any any -> any 22: Phát hiện lưu lượng đi cổng 22 (SSH) từ bất kỳ nguồn nào.
* msg: “SSH Brute Force Attack Detected”: Hiển thị cảnh báo khi phát hiện tấn công.
* flow:to\_server,established: Chỉ phát hiện khi có kết nối hợp lệ đến server.
* content:"SSH-"; depth:4: Kiểm tra nội dung gói tin có chứa chuỗi "SSH-" trong 4 byte đầu tiên (đặc trưng của giao thức SSH).
* detection\_filter:track by\_src, count 5, seconds 10: Phát hiện nếu có 5 lần thử kết nối SSH trong vòng 10 giây từ cùng một địa chỉ nguồn.
* sid:1000008: Mã số của quy tắc.
  + 1. Mô tả cách demo

**Bước 1:  Chuẩn bị danh sách tên người dùng**

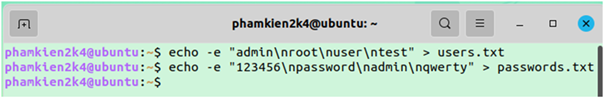
* Trên máy tấn công:

Tạo file chứa danh sách username:



echo -e "admin\nroot\nuser\ntest" > users.txt

Tạo file chứa danh sách mật khẩu:

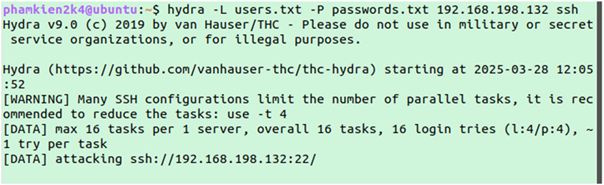


echo -e "123456\npassword\nadmin\nqwerty" > passwords.txt

**Bước 2: Thực hiện tấn công SSH Brute Force**

* Trên máy tấn công:

Dùng Hydra để thử nhiều user và mật khẩu:



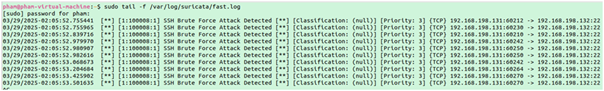
hydra -L users.txt -P passwords.txt 192.168.198.132 ssh

**Bước 3: Kiểm tra log trên máy mục tiêu:**

* Xem log Suricata để kiểm tra cảnh báo:

sudo tail -f /var/log/suricata/fast.log

* Kết quả đạt được:



* 1. Tấn công SQL Injection
     1. Mục tiêu:

Mô phỏng một cuộc tấn công SQL Injection nhằm vào trang đăng nhập (login.php) trên máy mục tiêu, sau đó sử dụng Suricata để phát hiện và ghi lại log về hành vi tấn công này. Cuộc tấn công sẽ khai thác lỗ hổng trong truy vấn SQL không được bảo vệ để bypass quá trình xác thực đăng nhập.

* + 1. Cấu hình môi trường
       1. Tạo cơ sở dữ liệu trên máy mục tiêu

CREATE DATABASE sql\_injection\_demo;

USE sql\_injection\_demo;

CREATE TABLE users (id INT AUTO\_INCREMENT PRIMARY KEY, username VARCHAR(50), password VARCHAR(50));

INSERT INTO users (username, password) VALUES ('admin', 'admin123');

* Giải thích:
* Tạo cơ sở dữ liệu sql\_injection\_demo để mô phỏng hệ thống thực tế.
* Tạo bảng users với các cột id, username, password.
* Thêm một tài khoản hợp lệ (admin/admin123) để kiểm tra đăng nhập bình thường.
  + - 1. Tạo trang đăng nhập
* Kết nối Database:

****

* Tạo kết nối tới cơ sở dữ liệu MySQL
* Sử dụng thư viện mysqli để kết nối
* Các tham số lần lượt là:

+ Địa chỉ máy chủ (localhost)

+ Tên người dùng (webuser)

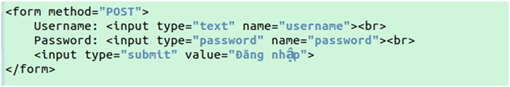
+ Mật khẩu (webpassword)

+ Tên cơ sở dữ liệu (sql\_injection\_demo)

* Nếu kết nối lỗi, chương trình sẽ dừng và hiển thị thông báo lỗi
  + - 1. Xử lí Form Đăng Nhập:



* Kiểm tra xem request có phải là POST không
* Lấy username và password từ form
* Tạo câu truy vấn SQL để kiểm tra đăng nhập
* Nếu tìm thấy bản ghi (num\_rows > 0), hiển thị "Đăng nhập thành công"
* Ngược lại, hiển thị "Sai tên đăng nhập hoặc mật khẩu"
  + - 1. Giao diện Form



* Tạo form HTML để nhập username và password
* Phương thức POST để gửi dữ liệu
* Các trường input để nhập thông tin
* Luật sử dụng:
* alert http any any -> any any (msg:"SQL Injection Attempt Detected"; content:"union select"; nocase; sid:1000003;)
* alert http any any -> any any (msg:"SQL Injection Attempt Detected"; content:"' OR '1'='1"; nocase; sid:100005;)
* alert http any any -> any any (msg:"SQL Injection Attempt Detected"; content:"admin' --"; nocase; sid:100006;)
* alert http any any -> any any (msg:"SQL Injection Attempt Detected"; content:" or 1=1--"; nocase; sid:100007;)
* Phân tích luật:
* alert http: Áp dụng cho giao thức HTTP.
* any any -> any any: Phát hiện trên mọi nguồn và đích.
* msg:"SQL Injection Attempt Detected": Thông báo khi phát hiện tấn công.
* content:"...": Kiểm tra chuỗi cụ thể trong payload:

+ "union select": Phát hiện kỹ thuật SQL Injection khai thác dữ liệu.

+ " OR '1'='1": Phát hiện chuỗi bypass xác thực.

+ "admin' --": Phát hiện chou dùng comment để bỏ qua kiểm tra.

+ " or 1=1--": Phát hiện biến thể bypass xác thực.

* nocase: Không phân biệt chữ hoa/thường.
* sid:1000003, sid:100005, ...: Mã số duy nhất của từng quy tắc.
  + 1. Mô tả demo

**Bước 1: Thực hiện tấn công SQL Injection thủ công qua form**

* Trên máy tấn công:
* Mở trình duyệt, truy cập http://192.168.198.132/login.php.
* Nhập vào form:
  + - Username: ' OR '1'='1'
    - Password: 123 (hoặc bất kỳ giá trị nào).
* Nhấn "Đăng nhập".
* Giải thích:
* Truy vấn SQL trong login.php trở thành:

SELECT \* FROM users WHERE usename = ‘’ OR ‘1’=’1’ AND password = ‘123’

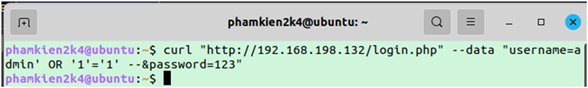
* '1'='1' luôn đúng, bypass kiểm tra và trả về tất cả bản ghi trong bảng users.
* Kết quả: Trang hiển thị "Đăng nhập thành công!" cùng thông tin tài khoản (nếu có).

A screenshot of a computer screen

AI-generated content may be incorrect.

**Bước 2: Sử dụng curl để tấn công**

* Trên máy tấn công, mở terminal:
* curl "http://192.168.198.132/login.php" --data "username=admin' OR '1'='1' --&password=123"



* Giải thích:
* Gửi request POST với payload SQL Injection.
* Truy vấn SQL:
* SELECT \* FROM users WHERE username = ‘admin’ OR ‘1’=’1’
* Kết quả: Nhận được phản hồi “Đăng nhập thành công!” từ server.

**Bước 3: Sử dụng sqlmap để tấn công tự động**

* Trên máy tấn công chạy:
* sqlmap -u "http://192.168.198.132/login.php" --method POST --data "username=admin&password=123" --level 2 --risk 2



* Giải thích:
* -u: URL mục tiêu.
* --method POST: Dùng phương thức POST.
* --data: Dữ liệu mẫu để sqlmap kiểm tra.
* --level 2 --risk 2: Tăng mức độ kiểm tra để phát hiện lỗ hổng.
* Kết quả: sqlmap sẽ thử các payload SQL Injection (bao gồm ' OR '1'='1', union select, v.v.) và báo cáo nếu tìm thấy lỗ hổng.

**Bước 4: Kiểm tra log trên máy mục tiêu**

* Trên máy mục tiêu, mở terminal:
* sudo tail -f /var/log/suricata/fast.log
* Quan sát với log khi thực hiện các bước tấn công trên:
* Với username ‘ OR ‘1’=’1’:



* Giải thích:
* Suricata phát hiện payload khớp với rule và ghi log vào fast.log.
* Log hiển thị thời gian, mã sid, và địa chỉ IP nguồn/đích.

KẾT LUẬN

Các kết quả đạt được

Nhóm thực hiện đề tài "Nghiên cứu và triển khai Suricata trong hệ thống giám sát an ninh mạng" đã hoàn thành việc tìm hiểu, cài đặt, thử nghiệm và đánh giá hệ thống Suricata trên môi trường thực tế. Báo cáo đã thực hiện đầy đủ các nội dung theo đề cương như sau:

Nghiên cứu về kiến trúc, nguyên lý hoạt động, các tính năng nổi bật của Suricata và vai trò của nó trong bảo mật mạng.

Tiến hành cài đặt và cấu hình Suricata trên hệ thống thử nghiệm để kiểm tra khả năng giám sát lưu lượng mạng và phát hiện xâm nhập.

Thử nghiệm và đánh giá hiệu suất của Suricata, so sánh với các hệ thống IDS/IPS khác, từ đó đưa ra các nhận xét và đề xuất cải tiến.

Xây dựng và kiểm thử các quy tắc tùy chỉnh để nâng cao khả năng phát hiện và phòng thủ trước các mối đe dọa mới.

Hướng phát triển

Đề tài này có thể được mở rộng theo các hướng sau:

Tích hợp Suricata với các hệ thống SIEM (Security Information and Event Management) để tăng cường khả năng phân tích và quản lý sự kiện bảo mật.

Tối ưu hóa hiệu suất Suricata khi triển khai trên các hệ thống mạng có lưu lượng lớn.

Phát triển thêm các quy tắc tùy chỉnh để mở rộng khả năng phát hiện các mối đe dọa mới và các cuộc tấn công tinh vi.

Nghiên cứu ứng dụng Suricata trong các hệ thống bảo mật mạng doanh nghiệp và tổ chức lớn nhằm đảm bảo khả năng giám sát và bảo vệ toàn diện hơn.

TÀI LIỆU THAM KHẢO

[1] Cấu hình Suricata: <https://adminvietnam.org/he-thong-ids-ips-cai-dat-va-cau-hinh-suricata/4801/>

[2] Trang chủ Suricata : [https://docs.suricata.io/en/latest](https://docs.suricata.io/en/latest/what-is-suricata.html)