



Application sécurité avancée

2021/2022

Projet: Analyse d'un réseau avec Snaffler sur windows

Proposé par: Nicolas vieux

Réalisée par: Célia Boulahouat & Rosa Hadroug

1-Introduction

Un renifleur réseau aussi appelé analyseur réseau ou analyseur de paquets «sniffer» est un logiciel ou appareil capable d'intercepter, de surveiller et d'enregistrer le trafic d'un réseau en temps réel. Le renifleur capture chacun des paquets qui transitent par le réseau et analyse son contenu dans le but :

- d'identifier et de localiser précisément les transmissions inhabituelles sur le réseau.
- de recueillir des données à des fins d'analyse de sécurité.
- de détecter les périodes où la bande passante est la plus et la moins utilisée.
- d'analyser les performances de votre réseau.

Selon la structure du réseau, on distingue deux principaux types de reniflage :

☐ Reniflage passif sur les réseaux avec «hub»:

Comme tous les appareils du concentrateur reçoivent tout le trafic du réseau, aucun mécanisme de régulation n'existe pour diriger le trafic vers son destinataire, le renifleur peut alors facilement et passivement absorber tout ce qui est envoyé. Ce type de reniflage est très difficile à détecter.

☐ Reniflage actif sur les réseaux avec «commutateur»:

Sur un concentrateur réseau, le renifleur passif ne peut voir que les données entrantes et sortantes de l'appareil hôte.

Pour accéder à tout le trafic qui passe sur le réseau, le renifleur actif doit tracer un chemin ou détourner la façon dont les commutateurs dirigent tout en incluant l'injection de trafic supplémentaire dans le réseau mais il est plus facile à détecter, car il indique lui-même sa présence.

Il existe plusieurs outils permettant de renifler un réseau on cite : acrylicWifi, tshark (Wireshark), TCPdump, Etherape, Ettercap....

2- Présentation de Snaffler

Snaffler est un outil destiné aux pentesters pour les aider à trouver principalement des creds, mais c'est flexible dans un environnement Windows/ environnement AD). Il peut également être utile à d'autres personnes faisant d'autres choses, mais il n'est explicitement PAS destiné à être un outil d'audit.

Il obtient une liste d'ordinateurs Windows à partir d'Active Directory, puis déploie ses appendices hargneux sur chacun d'entre eux pour déterminer lesquels ont des partages de fichiers, et si vous pouvez les lire.

Ensuite, d'autres appendices énumèrent tous les fichiers de ces partages et utilisent LEARNED ARTIFACTUAL INTELLIGENCE for MACHINES pour déterminer lesquels un petit pirate comme nous pourrait vouloir.

Installation de snaffler

```
C:\WINDOWS\system32>snaffler.exe -s -o snaffler.log
.:.....:..  .:..  .:..  .-:.....:..  .:.....:..  .:.....:..
:..:..:..:..  :..:..:..:..  :..:..:..:..  :..:..:..:..  :..:..:..:..
'==/[[[[[, [[[[[. '[[, [[ '[[, [[[[,== [[[[,== [[[[  [[cccc  [[[,/[[[
'[[  $ $$$ 'Y$c$$$cc$c$$$c'$$$'  '$$$'  '$$'  '$$'  '$$$$$c
88b  dP 888  Y88 888 888,888 888 888 888 888 888oo, __888oo, __ 888b '88bo,
'YmMY'  MMM  YM YMM  'MM,  'MM,  'MM,  'YUMMM' 'YUMMMMMMM 'W'
by l0ss and Sh3r4 - github.com/SnaffCon/Snaffler

[DESKTOP-GI155ST\T480S@DESKTOP-GI155ST] 2022-01-17 09:18:48Z [Info] Parsing args...
[DESKTOP-GI155ST\T480S@DESKTOP-GI155ST] 2022-01-17 09:18:48Z [Info] Parsed args successfully.
[DESKTOP-GI155ST\T480S@DESKTOP-GI155ST] 2022-01-17 09:18:48Z [Info] Getting computers and DFS targets from AD.
[DESKTOP-GI155ST\T480S@DESKTOP-GI155ST] 2022-01-17 09:19:48Z [Info] Status Update:
ShareFinder Tasks Completed: 0
ShareFinder Tasks Remaining: 0
ShareFinder Tasks Running: 0
TreeWalker Tasks Completed: 0
TreeWalker Tasks Remaining: 0
TreeWalker Tasks Running: 0
FileScanner Tasks Completed: 0
FileScanner Tasks Remaining: 0
FileScanner Tasks Running: 0
76,7MB RAM in use.

ShareScanner queue finished, rebalancing workload.
Insufficient FileScanner queue size, rebalancing workload.
Max ShareFinder Threads: 0
Max TreeWalker Threads: 21
Max FileScanner Threads: 39
```

PS: On a pas pu continuer de travailler sur snaffler en local

PrivescChek

On essaye à présent d'identifier les problèmes courants de configuration de Windows qui peuvent être exploités pour une escalade des privilèges locaux. Il recueille également diverses informations qui pourraient être utiles pour l'exploitation et/ou la post-exploitation.

PrivescChek machine celia

~~~ PrivescCheck Report ~~~		
OK	None	CONFIG > AlwaysInstallElevated
KO	High	CONFIG > PATH Folder Permissions -> 2 result(s)
NA	None	CONFIG > SCCM Cache Folder (info)
OK	None	CONFIG > SCCM Cache Folder
OK	None	CONFIG > WSUS Configuration
OK	None	CONFIG > PrintNightmare exploit
NA	Info	CONFIG > Driver Co-Installers -> 1 result(s)
NA	Info	CREDS > Vault Creds -> 2 result(s)
OK	None	CREDS > Unattend Files
OK	None	CREDS > WinLogon
OK	None	CREDS > GPP Passwords
NA	None	CREDS > Vault List
OK	None	CREDS > SAM/SYSTEM/SECURITY Files
OK	None	CREDS > SAM/SYSTEM/SECURITY in shadow copies
NA	Info	HARDENING > Credential Guard -> 1 result(s)
NA	Info	HARDENING > BitLocker -> 1 result(s)

On remarque la présence de deux folders contenant des fichiers de haute confidentialité (mentionné en rouge) dans le rapport de PrivescCheck. (2 menaces critiques)

En analysant les métadonnées de l'exécutable cible on retrouve 42 services enregistrés et on filtre ceux intégrés à Windows

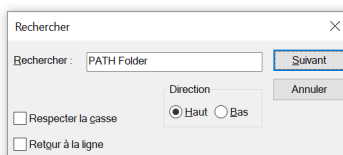
TEST	CONFIG > PATH Folder Permissions	VULN
DESC	Retrieve the list of SYSTEM %PATH% folders and check whether the current user has some write permissions in any of them.	
[*] Found 2 result(s).		
Path	: C:\MinGW\bin	
ModifiablePath	: C:\	
IdentityReference	: AUTORITE NT\Utilisateurs authentifi,s	
Permissions	: AppendData/AddSubdirectory	
Path	: D:\MATLAB\bin	
ModifiablePath	: D:\MATLAB\bin	
IdentityReference	: AUTORITE NT\Utilisateurs authentifi,s	
Permissions	: {Delete, WriteAttributes, Synchronize, ReadControl...}	

Rechercher

Rechercher : PATH Fol

☐ Respecter la casse

☐ Retour à la ligne



On remarque la présence de cette vulnérabilité dans 2 folders récupérant la liste des dossiers SYSTEM %PATH% et vérifie si l'utilisateur actuel a des droits d'écriture dans l'un d'entre eux.

aussi 2 DLL pas utilisés qui peuvent être hijackés

```
[*] Found 2 result(s).
```

```
Name       : cdpsgshims.dll
Description : Loaded by CDPSvc upon service startup
RunAs      : NT AUTHORITY\LocalService
RebootRequired : True

Name       : WptsExtensions.dll
Description : Loaded by the Task Scheduler upon service startup
RunAs      : LocalSystem
RebootRequired : True
```

## PrivescChk machine rosa

Sur cette machine on détecte deux (high) menaces

```
+-----+
|             ~~~ PrivescCheck Report ~~~             |
+-----+
| OK | None | CONFIG > AlwaysInstallElevated |
| KO | High | CONFIG > PATH Folder Permissions -> 1 result(s) |
| NA | None | CONFIG > SCCM Cache Folder (info) |
| OK | None | CONFIG > SCCM Cache Folder |
| OK | None | CONFIG > WSUS Configuration |
| OK | None | CONFIG > PrintNightmare exploit |
| NA | Info | CONFIG > Driver Co-Installers -> 1 result(s) |
| NA | Info | CREDS > Vault Creds -> 1 result(s) |
| OK | None | CREDS > Unattend Files |
| OK | None | CREDS > WinLogon |
| OK | None | CREDS > GPP Passwords |
| NA | None | CREDS > Vault List |
| OK | None | CREDS > SAM/SYSTEM/SECURITY Files |
| OK | None | CREDS > SAM/SYSTEM/SECURITY in shadow copies |
| NA | Info | HARDENING > Credential Guard -> 1 result(s) |
| NA | Info | HARDENING > BitLocker -> 1 result(s) |
| NA | Info | MISC > Hijackable DLLs -> 2 result(s) |
| NA | Info | SERVICES > Non-default Services -> 27 result(s) |
| OK | None | SERVICES > Service Permissions |
| OK | None | SERVICES > Registry Permissions |
| KO | High | SERVICES > Binary Permissions -> 6 result(s) |
| OK | None | SERVICES > Unquoted Path |
| OK | None | SERVICES > SCM Permissions |
| OK | None | UPDATES > System up to date? |
| NA | Info | USER > Groups -> 14 result(s) |
| NA | Info | USER > Identity -> 1 result(s) |
| NA | None | USER > Environment Variables |
| NA | Info | USER > Privileges -> 5 result(s) |
+-----+
```

## la première

Récupérer la liste des dossiers SYSTEM %PATH% et vérifier |  
 | | si l'utilisateur actuel a des droits d'écriture | | dans l'un d'entre eux.

```
+-----+-----+-----+
| TEST | CONFIG > PATH Folder Permissions | VULN |
+-----+-----+-----+
| DESC | Retrieve the list of SYSTEM %PATH% folders and check |
| | whether the current user has some write permissions |
| | in any of them. |
+-----+-----+-----+
[*] Found 1 result(s).

Path : C:\Users\user\AppData\Local\Programs\Python\Python310
ModifiablePath : C:\Users\user\AppData\Local\Programs\Python\Python310
IdentityReference : DESKTOP-BSSG8A7\user
Permissions : {WriteOwner, Delete, WriteAttributes, Synchroniz...
```

## la deuxième

Lister tous les services et vérifier si l'utilisateur actuel |  
 | | peut modifier l'exécutable cible ou écrire des fichiers dans |  
 | | son dossier parent.

```
+-----+-----+-----+
| TEST | SERVICES > Binary Permissions | VULN |
+-----+-----+-----+
| DESC | List all services and check whether the current user |
| | can modify the target executable or write files in |
| | its parent folder. |
+-----+-----+-----+
[*] Found 6 result(s).

Name : wampapache
ImagePath : "c:\wamp\bin\apache\apache2.4.41\bin\httpd.exe" -k run:
User : LocalSystem
ModifiablePath : C:\wamp\bin\apache\apache2.4.41\bin
IdentityReference : AUTORITE NT\Utilisateurs authenti.s
Permissions : Delete, WriteAttributes, Synchronize, ReadControl, ReadDa
ReadExtendedAttributes, Execute/Traverse
Status : Stopped
UserCanStart : False
UserCanStop : False

Name : wampapache
ImagePath : "c:\wamp\bin\apache\apache2.4.41\bin\httpd.exe" -k run:
User : LocalSystem
ModifiablePath : C:\wamp\bin\apache\apache2.4.41\bin\httpd.exe
IdentityReference : AUTORITE NT\Utilisateurs authenti.s
Permissions : Delete, WriteAttributes, Synchronize, ReadControl, ReadDa
ReadExtendedAttributes, Execute/Traverse
Status : Stopped
UserCanStart : False
UserCanStop : False

Name : wampapache64
```



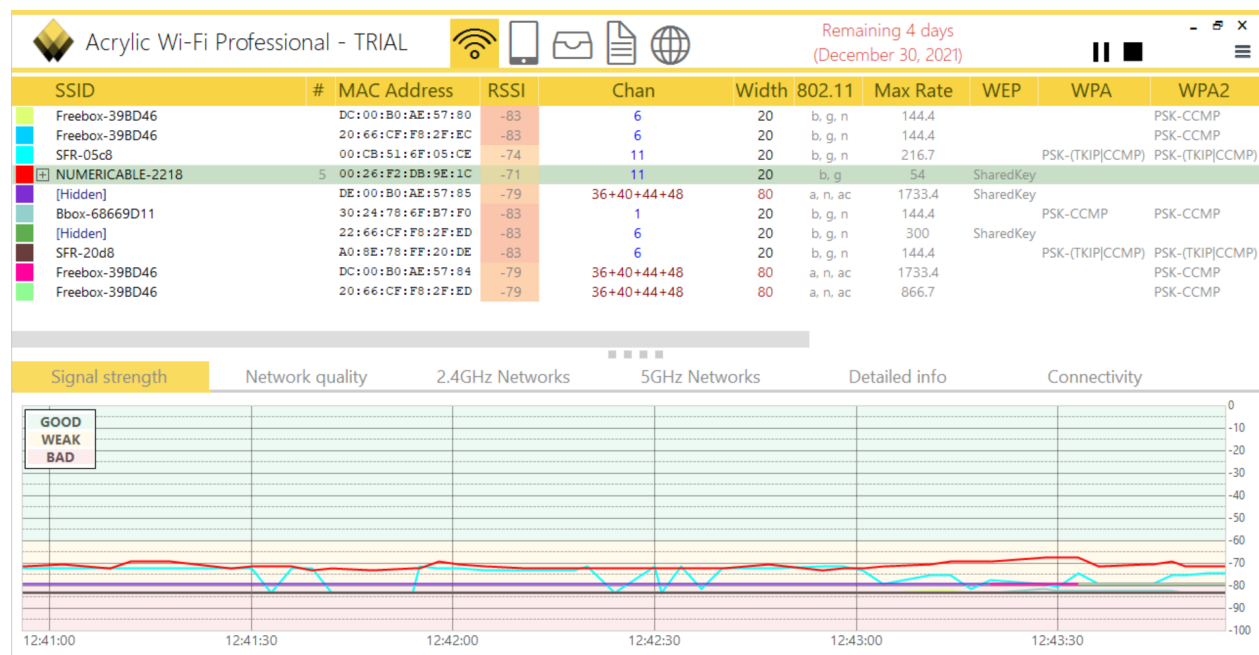
### **3- Acrylic Wi-Fi**

C'est un outil d'analyse et de visualisation du réseau wifi qui permet d'identifier, de diagnostiquer et de résoudre les problèmes wifi et aussi détecter les clients connectés, identifier le meilleur canal et améliorer la couverture

Cet outil permet de trouver le meilleur canal pour le réseau WiFi et toute mauvaise configuration AP ce qui aide à découvrir les points d'accès et les périphériques non autorisés.

Dans ce qui suit nous allons présenter certaines fonctions de base qui peuvent s'effectuer avec **Acrylic WiFi** pour voir les réseaux Wi-Fi de proximité, en activant le mode surveillance et en recueillant des informations sur:

- ☐ le niveau du signal Wi-Fi (RSSI).
- ☐ le canal Wi-Fi ou la fréquence d'émission Wi-Fi sur le réseau, de 2.4Ghz ou 5Ghz.
- ☐ l'onglet d'informations détaillées, disponible dans la version.



En bas, on a un graphique en temps réel de tous les signaux Wi-Fi que nous recevons et comment ils fluctuent dans le temps.

Ce graphique va nous permettre de voir s'il y a une sorte de coupure Wi-Fi, ou une baisse du signal sans fil, synonyme d'une sorte de problème avec le point d'accès, ou qu'il est simplement si loin que d'autres Wi-Fi les réseaux interfèrent.



Acrylic Wi-Fi Professional - TRIAL Remaining 4 days (December 30, 2021)

SSID	#	MAC Address	RSSI	Chan	Width	802.11	Max Rate	WE
Freebox-39BD46		DC:00:B0:AE:57:80	-83	6	20	b, g, n	144.4	
Freebox-39BD46		20:66:CF:F8:2F						
SFR-05c8		00:CB:51:6F:05						
NUMERICABLE-2218	5	00:26:F2:DB:9E:1C						
[Hidden]		DE:00:B0:AE:57						
Bbox-68669D11		30:24:78:6F:B7						
[Hidden]		22:66:CF:F8:2F						

Add STATION to inventory

MAC Address: 00:26:F2:DB:9E:1C

Name: AP NUMERICABLE-2218

OK Cancel

Element Name	Element Value
Ssid	NUMERICABLE-2218 - (Length: 16)
Supported Rates	1 Mb/s Mandatory 2 Mb/s Mandatory 5.5 Mb/s Mandatory 11 Mb/s Mandatory 18 Mb/s Optional 24 Mb/s Optional 36 Mb/s Optional
DSSS Parameter Set	Current Channel: 11
Extended Rates	6 Mb/s Optional 9 Mb/s Optional 12 Mb/s Optional 48 Mb/s Optional
Published SSIDs	NUMERICABLE-2218
Primary Channel	11 (2462MHz)
Channel Width	20MHz
Secondary Channel	11 (2462MHz)
All used Channels	11
802.11 Band	2.4 GHz

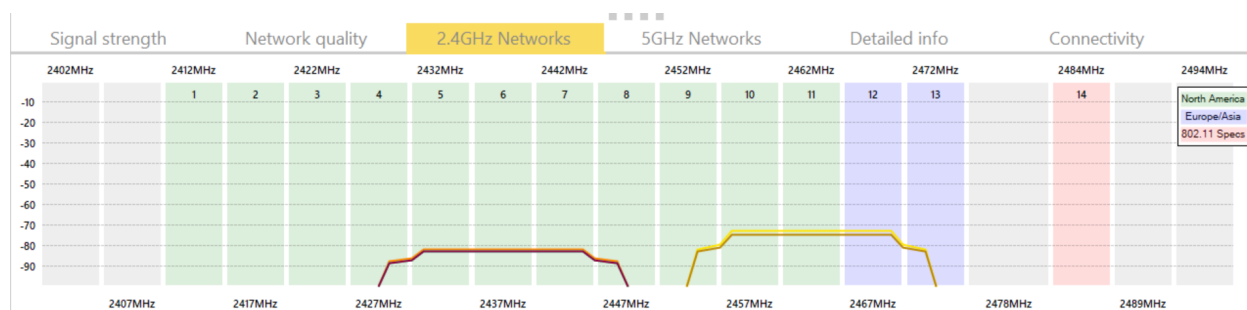
Pour la résolution d'incidents Wi-Fi, améliorer la performance Wi-Fi et résoudre des problèmes, **Acrylic Wi-Fi** comprend plusieurs fonctions:

- **Largeur de canal Wi-Fi** : informations sur des canaux de 20Mhz, 40Mhz et 80Mhz sur des réseaux 802.11n et 802.11ac
- **Couche Phy** : Version standard Wi-Fi prise en charge, connue également comme Phy type (802.11a/b/g/n/ac).
- **Vitesse Wi-Fi prise en charge** : informations sur les vitesses maximales (*Max rates*) prises en charge par le point d'accès Wi-Fi.
- **Renvoi de paquets**: frames Wi-Fi renvoyées à cause d'une erreur de communication, par exemple en raison d'interférences.

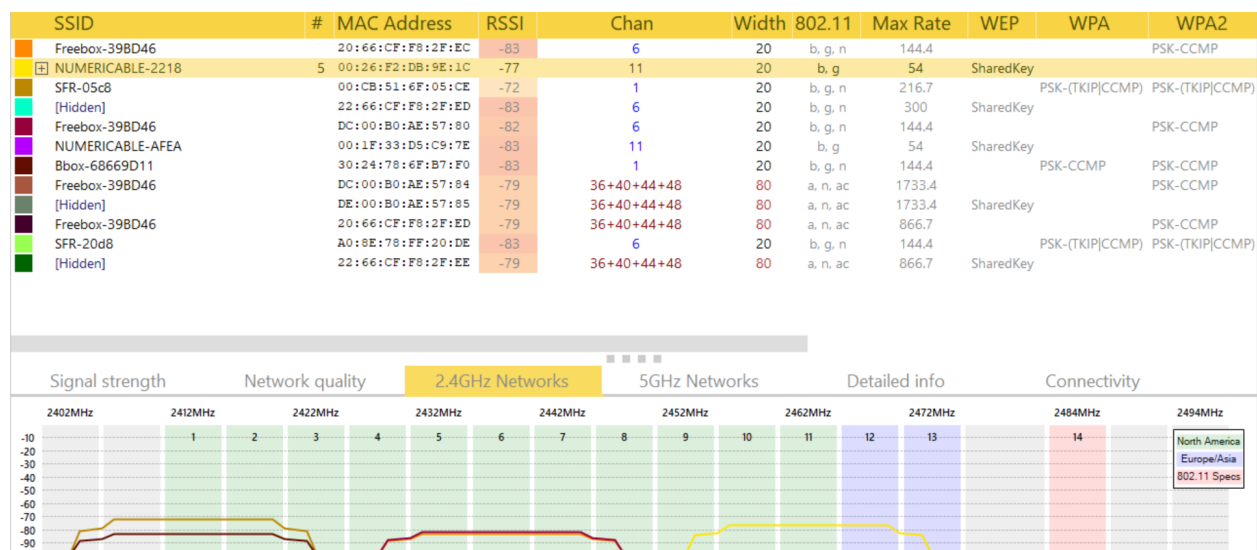
NUMERICABLE-2218	5	00:26:F2:DB:9E:1C	-70	11	20	b, g	54	SharedKey
[Client-Lan] - 192.168.0.1		00:26:F2:CC:22:1C						
[Client-Lan] - 192.168.0.11		D2:BD:A9:02:52:23						
[Client-Lan] - 192.168.0.13		20:64:CB:13:A0:73						
[Client-Lan] - 192.168.0.14		B4:A5:AC:A7:A7:5B						
[Client-Lan] - 192.168.0.16		5C:E0:C5:26:AB:BA						

Dans cette figure on a la liste de tous les terminaux(clients) connectés à votre SSID domicile (NUMERICABLE-2218)

Dans ce graphique on retrouve également en détail les MHz des différents canaux, et même quels canaux sont spécifiquement orientés pour l'Europe.



Dans l'onglet Réseaux 2,4 GHz, nous aurons un graphique de tous les réseaux sans fil, avec le canal utilisé, ainsi que la largeur de canal que nous avons dans ces réseaux de cette bande de fréquence



**Remarque:** Puisque cet outil est payant nous n'avons pas pu voir plus de fonctionnalité par exemple les mots de passe par défaut utilisés par les routeurs s'il ne nous le montre pas, nous pouvons toujours essayer d'entrer une liste de clés pour les tester automatiquement.

## 4-Utilisation de Wireshark sur la ligne de commande Linux avec TShark

TShark est un outil basé sur la ligne de commande, qui peut faire tout ce que fait wireshark. Sa structure de fonctionnement dispose de décodeurs et de filtres puissants.

Tshark est capable de capturer les informations des paquets de données de différentes couches de réseau et de les afficher dans différents formats.

## Installation de tshark

Si vous êtes sur Kali linux, vous trouverez le tshark inclus parmi les outils de la distribution.

Sinon on peut l'installer grâce à la commande `apt-get install tshark`. Après l'installation pour vérifier que tout s'est bien passé, on utilisera la commande `tshark -h` pour voir les différentes options disponibles avec tshark.

```
(kali@kali)-[~]
$ tshark -h
TShark (Wireshark) 3.4.7 (Git v3.4.7 packaged as 3.4.7-1)
Dump and analyze network traffic.
See https://www.wireshark.org for more information.

Usage: tshark [options] ...

Capture interface:
  -i <interface>, --interface <interface>      name or idx of interface (def: first non-loopback)
  -f <capture filter>                          packet filter in libpcap filter syntax
  -s <snaplen>, --snapshot-length <snaplen>     packet snapshot length (def: appropriate maximum)
  -p, --no-promiscuous-mode                    don't capture in promiscuous mode
  -I, --monitor-mode                           capture in monitor mode, if available
  -B <buffer size>, --buffer-size <buffer size> size of kernel buffer (def: 2MB)
  -y <link type>, --linktype <link type>       link layer type (def: first appropriate)
  --time-stamp-type <type>                     timestamp method for interface
  -D, --list-interfaces                        print list of interfaces and exit
  -L, --list-data-link-types                  print list of link-layer types of iface and exit
  --list-time-stamp-types                     print list of timestamp types for iface and exit

Capture stop conditions:
  -c <packet count>                          stop after n packets (def: infinite)
  -a <autostop cond.> ..., --autostop <autostop cond.> ...
      duration:NUM - stop after NUM seconds
      filesize:NUM - stop this file after NUM KB
      files:NUM - stop after NUM files
      packets:NUM - stop after NUM packets
```

Pour commencer on peut lister les interfaces dont tshark peut capturer le trafic avec la commande: `tshark -D`

```
(root@kali)-[/home/kali]
# tshark -D
Running as user "root" and group "root". This could be dangerous.
1. eth0
2. eth1
3. any
4. lo (Loopback)
5. bluetooth-monitor
6. nflog
7. nfqueue
8. dbus-system
9. dbus-session
10. ciscodump (Cisco remote capture)
11. dpauwmon (DisplayPort AUX channel capture)
12. randpkt (Random packet generator)
13. sdjournal (systemd Journal Export)
14. sshdump (SSH remote capture)
15. udpdump (UDP Listener remote capture)
```

avant de lancer la capture nous d'abord générer du trafic sur l'interface avec un `ping` vers kali-linux.fr.

```
(kali@kali)-[~]
$ ping kali-linux.fr
PING kali-linux.fr (64.91.241.196) 56(84) bytes of data.
64 bytes from host.logoinstant.com (64.91.241.196): icmp_seq=1 ttl=47 time=148 ms
64 bytes from host.logoinstant.com (64.91.241.196): icmp_seq=2 ttl=47 time=136 ms
64 bytes from host.logoinstant.com (64.91.241.196): icmp_seq=3 ttl=47 time=143 ms
64 bytes from host.logoinstant.com (64.91.241.196): icmp_seq=4 ttl=47 time=155 ms
64 bytes from host.logoinstant.com (64.91.241.196): icmp_seq=5 ttl=47 time=178 ms
64 bytes from host.logoinstant.com (64.91.241.196): icmp_seq=6 ttl=47 time=209 ms
64 bytes from host.logoinstant.com (64.91.241.196): icmp_seq=7 ttl=47 time=227 ms
64 bytes from host.logoinstant.com (64.91.241.196): icmp_seq=8 ttl=47 time=126 ms
64 bytes from host.logoinstant.com (64.91.241.196): icmp_seq=9 ttl=47 time=128 ms
64 bytes from host.logoinstant.com (64.91.241.196): icmp_seq=10 ttl=47 time=152 ms
64 bytes from host.logoinstant.com (64.91.241.196): icmp_seq=11 ttl=47 time=150 ms
64 bytes from host.logoinstant.com (64.91.241.196): icmp_seq=12 ttl=47 time=149 ms
64 bytes from host.logoinstant.com (64.91.241.196): icmp_seq=13 ttl=47 time=223 ms
64 bytes from host.logoinstant.com (64.91.241.196): icmp_seq=14 ttl=47 time=134 ms
64 bytes from host.logoinstant.com (64.91.241.196): icmp_seq=15 ttl=47 time=146 ms
64 bytes from host.logoinstant.com (64.91.241.196): icmp_seq=16 ttl=47 time=164 ms
```

Nous avons une longue liste d'interface. Mais pour notre démonstration nous allons utiliser l'interface «eth1».

Pour lancer la capture du trafic sur cet interface nous allons utiliser la commande `tshark -i eth1`

Comme nous pouvons le voir le trafic est en train d'être capturé par tshark.

```
(root@kali)-[/home/kali]
# tshark -i eth1
Running as user "root" and group "root". This could be dangerous.
Capturing on 'eth1'
1 0.000000000 10.0.3.15 → 142.250.201.163 TLSv1.2 93 Application Data
2 0.000743789 142.250.201.163 → 10.0.3.15 TCP 60 443 → 33032 [ACK] Seq=1 Ack=40 Win=65535 Len=0
3 0.020162755 142.250.201.163 → 10.0.3.15 TLSv1.2 93 Application Data
4 0.020206061 10.0.3.15 → 142.250.201.163 TCP 54 33032 → 443 [ACK] Seq=40 Ack=40 Win=62780 Len=0
5 0.174902054 10.0.3.15 → 64.91.241.196 ICMP 98 Echo (ping) request id=0xc653, seq=158/40448, ttl=64
6 0.437015157 64.91.241.196 → 10.0.3.15 ICMP 98 Echo (ping) reply id=0xc653, seq=158/40448, ttl=47 (request in 5)
7 0.437505414 10.0.3.15 → 89.2.0.1 DNS 86 Standard query 0x8e70 PTR 196.241.91.64.in-addr.arpa
8 0.451668933 89.2.0.1 → 10.0.3.15 DNS 120 Standard query response 0x8e70 PTR 196.241.91.64.in-addr.arpa PTR host.logoinstant.com
9 1.001455934 10.0.3.15 → 172.217.19.234 TLSv1.2 93 Application Data
10 1.001663006 10.0.3.15 → 142.250.74.237 TLSv1.2 93 Application Data
11 1.003232417 172.217.19.234 → 10.0.3.15 TCP 60 443 → 40460 [ACK] Seq=1 Ack=40 Win=65535 Len=0
12 1.003232023 142.250.74.237 → 10.0.3.15 TCP 60 443 → 43846 [ACK] Seq=1 Ack=40 Win=65535 Len=0
13 1.111995131 172.217.19.234 → 10.0.3.15 TLSv1.2 93 Application Data
14 1.118591819 142.250.74.237 → 10.0.3.15 TLSv1.2 93 Application Data
15 1.118645294 10.0.3.15 → 142.250.74.237 TCP 54 43846 → 443 [ACK] Seq=40 Ack=40 Win=65535 Len=0
16 1.166333393 10.0.3.15 → 172.217.19.234 TCP 54 40460 → 443 [ACK] Seq=40 Ack=40 Win=65535 Len=0
17 1.175955814 10.0.3.15 → 64.91.241.196 ICMP 98 Echo (ping) request id=0xc653, seq=159/40704, ttl=64
18 1.406089316 64.91.241.196 → 10.0.3.15 ICMP 98 Echo (ping) reply id=0xc653, seq=159/40704, ttl=47 (request in 17)
19 1.406529768 10.0.3.15 → 89.2.0.1 DNS 86 Standard query 0x0901 PTR 196.241.91.64.in-addr.arpa
20 1.420193765 89.2.0.1 → 10.0.3.15 DNS 120 Standard query response 0x0901 PTR 196.241.91.64.in-addr.arpa PTR host.logoinstant.com
21 2.177465133 10.0.3.15 → 64.91.241.196 ICMP 98 Echo (ping) request id=0xc653, seq=160/40960, ttl=64
22 2.383659531 64.91.241.196 → 10.0.3.15 ICMP 98 Echo (ping) reply id=0xc653, seq=160/40960, ttl=47 (request in 21)
23 2.385017004 10.0.3.15 → 89.2.0.1 DNS 86 Standard query 0xebef PTR 196.241.91.64.in-addr.arpa
24 2.403632133 89.2.0.1 → 10.0.3.15 DNS 120 Standard query response 0xebef PTR 196.241.91.64.in-addr.arpa PTR host.logoinstant.com
25 2.428768606 10.0.3.15 → 142.250.74.227 TCP 54 50906 → 80 [ACK] Seq=1 Ack=1 Win=63882 Len=0
26 2.428837354 10.0.3.15 → 142.250.74.227 TCP 54 50940 → 80 [ACK] Seq=1 Ack=1 Win=63882 Len=0
27 2.429156531 10.0.3.15 → 93.184.220.29 TCP 54 39504 → 80 [ACK] Seq=1 Ack=1 Win=63920 Len=0
28 2.430550506 142.250.74.227 → 10.0.3.15 TCP 60 [TCP ACKed unseen segment] 80 → 50906 [ACK] Seq=1 Ack=2 Win=65535 Len=0
29 2.430550817 142.250.74.227 → 10.0.3.15 TCP 60 [TCP ACKed unseen segment] 80 → 50940 [ACK] Seq=1 Ack=2 Win=65535 Len=0
```

Nous pouvons aussi limiter le nombre de paquets que nous allons intercepter avec l'option **-c**. La commande donne

```
(root@kali)~/home/kali
# tshark -i eth1 -c 10
Running as user "root" and group "root". This could be dangerous.
Capturing on 'eth1'
 1 0.000000000 10.0.3.15 → 89.2.0.1      DNS 73 Standard query 0xae9d A kali-linux.fr
 2 0.000018260 10.0.3.15 → 89.2.0.1      DNS 73 Standard query 0x809b AAAA kali-linux.fr
 3 0.031036779 89.2.0.1 → 10.0.3.15      DNS 89 Standard query response 0xae9d A kali-linux.fr A 64.91.241.196
 4 0.031036895 89.2.0.1 → 10.0.3.15      DNS 142 Standard query response 0x809b AAAA kali-linux.fr SOA ns1.logoinstant.com
 5 0.031674196 10.0.3.15 → 64.91.241.196 ICMP 98 Echo (ping) request id=0x7206, seq=1/256, ttl=64
 6 0.183430426 64.91.241.196 → 10.0.3.15 ICMP 98 Echo (ping) reply id=0x7206, seq=1/256, ttl=47 (request in 5)
 7 0.184217230 10.0.3.15 → 89.2.0.1      DNS 86 Standard query 0xcb0b PTR 196.241.91.64.in-addr.arpa
 8 0.212030128 89.2.0.1 → 10.0.3.15      DNS 120 Standard query response 0xcb0b PTR 196.241.91.64.in-addr.arpa PTR host.log
 9 1.032845976 10.0.3.15 → 64.91.241.196 ICMP 98 Echo (ping) request id=0x7206, seq=2/512, ttl=64
10 1.207837393 64.91.241.196 → 10.0.3.15 ICMP 98 Echo (ping) reply id=0x7206, seq=2/512, ttl=47 (request in 9)
10 packets captured
```

Nous pouvons aussi avoir plus de détails sur les différents paquets avec l'option **-V**.

```
(root@kali)~/home/kali
# tshark -i eth1 -c 10 -V
Running as user "root" and group "root". This could be dangerous.
Capturing on 'eth1'
Frame 1: 98 bytes on wire (784 bits), 98 bytes captured (784 bits) on interface eth1, id 0
  Interface id: 0 (eth1)
    Interface name: eth1
    Encapsulation type: Ethernet (1)
    Arrival Time: Dec 26, 2021 07:33:31.735439012 EST
    [Time shift for this packet: 0.000000000 seconds]
    Epoch Time: 1640522011.735439012 seconds
    [Time delta from previous captured frame: 0.000000000 seconds]
    [Time delta from previous displayed frame: 0.000000000 seconds]
    [Time since reference or first frame: 0.000000000 seconds]
    Frame Number: 1
    Frame Length: 98 bytes (784 bits)
    Capture Length: 98 bytes (784 bits)
    [Frame is marked: False]
    [Frame is ignored: False]
    [Protocols in frame: eth:ethertype:ip:icmp:data]
    Ethernet II, Src: RealtekU_12:35:02 (52:54:00:12:35:02), Dst: PcsCompu_76:b2:12 (08:00:27:76:b2:12)
      Destination: PcsCompu_76:b2:12 (08:00:27:76:b2:12)
        Address: PcsCompu_76:b2:12 (08:00:27:76:b2:12)
          .... 0. .... = LG bit: Globally unique address (factory default)
          .... 0. .... = IG bit: Individual address (unicast)
        Source: RealtekU_12:35:02 (52:54:00:12:35:02)
          Address: RealtekU_12:35:02 (52:54:00:12:35:02)
            .... 1. .... = LG bit: Locally administered address (this is NOT the factory def
            .... 0. .... = IG bit: Individual address (unicast)
        Type: IPv4 (0x0800)
      Internet Protocol Version 4, Src: 64.91.241.196, Dst: 10.0.3.15
        0100 .... = Version: 4
        .... 0101 = Header Length: 20 bytes (5)
        Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
          0000 00.. = Differentiated Services Codepoint: Default (0)
          .... 0000 = Explicit Congestion Notification: Not ECN-Capable Transport (0)
        Total Length: 84
        Identification: 0x273a (10042)
        Flags: 0x00
          0 ... .... = Reserved bit: Not set
          .0.. .... = Don't fragment: Not set
          ..0. .... = More fragments: Not set
        Fragment Offset: 0
```

On peut capturer uniquement les paquets de l'IP source ou de destination spécifique

avant ça on lance un ping vers l'adresse ip souhaité pour notre cas c'est l'adresse ip du domain opensource.com

```
(kali㉿kali)-[~]
$ nslookup opensource.com
Server:      89.2.0.1
Address:     89.2.0.1#53

Non-authoritative answer:
Name:   opensource.com
Address: 54.204.39.132

(kali㉿kali)-[~]
$ ping 54.204.39.132

PING 54.204.39.132 (54.204.39.132) 56(84) bytes of data.
64 bytes from 54.204.39.132: icmp_seq=1 ttl=44 time=221 ms
64 bytes from 54.204.39.132: icmp_seq=2 ttl=44 time=126 ms
64 bytes from 54.204.39.132: icmp_seq=3 ttl=44 time=349 ms
64 bytes from 54.204.39.132: icmp_seq=4 ttl=44 time=271 ms
64 bytes from 54.204.39.132: icmp_seq=5 ttl=44 time=245 ms
64 bytes from 54.204.39.132: icmp_seq=6 ttl=44 time=190 ms
64 bytes from 54.204.39.132: icmp_seq=7 ttl=44 time=418 ms
64 bytes from 54.204.39.132: icmp_seq=8 ttl=44 time=646 ms
64 bytes from 54.204.39.132: icmp_seq=9 ttl=44 time=126 ms
64 bytes from 54.204.39.132: icmp_seq=10 ttl=44 time=383 ms
^C
--- 54.204.39.132 ping statistics ---
10 packets transmitted, 10 received, 0% packet loss, time 9060ms
rtt min/avg/max/mdev = 125.761/297.505/646.472/150.371 ms
```

a l'aide de cette commande on peut capturer notre reseau provenant de l'address 10.0.3.15 vers 54.204.39.132

```
(root㉿kali)-[/home/kali]
# tshark -i eth1 -c 10 host 54.204.39.132
Running as user "root" and group "root". This could be dangerous.
Capturing on 'eth1'
 1 0.000000000 10.0.3.15 → 54.204.39.132 ICMP 98 Echo (ping) request id=0x5b0d, seq=1/256, ttl=64
 2 0.221002284 54.204.39.132 → 10.0.3.15 ICMP 98 Echo (ping) reply id=0x5b0d, seq=1/256, ttl=44 (request in 1)
 3 1.015454297 10.0.3.15 → 54.204.39.132 ICMP 98 Echo (ping) request id=0x5b0d, seq=2/512, ttl=64
 4 1.141645564 54.204.39.132 → 10.0.3.15 ICMP 98 Echo (ping) reply id=0x5b0d, seq=2/512, ttl=44 (request in 3)
 5 2.020925857 10.0.3.15 → 54.204.39.132 ICMP 98 Echo (ping) request id=0x5b0d, seq=3/768, ttl=64
 6 2.369776211 54.204.39.132 → 10.0.3.15 ICMP 98 Echo (ping) reply id=0x5b0d, seq=3/768, ttl=44 (request in 5)
 7 3.021935887 10.0.3.15 → 54.204.39.132 ICMP 98 Echo (ping) request id=0x5b0d, seq=4/1024, ttl=64
 8 3.292796450 54.204.39.132 → 10.0.3.15 ICMP 98 Echo (ping) reply id=0x5b0d, seq=4/1024, ttl=44 (request in 7)
 9 4.022568591 10.0.3.15 → 54.204.39.132 ICMP 98 Echo (ping) request id=0x5b0d, seq=5/1280, ttl=64
10 4.267181123 54.204.39.132 → 10.0.3.15 ICMP 98 Echo (ping) reply id=0x5b0d, seq=5/1280, ttl=44 (request in 9)
10 packets captured
```

On peut aussi capturer uniquement des paquets réseau de protocoles spécifiques

si on veut chercher par exemple le protocole udp , on l'ajoute à la commande **tshark** . L'exemple suivant recherche uniquement les paquets UDP, mais il capture les paquets DNS. C'est parce que les paquets DNS utilisent le protocole UDP ci-dessous :

```
(root㉿kali)-[/home/kali]
# tshark -i eth1 -c 10 udp
Running as user "root" and group "root". This could be dangerous.
Capturing on 'eth1'
 1 0.000000000 10.0.3.15 → 89.2.0.1 DNS 87 Standard query 0x5679 A safebrowsing.googleapis.com
 2 0.016345272 89.2.0.1 → 10.0.3.15 DNS 103 Standard query response 0x5679 A safebrowsing.googleapis.com A 172.217.19.234
```



Nous avons aussi la possibilité d'écrire la sortie de la commande dans un fichier pcap. Pour le faire on utilise l'option **-w**

```
(root@kali)-[/home/kali]
# sudo tshark -w /tmp/nlog.pcap -i eth1 host 54.204.39.132
Running as user "root" and group "root". This could be dangerous.
Capturing on 'eth1'
22 ^C
```

Ensuite on doit utiliser l'option **-r** suivi du nom du fichier pcap pour le lire, ce qui donne

```
(root@kali)-[/home/kali]
# tshark -r /tmp/nlog.pcap
Running as user "root" and group "root". This could be dangerous.
 1 0.000000000 10.0.3.15 → 54.204.39.132 ICMP 98 Echo (ping) request id=0x09b5, seq=1/256, ttl=64
 2 0.116576835 54.204.39.132 → 10.0.3.15 ICMP 98 Echo (ping) reply id=0x09b5, seq=1/256, ttl=44 (request in 1)
 3 1.003010394 10.0.3.15 → 54.204.39.132 ICMP 98 Echo (ping) request id=0x09b5, seq=2/512, ttl=64
 4 1.116664533 54.204.39.132 → 10.0.3.15 ICMP 98 Echo (ping) reply id=0x09b5, seq=2/512, ttl=44 (request in 3)
 5 2.008567372 10.0.3.15 → 54.204.39.132 ICMP 98 Echo (ping) request id=0x09b5, seq=3/768, ttl=64
 6 2.126793456 54.204.39.132 → 10.0.3.15 ICMP 98 Echo (ping) reply id=0x09b5, seq=3/768, ttl=44 (request in 5)
 7 3.010603855 10.0.3.15 → 54.204.39.132 ICMP 98 Echo (ping) request id=0x09b5, seq=4/1024, ttl=64
 8 3.125632172 54.204.39.132 → 10.0.3.15 ICMP 98 Echo (ping) reply id=0x09b5, seq=4/1024, ttl=44 (request in 7)
 9 4.013251748 10.0.3.15 → 54.204.39.132 ICMP 98 Echo (ping) request id=0x09b5, seq=5/1280, ttl=64
10 4.126953207 54.204.39.132 → 10.0.3.15 ICMP 98 Echo (ping) reply id=0x09b5, seq=5/1280, ttl=44 (request in 9)
11 5.014518558 10.0.3.15 → 54.204.39.132 ICMP 98 Echo (ping) request id=0x09b5, seq=6/1536, ttl=64
12 5.202724502 54.204.39.132 → 10.0.3.15 ICMP 98 Echo (ping) reply id=0x09b5, seq=6/1536, ttl=44 (request in 11)
13 6.017066564 10.0.3.15 → 54.204.39.132 ICMP 98 Echo (ping) request id=0x09b5, seq=7/1792, ttl=64
14 6.253478523 54.204.39.132 → 10.0.3.15 ICMP 98 Echo (ping) reply id=0x09b5, seq=7/1792, ttl=44 (request in 13)
15 7.018751377 10.0.3.15 → 54.204.39.132 ICMP 98 Echo (ping) request id=0x09b5, seq=8/2048, ttl=64
16 7.132274341 54.204.39.132 → 10.0.3.15 ICMP 98 Echo (ping) reply id=0x09b5, seq=8/2048, ttl=44 (request in 15)
17 8.020313114 10.0.3.15 → 54.204.39.132 ICMP 98 Echo (ping) request id=0x09b5, seq=9/2304, ttl=64
18 8.136286256 54.204.39.132 → 10.0.3.15 ICMP 98 Echo (ping) reply id=0x09b5, seq=9/2304, ttl=44 (request in 17)
19 9.021781676 10.0.3.15 → 54.204.39.132 ICMP 98 Echo (ping) request id=0x09b5, seq=10/2560, ttl=64
20 9.135630731 54.204.39.132 → 10.0.3.15 ICMP 98 Echo (ping) reply id=0x09b5, seq=10/2560, ttl=44 (request in 19)
21 10.109980106 10.0.3.15 → 54.204.39.132 ICMP 98 Echo (ping) request id=0x09b5, seq=11/2816, ttl=64
22 10.224217432 54.204.39.132 → 10.0.3.15 ICMP 98 Echo (ping) reply id=0x09b5, seq=11/2816, ttl=44 (request in 21)
```

## 5-EtherApe

Etherape est un outil gratuit et open source de reniflage de paquets/de surveillance du trafic réseau, développé pour Unix/Linux. Basé sur etherman qui propose des modes couche de liaison, tcp et ip. Etherape est un moniteur en temps réel dont la représentation graphique change instantanément au fur et à mesure que le trafic réseau entre et sort. Vous pouvez utiliser Etherape en direct ou le lire à partir d'un fichier de vidage. Etherape prend en charge les périphériques Ethernet, FDDI, Token Ring, WLAN, RNIS, PPP et SLIP.

### Installation d'EtherApe

L'installation d'EtherApe sur linux est simple, il sera installé à l'aide de la simple commande ci-dessous, puis il faut appuyer sur la touche « y » pour continuer.

```
(root@kali)-[/home/kali]
# sudo apt-get install etherape
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  etherape-data libgoocanvas-2.0-9 libgoocanvas-2.0-common
The following NEW packages will be installed:
  etherape etherape-data libgoocanvas-2.0-9 libgoocanvas-2.0-common
0 upgraded, 4 newly installed, 0 to remove and 1040 not upgraded.
Need to get 1,057 kB of archives.
After this operation, 5,396 kB of additional disk space will be used.
Do you want to continue? [Y/n] Y
Get:1 http://archive-4.kali.org/kali kali-rolling/main amd64 libgoocanvas-2.0-common all 2.0.4-1 [141 kB]
Get:2 http://archive-4.kali.org/kali kali-rolling/main amd64 libgoocanvas-2.0-9 amd64 2.0.4-1+b1 [126 kB]
Get:3 http://archive-4.kali.org/kali kali-rolling/main amd64 etherape-data all 0.9.20-1 [699 kB]
Get:4 http://archive-4.kali.org/kali kali-rolling/main amd64 etherape amd64 0.9.20-1 [91.3 kB]
Fetched 1,057 kB in 1s (829 kB/s)
Selecting previously unselected package libgoocanvas-2.0-common.
(Reading database ... 277287 files and directories currently installed.)
Preparing to unpack .../libgoocanvas-2.0-common_2.0.4-1_all.deb ...
Unpacking libgoocanvas-2.0-common (2.0.4-1) ...
```

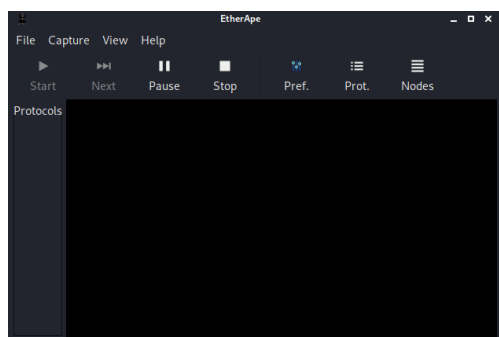
### Démarrage d'EtherApe

Après l'installation, on peut démarrer EtherApe sur notre machine en utilisant la commande ci-dessous

```
(root@kali)-[/home/kali]
# etherape
```



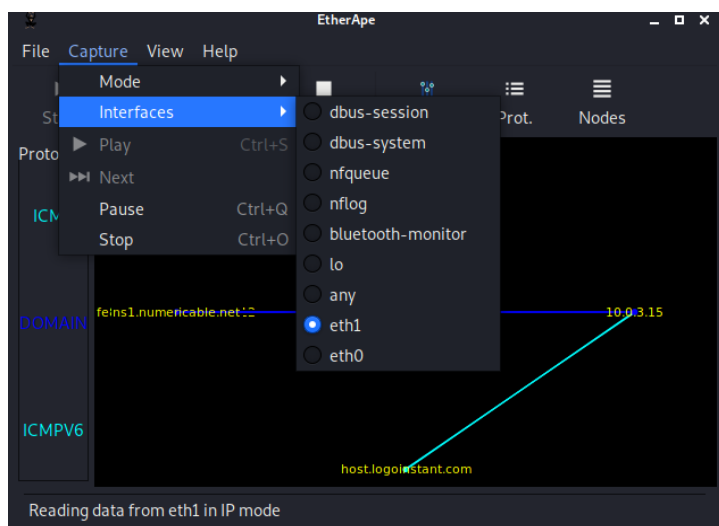
Lorsque on lance Etherape pour la première fois, une fenêtre vide s'affiche avec des boutons et des menus. À ce stade, on ne capture aucun paquet car on n'a pas indiqué à Etherape quelles interfaces utiliser.



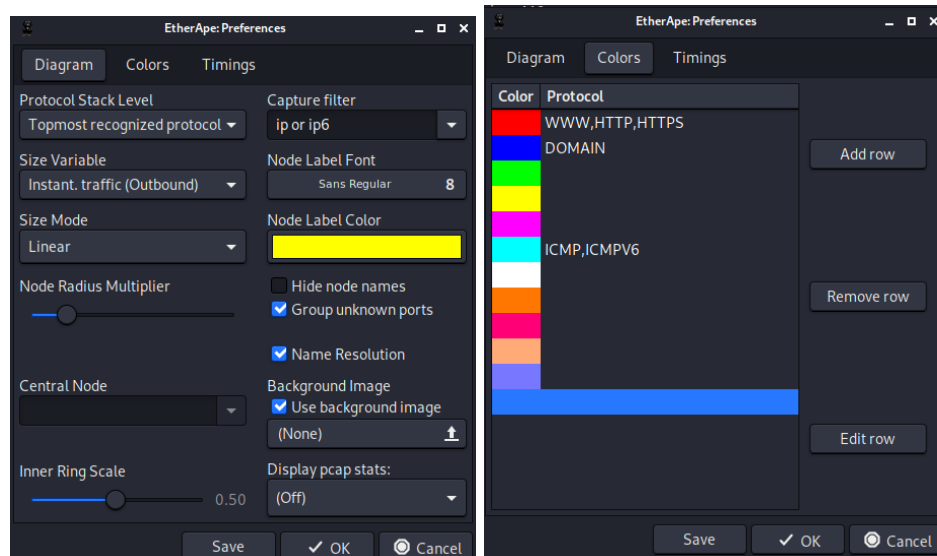
## Configuration d'Etherape

Pour configurer l'interface cliquez sur le menu Capture puis cliquez sur le sous-menu Interfaces. Sélectionnez l'interface que votre machine utilise pour continuer. Une fois que vous avez configuré l'interface, sélectionnez le type de mode dans le même menu que vous avez trouvé l'entrée Interfaces.

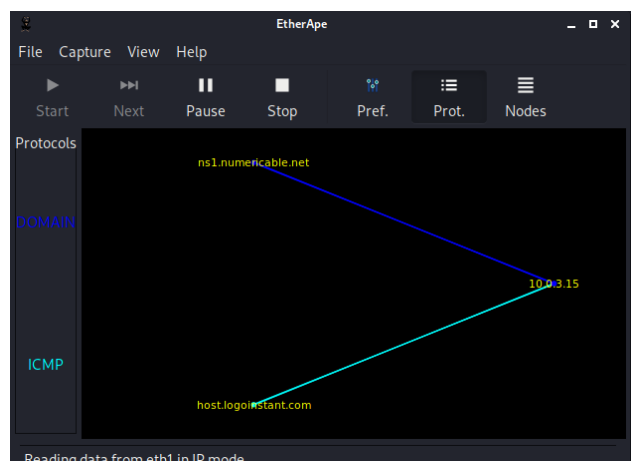
Etherape commencera immédiatement à capturer les paquets. Vous verrez la grande fenêtre noire se remplir rapidement de trafic. Selon votre réseau, votre fenêtre peut se remplir très rapidement. Vous remarquerez également qu'Etherape contient une légende codée par couleur.



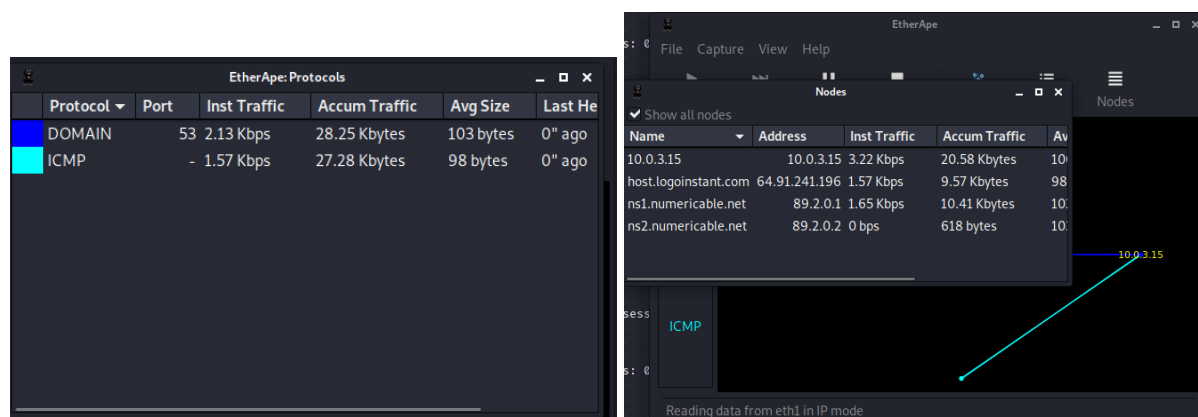
Vous pouvez également créer des filtres pour votre moniteur, à partir de la barre d'options supérieure en sélectionnant **"Fichier >> Préférences"**. Ici, vous pouvez apporter les modifications appropriées sous le schéma, le schéma de coloration et les horaires, puis cliquer sur le bouton « enregistrer » pour mettre en œuvre les modifications.



Etherape commence à capturer les paquets. La grande fenêtre noire se remplit rapidement de trafic. Selon votre réseau, votre fenêtre peut se remplir très rapidement. On remarque également qu'Etherape contient une légende codée par couleur.



Si on clique sur **"Afficher >> Protocoles"**, on peut voir les multiples protocoles et couleurs sur le côté gauche de l'écran. Chaque protocole représente la couleur équivalente. Comme on peut voir différentes statistiques indiquant que chaque protocole écoute sur quel port et quelle quantité de trafic est générée.

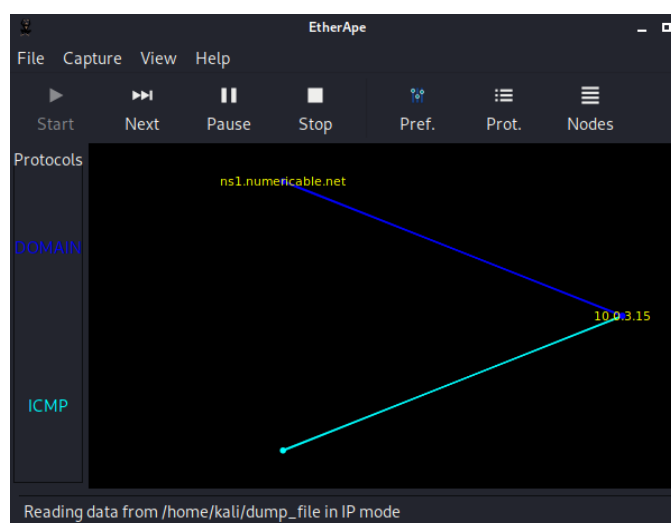


## Lecture à partir de fichiers

Etherape peut lire les fichiers de l'application tcpdump . Mais il faut l'exécuter avec les options -n et -w .

La commande tcpdump serait exécutée comme suit : `tcpdump -n -w dump_file`

Une fois qu'on a suffisamment d'informations, on arrête le vidage avec ctrl-c. Une fois qu'on a le fichier de vidage, on peut l'ouvrir en allant dans Fichier, puis en sélectionnant Ouvrir. Les paquets capturés s'afficheront en temps réel tels qu'ils ont été capturés lors de l'exécution de tcpdump.



Etherape est un excellent outil pour surveiller le trafic réseau. Non seulement il est simple à utiliser, mais il nous offre une sortie instantanée lorsque le trafic entre et sort de notre réseau. Il fournit des graphiques de tous les

hôtes qui ont des packages entrants/sortants avec notre ordinateur, le logiciel a la capacité de créer un graphique basé sur tous les packages en cours de traitement sur notre interface réseau. Ainsi, en utilisant EtherApe, on peut résoudre les problèmes liés à la sécurité de notre ordinateur, tels que les virus, les attaques DoS, les attaques par force brute, et il est également possible de vérifier pourquoi notre réseau informatique est lent.

## 6-Ettercap

### Principe du protocole ARP

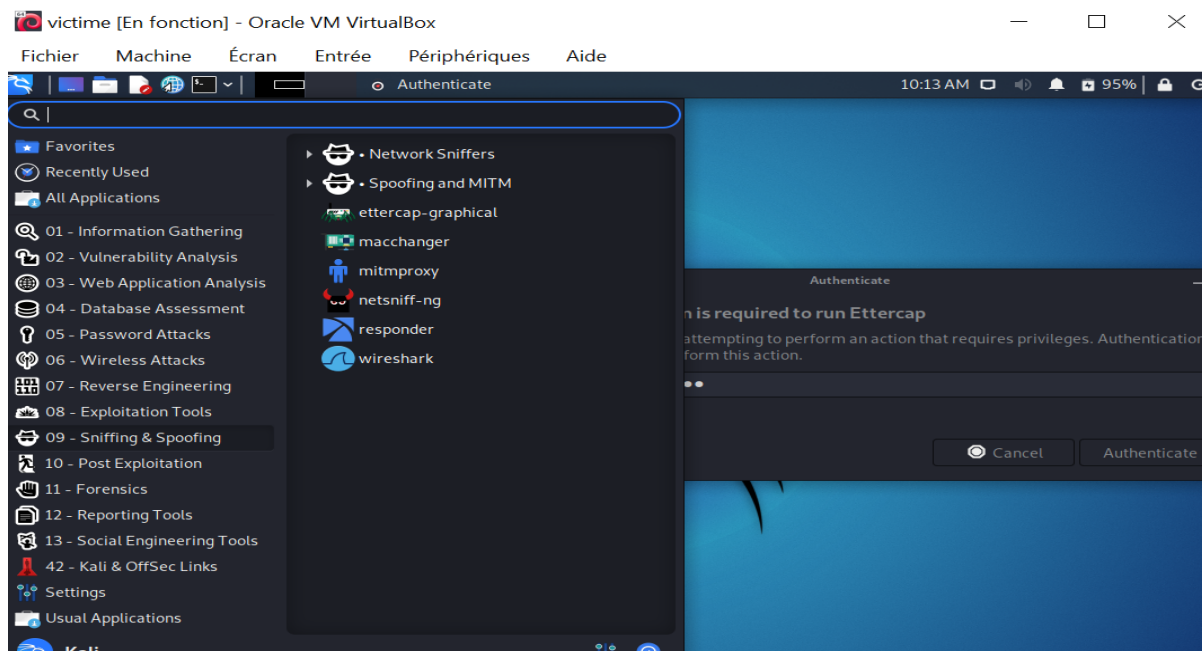
ARP est l'acronyme de Address Resolution Protocol (protocole de résolution d'adresse). Il fonctionne sur la couche réseau et est utilisé pour convertir une adresse IP en une adresse MAC (adresse physique).

Lorsqu'un nouvel ordinateur ou un nouveau périphérique est connecté au réseau, il diffuse son adresse MAC sur le réseau TCP/IP, puis tous les périphériques connectés trouvent l'adresse MAC de la nouvelle machine et la saisissent dans la table ARP.

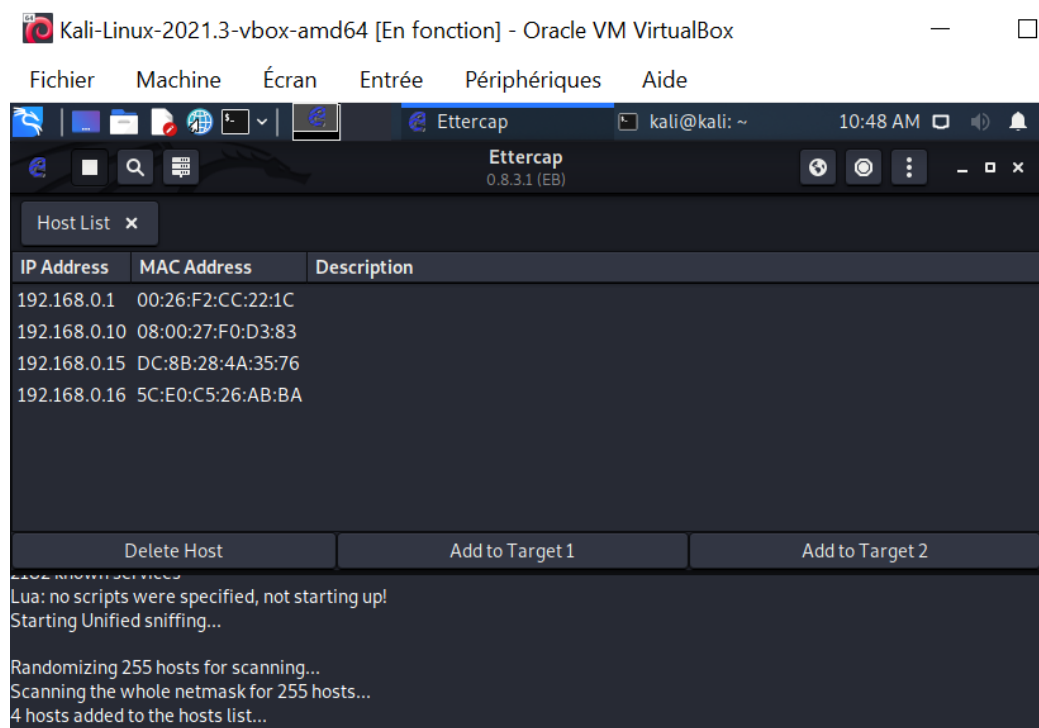
Il existe de nombreux outils disponibles pour les attaques de type "Man in the middle", dans ce qui suit nous travaillerons sur une suite complète pour ce genre d'attaque (MITM) qui est Ettercap, elle est préinstallée dans la plupart des systèmes d'exploitation de cybersécurité, notamment Kali Linux.

- ☐ Sur Kali Linux, ettercap est installé par défaut, pour l'ouvrir:

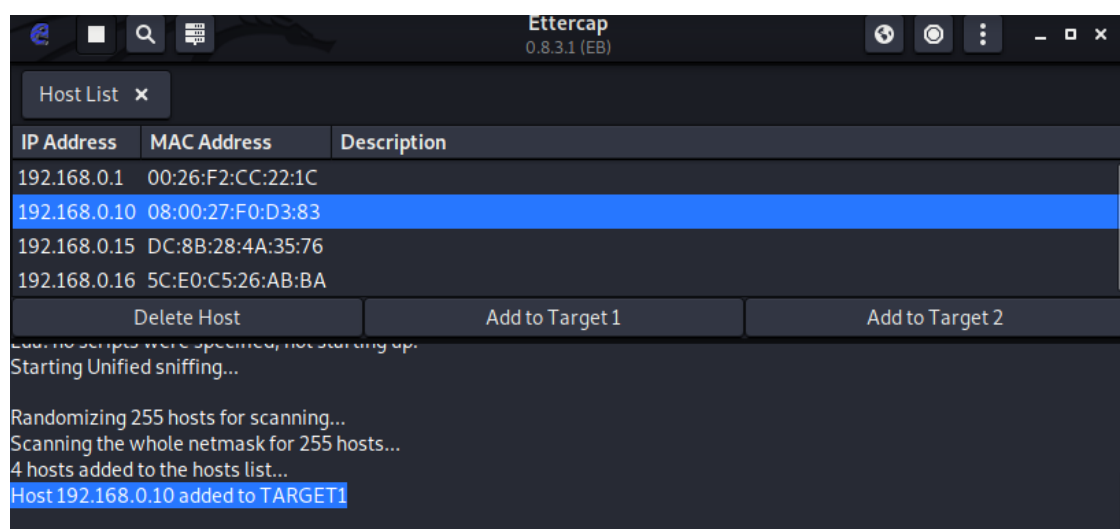
**Application > Sniffing & Spoofing > ettercap-graphical**



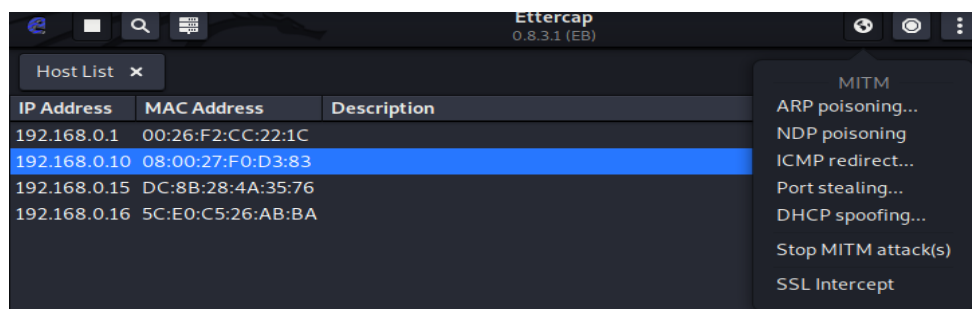
- ☐ en cliquant sur l'icône à 3 points > Hosts > Scan for hosts, on scanne l'ensemble du réseau sur lequel on est connecté puis dans le même menu on clique sur la liste des hôtes pour afficher le résultat du scan (hôtes disponibles dans le réseau)



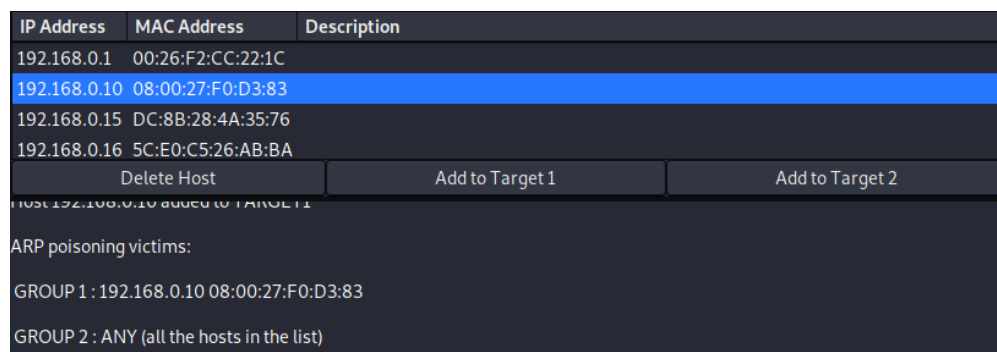
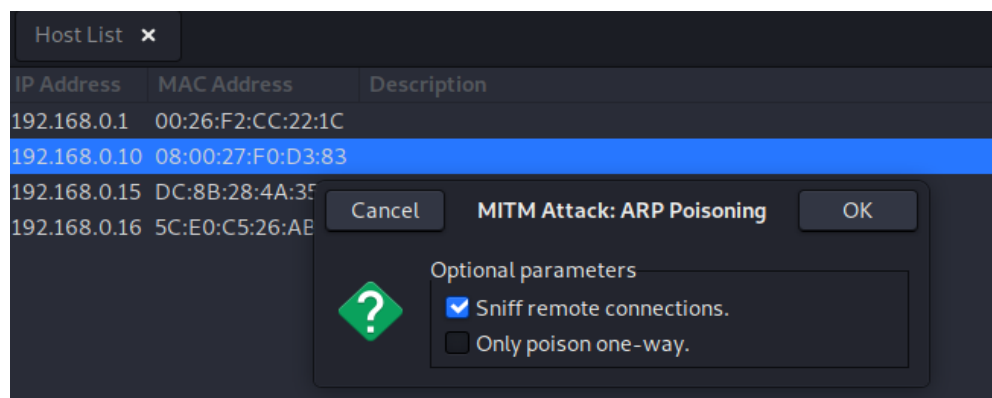
- ☐ Puis on ajoute l'hôte 192.168.0.10 dans target 1



☐ On clique sur l'option ARP poisoning, l'attaque par empoisonnement ARP va commencer




☐ a présent on peut renifler les données en cliquant sur Start > Start Sniffing



☐ on vérifie que l'empoisonnement arp est activé ou en allant dans Plugins > Manage the plugins  
puis on clique sur chk_poison le résultat est dans les captures suivantes :

Host List x Plugins x			
Name	Version	Info	
arp_cop	1.1	Report suspicious ARP activity	
autoadd	1.2	Automatically add new victims in the target range	
chk_poison	1.1	Check if the poisoning had success	
dns_spoof	1.3	Sends spoofed dns replies	
dos_attack	1.0	Run a d.o.s. attack against an IP address	
dummy	3.0	A plugin template (for developers)	
find_conn	1.0	Search connections on a switched LAN	
find_ettercap	2.0	Try to find ettercap activity	
find_ip	1.0	Search an unused IP address in the subnet	


Ettercap  
0.8.3.1 (EB)

Host List x Plugins x

Name	Version	Info
arp_cop	1.1	Report suspicious ARP activity
autoadd	1.2	Automatically add new victims in the target range
chk_poison	1.1	Check if the poisoning had success
dns_spoof	1.3	Sends spoofed dns replies
dos_attack	1.0	Run a d.o.s. attack against an IP address

ARP poisoning victims:

GROUP 1 : 192.168.0.10 08:00:27:F0:D3:83

GROUP 2 : ANY (all the hosts in the list)

Activating chk_poison plugin...

chk_poison: Checking poisoning status...

chk_poison: Poisoning process successful!

☐ Lorsque l'utilisateur accède à une page et saisit son identifiant de connexion, il est capturé par l'attaquant.

chk_poison: Poisoning process successful!

```
HTTP: 192.168.56.1:80 -> USER: vijay PASS: INFO: http://192.168.56.1/phpmyadmin/
CONTENT: token=1109610fa726265f17e5cb9a4fa7aed6&pred_username=userdefined&username=vijay&pred_hostname=any&hostname=%
25&pred_password=userdefined&pma_pw=ajay2vijay&pma_pw2=ajay2vijay&generated_pw=&createdb-3=on&dbname=cybers&grant_count=27&Select_priv=Y&Insert_priv=Y&Update_priv=Y&Delete_priv=Y
```

## 7-Tableau comparatif

Acrylic-WIFI	Tshark	EtherApe	Ettercap	Snaffler
<ul style="list-style-type: none"> <li>-Détection de clients connectés</li> <li>- Identification du meilleur canal et amélioration de la couverture</li> <li>- Trouver les mots de passe par défaut utilisés par les routeurs</li> </ul>	<ul style="list-style-type: none"> <li>-capture et l'analyse des paquets en temps réel</li> <li>-Similaire a wireshark</li> <li>- Capture paquets de différentes couches de réseau et de les affiche dans différents formats.</li> </ul>	<ul style="list-style-type: none"> <li>-Moniteur graphique de réseau pour Unix en temps réel.</li> <li>-Trouver qui consomme la bande passante et découvrir que l'on est la cible d'un worm.</li> </ul>	<ul style="list-style-type: none"> <li>-Capable d'accomplir des attaques man in the middle sur le protocole ARP en modifiant les correspondance @mac - @ip dans les tables arp chez les utilisateurs d'un LAN.</li> </ul>	<ul style="list-style-type: none"> <li>-Recherche de fichiers confidentiels dans un environnement Windows/AD massif</li> <li>-</li> </ul>

## Conclusion

Dans notre étude nous avons essayé de découvrir quelques outils permettant d'étudier le trafic d'un réseau, ces derniers sont d'une grande utilité aux administrateurs réseau afin de leur permettre de diagnostiquer les problèmes sur leur réseau ainsi que pour connaître le trafic qui y circule. Ainsi les détecteurs d'intrusion (*IDS*) sont basés sur un sniffeur pour la capture des trames, et utilisent une base de données de règles (*rules*) pour détecter des trames suspectes.

Ce fut une très bonne découverte que nous utiliserons sans doute durant notre stage et par la suite durant notre carrière professionnelle.