Système de détection d'intrusions

SOMMAIRE

1. Contexte et cahier des charges

VIRONAX est acteur français dans la recherche de vaccins. C'est une PME récente d'une cinquantaine de personnes.

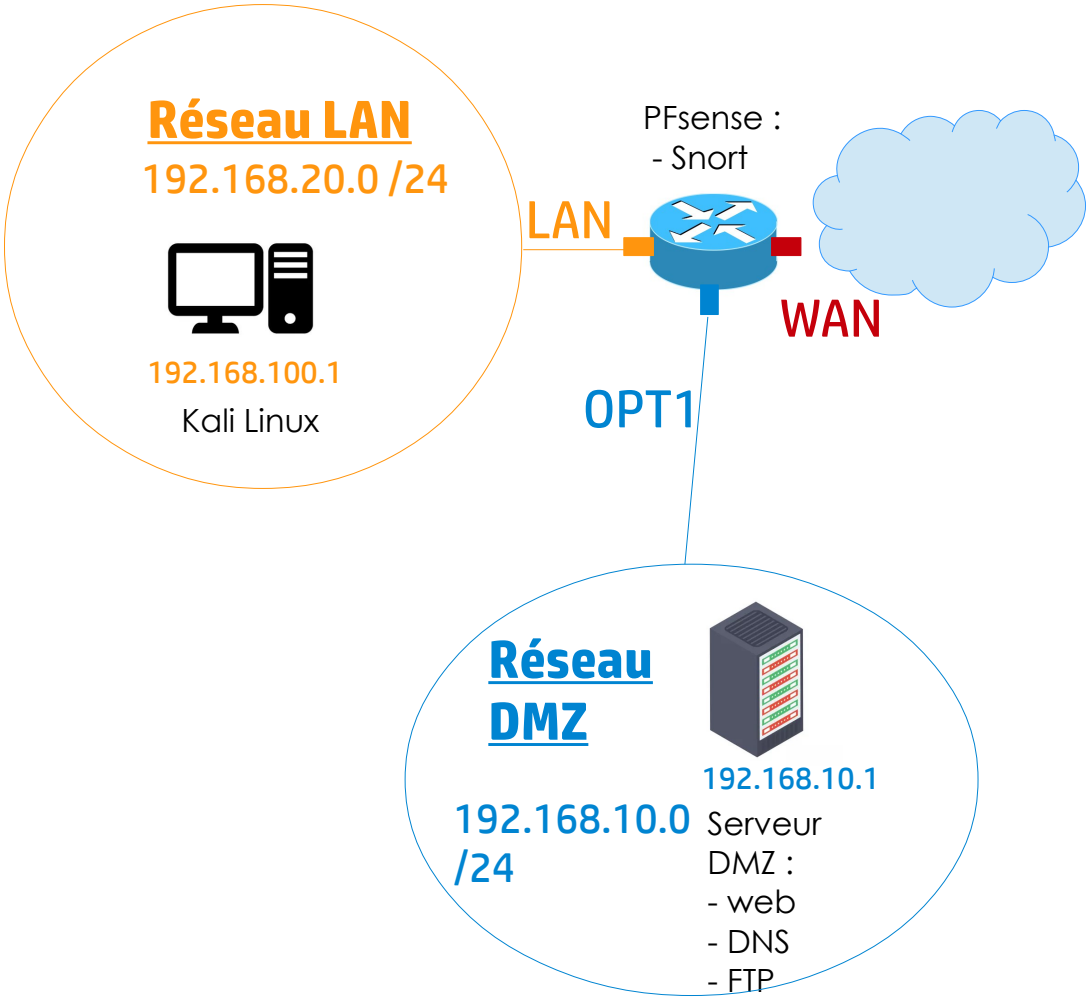
Nous nous trouvons dans la petite équipe du service informatique. Notre maitre de stage, qui n'a pas encore installé d'IDS, nous missionne pour faire des recherches sur des IDS, que nous en choisissons un et que nous fassions des tests de pénétration pour voir comment le logiciel réagit.

Notre tuteur nous demande pour l'instant de créer des scénarios pour 2 types d'attaques informatiques et d'en faire une documentation.

1.1. Objectifs

- Définir ce qu'est un IDS et en sélectionner un
- Définir 2 types d'attaques informatiques
- Proposer et simuler des scenarii d'attaques pour ces piratages
- Voir comment l'IDS réagit pour chaque scénario
- Rédiger une documentation accessible par tous

1.2. Architecture



2. Organisation

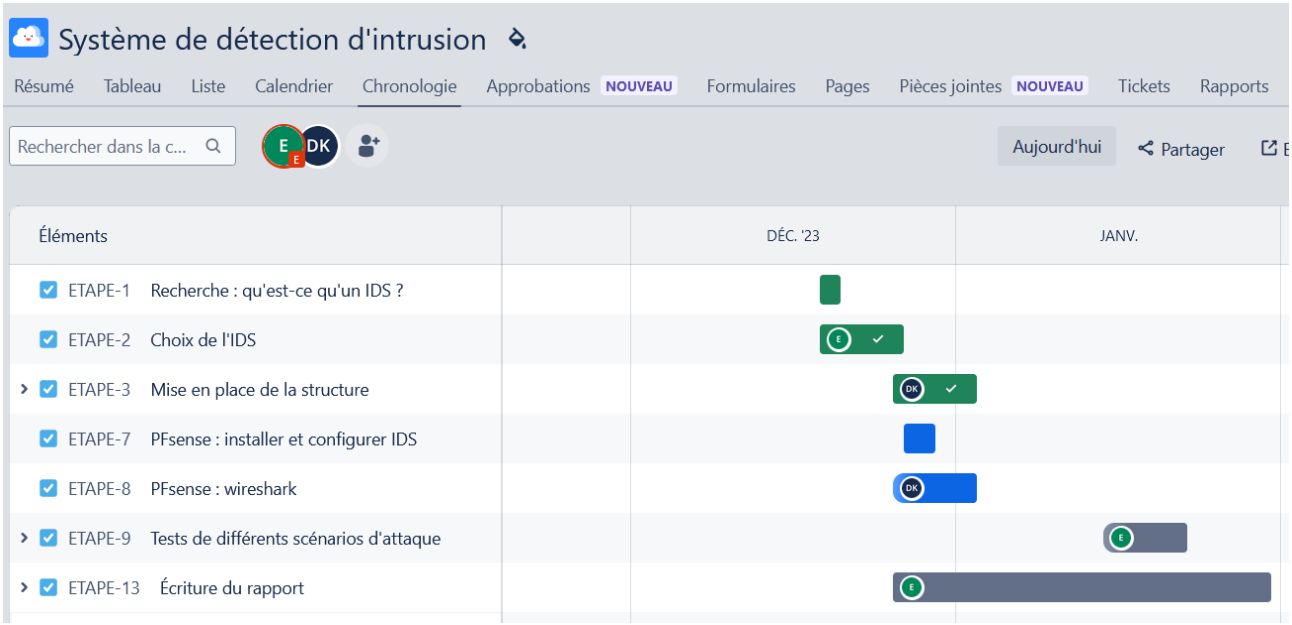
Nous sommes deux stagiaires sur la mission. Afin de mieux nous organiser, nous répartir les tâches et répondre correctement aux objectifs, nous avons décidé d'utiliser un outil de gestion de projets.

Notre tuteur/formateur nous a proposé l'outil de gestion utilisé par l'entreprise : JIRA.

Nous avons fait une réunion pour découper le projet en plusieurs étape, la répartition des tâches et sur les délais que nous avions à tenir.

Étapes du projet :

Clé	Résumé
ETAPE-1	Recherche : qu'est-ce qu'un IDS ?
ETAPE-2	Choix de l'IDS
ETAPE-3	Mise en place de la structure
ETAPE-4	Installation Kali Linux
ETAPE-5	Installation serveur DMZ
ETAPE-6	Installation PFsense
ETAPE-7	PFsense : installer et configurer IDS
ETAPE-8	PFsense : wireshark
ETAPE-9	Tests de différents scénarios d'attaque



Sur cette capture d'écran, nous voyons la répartition des différentes étapes du projet :

étapes 1, 4, 5, 6 : la 2ème stagiaire

étapes 2, 3, 7, 8, 9 : moi-même

L'étape de l'écriture du rapport est une tâche individuelle.

Après l'étape 8, nous nous sommes réunis pour parler du délai qui nous restait. Nous étions allés plus vite que prévu sur certaines étapes et avons pris du retard sur d'autres.

Nous nous sommes finalement rendu compte que nous étions dans les temps et avons pris la décisions de faire tous les deux les tests des scénarios d'attaques.

Cela a eu pour but de pour voir si la reproduction des scénarios étaient possible par tous ceux qui auront la documentation en main.

J'ai mis en place et tester les scénarios en premier et ma collègue les a reproduit avec la documentation que j'avais mis en place.

2. Qu'est-ce qu'un IDS ?

IDS, "**Intrusion detection System**" est un **système de détection d'intrusion**. Il permet

er le trafic réseau, de détecter des activités suspectes voire
es ou encore des violations des politiques de sécurité et d'en faire une
ministrateur lit les résultats et décide ce qui doit être mis en place face à une ou des attaques données.

Il existe deux catégories d'IDS :

- détection par signatures : reconnaissance de comportements de programmes malveillants par analyse des paquets réseau. Ce système maintient une base de données signatures d'attaque avec laquelle il compare les paquets réseau. Les dernières attaques non répertoriées peuvent échapper à sa vigilance.
- détections par anomalies : compare l'activité du réseau à un modèle de référence. Le système détecte et signale les moindres écarts. Il fait appel à l'apprentissage automatique pour affiner en permanence le modèle de référence. Ce système peut être facilement sujet aux faux positifs (exemple : un utilisateur autorisé qui vient de se connecter pour la première fois à une ressource).

Nous pouvons installer deux types d'IDS. Un système de détection d'intrusion réseau (**NIDS**), qui est installé à un point stratégique du réseau. Il surveille le trafic entrant et sortant vers les appareils du réseau. Le deuxième est un système de détection d'intrusion hôte (**HIDS**). Il est installé sur un poste spécifique (ordinateur, routeur, serveur) et surveille uniquement l'activité de l'appareil sur lequel il est installé.

Un IDS peut donc aider à accélérer et automatiser la détection de menaces sur le réseau. Il peut aussi aider dans le respect de la conformité des réglementations.

3. Choix de l'IDS

Nous allons tout d'abord nous centrer sur des IDS open Source.

ESICAD

pen source utilisé par des petites-moyennes entreprises (50-200 [illegible]). Il a été développé depuis plus longtemps que Suricata et est toujours en cours d'évolution pour s'adapter au marché.

Il utilise des mécanismes de surveillance en temps réel. Il se base sur des règles qui combinent des méthodes d'anomalie, de protocole et d'inspection de signature pour détecter un activité potentiellement malveillante.
Il peut être déployé dans n'importe quel environnement réseau et n'importe quel système d'exploitation. Il est flexible car son code open source permet d'ajouter de nouvelles fonctionnalités.

SURICATA : est aussi un open source qui utilise la détection par signatures. Suricata a été développé plus récemment que Snort et il a donc plus de fonctionnalités que celui-ci. Il possède un moteur IDS mutlithread supportant le langage de signature de Snort. Il a aussi une faculté permettant d'extraire des fichiers pour les analyser ultérieurement et il détecte automatiquement les protocoles.

Ces 2 IDS sont aussi des IPS (système de prévention et d'intrusion qui analyse également les paquets mais peuvent également les bloquer, ce qui peut stopper automatiquement certaines attaques), mais nous allons seulement nous servir de leur fonction IDS dans le projet. Nous pouvons tout de même souligner que si notre tuteur souhaite mettre en place plus tard un IPS, il pourra juste paramétrer un des IDS en place.

Suricata nous paraît un peu plus complet mais pour le cadre de notre petite entreprise, Snort nous semble suffisant et peut-être plus flexible avec ses différentes méthodes de détection et le fait qu'il puisse être déployé sur une large catégorie de systèmes d'exploitation.

4. Prérequis pour la réalisation des scenarii

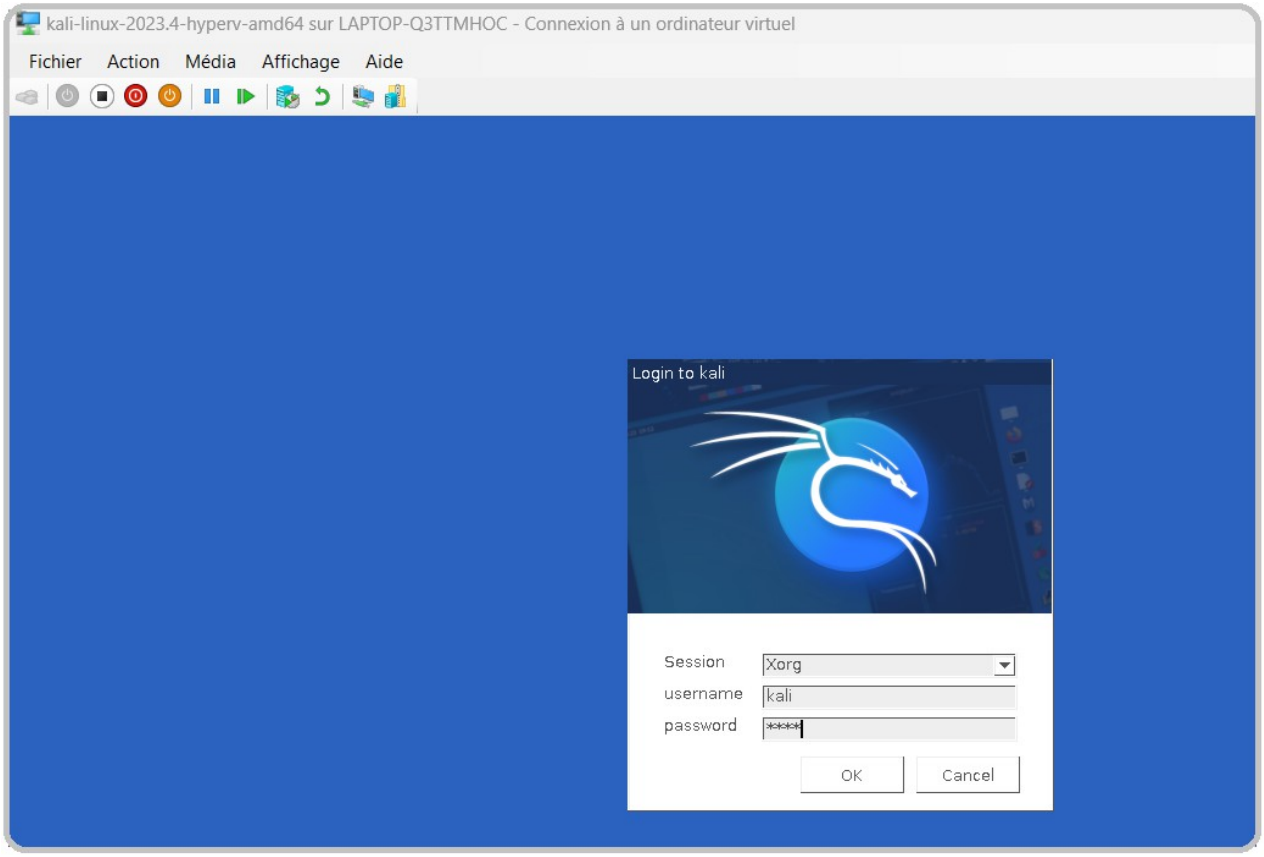
4.1.Kali Linux

Pour nos tests de pénétration, nous allons utiliser une distribution Linux dans laquelle de nombreux outils facilitant ces tests sont présents : Kali Linux.
Elle a été créée dans le but premier d'identifier des failles de sécurité, de récupérer des données perdues, d'analyser les mots de passe et de décoder des documents chiffrés. Plus de 600 outils de sécurités différents sont proposés avec cette distribution open source. Ce système est considéré très fiable et est même utilisé par des pirates « éthiques ».

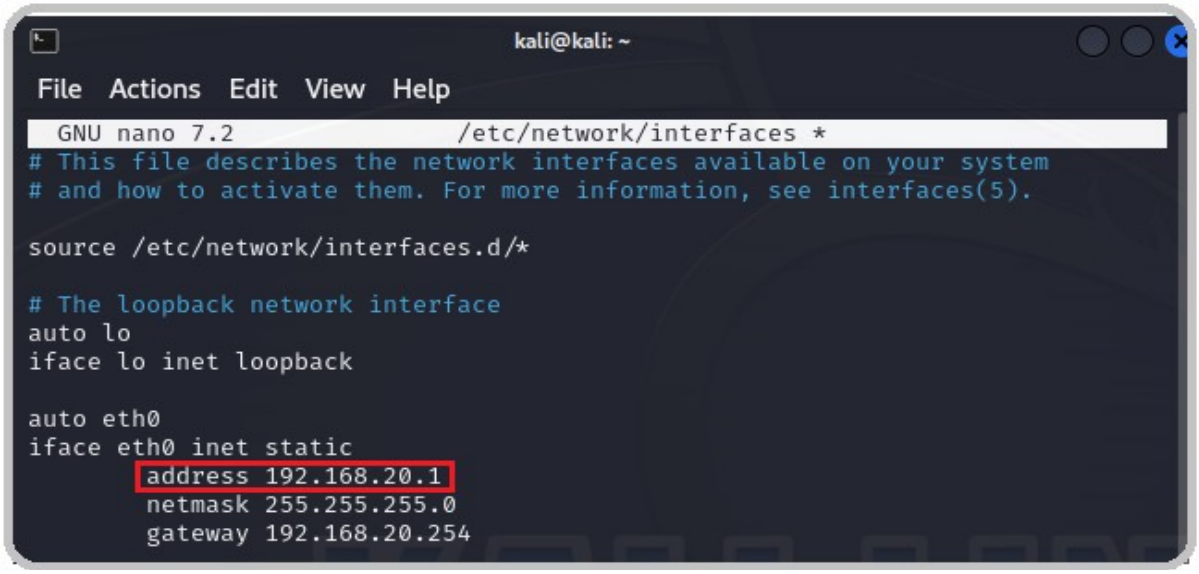
Il est possible d'installer KaliLinux à partir d'une image préfaite. Nous utilisons Hyper-V comme virtualiseur. Nous choisissons donc une image KaliLinux préfaite sur Hyper-V.

- Après installation, lancement :

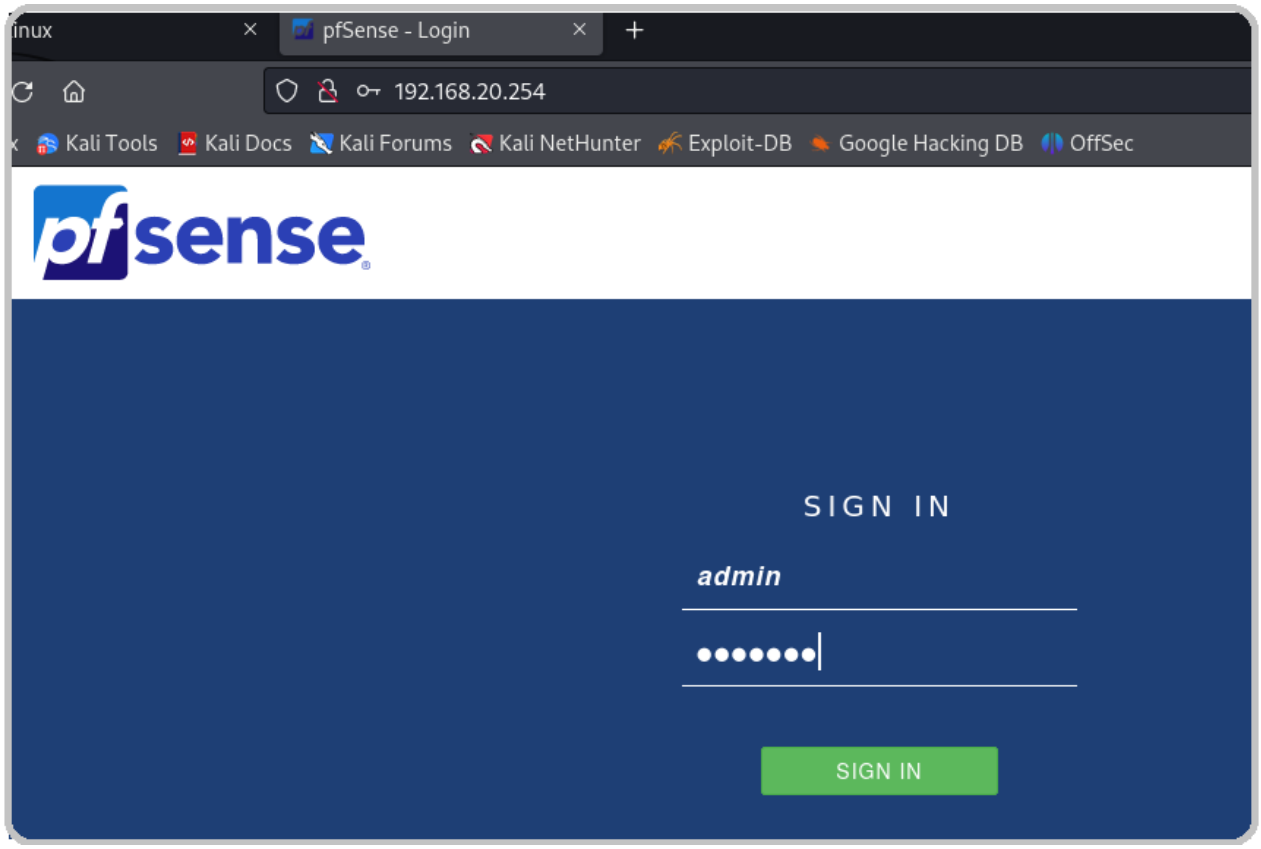
identifiant par défaut : kali
mot de passe par défaut : kali



- Configuration IP



– Connexion à PFsense

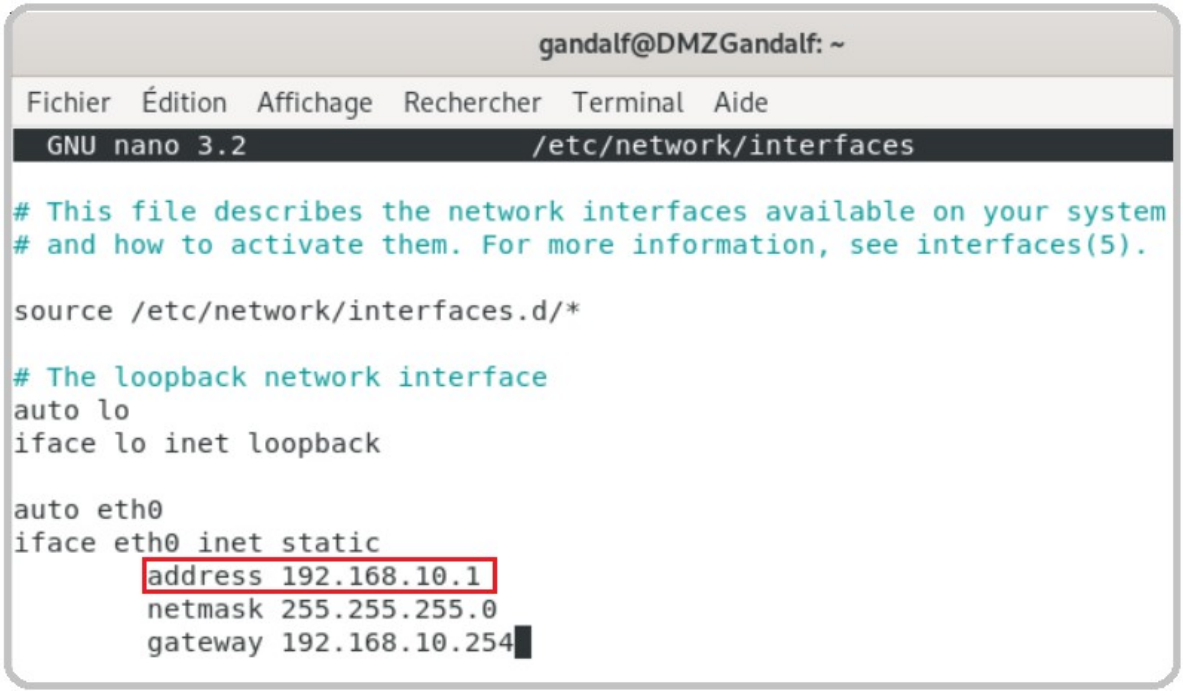


4.2. Machine DMZ

Nous allons à présent installer et configurer notre serveur présent dans le réseau DMZ. Il va regrouper un serveur DNS, un serveur web et un serveur FTP. Nous avons réuni ces trois serveurs sur une seule machine pour plus de facilité mais ils auraient très bien pu être présent sur 3 ordinateurs différents.

Comme l'installation et le paramétrage de ce serveur n'est pas le cœur de l'objectif de ce projet, nous ne vous détaillons pas la procédure.
Nous vous mettons tout de même à disposition les fichiers de configuration et tests de validation pour vous montrer le bon fonctionnement des différents services.

- Configuration IP



- Serveur web avec Apache2

- Serveur DNS

Vérification du bon fonctionnement

- zone directe

```
root@DMZGandalf:/etc/bind# dig DMZGandalf.lotr.lan

; <<>> DiG 9.11.5-P4-5.1+deb10u9-Debian <<>> DMZGandalf.lotr.lan
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 12674
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 1, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: e96c7d09c416acccfeef2cf86597c3fd4bf06623b001f32a (good)
;; QUESTION SECTION:
;DMZGandalf.lotr.lan.           IN      A

;; ANSWER SECTION:
DMZGandalf.lotr.lan.    10800   IN      A       192.168.10.1

;; AUTHORITY SECTION:
lotr.lan.               10800   IN      NS      DMZGandalf.lotr.lan.

;; Query time: 0 msec
;; SERVER: 192.168.10.1#53(192.168.10.1)
;; WHEN: ven. janv. 05 09:55:25 CET 2024
;; MSG SIZE  rcvd: 106
```

- zone indirecte

```
root@DMZGandalf:/etc/bind# dig -x 192.168.10.1

; <<>> DiG 9.11.5-P4-5.1+deb10u9-Debian <<>> -x 192.168.10.1
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 61134
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 1, ADDITIONAL: 2

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: 18321046b0bb92a1c48d144d6597c4cb5db1575bc8596042 (good)
;; QUESTION SECTION:
;1.10.168.192.in-addr.arpa.     IN      PTR

;; ANSWER SECTION:
1.10.168.192.in-addr.arpa. 10800 IN     PTR     DMZGandalf.lotr.lan.

;; AUTHORITY SECTION:
10.168.192.in-addr.arpa. 10800  IN      NS      DMZGandalf.lotr.lan.

;; ADDITIONAL SECTION:
DMZGandalf.lotr.lan.    10800   IN      A       192.168.10.1

;; Query time: 0 msec
```

- ping site internet

```
root@DMZGandalf:/etc/bind# ping www.lotr.lan
PING www.lotr.lan (192.168.10.1) 56(84) bytes of data.
64 bytes from DMZGandalf.lotr.lan (192.168.10.1): icmp_seq=1 ttl=64 time=0.011 ms
64 bytes from DMZGandalf.lotr.lan (192.168.10.1): icmp_seq=2 ttl=64 time=0.050 ms
64 bytes from DMZGandalf.lotr.lan (192.168.10.1): icmp_seq=3 ttl=64 time=0.046 ms
64 bytes from DMZGandalf.lotr.lan (192.168.10.1): icmp_seq=4 ttl=64 time=0.033 ms
64 bytes from DMZGandalf.lotr.lan (192.168.10.1): icmp_seq=5 ttl=64 time=0.054 ms
^C
--- www.lotr.lan ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 91ms
rtt min/avg/max/mdev = 0.011/0.038/0.054/0.017 ms
root@DMZGandalf:/etc/bind#
```

- Serveur FTP

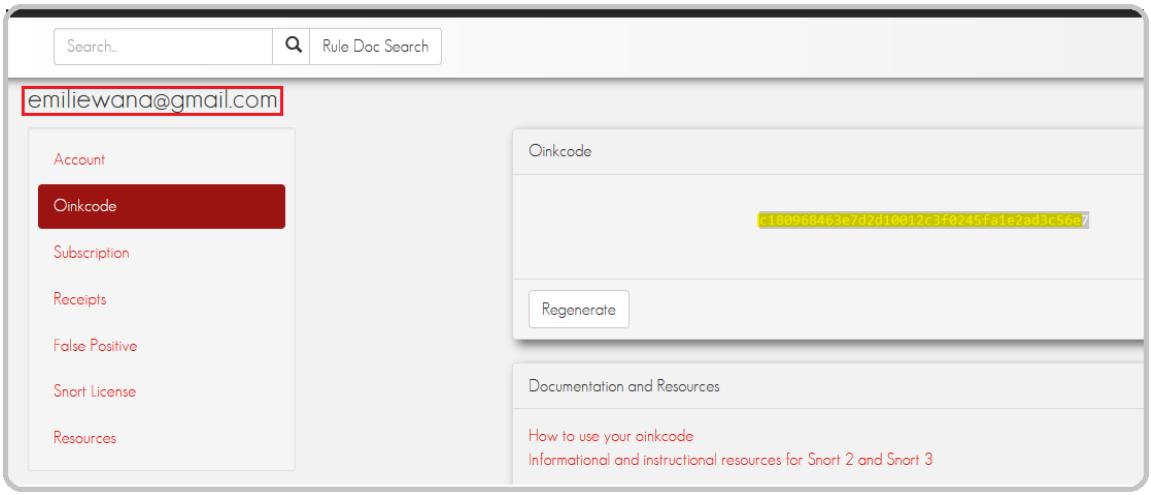
```
gandalf@DMZGandalf: ~
  GNU nano 5.4                          ftp-perso.conf *
#Nom du serveur
ServerName "DMZGandalf.lotr.lan"

#Message de connexion
DisplayLogin "La connexion FTP s'est bien effectuée"

#Désactiver IPV6
UseIPV6 off

#Spécifie le répertoire FTP auquel l'utolisateur est autorisé à accéder
DefaultRoot état

#Autoriser la connexion uniquement aux membres du groupe "ftp"
<Limit LOGIN>
Denygroup !ftp
</Limit>
```

4.3. Installation Snort avec PFsense

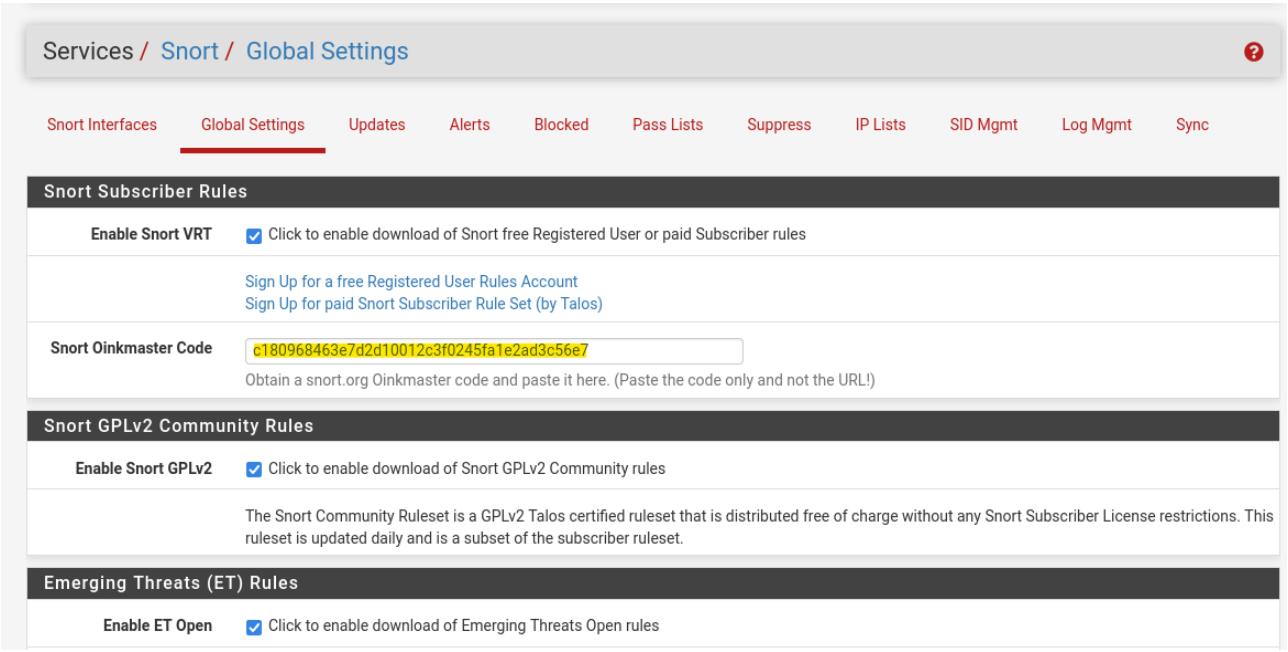
Nous téléchargeons et installons Snort, dans les "available packages" proposés par PFsense.

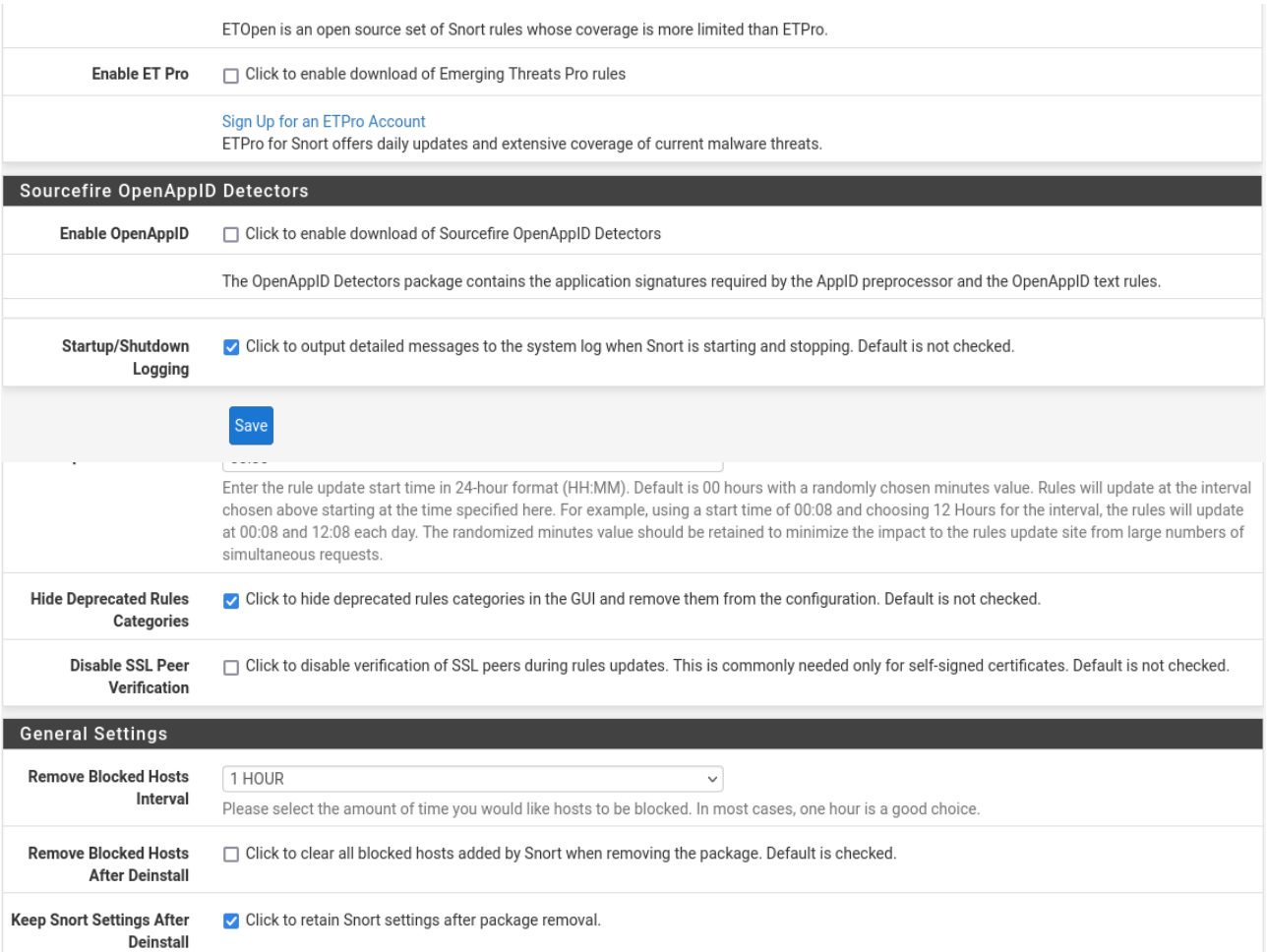
Paramétrage

- Création d'un compte sur Snort et récupération du « Oinkmaster Code » qui est demandé dans la configuration de Snort

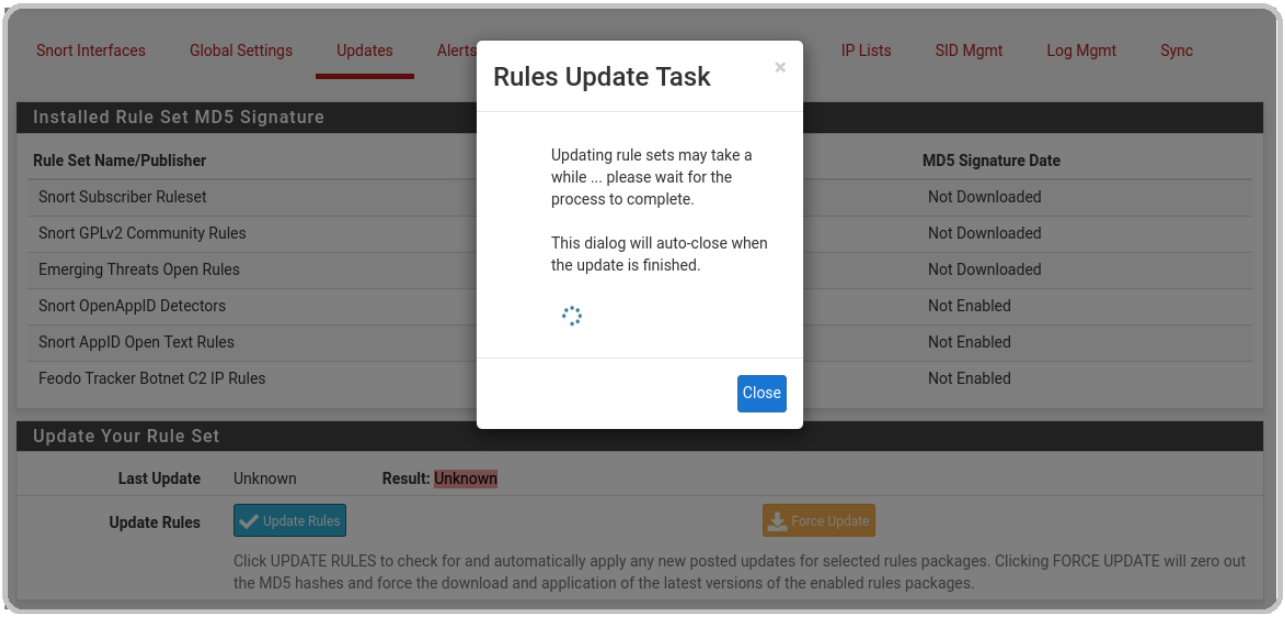


- Dans "Services" / "Snort" / "Global Settings", nous activons le code Oinkmaster, les règles créés par la communauté, les règles pour les menaces les plus récentes notamment.

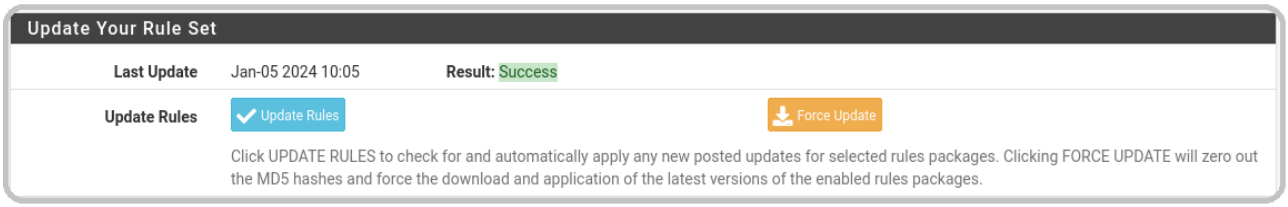




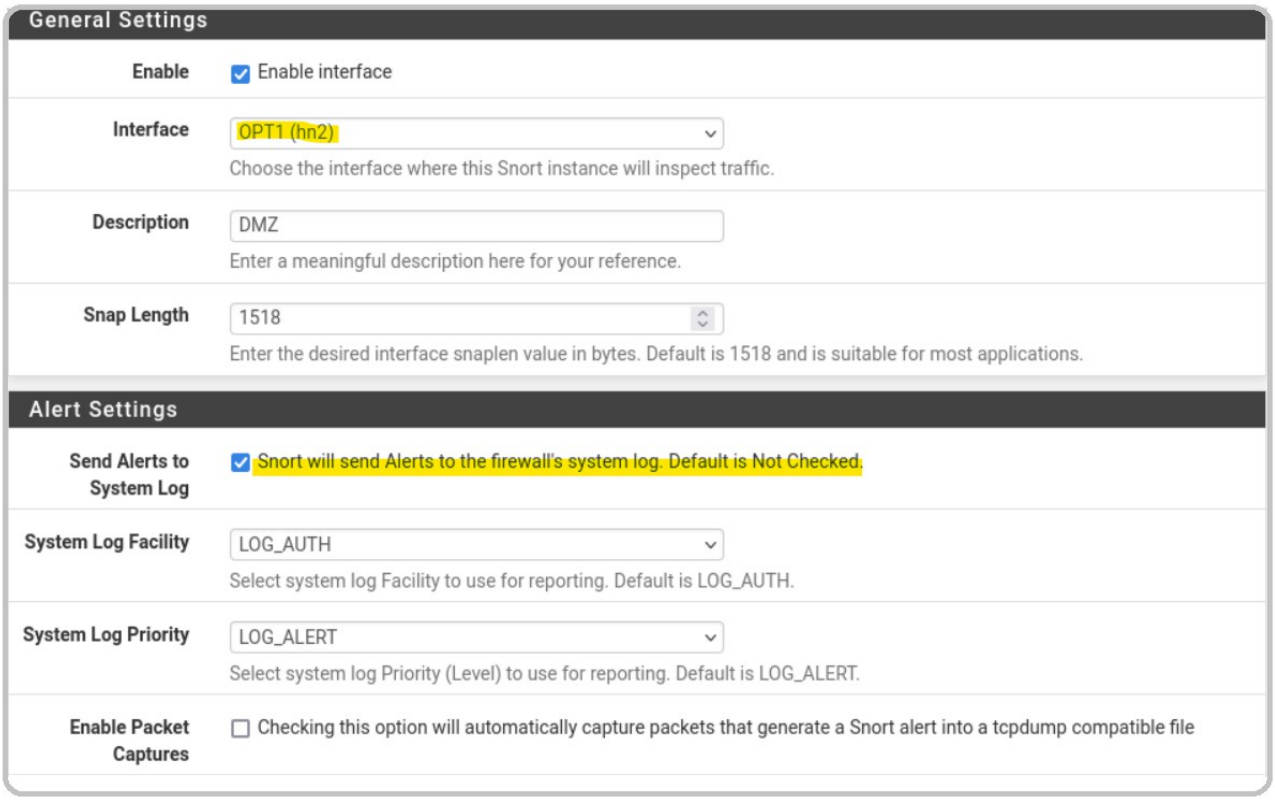
– Dans l'onglet "Update", nous cliquons sur "Update Rules".



Lorsque c'est terminé



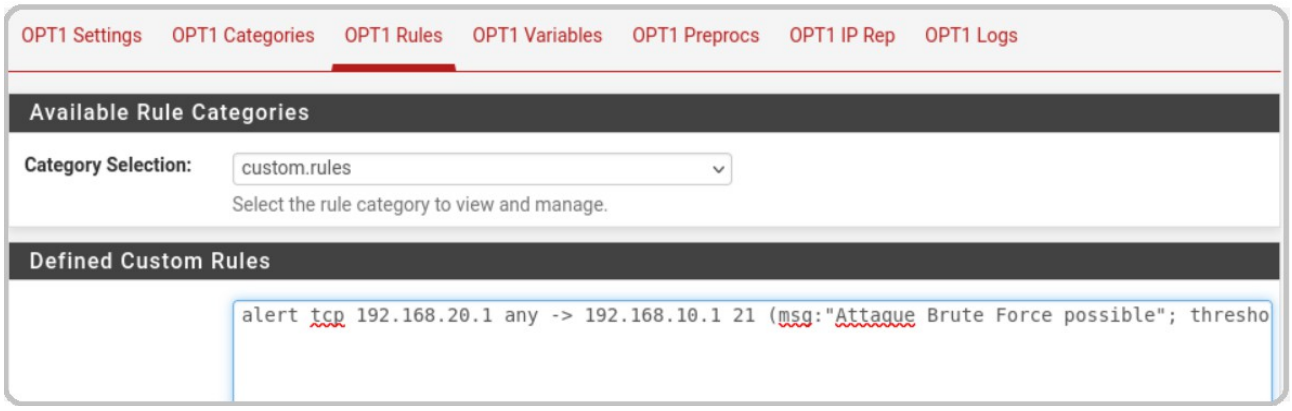
- Ajout de l'interface à surveiller, ici OPT1(192.168.10.0/24), là où se trouve notre réseau DMZ qui va subir des attaques. Dans une situation réelle et non de test, nous aurions activé Snort sur les 3 interfaces car les attaques peuvent venir des interfaces LAN et WAN. Dans ce cas, l'IDS détecte immédiatement et si l'IPS avait été activé, il l'aurait bloquée avant que l'attaque n'arrive sur l'interface OPT1.



- Démarrage de l'interface

- Pour l'attaque Brute Force, nous allons entrer une règle personnalisée. Pour cela, nous allons dans l'interface Snort OPT1 / OPT1 Rules et nous sélectionnons la catégorie "custom.rules"
  Nous entrons la règle suivante :

alert tcp 192.168.20.1 any → 192.168.10.1 21 (msg : « Attaque Brute Force possible » ; threshold : type threshold, track by_dst, count 3, seconds 60 ; sid : 1000001;)



4.4. Utilisation Wireshark

C'est une application open source. C'est un analyseur de réseau qui capte et affiche, par une interface, les données circulant dans le réseau (dans les deux sens).

Un administrateur ou technicien peuvent s'en servir pour déterminer si une machine est infectée ou débugger des problèmes de connectivité grâce à des filtres qui permettent d'affiner la recherche.

Dans notre cas, nous nous servons de Wireshark pour voir ce qui se passe sur le réseau lorsqu'une attaque informatique est lancée.

5. Différents types d'attaques et scenarii

Il existe un très grand nombre de cyberattaques différentes, de la Bruteforce en passant par les malwares ou chevaux de Troie, des injections SQL, etc.
Les objectifs de ces attaques sont différentes comme le vol de données, empêcher l'accès à un service, mais elles mettent toujours à mal le système visé.
Nous allons sélectionner quelques types d'attaques, faire des tests d'intrusion grâce à Kali Linux, ensuite regarder la trame et enfin voir comment notre IDS les détecte.

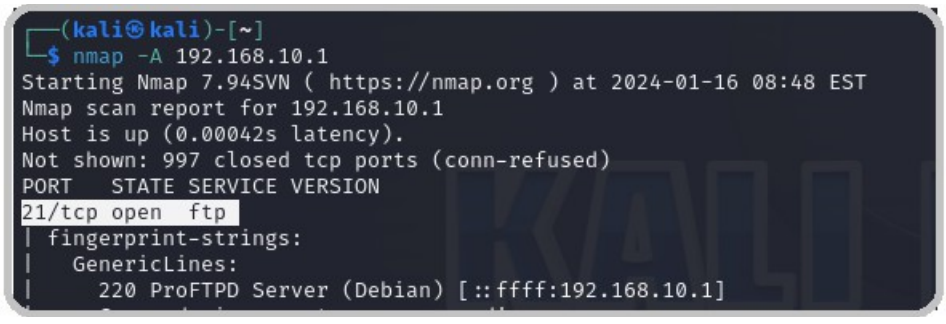
5.1. Bruteforce

Selon la cnil, dans une "attaque par force brute" (bruteforce attack), le pirate tente de se connecter à un service qui possède un mot de passe ou une clé, en essayant différentes combinaisons possibles jusqu'à ce que cela fonctionne.

Exemple d'attaque avec récupération des identifiants et mot de passe pour le service FTP

Pour effectuer une attaque Bruteforce, nous allons utiliser l'outil **Hydra** déjà installé sous KaliLinux.
Hydra est un outil open source qui permet de forcer des noms d'utilisateurs, de mots de passe de différents protocoles (FTP, SSH, Telnet, SMB...) avec l'aide de dictionnaires de mots ou par essai de tous les caractères possibles.

- Avant d'initier l'attaque, nous utilisons la commande nmap pour vérifier si le port 21 est ouvert.



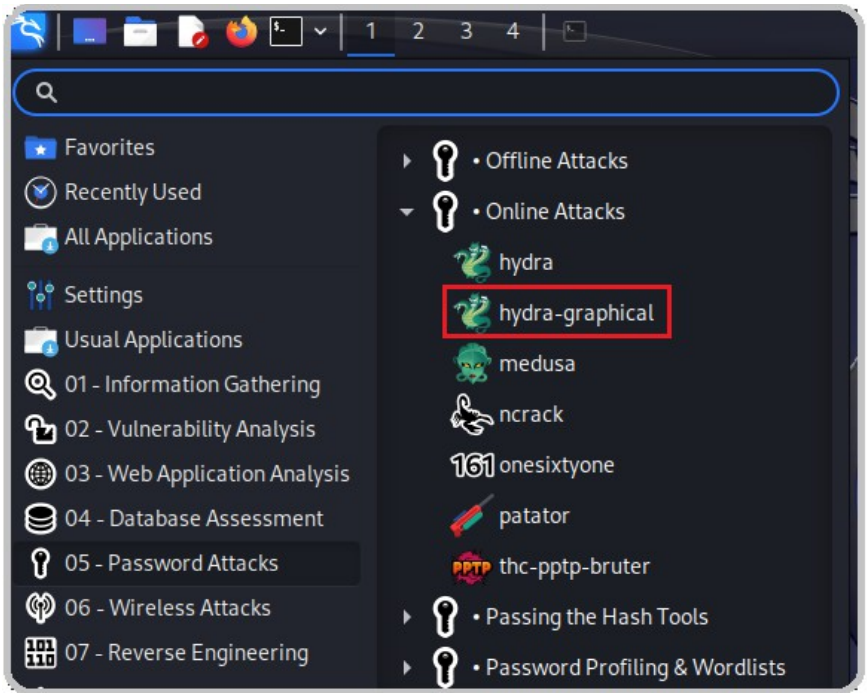
→ c'est le cas

– Sur Snort, dans les alertes, nous pouvons déjà remarquer des lignes liées à la commande "nmap". Cela est un premier indicateur de l'attaque.

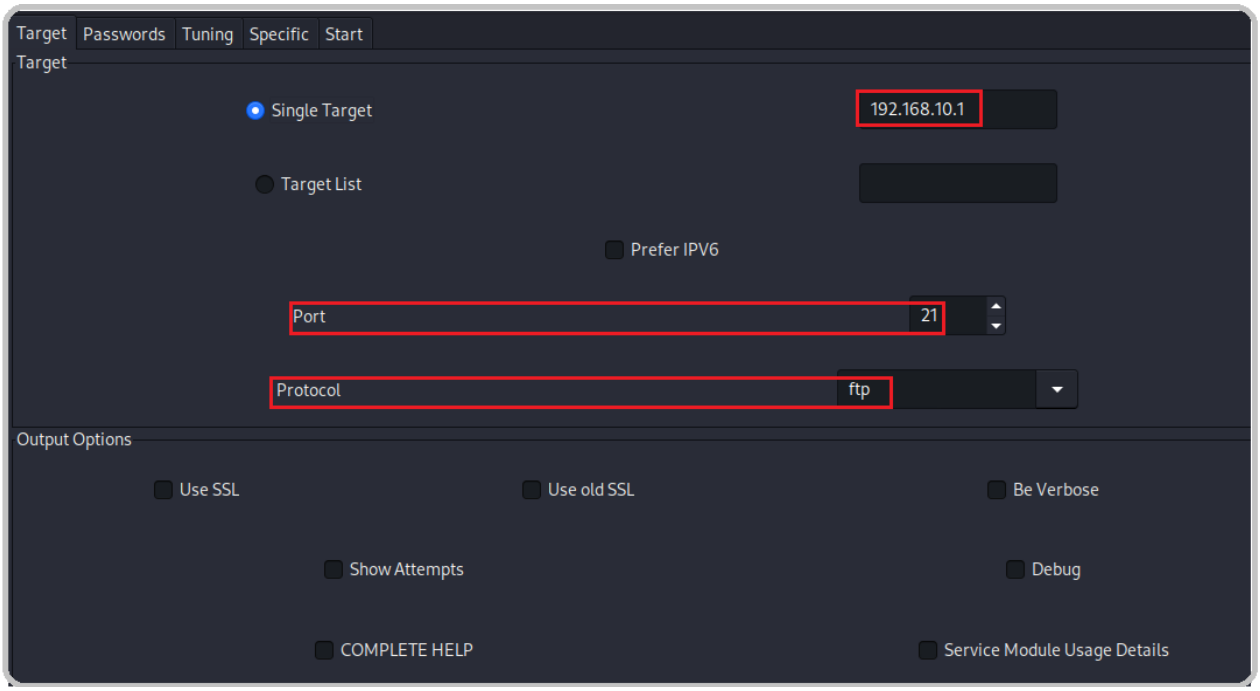
9 Entries in Active Log

Date	Action	Pri	Proto	Class	Source IP	SPort	Destination IP	DPort	GID:SID	Description
2024-01-28 10:23:52	⚠	3	TCP	Unknown Traffic	192.168.20.1	36530	192.168.10.1	80	119:31	(http_inspect) UNKNOWN METHOD
2024-01-28 10:23:16	⚠	2	TCP	Potentially Bad Traffic	192.168.20.1	60982	192.168.10.1	21	125:2	(ftp_telnet) Invalid FTP Command
2024-01-28 10:22:49	⚠	2	TCP	Potentially Bad Traffic	192.168.20.1	41912	192.168.10.1	21	125:2	(ftp_telnet) Invalid FTP Command
2024-01-28 10:22:54	⚠	2	TCP	Potentially Bad Traffic	192.168.20.1	50774	192.168.10.1	21	125:2	(ftp_telnet) Invalid FTP Command
2024-01-28 10:22:44	⚠	2	TCP	Potentially Bad Traffic	192.168.20.1	41910	192.168.10.1	21	125:2	(ftp_telnet) Invalid FTP Command
2024-01-28 10:22:34	⚠	2	TCP	Potentially Bad Traffic	192.168.20.1	49402	192.168.10.1	21	125:2	(ftp_telnet) Invalid FTP Command
2024-01-28 10:21:44	⚠	2	TCP	Potentially Bad Traffic	192.168.20.1	56872	192.168.10.1	21	125:2	(ftp_telnet) Invalid FTP Command
2024-01-28 10:21:39	⚠	2	TCP	Potentially Bad Traffic	192.168.20.1	58194	192.168.10.1	21	125:2	(ftp_telnet) Invalid FTP Command
2024-01-28	⚠	2	TCP	Potentially Bad Traffic	192.168.20.1	58182	192.168.10.1	21	125:2	(ftp_telnet) Invalid FTP Command

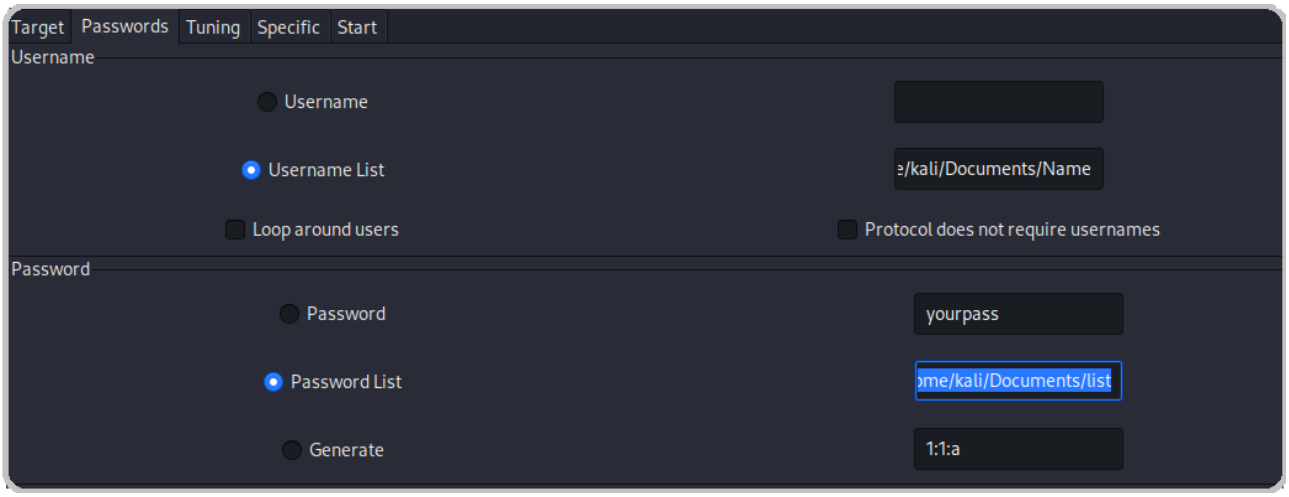
– Ensuite, nous exécutons la version Hydra avec interface

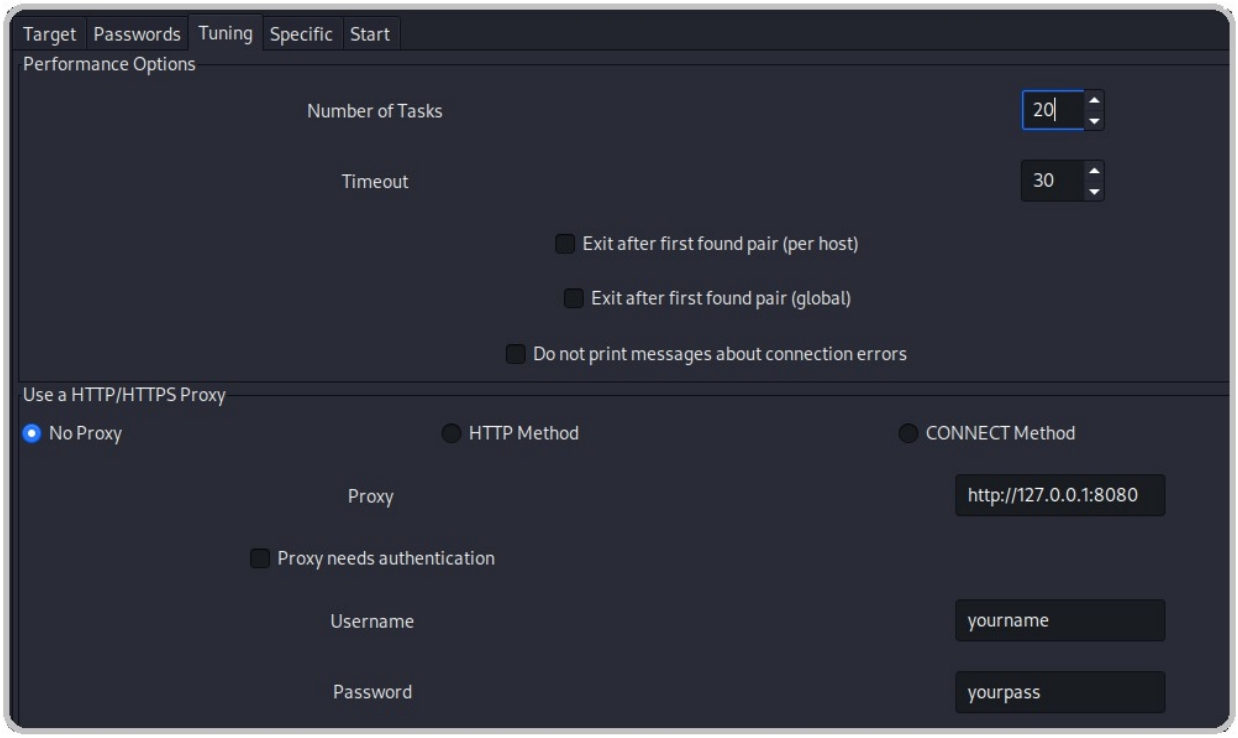


- Nous entrons l'adresse IP de l'ordinateur cible, le port et le protocole

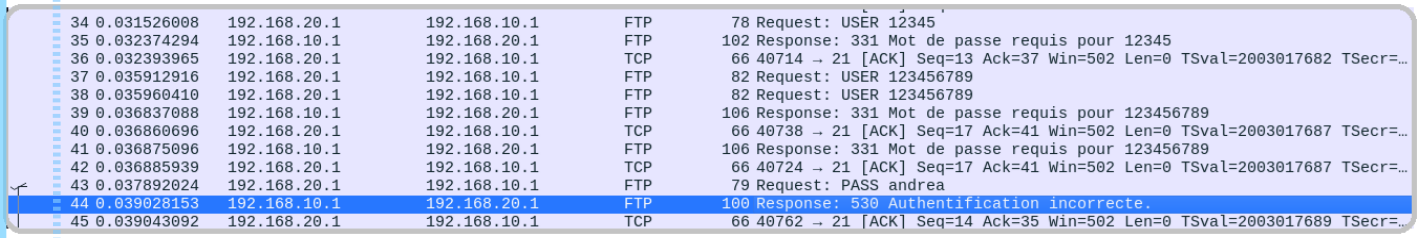


- Ensuite, pour trouver le nom d'utilisateur et le mot de passe, nous nous servons de dictionnaires de mots. Pour notre exemple, nous avons ajouté dans les listes les utilisateurs et les mots de passe pour que le logiciel Hydra les trouve.



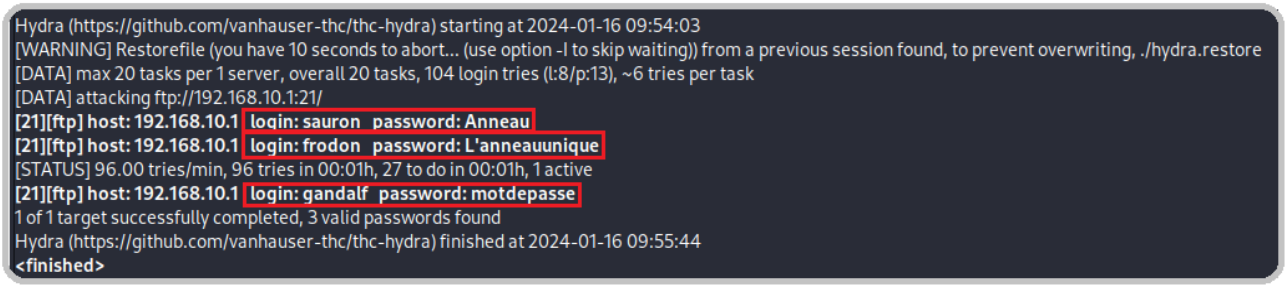


– Nous lançons la recherche des identifiants et regardons en parallèle sur Wireshark ce qui se passe.

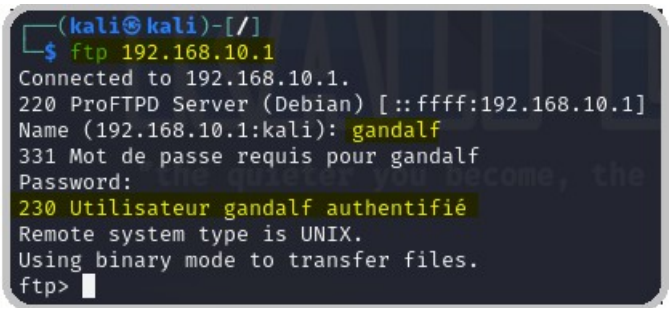


→ nous remarquons bien qu'Hydra tente différents utilisateurs et mot de passe qui sont pour l'instant incorrects.

– Voici enfin le résultat de la recherche de comptes ftp par Hydra.



- Vérification : ces identifiants sont-ils corrects ?



→ oui

- Comment Snort réagit-il ?

Most Recent 250 Entries from Active Log

Date	Action	Pri	Proto	Class	Source IP	SPort	Destination IP	DPort	GID:SID	Description
2024-02-06 14:25:07	⚠	0	TCP		192.168.20.1	38778	192.168.10.1	21	1:1000001	Attaque Brute Force possible
2024-02-06 14:25:07	⚠	0	TCP		192.168.20.1	38818	192.168.10.1	21	1:1000001	Attaque Brute Force possible
2024-02-06 14:25:07	⚠	0	TCP		192.168.20.1	38916	192.168.10.1	21	1:1000001	Attaque Brute Force possible
2024-02-06 14:25:07	⚠	0	TCP		192.168.20.1	38904	192.168.10.1	21	1:1000001	Attaque Brute Force possible
2024-02-06 14:25:07	⚠	0	TCP		192.168.20.1	38792	192.168.10.1	21	1:1000001	Attaque Brute Force possible

Nous venons donc d'effectuer une attaque brute force sur notre serveur DMZ. Cette attaque a réussi car nous avons, grâce à Hydra, trouvé 3 identifiants et mot de passe corrects pour se connecteur au serveur FTP.

5.2. DdoS ou Dos

Une attaque DDoS (Distributed Denial of Service) plus précisément par déni de service distribué consiste en rendre inaccessible un service en envoyant un nombre important de requêtes simultanément pour provoquer une saturation ou en exploitant une faille de sécurité provoquant une panne ou un fonctionnement très dégradé du service.

Les attaques DDos sont similaires aux attaques Dos mais ont quelques différences. Une attaque DDos implique plusieurs systèmes, souvent à l'aide d'un botnet, attaquant un système alors que pour une Dos, un seul système attaque. Une attaque DDos peut être plus rapide car elle envoie plus de requêtes et il est moins facile de détecter d'où elle provient.

Flooding

C'est une forme d'attaque DDoS : saturer un système avec des demandes
Il existe différentes attaques de flood comme le "ping flood", "HTTP flood", "UDP flood"...

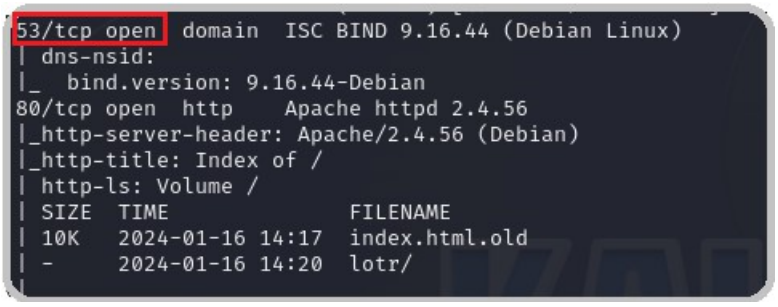
Nous allons tester dans notre cas l'attaque « SYN flood » sur le port http pour empêcher l'accès au serveur web. Le pirate envoie énormément de paquets sur le poste serveur ciblé et son port, ici 80. Le serveur doit répondre à chaque requête ce qui à terme peut le submerger et bloquer momentanément l'accès au serveur web.
Généralement, le pirate se sert de botnet au vue des capacités actuelles des serveurs mais dans notre cas, nous allons attaquer seulement avec la machine Kali Linux.

Exemple d'attaque de déni de service sur le port 80 (HTTP)

Pour cela, nous allons utiliser l'outil Hping3. C'est une application qui s'exécute par terminal sous Linux. Elle permet l'envoi de paquets TCP, UDP. Elle sert à tester des réseaux et des hôtes mais en premier lieu, lors de sa création, elle était surtout utilisée pour détecter des problèmes de cybersécurité. Hping3 peut envoyer des paquets via un port spécifique, masquer l'adresse IP source, émettre un nombre de paquets très important et bien d'autre grâce à un grand nombre de paramètres.
Elle est déjà installée sur Kali Linux.

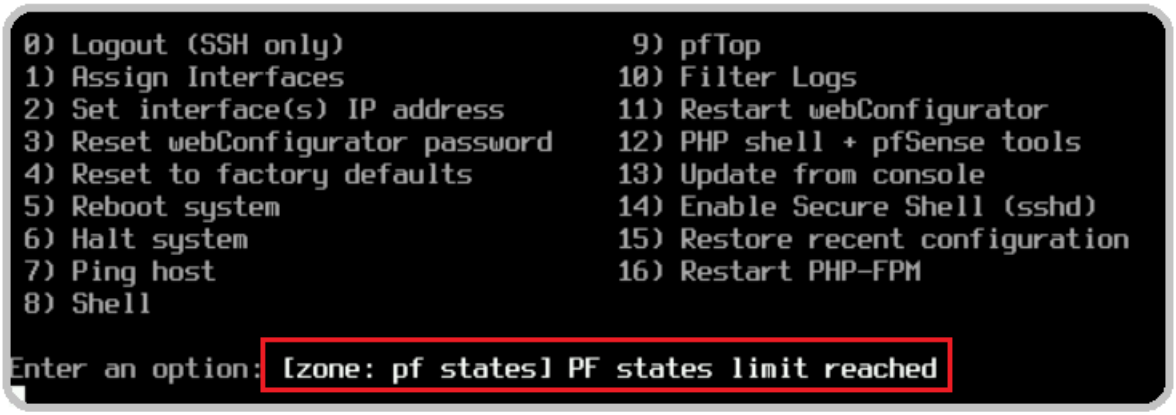
Avec cette application, nous allons effectuer une attaque sur le routeur PFsense, sur le port 80. Cela aura pour conséquence que personne ne pourra accéder à l'interface internet PFsense.

- Avant d'initier l'attaque, nous utilisons la commande nmap pour vérifier que le port 80 est bien ouvert

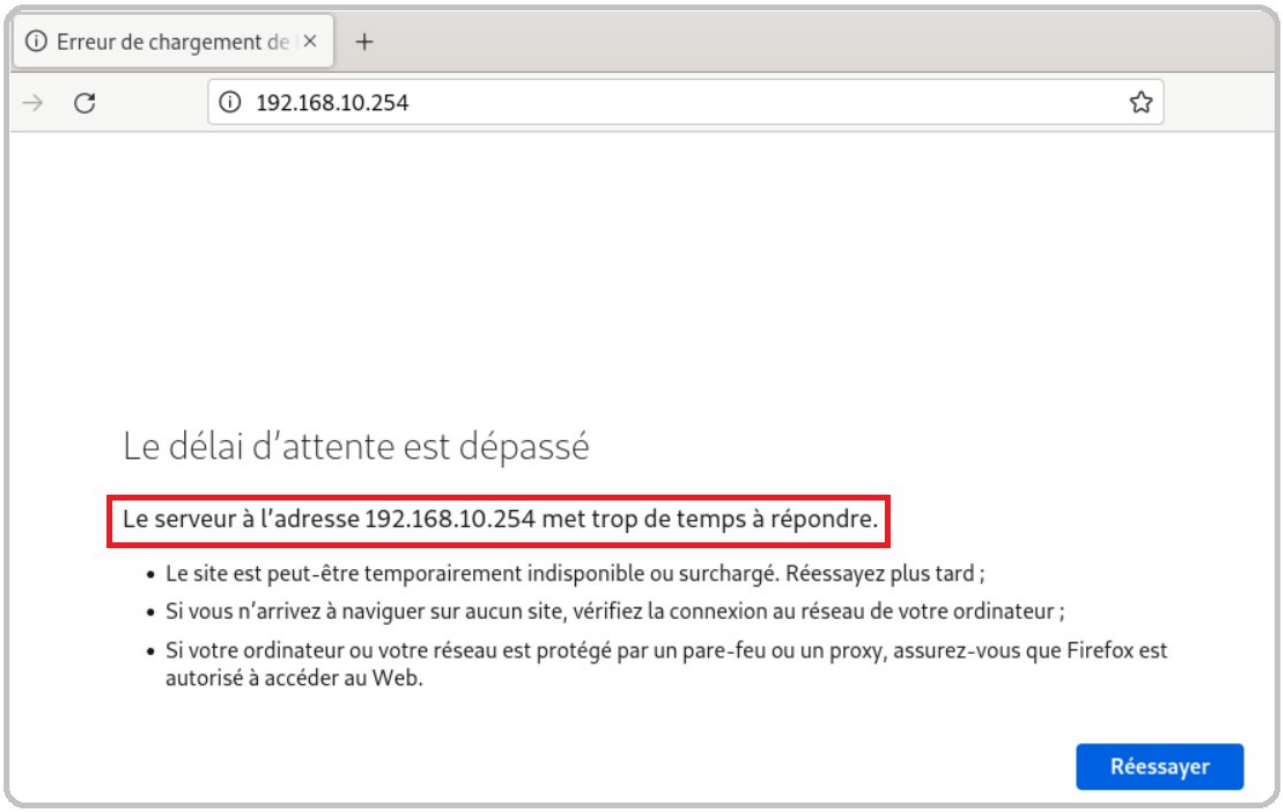


- Commande

- Nous remarquons tout d'abord sur la machine PFsense un message qui s'affiche quelques instants après le lancement de l'attaque

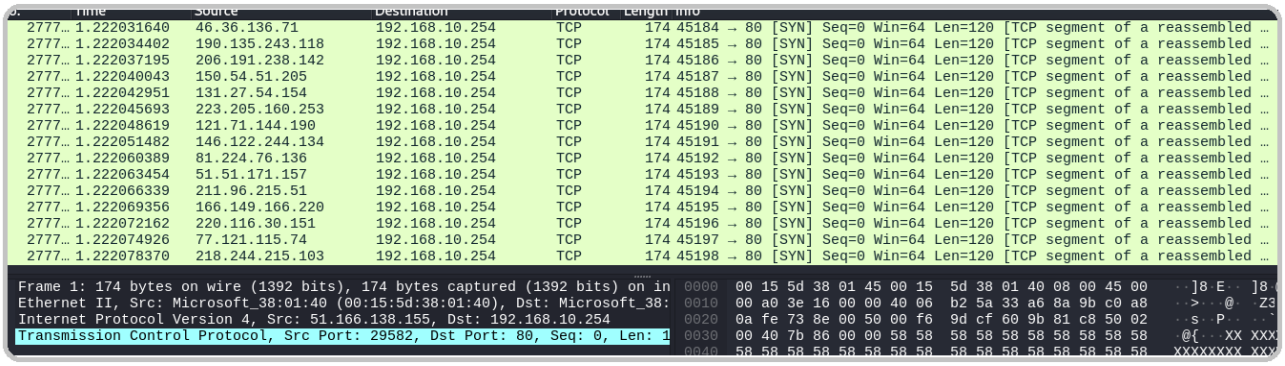


- Ensuite, nous essayons de nous connecter à l'interface PFsense via internet sur notre poste DMZ (192.168.10.1)



→ c'est impossible

- Sur Wireshark, nous visualisons bien l'activité qui se déroule vers l'interface PFsense



Normalement, nous aurions dû attaquer 192.168.10.1, port 80 pour ne plus accéder au service web. Nous avons fait des tests, et même si Wireshark détectait correctement les paquets, et Snort l'attaque, le serveur web n'était pas "down". Cela peut s'expliquer par le fait que la machine pirate n'envoie pas suffisamment de paquets et que la machine attaquée est trop puissante. Dans ce cas présent, il aurait fallu effectuer une attaque Ddos, avec plusieurs machines en même temps pour faire tomber le serveur web.

Nous vous mettons tout de même la trame Wireshark et ce qu'affiche Snort lors de l'attaque.

- Commande de l'attaque

```
┌──(kali㉿kali)-[~]
└─$ sudo hping3 -c 300000 -d 120 -S -w 64 -p 80 --rand-source --flood 192.168.10.1
[sudo] password for kali:
HPING 192.168.10.1 (eth0 192.168.10.1): S set, 40 headers + 120 data bytes
hping in flood mode, no replies will be shown
^C
--- 192.168.10.1 hping statistic ---
142054789 packets transmitted, 0 packets received, 100% packet loss
round-trip min/avg/max = 0.0/0.0/0.0 ms
```

- Wireshark

No.	Time	Source	Destination	Protocol	Length	Info
3708…	2.052360070	211.4.217.246	192.168.10.1	TCP	174	40663 → 80 [SYN] Seq=0 Win=64 Len=120 [TCP segment of a reassembled …
3708…	2.052362170	140.125.140.159	192.168.10.1	TCP	174	40664 → 80 [SYN] Seq=0 Win=64 Len=120 [TCP segment of a reassembled …
3708…	2.052364255	16.43.49.141	192.168.10.1	TCP	174	40665 → 80 [SYN] Seq=0 Win=64 Len=120 [TCP segment of a reassembled …
3708…	2.052366373	70.32.205.149	192.168.10.1	TCP	174	40666 → 80 [SYN] Seq=0 Win=64 Len=120 [TCP segment of a reassembled …
3708…	2.052368474	47.70.134.144	192.168.10.1	TCP	174	40667 → 80 [SYN] Seq=0 Win=64 Len=120 [TCP segment of a reassembled …
3708…	2.052370613	106.141.68.4	192.168.10.1	TCP	174	40668 → 80 [SYN] Seq=0 Win=64 Len=120 [TCP segment of a reassembled …
3708…	2.052373816	200.159.159.32	192.168.10.1	TCP	174	40669 → 80 [SYN] Seq=0 Win=64 Len=120 [TCP segment of a reassembled …
3708…	2.052375931	15.21.49.143	192.168.10.1	TCP	174	40670 → 80 [SYN] Seq=0 Win=64 Len=120 [TCP segment of a reassembled …
3708…	2.052378019	66.189.205.63	192.168.10.1	TCP	174	40671 → 80 [SYN] Seq=0 Win=64 Len=120 [TCP segment of a reassembled …
3708…	2.052380373	150.216.66.15	192.168.10.1	TCP	174	40672 → 80 [SYN] Seq=0 Win=64 Len=120 [TCP segment of a reassembled …
3708…	2.052382722	222.71.132.92	192.168.10.1	TCP	174	40673 → 80 [SYN] Seq=0 Win=64 Len=120 [TCP segment of a reassembled …
3708…	2.052384907	217.165.45.1	192.168.10.1	TCP	174	40674 → 80 [SYN] Seq=0 Win=64 Len=120 [TCP segment of a reassembled …
3708…	2.052387001	160.140.217.22	192.168.10.1	TCP	174	40675 → 80 [SYN] Seq=0 Win=64 Len=120 [TCP segment of a reassembled …

– Snort

14 Entries in Active Log

Date	Action	Pri	Proto	Class	Source IP	SPort	Destination IP	DPort	GID:SID	Description
2024-01-28 10:37:24	⚠	2	TCP	Detection of a Non-Standard Protocol or Event	38.178.130.228	22	192.168.10.1	80	128:4	(spp_ssh) Protocol mismatch
2024-01-28 10:37:14	⚠	2	TCP	Detection of a Non-Standard Protocol or Event	47.31.131.248	22	192.168.10.1	80	128:4	(spp_ssh) Protocol mismatch
2024-01-28 10:37:12	⚠	3	TCP	Unknown Traffic	192.168.10.1	35626	192.168.10.254	80	119:34	(http_inspect) TOO MANY PIPELINED REQUESTS
2024-01-28 10:36:45	⚠	2	TCP	Detection of a Non-Standard Protocol or	6.139.224.8	22	192.168.10.1	80	128:4	(spp_ssh) Protocol mismatch