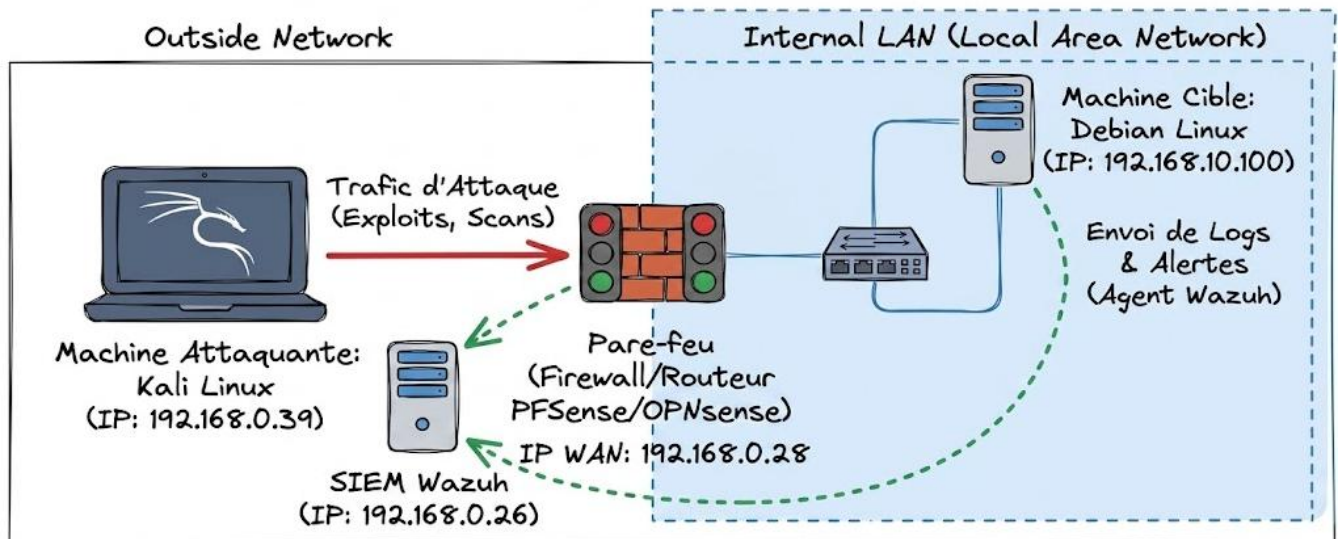


Mini-lab de détection d'intrusion

BTS SIO SISR – Maël Philibert



Sommaire :

1. Contexte et objectif du projet.....	2
2. Architecture du laboratoire.....	2
3. Plan d'adressage et rôles	2
4. Mise en place de l'environnement.....	3
5. Scénario d'attaque web (nikto).....	4
6. La réponse : blocage manuel au pare-feu).....	5
7. Investigation de l'incident.....	6
8. Difficultés rencontrées.....	7
9. Conclusion.....	8
10. Bonus.....	9

1. Contexte et objectifs du projet

Ce projet s'inscrit dans le cadre de mon BTS SIO SISR ainsi que dans mon apprentissage en **Autodidacte**. Il consiste à monter, dans un environnement virtualisé un petit laboratoire de sécurité représentant une situation réaliste. On y trouve une machine cible exposée à internet, un pare-feu Pfsense la protégeant avec un SIEM Wazuh et de l'autre coté un attaquant extérieur avec une machine Kali Linux.

L'objectif n'est pas seulement de déclencher des alertes, mais surtout de mener une véritable investigation : reconstituer ce qui s'est passé (qui a attaqué, avec quelle technique, quand, quelle règle a réagi, l'attaque a-t-elle réussi ou échoué ?).

2. Architecture du laboratoire

Mon infrastructure repose sur VirtualBox et se divise en deux réseaux distincts :

- **Le réseau « maison » / WAN 192.168.0.X** , en mode « accès par pont » c'est le coté internet, on y trouve le WAN du pare-feu, la machine attaquante Kali Linux, et le SIEM Wazuh. J'ai choisi de placer Wazuh de ce côté (en mode « accès par pont ») pour que son interface d'administration reste directement accessible depuis le navigateur d'un poste windows.
- **Le réseau interne protégé / LAN (192.168.10.x)**, en réseau privé hôte (host-only) VirtualBox. C'est le « côté à défendre » : on y trouve la machine cible Debian et le poste d'administration Windows, isolés derrière le pare-feu

Choix d'architecture assumé : l'attaquant (Kali) est placé **côté WAN**, et non sur le LAN à côté de la cible. Placer l'attaquant sur le même réseau que sa victime n'aurait aucun sens dans une démonstration de sécurité : un vrai attaquant vient de l'extérieur et doit **traverser le pare-feu** .

3. Plan d'adressage et rôles

Machine	Rôle	Adresse
Pfsense 2.9.0	Pare-feu + IDS Suricata	Wan 192.168.0.28 / LAN 192.168.10.1
Wazuh	SIEM (manager, dashboard)	192.168.0.26
Cible Debian 13	SSH + Apache + Agent wazuh	192.168.10.100
Poste Windows	Administration	192.168.10.15
Kali Linux	Attaquant	192.168.0.39

4. Mise en place de l'environnement

4.1 pfSense et son IDS Suricata

Les cinq machines sont hébergées dans VirtualBox. Chaque VM utilise deux types de cartes réseau selon sa place sauf le pare-feu : adaptateur ponté pour le côté WAN, réseau host-only pour le côté LAN. Le pare-feu, lui, possède une carte de chaque type puisqu'il fait le lien entre les deux réseaux.

```
Starting package suricata...done.
pfSense 2.9.0-RELEASE amd64 20260817-1836
bootup complete

FreeBSD/amd64 (pfSense.home.arpa) (ttyv0)

VirtualBox Virtual Machine - Netgate Device ID: 2ffb78cde7bef2d7a3d4

*** Welcome to pfSense 2.9.0-RELEASE (amd64) on pfSense ***

WAN (wan) -> em0 -> v4/DHCP4: 192.168.0.28/24
v6/DHCP6: 2a01:e0a:c0f:6030:5fcc:f041:a148:f8a9/64
LAN (lan) -> em1 -> v4: 192.168.10.1/24

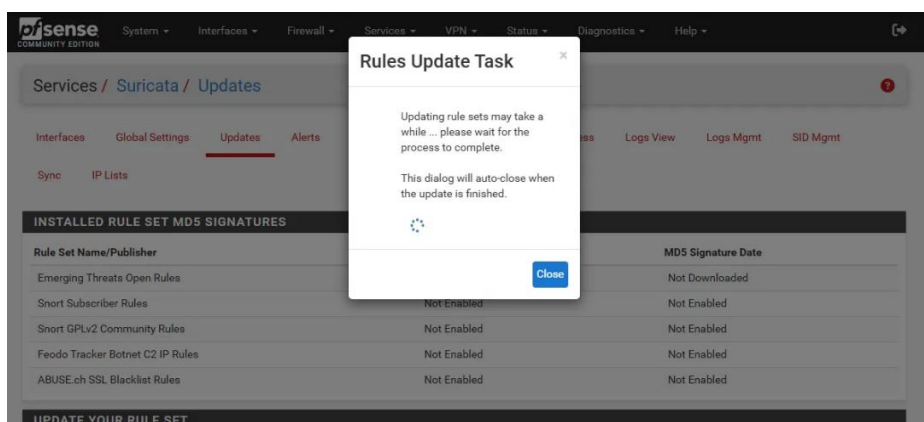
0) Logout / Disconnect SSH          9) pfTop
1) Assign Interfaces                10) Filter Logs
2) Set interface(s) IP address      11) Restart GUI
3) Reset admin account and password 12) PHP shell + pfSense tools
4) Reset to factory defaults        13) Update from console
5) Reboot system                    14) Enable Secure Shell (sshd)
6) Halt system                      15) Restore recent configuration
7) Ping host                        16) Restart PHP-FPM
8) Shell

Enter an option: █
```

pfSense joue deux rôles : pare-feu (il sépare et filtre les deux réseaux) et hôte de la sonde IDS Suricata. Trois réglages ont été indispensables :

- **Désactivation du blocage RFC1918 sur le WAN** : sans cela, pfSense bloquerait par défaut le trafic venant d'adresses privées comme 192.168.0.39 (mon attaquant Kali), et l'attaque ne pourrait même pas arriver jusqu'au pare-feu.
- **Désactivation de trois « hardware offloads »** : ces optimisations de carte réseau perturbent l'analyse de paquets de Suricata.

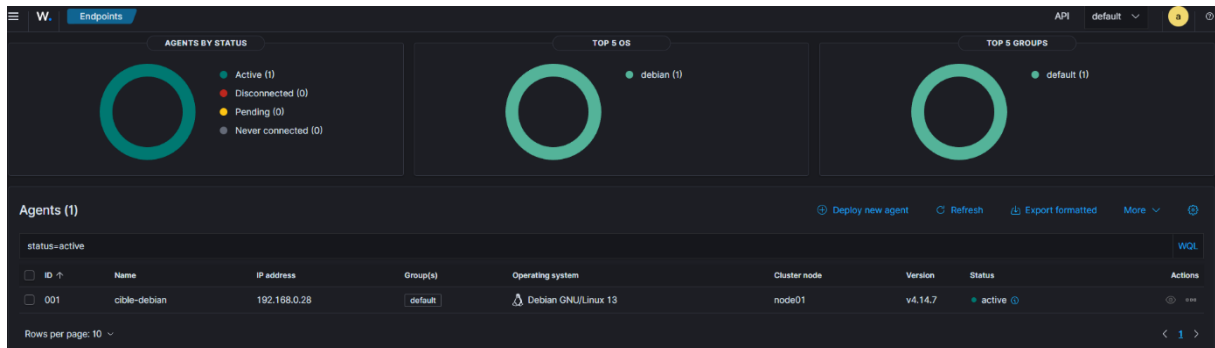
Suricata **7.0.9** a été installé depuis le gestionnaire de paquets de pfSense (Available Packages). Les règles **ET Open** (Emerging Threats) ont été téléchargées, toutes les catégories sélectionnées, et la sonde activée sur l'interface (voyant vert), en **mode IDS** (détection seule, sans blocage) afin de laisser passer l'attaque tout en l'observant.



- **Création des redirections de port (port forward)** : c'est ce qui expose volontairement le SSH (port 22) et le web (port 80) de la cible vers l'extérieur, pour que l'attaquant ait une porte à attaquer.

• 4.2 La machine cible : Debian 13

La cible est un serveur Debian 13 exposant deux services : SSH (openssh-server) et un site web (Apache2). Un agent Wazuh y a été installé et pointe vers le manager (192.168.0.26, port 1514) ; la connectivité a été confirmée et l'agent apparaît « Active » dans le dashboard.

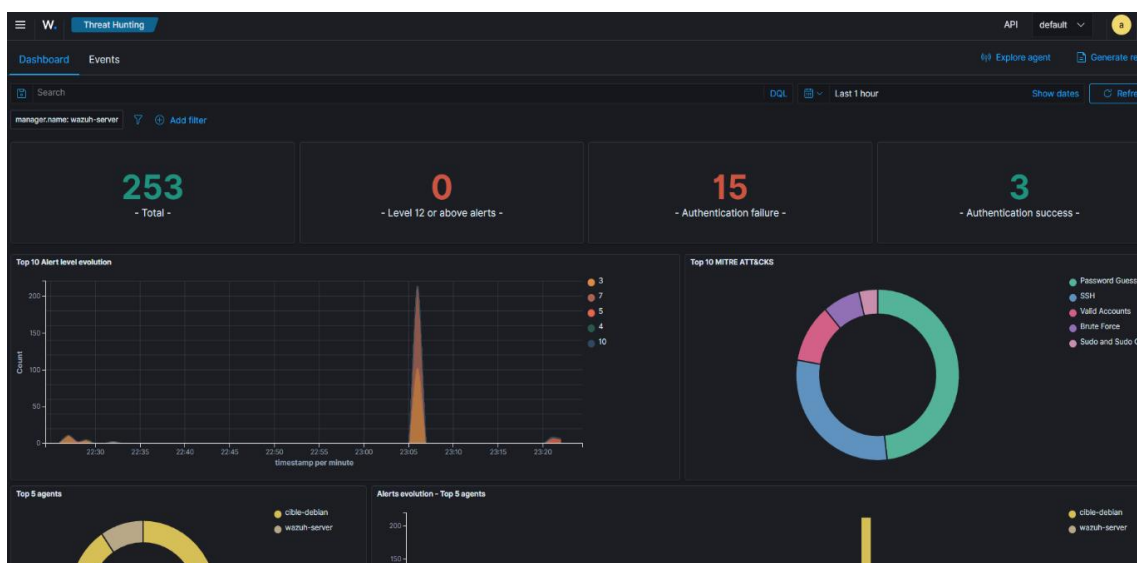


4.3 La machine attaquante : Kali Linux

Kali est positionnée côté WAN (192.168.0.39), en attaquant externe crédible. Elle ne « voit » la cible qu'à travers l'adresse publique du pare-feu (192.168.0.28) et ses ports redirigés — exactement comme le ferait un attaquant sur Internet.

4.4 Import de Wazuh

Après import, quelques réglages ont été nécessaires : suppression d'un contrôleur disquette vide qui provoquait une erreur « paramètre invalide », passage du contrôleur graphique en VMSVGA, et surtout ajustement de la mémoire vive car l'indexer demande beaucoup de RAM. S'il manque de la RAM au démarrage, il « time out » et le dashboard reste bloqué sur « not ready yet ». La RAM de la VM a été fixée à **5120 Mo**, (voir section 8).



5. Scénario d'attaque web simple (nikto)

Le serveur Apache de la cible est exposé sur Internet grâce à une redirection de port sur pfSense (WAN:80 → 192.168.10.100:80). Sans ce NAT, le serveur web serait invisible de l'extérieur : c'est justement le but d'une redirection de port, exposer un service interne.

L'attaque :

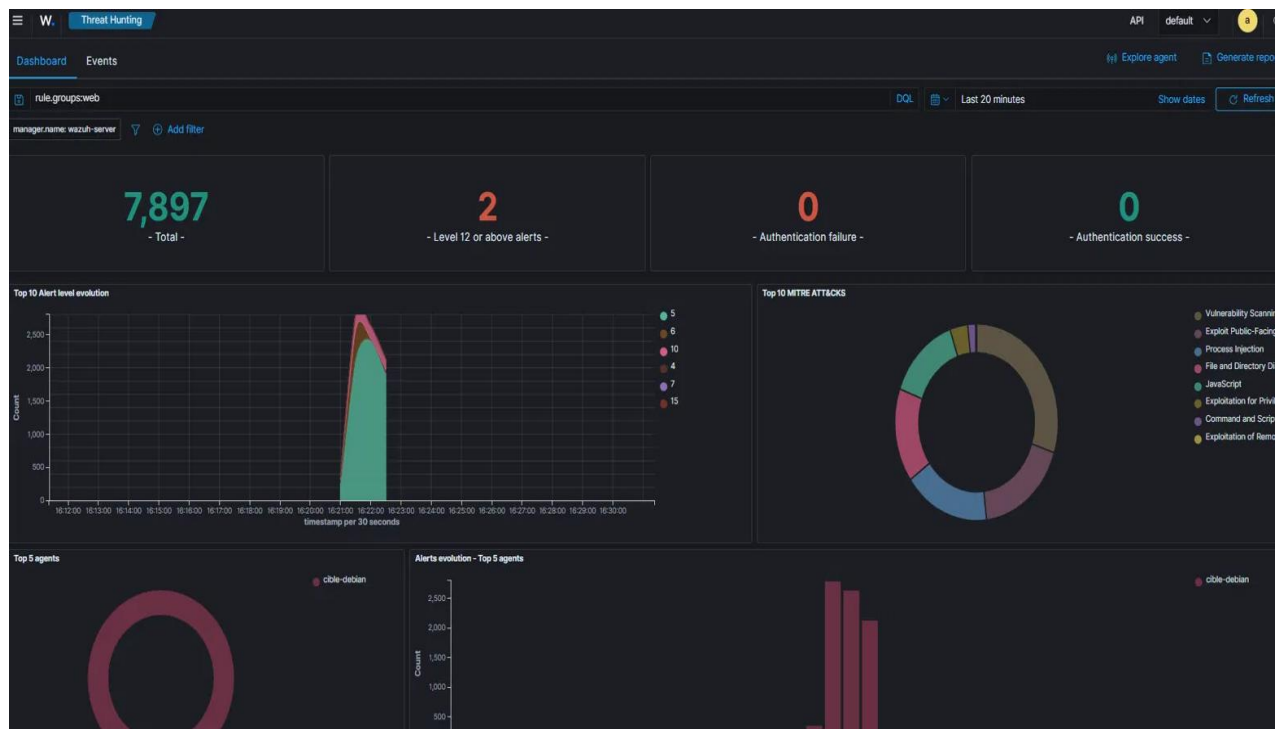
Depuis Kali (192.168.0.39, côté WAN), j'ai lancé un **scan de vulnérabilités web avec nikto**, en visant l'adresse WAN de pfSense :

nikto -h http://192.168.0.28

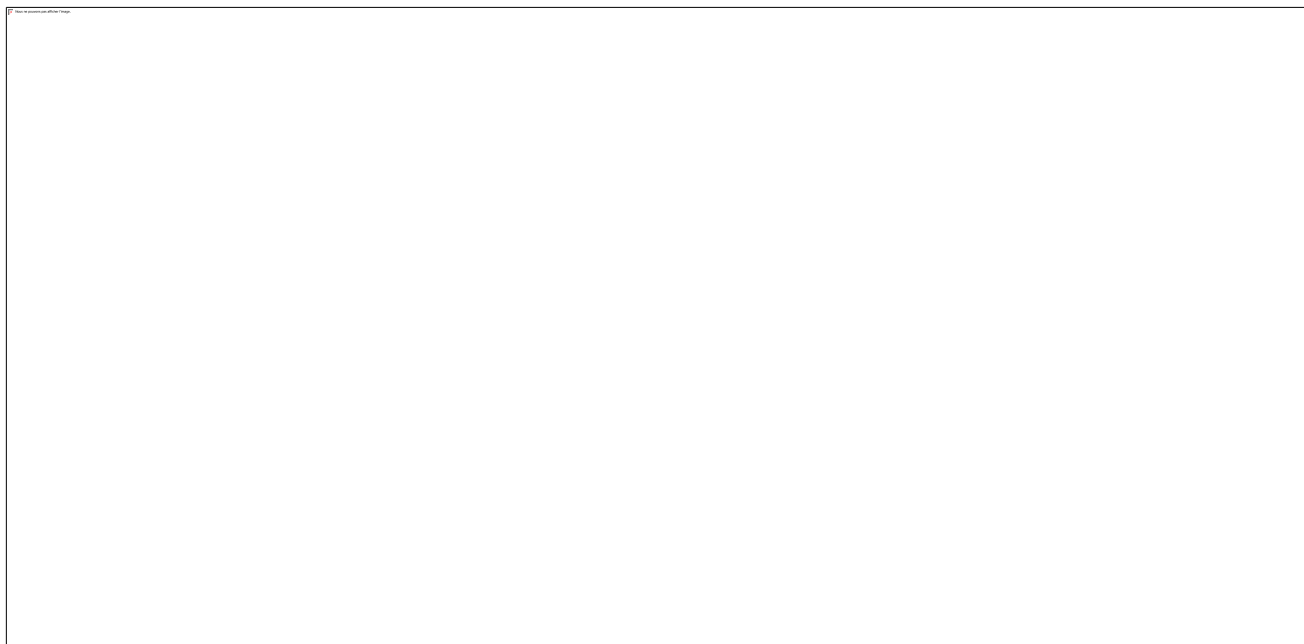
Nikto envoie des **milliers de requêtes** pour tester la présence de fichiers et de répertoires sensibles (par exemple des plugins WordPress), ce qui génère en masse des erreurs HTTP côté serveur.

Le résultat est immédiat et massif : en filtrant sur rule.groups:web, le dashboard remonte **près de 7 897 événements** sur la fenêtre du scan, avec un pic net au moment de l'attaque (vers 16H21-16H22).

Le graphique MITRE ATT&CK associé fait apparaître les techniques attendues pour ce type d'attaque : **Vulnerability Scanning (T1595)** et **Exploit Public-Facing Application (T1190)**.



En ouvrant un événement en détail on peut lire l'attaque champ par champ et avoir **plusieurs informations** :



data.srcip	192.168.0.39 → l'attaquant (Kali)
agent.ip / agent.name	192.168.10.100 / cible-debian → la victime
data.protocol / data.id = GET / 404	requête sur une ressource inexistante
rule.id / rule.level	31101 / 5 → « Web server 400 error code »
rule.firedtimes	6 727
full_log	la requête GET brute avec son code 404 → la preuve concrète.

6. La réponse : blocage manuel au pare-feu

Une fois l'attaquant identifié, j'ai appliqué la remédiation directement sur pfSense (**Firewall → Rules → WAN → Add**) : une règle d'action **Block**, avec pour **Source l'adresse de l'attaquant (192.168.0.39)**

J'ai veillé à placer cette règle au-dessus des règles NAT, car sur pfSense la première règle qui correspond l'emporte, et à cliquer sur Apply Changes pour l'activer.



Edit Firewall Rule

Action Block
Choose what to do with packets that match the criteria specified below.
Hint: the difference between block and reject is that with reject, a packet (TCP RST or ICMP port unreachable for UDP) is returned to the sender, whereas with block the packet is dropped silently. In either case, the original packet is discarded.

Disabled ☐ Disable this rule
Set this option to disable this rule without removing it from the list.

Interface WAN
Choose the interface from which packets must come to match this rule.

Address Family IPv4
Select the Internet Protocol version this rule applies to.

Protocol Any
Choose which IP protocol this rule should match.

Source
Source ☐ Invert match Address or Alias 192.168.0.39 /

Destination
Destination ☐ Invert match Any Destination Address /

Extra Options
Log ☒ Log packets that are handled by this rule
Hint: the firewall has limited local log space. Don't turn on logging for everything. If doing a lot of logging, consider using a remote syslog server (see the Status: System Logs: Settings page).

Description Blocage de l'attaquant web 16.21
A description may be entered here for administrative reference. A maximum of 57 characters will be used in the ruleset label and displayed in the firewall log.

Advanced Options [Display Advanced](#)

[Save](#)

Pour prouver que le blocage est réellement effectif, je relance l'attaque et j'ai consulté les journaux du pare-feu (**Status** → **System Logs** → **Firewall**) en filtrant sur la source 192.168.0.39 et la destination de l'ip du pare-feu.

Chaque tentative de l'attaquant vers 192.168.10.100:80 y apparaît en **rouge (X)** = paquet **bloqué**. C'est la preuve que la remédiation fonctionne : l'attaquant ne peut plus atteindre le serveur web, nous pouvons le voir ci-dessous :

X	Aug 29 16:46:46	WAN	192.168.0.39:38042	192.168.10.100:80	TCP:S
X	Aug 29 16:46:50	WAN	192.168.0.49:59727	255.255.255.255:6667	UDP
X	Aug 29 16:46:54	WAN	192.168.0.49:59727	255.255.255.255:6667	UDP
X	Aug 29 16:46:59	WAN	192.168.0.49:59727	255.255.255.255:6667	UDP
X	Aug 29 16:47:04	WAN	192.168.0.49:59727	255.255.255.255:6667	UDP
X	Aug 29 16:47:09	WAN	192.168.0.49:59727	255.255.255.255:6667	UDP
X	Aug 29 16:47:11	WAN	192.168.0.39:43902	192.168.10.100:80	TCP:S
X	Aug 29 16:47:12	WAN	192.168.0.39:43902	192.168.10.100:80	TCP:S
X	Aug 29 16:47:13	WAN	192.168.0.39:43902	192.168.10.100:80	TCP:S
X	Aug 29 16:47:14	WAN	192.168.0.39:43902	192.168.10.100:80	TCP:S
X	Aug 29 16:47:15	WAN	192.168.0.49:59727	255.255.255.255:6667	UDP
X	Aug 29 16:47:15	WAN	192.168.0.39:43902	192.168.10.100:80	TCP:S
X	Aug 29 16:47:16	WAN	192.168.0.39:43902	192.168.10.100:80	TCP:S
X	Aug 29 16:47:18	WAN	192.168.0.39:43902	192.168.10.100:80	TCP:S
X	Aug 29 16:47:22	WAN	192.168.0.39:43902	192.168.10.100:80	TCP:S
X	Aug 29 16:47:24	WAN	192.168.0.49:59727	255.255.255.255:6667	UDP
X	Aug 29 16:47:29	WAN	192.168.0.49:59727	255.255.255.255:6667	UDP
X	Aug 29 16:47:30	WAN	192.168.0.39:43902	192.168.10.100:80	TCP:S
X	Aug 29 16:47:34	WAN	192.168.0.49:59727	255.255.255.255:6667	UDP
X	Aug 29 16:47:39	WAN	192.168.0.49:59727	255.255.255.255:6667	UDP
X	Aug 29 16:47:44	WAN	192.168.0.49:59727	255.255.255.255:6667	UDP
X	Aug 29 16:47:46	WAN	192.168.0.39:43902	192.168.10.100:80	TCP:S

Cette deuxième démonstration boucle ainsi le cycle complet d'un SOC : **détecter** → **investiguer** → **répondre** → **vérifier**.

Pour aller plus loin pour la réponse :

Le blocage que j'ai réalisé est manuel mais dans un environnement de production, on chercherait à l'automatiser et à renforcer la protection car l'attaquant peut changer d'ip rapidement avec un **VPN** par exemple. J'ai d'autre solution comme :

1. **Automatiser le blocage avec fail2ban** : installé sur la cible, il bannit automatiquement toutes les IP qui génèrent trop d'erreurs (par exemple un trop grand nombre d'erreurs 400/404 en peu de temps), sans intervention humaine.
2. **Piloter le blocage depuis le SIEM avec Wazuh Active Response** : c'est Wazuh lui-même qui, en détectant l'attaque, déclenche la commande de blocage — la détection et la réponse sont alors reliées dans le même outil.

7. Investigation de l'incident

C'est le cœur du projet. Au-delà de l'alerte, il s'agit de reconstruire le récit complet de l'attaque et d'en expliquer chaque élément.

Le 29/08 à 14:22 UTC (16:22 à Paris), Wazuh a détecté une vague massive d'événements sur l'agent **cible-debian** (192.168.10.100).

Un attaquant situé à l'extérieur du pare-feu, sur **192.168.0.39** (la machine Kali, côté WAN), a mené un **scan de vulnérabilités web (nikto)** contre le serveur Apache de la cible, exposé sur Internet via la redirection de port pfSense. En quelques minutes, l'outil a généré près de **7 897 requêtes malveillantes** (erreurs 400/404 en masse, tentative XSS).

Il ne s'agit pas d'une intrusion réussie, mais d'une **phase de reconnaissance** : l'attaquant cartographie la surface d'attaque du serveur apache, une étape qui précède typiquement une tentative d'exploitation.

8. Difficultés rencontrées

Plusieurs incidents techniques ont jalonné le montage du lab. Les avoir résolus et documentés fait partie de la démonstration : un environnement de sécurité doit être fiable le jour J.

- **Perte d'IP côté Wazuh** (plus d'IPv4) : résolu par un redémarrage réseau ; solution durable = IP statique 192.168.1.82 ou réservation DHCP.
- **Cible Debian en 169.254.x.x** : signe d'absence de DHCP car pfSense n'était pas prêt au démarrage. D'où l'ordre de démarrage : pfSense d'abord.
- **Indexer en échec / disque plein** : le module Vulnerability Detection se remplissait ~19 Go dans /var/ossec/queue/vd. Désactivé pour libérer l'espace et fiabiliser l'indexer.
- Pour Wazuh en cas de manque de RAM au démarrage, il « time out » et le dashboard reste bloqué sur « not ready yet », il faut donc 5000 mo de RAM attribué pour un bon démarrage

9. Conclusion

Pour finir, mon laboratoire fonctionne de bout en bout. J'ai monté une attaque web avec nikto depuis Kali, qui traverse le pare-feu, et Wazuh l'a bien détectée : j'ai pu voir les milliers de requêtes remonter dans le dashboard, retrouver l'IP de l'attaquant, puis la bloquer sur pfSense et vérifier dans les logs que le blocage marchait vraiment. J'ai ainsi fait le tour complet du travail d'un SOC : détecter, investiguer, répondre, vérifier.

En plus, j'ai réalisé une attaque par force brute **SSH** : là, contrairement au scan web, l'attaquant a réussi à se connecter, donc c'est une vraie compromission (alerte critique de niveau 12, deux techniques MITRE). Pour éviter ça, la meilleure solution aurait été d'utiliser des **clés SSH au lieu d'un mot de passe** : sans mot de passe à deviner, la force brute ne peut plus fonctionner.

Ce qui m'a le plus appris, ce n'est pas seulement de voir des alertes s'afficher, mais de réussir à **comprendre et raconter ce qui s'est passé** : qui a attaqué, comment, quand, et si l'attaque a marché ou non. Pour moi c'est ça le plus important en cybersécurité, et ça m'a donné envie de continuer dans ce domaine. Si j'allais plus loin, j'aimerais automatiser les blocages (fail2ban ou Wazuh Active Response) pour une réponse encore plus rapide, sans intervention manuelle.

10. Bonus

Liste des outils utilisés :
VirtualBox (Oracle)
pfSense 2.9.0
Suricata
Wazuh
Debian 13
Apache2
OpenSSH
Kali Linux
Nmap
Hydra
Nikto
gobuster

LIKEDIN: <https://www.linkedin.com/in/maelphi>

PORTFOLIO: <https://maelphilibert.fr/>

