

TD – PROTOCOLE ENCAPSULATION

2. TESTS DU RESEAU REEL

L'objectif ici est de faire quelques manipulations simples sur notre machine afin de commencer à apprêhender comment les machines communiquent sur un réseau.

2.1. DETERMINER SES ADRESSES

 Sous Linux

Il vous faut lancer un terminal. A l'invite de commande, tapez :

```
ifconfig
```

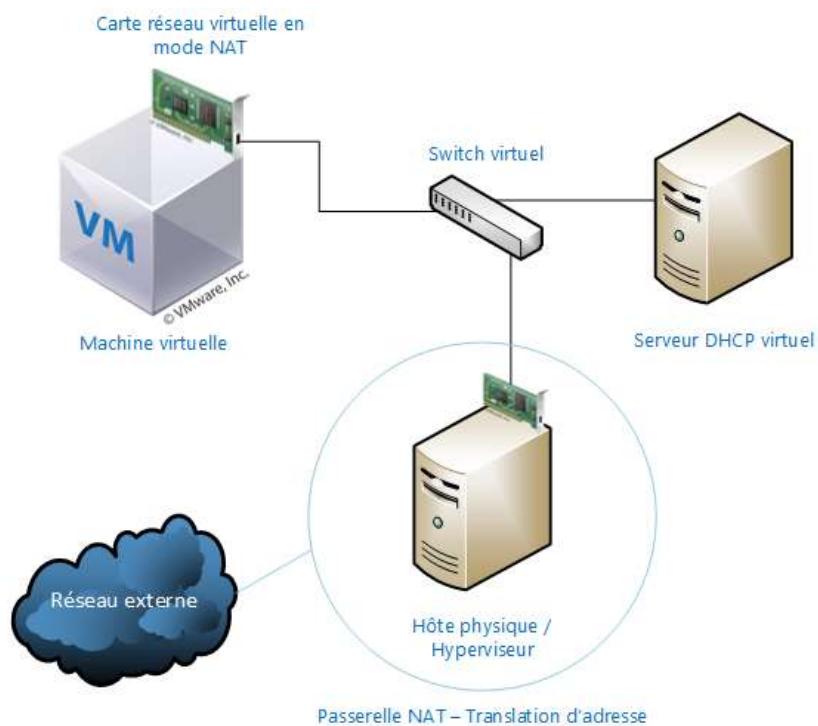
Ou bien :

```
ip
```

En résultat, vous devriez trouver toutes les informations concernant votre appareil, son ou ses adresses sur le réseau. En effet, votre ordinateur peut disposer de plusieurs interfaces réseaux (wifi, câble) et bien sûr, il dispose d'adresse IPv4 ou IPv6 et d'adresse MAC. Notez ces adresses avec le nom des interfaces correspondantes.

La commande ifconfig peut nécessiter d'être super utilisateur suivant les distributions, auquel cas, on utilisera la commande ip.

2.1) Le but de ma démarche ici va être de déterminer l'adresse IP de ma machine. Mais étant donné que j'utilise une machine virtuelle Linux sous un pc hôte physique, mon adresse affichée n'est pas mon adresse réelle sur le réseau ! En effet la machine virtuelle met en place une carte réseau virtuelle en mode **NAT** directement dans la virtualisation. Le **serveur DHCP virtuel** me configure donc automatiquement une adresse IP différente de celle de ma carte réseau physique !



Comme on peut le voir ci-dessous, la console m'affiche mon adresse IP « virtuelle » lorsque j'interroge mon système d'exploitation via la commande « ifconfig » :

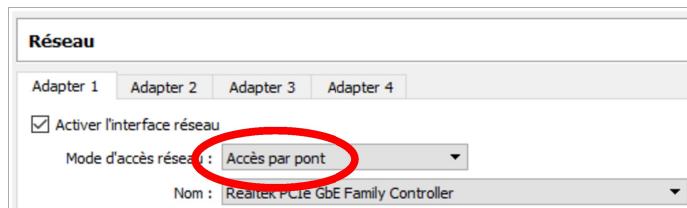
```
root@rmellaza-VirtualBox:/home/rmellaza/Bureau# ifconfig
enp0s3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
        inet 10.0.2.15 netmask 255.255.255.0 broadcast 10.0.2.255
                ether fe80::10d7:fbc4:8045:4dce prefixlen 64 scopeid 0x20<link>
                ether 08:00:27:34:05:31 txqueuelen 1000 (Ethernet)
                RX packets 138954 bytes 190656588 (190.6 MB)
                RX errors 0 dropped 0 overruns 0 frame 0
                TX packets 25782 bytes 2544136 (2.5 MB)
                TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
        inet 127.0.0.1 netmask 255.0.0.0
        inet6 ::1 prefixlen 128 scopeid 0x10<host>
            loop txqueuelen 1000 (Boucle locale)
            RX packets 1804 bytes 177246 (177.2 KB)
            RX errors 0 dropped 0 overruns 0 frame 0
            TX packets 1804 bytes 177246 (177.2 KB)
            TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

Je précise qu'il est préférable de réaliser cette commande dans une console exécutée en tant qu'administrateur « root ».

Heureusement il est possible de configurer ma machine virtuelle pour qu'elle utilise la même adresse IP que ma machine physique ! En effet plutôt que de configurer ma carte réseau virtuelle en mode NAT je vais la configurer en mode accès par pont « **Bridged Networking** ». L'adaptateur réseau virtuel est relié à ma carte réseau physique sur mon hôte, afin que je puisse accéder à ma machine virtuelle invitée de la même manière que si j'accède à l'hôte.

Cette configuration se réalise directement dans les paramètres de la machine virtuelle dans le logiciel « VirtualBox » :



Et voilà comme on peut le voir ci-dessous, en retapant la commande, mon adresse IP affichée correspond maintenant bel et bien à celle de ma machine hôte :

```
root@rmellaza-VirtualBox:/home/rmellaza/Bureau# ifconfig
enp0s3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
        inet 172.16.123.193 netmask 255.255.240.0 broadcast 172.16.127.255
                ether fe80::10d7:fbc4:8045:4dce prefixlen 64 scopeid 0x20<link>
                ether 08:00:27:34:05:31 txqueuelen 1000 (Ethernet)
                RX packets 139544 bytes 190787178 (190.7 MB)
                RX errors 0 dropped 72 overruns 0 frame 0
                TX packets 25904 bytes 2590191 (2.5 MB)
                TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
        inet 127.0.0.1 netmask 255.0.0.0
        inet6 ::1 prefixlen 128 scopeid 0x10<host>
            loop txqueuelen 1000 (Boucle locale)
            RX packets 1890 bytes 186779 (186.7 KB)
            RX errors 0 dropped 0 overruns 0 frame 0
            TX packets 1890 bytes 186779 (186.7 KB)
            TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

2.2. COMMUNIQUER AVEC SON VOISIN

Pour communiquer avec son voisin, on a besoin de son adresse IP. Ensuite, dans un premier temps, il suffit de faire un « ping » sur cette adresse. La commande ping est disponible sur Linux et Windows mais sous Linux, elle ne s'arrête pas d'elle-même. Du coup, 2 solutions, la barbare avec la combinaison Ctrl+C ou la civilisée en lui passant l'option -c (-n sous Windows) et en spécifiant le nombre de ping à faire.

```
ping -c 4 xxx.xxx.xxx.xxx
```

xxx.xxx.xxx.xxx est bien sur l'adresse IP de votre voisin.

2.2) Le but de l'exercice ici va être d'utiliser l'adresse IP d'une autre machine du réseau (en l'occurrence celle d'un camarade de classe dans mon cas), et de réaliser la commande « ping ». Cette commande calcule les temps d'aller-retour et les statistiques de perte de paquets, et affiche un bref résumé à la fin.

En premier lieu je vais vérifier que mes services réseaux sont opérationnels.

Pour cela je vais « pinguer » mon adresse de boucle locale (adresse réservée), qui est « **127.0.0.1** » :

```
root@rmellaza-VirtualBox:/home/rmellaza/Bureau# ping -c 100 127.0.0.1
PING 127.0.0.1 (127.0.0.1) 56(84) bytes of data.
64 octets de 127.0.0.1 : icmp_seq=1 ttl=64 temps=0.117 ms
64 octets de 127.0.0.1 : icmp_seq=2 ttl=64 temps=0.078 ms
64 octets de 127.0.0.1 : icmp_seq=3 ttl=64 temps=0.077 ms
64 octets de 127.0.0.1 : icmp_seq=4 ttl=64 temps=0.235 ms
64 octets de 127.0.0.1 : icmp_seq=5 ttl=64 temps=0.146 ms
64 octets de 127.0.0.1 : icmp_seq=6 ttl=64 temps=0.028 ms
64 octets de 127.0.0.1 : icmp_seq=7 ttl=64 temps=0.216 ms
64 octets de 127.0.0.1 : icmp_seq=8 ttl=64 temps=0.071 ms
64 octets de 127.0.0.1 : icmp_seq=9 ttl=64 temps=0.066 ms
64 octets de 127.0.0.1 : icmp_seq=10 ttl=64 temps=0.074 ms
64 octets de 127.0.0.1 : icmp_seq=11 ttl=64 temps=0.107 ms
64 octets de 127.0.0.1 : icmp_seq=12 ttl=64 temps=0.106 ms
64 octets de 127.0.0.1 : icmp_seq=13 ttl=64 temps=0.074 ms
64 octets de 127.0.0.1 : icmp_seq=14 ttl=64 temps=0.076 ms
64 octets de 127.0.0.1 : icmp_seq=15 ttl=64 temps=0.079 ms
64 octets de 127.0.0.1 : icmp_seq=16 ttl=64 temps=0.141 ms
64 octets de 127.0.0.1 : icmp_seq=17 ttl=64 temps=0.135 ms
64 octets de 127.0.0.1 : icmp_seq=18 ttl=64 temps=0.075 ms
64 octets de 127.0.0.1 : icmp_seq=19 ttl=64 temps=0.023 ms
64 octets de 127.0.0.1 : icmp_seq=20 ttl=64 temps=0.062 ms
64 octets de 127.0.0.1 : icmp_seq=21 ttl=64 temps=0.077 ms
64 octets de 127.0.0.1 : icmp_seq=22 ttl=64 temps=0.078 ms
64 octets de 127.0.0.1 : icmp_seq=23 ttl=64 temps=0.075 ms
64 octets de 127.0.0.1 : icmp_seq=24 ttl=64 temps=0.078 ms
^C
--- statistiques ping 127.0.0.1 ---
24 paquets transmis, 24 reçus, 0 % paquets perdus, temps 23545 ms
rtt min/moy/max/mdev = 0,023/0,095/0,235/0,048 ms
```

Tout a l'air en ordre ! Je vais pouvoir commencer mes tests réseaux bien réels.

J'utilise donc l'adresse IP de mon camarade qui est : « **172.16.123.200** », je rappelle que la mienne est : « **172.16.123.193** ».

```
root@rmellaza-VirtualBox:/home/rmellaza/Bureau# ping -c 10 172.16.123.200
PING 172.16.123.200 (172.16.123.200) 56(84) bytes of data.
64 octets de 172.16.123.200 : icmp_seq=1 ttl=64 temps=1.87 ms
64 octets de 172.16.123.200 : icmp_seq=2 ttl=64 temps=1.96 ms
64 octets de 172.16.123.200 : icmp_seq=3 ttl=64 temps=1.77 ms
64 octets de 172.16.123.200 : icmp_seq=4 ttl=64 temps=2.23 ms
64 octets de 172.16.123.200 : icmp_seq=5 ttl=64 temps=1.97 ms
64 octets de 172.16.123.200 : icmp_seq=6 ttl=64 temps=1.48 ms
64 octets de 172.16.123.200 : icmp_seq=7 ttl=64 temps=1.34 ms
64 octets de 172.16.123.200 : icmp_seq=8 ttl=64 temps=1.87 ms
64 octets de 172.16.123.200 : icmp_seq=9 ttl=64 temps=1.93 ms
64 octets de 172.16.123.200 : icmp_seq=10 ttl=64 temps=1.87 ms
^C
--- statistiques ping 172.16.123.200 ---
10 paquets transmis, 10 reçus, 0 % paquets perdus, temps 9013 ms
rtt min/moy/max/mdev = 1,344/1,828/2,231/0,239 ms
```

Nous verrons par la suite via le logiciel Wireshark s'il a bien reçu les paquets envoyés par ma machine !

Les arguments possibles de la fonction « ping » sont :

```
ping -c <number of echo request to send> <buffer size> <ip address>
```

2.3. CAPTURER DES TRAMES

2.3.1. LE VOISIN

Pour capturer des trames, rien ne vaut Wireshark. C'est un logiciel d'analyse de trame. Il peut capturer les trames réseaux tout comme, avec le bon plug in, les trames USB.



Sous Linux

Le problème peut aussi se poser ici. Dans un terminal, si vous passez en mode super utilisateur, soit par la commande su, soit par la commande sudo -s, il vous suffira ensuite de taper :

```
wireshark &
```

Ou bien restez sur votre compte utilisateur et tapez :

```
sudo wireshark &
```

Le & vous permet de reprendre la main sur votre terminal. Si vous l'oubliez, ce n'est pas grave, ouvrez en un autre.

2.3) Lors du lancement de Wireshark via la console il est important d'être connecté en tant qu'administrateur car sinon toutes les interfaces réseaux n'apparaîtront pas...

Si l'on revient donc sur le sujet du ping que j'ai envoyé à mon camarade (et donc le retour que j'ai aussi reçu), on va donc analyser les trames que j'ai précédemment capturées.

Mais un problème se pose ! Il y a vraiment beaucoup de trames sur le réseau lors de la capture, heureusement il existe deux commandes pour trier les différents échanges de données, il s'agit de :

```
ip.dst : destination  
ip.src : source
```

Sachant que lors de l'exécution de la fonction « **ping** » les paquets réalisent un aller-retour vers la machine ciblée et la machine source, alors je recherche les paquets partant de ma machine vers celle de mon camarade.

Par conséquent, j'utilise le filtre ICMP : « **ip.dst : 172.16.123.200** » :

No.	Time	Source	Destination	Protocol	Length	Info	
2651	41.043075948	172.16.123.193	172.16.123.200	ICMP	98	Echo (ping) reply	id=0x0002, seq=3/768, ttl=64 (request i...
2723	42.044693809	172.16.123.193	172.16.123.200	ICMP	98	Echo (ping) reply	id=0x0002, seq=4/1024, ttl=64 (request i...
2771	43.047067550	172.16.123.193	172.16.123.200	ICMP	98	Echo (ping) reply	id=0x0002, seq=5/1280, ttl=64 (request i...
2828	44.048920831	172.16.123.193	172.16.123.200	ICMP	98	Echo (ping) reply	id=0x0002, seq=6/1536, ttl=64 (request i...
2865	45.060815250	172.16.123.193	172.16.123.200	ICMP	98	Echo (ping) reply	id=0x0002, seq=7/1792, ttl=64 (request i...
2910	46.052924697	172.16.123.193	172.16.123.200	ICMP	98	Echo (ping) reply	id=0x0002, seq=8/2048, ttl=64 (request i...
2968	47.055245559	172.16.123.193	172.16.123.200	ICMP	98	Echo (ping) reply	id=0x0002, seq=9/2304, ttl=64 (request i...
3013	48.057423143	172.16.123.193	172.16.123.200	ICMP	98	Echo (ping) reply	id=0x0002, seq=10/2560, ttl=64 (request i...
3058	49.058856857	172.16.123.193	172.16.123.200	ICMP	98	Echo (ping) reply	id=0x0002, seq=11/2816, ttl=64 (request i...
3099	50.060241233	172.16.123.193	172.16.123.200	ICMP	98	Echo (ping) reply	id=0x0002, seq=12/3072, ttl=64 (request i...
3201	51.063178343	172.16.123.193	172.16.123.200	ICMP	98	Echo (ping) reply	id=0x0002, seq=13/3328, ttl=64 (request i...
3249	52.065399125	172.16.123.193	172.16.123.200	ICMP	98	Echo (ping) reply	id=0x0002, seq=14/3584, ttl=64 (request i...
3320	53.066913839	172.16.123.193	172.16.123.200	ICMP	98	Echo (ping) reply	id=0x0002, seq=15/3840, ttl=64 (request i...
3373	54.069244532	172.16.123.193	172.16.123.200	ICMP	98	Echo (ping) reply	id=0x0002, seq=16/4096, ttl=64 (request i...
3422	55.071479348	172.16.123.193	172.16.123.200	ICMP	98	Echo (ping) reply	id=0x0002, seq=17/4352, ttl=64 (request i...
3483	56.073931203	172.16.123.193	172.16.123.200	ICMP	98	Echo (ping) reply	id=0x0002, seq=18/4608, ttl=64 (request i...
3535	57.075848100	172.16.123.193	172.16.123.200	ICMP	98	Echo (ping) reply	id=0x0002, seq=19/4864, ttl=64 (request i...
3569	58.077731837	172.16.123.193	172.16.123.200	ICMP	98	Echo (ping) reply	id=0x0002, seq=20/5120, ttl=64 (request i...
3614	59.079924345	172.16.123.193	172.16.123.200	ICMP	98	Echo (ping) reply	id=0x0002, seq=21/5376, ttl=64 (request i...
3666	60.082505073	172.16.123.193	172.16.123.200	ICMP	98	Echo (ping) reply	id=0x0002, seq=22/5632, ttl=64 (request i...
3720	61.084458691	172.16.123.193	172.16.123.200	ICMP	98	Echo (ping) reply	id=0x0002, seq=23/5888, ttl=64 (request i...
3765	62.086293462	172.16.123.193	172.16.123.200	ICMP	98	Echo (ping) reply	id=0x0002, seq=24/6144, ttl=64 (request i...
3812	63.087506392	172.16.123.193	172.16.123.200	ICMP	98	Echo (ping) reply	id=0x0002, seq=25/6400, ttl=64 (request i...
3861	64.089761542	172.16.123.193	172.16.123.200	ICMP	98	Echo (ping) reply	id=0x0002, seq=26/6656, ttl=64 (request i...
3901	65.091427875	172.16.123.193	172.16.123.200	ICMP	98	Echo (ping) reply	id=0x0002, seq=27/6912, ttl=64 (request i...
3937	66.094178713	172.16.123.193	172.16.123.200	ICMP	98	Echo (ping) reply	id=0x0002, seq=28/7168, ttl=64 (request i...
3984	67.095748525	172.16.123.193	172.16.123.200	ICMP	98	Echo (ping) reply	id=0x0002, seq=29/7424, ttl=64 (request i...
4055	68.097386771	172.16.123.193	172.16.123.200	ICMP	98	Echo (ping) reply	id=0x0002, seq=30/7680, ttl=64 (request i...

Et comme on peut le voir ci-dessus, cela fonctionne parfaitement, car on voit bien apparaître les multiples pings à la destination de la machine 172.16.123.200, qui utilisent bien le protocole ICMP comme c'est le cas avec la commande « ping » et enfin l'information est explicitement donnée sur la nature de l'échange !

2.3.2. LES GAFA

Toujours sur le même principe, pinguez maintenant les GAFA (Google, Apple, Facebook et Amazon).

Vous ne connaissez pas l'adresse IP de leurs serveurs ? Pas de problème.



Point cours

La commande ping peut également utiliser une adresse telle qu'on les utilise dans un navigateur.

```
ping -n 4 google.fr
```

La commande ping vous retournera la réponse d'un des serveurs de Google (dans cet exemple) grâce au service DNS (Domain Name Server). Ce service fait le lien entre adresse IP et adresse web. Il vous est généralement fourni par votre fournisseur d'accès.

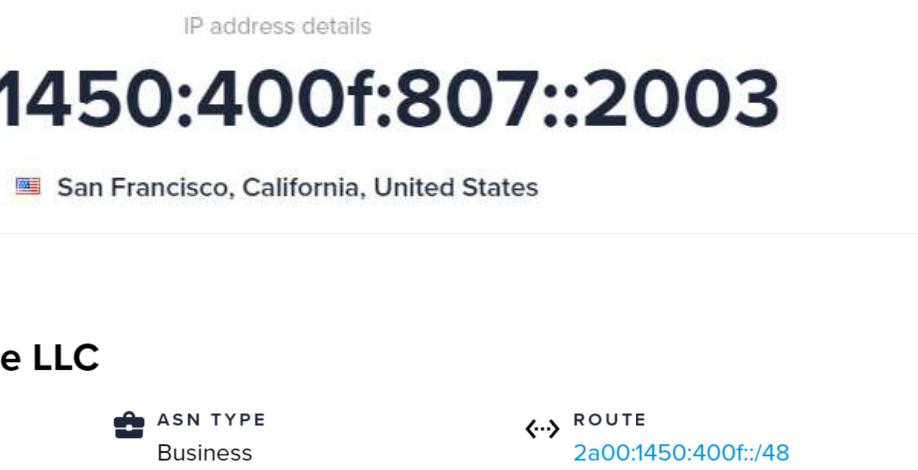
2.3.2) Je vais donc commencer par essayer de pinguer un des serveurs de Google, à noter que je ne renseigne pas directement son adresse IP mais bien son adresse web, en effet le service DNS (voir **TP Filius**) va me permettre de faire le lien entre l'adresse web que je renseigne et l'adresse IP correspondante !

- **Google.fr** :

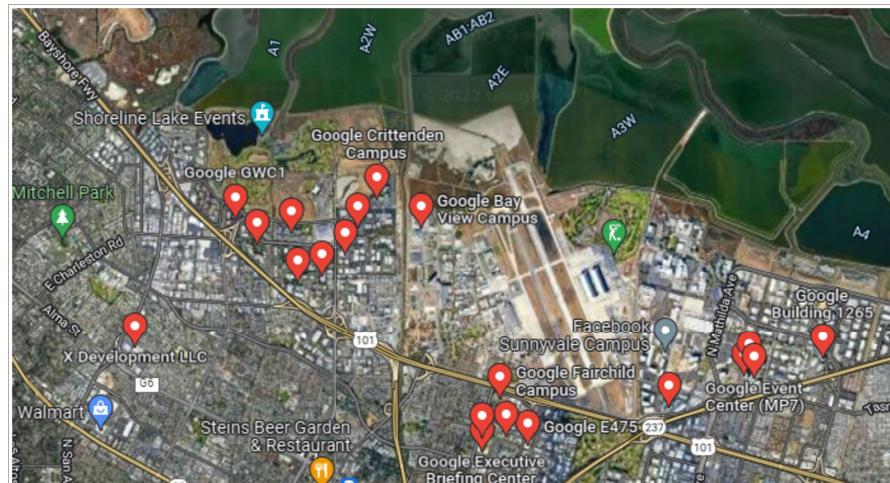
```
(romainmza@kali)-[~]
$ sudo ping -n -c 4 google.fr
PING google.fr(2a00:1450:4007:807::2003) 56 data bytes
64 bytes from 2a00:1450:4007:807::2003: icmp_seq=1 ttl=115 time=13.8 ms
64 bytes from 2a00:1450:4007:807::2003: icmp_seq=2 ttl=115 time=12.3 ms
64 bytes from 2a00:1450:4007:807::2003: icmp_seq=3 ttl=115 time=12.5 ms
64 bytes from 2a00:1450:4007:807::2003: icmp_seq=4 ttl=115 time=12.6 ms

--- google.fr ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3003ms
rtt min/avg/max/mdev = 12.273/12.801/13.846/0.616 ms
```

Comme on peut le voir, la console me retourne l'adresse IP :



Cette adresse est donc bien celle d'un des serveurs de l'entreprise Google, qui se trouve à San Francisco, en Californie, plus précisément dans un de ces data-centers :



- [Apple.com](#) :

```
(romainmza㉿kali)-[~]
$ sudo ping -n -c 4 apple.com
PING apple.com (17.253.144.10) 56(84) bytes of data.
64 bytes from 17.253.144.10: icmp_seq=2 ttl=58 time=11.0 ms
64 bytes from 17.253.144.10: icmp_seq=3 ttl=58 time=11.3 ms
64 bytes from 17.253.144.10: icmp_seq=4 ttl=58 time=11.1 ms

--- apple.com ping statistics ---
4 packets transmitted, 3 received, 25% packet loss, time 3020ms
rtt min/avg/max/mdev = 10.971/11.114/11.310/0.143 ms
```

Comme on peut le voir, la console me retourne l'adresse IP :

IP address details

17.253.144.10

 Cupertino, California, United States

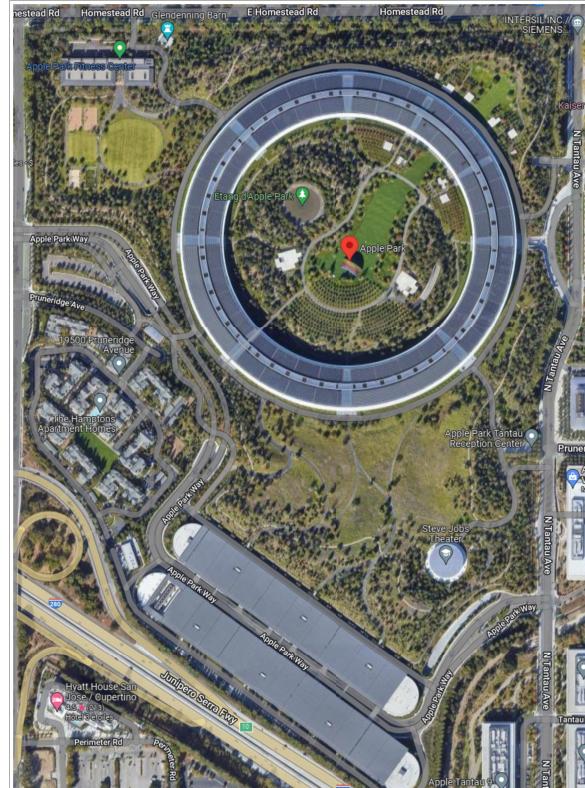
ASN

AS714 - Apple Inc.

 DOMAIN
apple.com

 ASN TYPE
Business

 ROUTE
17.253.144.0/21



Cette adresse est donc bien celle d'un des serveurs de l'entreprise Apple, qui se trouve à Cupertino, en Californie, plus précisément dans l'immense « Apple Park » :

- **Facebook.com :**

```
(romainmza㉿kali)-[~]
$ sudo ping -n -c 4 facebook.com
[sudo] Mot de passe de romainmza :
PING facebook.com(2a03:2880:f130:83:face:b00c:0:25de) 56 data bytes
64 bytes from 2a03:2880:f130:83:face:b00c:0:25de: icmp_seq=1 ttl=54 time=14.1 ms
64 bytes from 2a03:2880:f130:83:face:b00c:0:25de: icmp_seq=2 ttl=54 time=12.7 ms
64 bytes from 2a03:2880:f130:83:face:b00c:0:25de: icmp_seq=3 ttl=54 time=15.1 ms
64 bytes from 2a03:2880:f130:83:face:b00c:0:25de: icmp_seq=4 ttl=54 time=13.5 ms

--- facebook.com ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3006ms
rtt min/avg/max/mdev = 12.712/13.840/15.077/0.870 ms
```

Comme on peut le voir, la console me retourne l'adresse IP :

IP address details

2a03:2880:f130:83:face:b00c:0:25de

DUBLIN, IRELAND

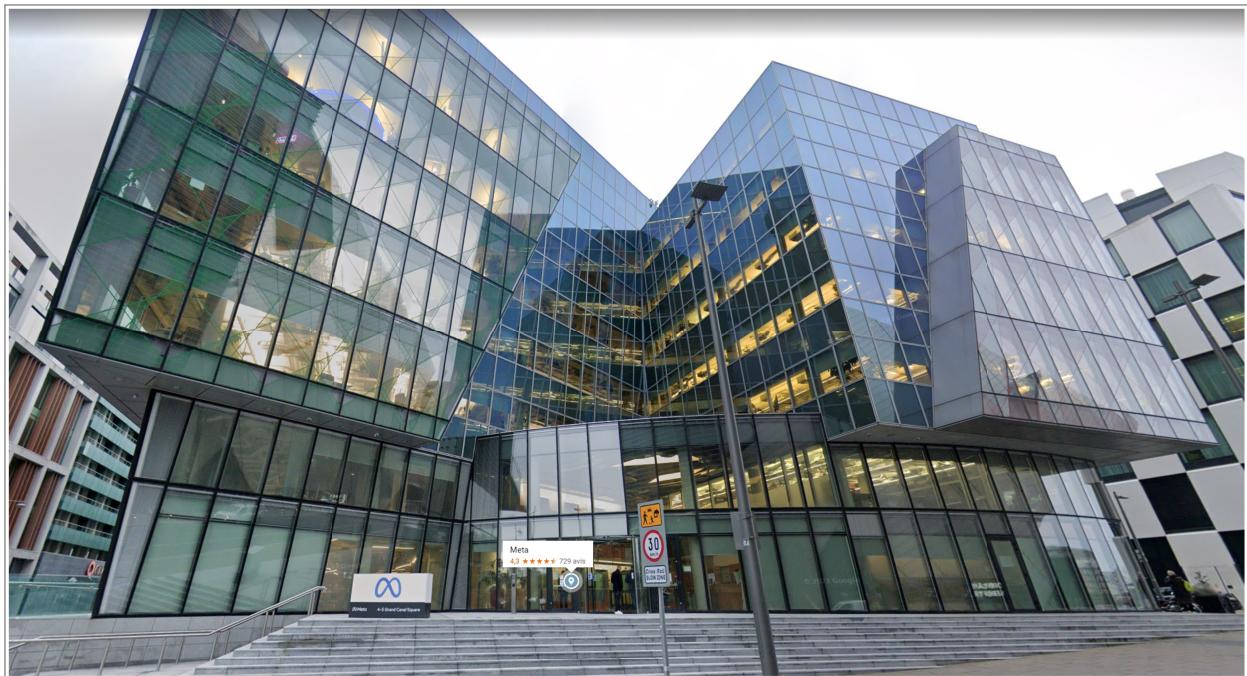
ASN

AS32934 - Facebook, Inc.

DOMAIN facebook.com	ASN TYPE Business	ROUTE 2a03:2880:f130::/48
------------------------	----------------------	------------------------------

Ici on peut remarquer deux choses très intéressantes :

- Contrairement aux autres GAFA, Meta (anciennement Facebook) dispose d'un siège européen à Dublin, que l'on peut voir ci-dessous :



Ce site supervise l'ensemble des activités de Facebook dans la zone EMEA (Europe, Moyen-Orient et Afrique), qu'il s'agisse de la vente de publicités ou de l'assistance technique pour le réseau. Il bénéficie au passage de conditions financières attractives offertes par le gouvernement, avec notamment des allègements de taxes.

L'entreprise compte bien continuer dans sa lancée, car elle construit en ce moment même un campus dans Dublin :



- Le deuxième point très intéressant et intriguant à mentionner et le fait que dans l'adresse IPv6 du serveur de Facebook que j'ai « pingué » on retrouve la mention « Facebook » sous la forme « face:b00c » !
- **Amazon.com** :

```
(romainmza㉿kali)-[~]
└─$ sudo ping -n -c 4 amazon.com
[sudo] Mot de passe de romainmza :
PING amazon.com (205.251.242.103) 56(84) bytes of data.
64 bytes from 205.251.242.103: icmp_seq=1 ttl=231 time=87.3 ms
64 bytes from 205.251.242.103: icmp_seq=2 ttl=231 time=87.6 ms
64 bytes from 205.251.242.103: icmp_seq=3 ttl=231 time=87.6 ms
64 bytes from 205.251.242.103: icmp_seq=4 ttl=231 time=87.4 ms

--- amazon.com ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3006ms
rtt min/avg/max/mdev = 87.345/87.478/87.625/0.114 ms
```

Comme on peut le voir, la console me retourne l'adresse IP :

IP address details

205.251.242.103

Ashburn, Virginia, United States

ASN

AS16509 - Amazon.com, Inc.

 DOMAIN
amazon.com

 ASN TYPE
Hosting

 ROUTE
205.251.240.0/22

Il faut savoir que la société Amazon dispose d'une branche nommée AWS qui est une référence mondiale dans le domaine du stockage web et du cloud. Dans notre cas on voit que l'adresse IP obtenue correspond à un serveur se situant à Ashburn, en Virginie.

En l'occurrence il existe un énorme data-center appartenant à AWS à Ashburn.

En faisant quelques recherches je me suis rendu compte que en réalité beaucoup d'informaticiens appellent Ashburn comme la capitale mondiale des data-centers. Lien vers l'article complet traitant de ce sujet :

<https://www.datacenters.com/news/why-is-ashburn-the-data-center-capital-of-the-world>

En résumé, cette aussi haute concentration de data-centers de sociétés différentes à Ashburn s'explique par plusieurs facteurs d'échanges et connexions:

- **L'électricité**, la Virginie est l'un des États américains qui en produit le plus ! Le coût de l'électricité y est donc moins cher, et sachant que selon l'auteur de l'article, les data-centers de Ashburn consomment autant que 40 % de la consommation du Royaume-Uni, c'est un facteur à prendre en compte !
- **Le coût du terrain**, il est raisonnable pour des grandes firmes.
- **La fibre optique**, le réseau de fibre optique en Virginie est l'un des mieux développé au monde, l'État a investi dans cette technologie dès les années 90 !
- **Le réseau d'eau**, essentiel pour le refroidissement des bâtiments !

On constate que les plus grandes entreprises du domaine du cloud y sont présentes :

- Amazon Web Service (AWS)
- Google
- Microsoft
- CyrusOne
- Digital Reality
- Equinix
- Intel

Pour vous donner une idée voici la surface totale occupée par les data-centers :



En comparaison, avec exactement la même échelle voici une vue satellitaire de la ville de Brest :



On se rend bien compte de la taille démesurée des bâtiments accueillant les serveurs, d'autant plus qu'il y a un aéroport spécialisé dans le fret ainsi qu'une gare de marchandises qui desservent directement les différents data-centers !



Point cours

Votre machine a besoin des adresses MAC pour pouvoir communiquer sur le réseau.

Elle stocke de manière temporaire les adresses qu'elle connaît dans une table dite table ARP (Address Resolution Protocol).

Cette table fait la correspondance entre adresse IP et adresse MAC.

Vous allez regarder dans la table ARP de votre PC



Sous Linux

C'est normalement la même commande que sous Windows. Mais il se peut que votre Linux n'accepte pas arp -a directement. Il vous faut soit taper

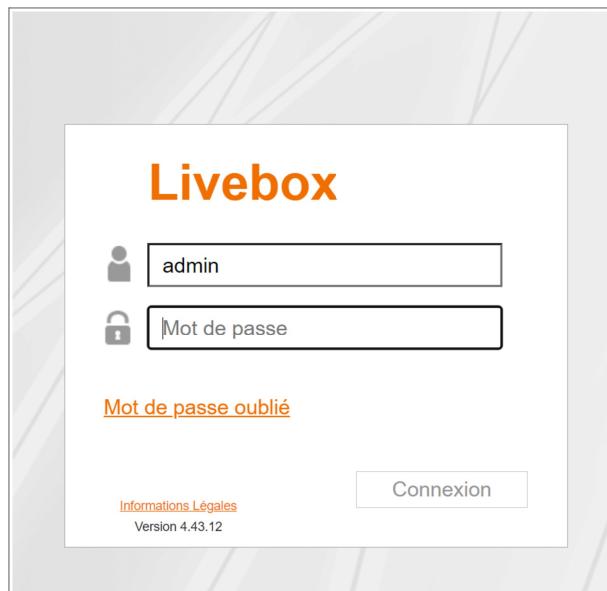
```
sudo arp -a
```

soit le chemin complet :

```
/usr/sbin/arp -a
```

```
(romainmza㉿kali)-[~]
$ sudo arp -a
[sudo] Mot de passe de romainmza :
livebox.home (192.168.1.1) at 5c:b1:3e:ff:65:90 [ether] on eth0
```

Comme on peut le voir ci-dessus la commande « **arp -a** » me renvoie bien l'adresse IP de ma box internet qui est une « livebox » d'Orange.
De plus si je rentre cette adresse IP dans mon navigateur web alors j'accède au panneau d'administration de ma box internet :





Point cours

Votre machine est dans un réseau. Pour communiquer avec d'autres machines de ce réseau, elle n'a à priori besoin d'aucun intermédiaire et peut même se contenter de connaître uniquement une adresse MAC. Par contre, pour aller sur les sites comme ceux des GAFA, il vous faut sortir du réseau. Vous passerez dès lors par une passerelle (un routeur) qui fera la jonction entre votre réseau local et les autres réseaux.

C'est ce que fait votre box chez vous.

Vous allez déterminer l'adresse de la passerelle de votre réseau.

Je réalise donc la commande « **route -n** » et « **route** » dans la console Linux :

```
(romainmza㉿kali)-[~]
$ route -n
Table de routage IP du noyau
Destination     Passerelle      Genmask         Indic Metric Ref    Use Iface
0.0.0.0         192.168.1.1   0.0.0.0         UG        100    0      0 eth0
192.168.1.0     0.0.0.0       255.255.255.0  U          100    0      0 eth0

(romainmza㉿kali)-[~]
$ route
Table de routage IP du noyau
Destination     Passerelle      Genmask         Indic Metric Ref    Use Iface
default         livebox.home   0.0.0.0         UG        100    0      0 eth0
192.168.1.0     0.0.0.0       255.255.255.0  U          100    0      0 eth0
```

Ma box internet qui est donc la passerelle de mon réseau apparaît bien !

On va maintenant voir par quels routeurs passent les paquets lorsqu'on se connecte à « amazon.com » :

```
(romainmza㉿kali)-[~]
$ traceroute -4 amazon.com
traceroute to amazon.com (54.239.28.85), 30 hops max, 60 byte packets
 1  livebox.home (192.168.1.1)  1.313 ms  1.288 ms  1.438 ms
 2  80.10.238.225 (80.10.238.225)  9.771 ms  9.760 ms  9.752 ms
 3  ae99-0.ncren102.rbcii.orange.net (193.251.108.86)  9.628 ms  9.805 ms  10.760 ms
 4  ae43-0.niidf302.rbcii.orange.net (193.252.159.161)  15.573 ms  14.659 ms  14.834 ms
 5  ae40-0.niidf301.rbcii.orange.net (193.252.103.37)  16.014 ms  16.117 ms  16.802 ms
 6  81.253.184.6 (81.253.184.6)  17.585 ms  11.636 ms  12.035 ms
 7  81.52.166.173 (81.52.166.173)  95.114 ms 81.52.166.169 (81.52.166.169)  90.573 ms  90.312 ms
 8  81.52.187.238 (81.52.187.238)  88.593 ms  89.274 ms  89.445 ms
 9  * * *
10  * * *
11  * * *
12  * * *
13  * * *
14  * * *
15  * * *
16  * * *
17  * * *
18  * * *
19  * * *
20  * * *
21  * * *
22  * * *
23  * * *
24  * * *
25  * * *
26  * * *
27  * * *
28  * * *
29  * * *
30  * * *
```

Comme on le voit sur la page précédente, la cible est toujours un serveur AWS à Ashburn, nous allons voir quels chemins mes paquets empruntent pour s'y rendre !

Je vais lister dans l'ordre les différents points de passages :

1. Passerelle de mon réseau (livebox)
2. Rennes (Orange)
3. Alexandria, Virginie
4. Ashburn, Virginie

Bonus) Je vais essayer de retrouver l'emplacement de stockage des données du site web de mon lycée qui est à l'adresse internet suivante :

<https://www.lycee-vauban-brest.ac-rennes.fr/>

Je ping donc ce nom domaine afin de récupérer l'adresse IP du serveur qui prend en charge ma requête :

```
root@rmellaza-VirtualBox:/home/rmellaza/Bureau# ping -n 10000 www.lycee-vauban-brest.ac-rennes.fr
PING www.lycee-vauban-brest.ac-rennes.fr (149.202.23.242) 56(124) bytes of data.
^C
--- statistiques ping www.lycee-vauban-brest.ac-rennes.fr ---
264 paquets transmis, 0 reçus, 100 % paquets perdus, temps 269319 ms
```

L'adresse du serveur est donc : « **149.202.23.242** », après quelques recherches je détermine que cette adresse appartient à un serveur OVH. Ce dernier se situe à Roubaix, dans ce data-center, situé 2 rue Kellermann :

