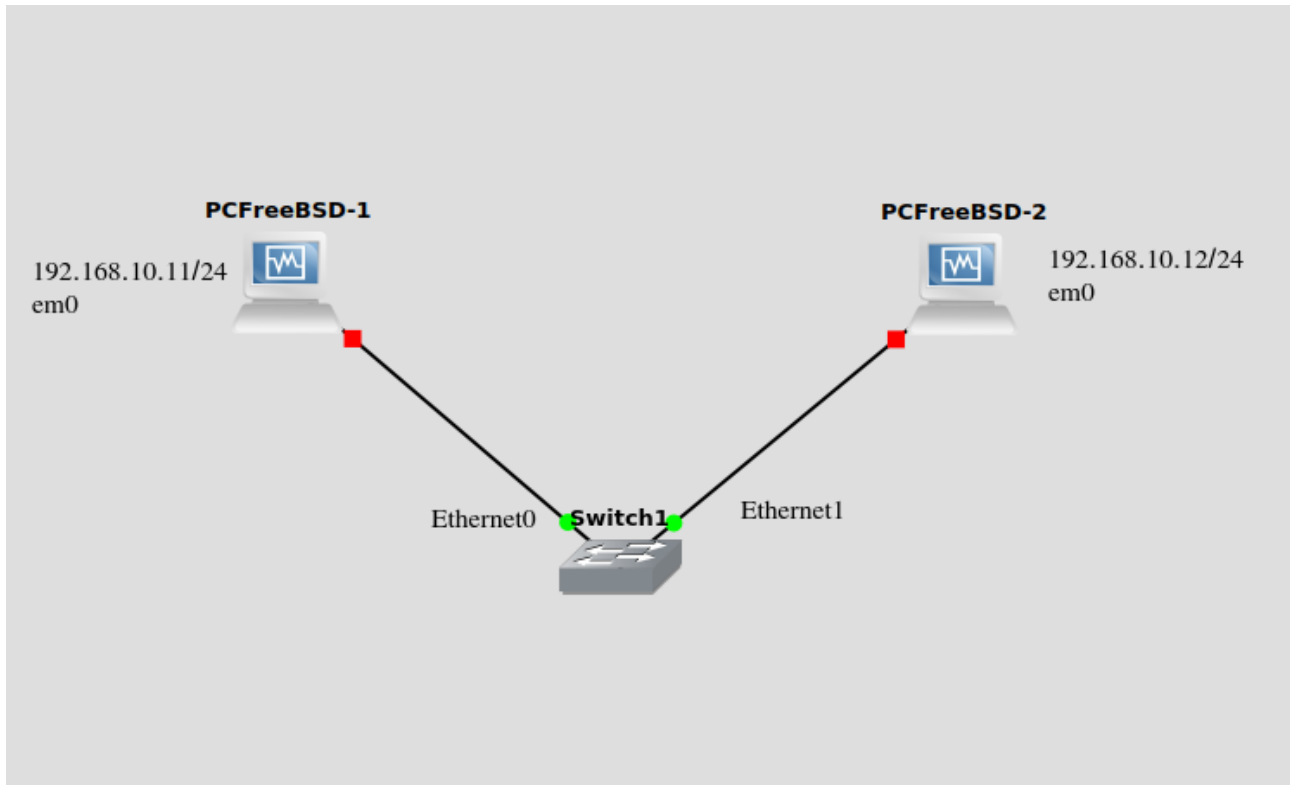


TP1 RX

2.1.2.1 :

J'ai choisi le masque /24 et le réseau 192.168.10.0 . Le PC1 aura l'ip 192.168.10.11 et le PC2 aura l'ip 192.168.10.12



2.1.2.2 :

```
em0: flags=8843<UP,BROADCAST,RUNNING,SIMPLEX,MULTICAST> metric 0 mtu 1500
options=81009b<RXCSUM,TXCSUM,VLAN_MTU,VLAN_HWTAGGING,VLAN_HWCSUM,V
LAN_HWFILTER>
ether 08:00:27:71:c1:22
inet 192.168.10.11 netmask 0xfffff00 broadcast 192.168.10.255
media: Ethernet autoselect (1000baseT <full-duplex>)
status: active
nd6 options=29<PERFORMNUD,IFDISABLED,AUTO_LINKLOCAL>
```

/24 -> 255.255.255.0 -> 0xfffff00

l'adresse de broadcast correspond à la dernière adresse (ici en tous cas) donc
192.168.10.255

Sur chaque machine on rajoute dans le fichier /etc/hosts :

```
192.168.10.11 PC1
192.168.10.12 PC2
192.168.10.13 PC3
```

192.168.10.14 PC4

2.2.3:

Une machine envoie un paquet ICMP à une autre machine qui répond simplement en renvoyant une réponse correspondante au paquet reçu.

2.2.4:

Arp sert à résoudre les adresses IP en réseau local. Ainsi la table arp contient la correspondance entre adresse IP et adresse Ethernet. Elle apparaît qu'une seule fois au début lorsqu'une machine veut communiquer pour la première fois avec une autre machine en connaissant son IP car ensuite, la paire adresse IP / adresse Ethernet sera gardée en cache (de manière permanente ou pas (expiration)).

2.3.5:

Pour la partie 2.3, je me base entièrement sur des captures d'écran des manipulations faites en présentiel.

The image shows two terminal windows. The left window displays the output of the command `netstat -I bge0 10`, showing network statistics for interface bge0. The right window displays the output of the command `arp -n`, showing the ARP table.

input				output			
packets	errs	idrops	bytes	packets	errs	bytes	colls
22086	0	0	11970612	0	0	0	0
22086	0	0	11970612	0	0	0	0
22086	0	0	11970612	0	0	0	0
22095	0	0	11975490	0	0	0	0
22074	0	0	11964108	0	0	0	0
22085	0	0	11970070	0	0	0	0
22086	0	0	11970612	0	0	0	0
22085	0	0	11970070	0	0	0	0
22085	0	0	11970070	0	0	0	0
22085	0	0	11970070	0	0	0	0
22085	0	0	11970070	0	0	0	0
22085	0	0	11970070	0	0	0	0

IP	MAC
115.831	8831.7
116.819	8834.0
117.819	8836.0
118.819	8832.0
119.819	8836.0
120.819	8832.0
121.819	8836.0
122.819	8832.0
123.826	8834.2
124.819	8833.8
125.819	8836.0
126.819	8832.0
127.819	8836.0
128.819	8836.0
129.819	8832.0
130.819	8832.0
131.819	8836.0
132.819	8832.0
133.819	8836.0
134.819	8832.0
135.819	8836.0
136.819	8836.0
137.819	8832.0
138.819	8832.0
139.820	8839.2
140.819	8832.8
141.819	8832.0
142.819	8836.0
143.819	8832.0
144.819	8836.0
145.819	8832.0
146.819	8836.0
147.819	8832.0
148.819	8836.0
149.821	8843.2
150.819	8824.8
151.819	8832.0
152.819	8836.0
153.819	8832.0
154.819	8836.0

On remarque qu'il n'y a pas de collision et c'est compréhensible : il n'y a que 2 machines sur réseau. De plus, seule 1 machine envoie des données sur le réseau, donc pas de partage, donc pas de collisions.

2.3,6:

Terminal

Fichier	Édition	Affichage	Terminal	Onglets	Aide			
0	0	0	0	22085	0	12186328	0	
0	0	0	0	22086	0	11909366	0	
0	0	0	0	22085	0	11909366	0	
0	0	0	0	22084	0	11909366	0	
2	0	0	184	22084	0	11909366	6	
0	0	0	0	22086	0	12186328	0	
0	0	0	0	22085	0	11909366	0	
2	0	0	184	22080	0	11909366	49	
2	0	0	184	22084	0	11909366	6	
0	0	0	0	22091	0	12186328	0	
6	0	0	468	22080	0	11908916	0	
3	0	0	234	22085	0	11909366	0	
0	0	0	0	22085	0	11909366	0	
input				output				
packets	errs	idrops	bge0	bytes	packets	errs	bytes	colls
0	0	0	0	0	22086	0	11909366	0
0	0	0	0	0	22085	0	12186328	0
0	0	0	0	0	22085	0	11909366	0
0	0	0	0	0	22085	0	11909366	0
0	0	0	0	0	22085	0	11909366	0
0	0	0	0	0	22085	0	12186328	0
1	0	0	60	15514	0	8308860	469	
0	0	0	0	10495	0	5816202	727	
0	0	0	0	12652	0	6647088	775	
0	0	0	0	11247	0	6093164	776	
0	0	0	0	11116	0	6093164	799	
0	0	0	0	11380	0	6093164	723	
0	0	0	0	14027	0	7754936	826	
0	0	0	0	11859	0	6370126	764	
0	0	0	0	10481	0	5816202	815	
0	0	0	0	10714	0	5816202	760	
0	0	0	0	10940	0	5816202	734	
0	0	0	0	10875	0	5816202	820	
0	0	0	0	11226	0	6093164	822	
0	0	0	0	10077	0	5539240	790	
input				output				
packets	errs	idrops	bge0	bytes	packets	errs	bytes	colls
0	0	0	0	0	9981	0	5262278	744
0	0	0	0	0	10952	0	6093164	737
0	0	0	0	0	10896	0	5816202	777
0	0	0	0	0	10517	0	5816202	744
0	0	0	0	0	8055	0	4431392	781
0	0	0	0	0	10356	0	5539240	751
0	0	0	0	0	10153	0	5539240	761
0	0	0	0	0	11221	0	6093164	738
0	0	0	0	0	12591	0	6647088	789
0	0	0	0	0	10451	0	5816202	783
0	0	0	0	0	11467	0	6093164	752

Terminal

Fichier	Édition	Affichage	Terminal	Onglets	Aide	
6963						
42077.036	1022		952540		7893	4799
6964						
42077.925	1022		953562		4595	4702
6960						
42079.093	2044		955606		7001	4835
6960						
42080.367	1533		957139		4813	5015
6955						
42081.758	511		957650		1469	4477
6941						
42082.262	511		958161		4062	4447
6939						
42082.982	1022		959183		5675	4254
6937						
42084.330	2044		961227		6063	4925
6935						
42085.028	1533		962760		8785	5387
6937						
42086.124	2044		964804		7462	5532
6938						
42087.206	1533		966337		5667	5426
6936						
42088.292	511		966848		1882	5126
6926						
42088.965	1022		967870		6078	4969
6925						
42090.263	511		968381		1574	4544
6913						
42091.431	511		968892		1750	4649
6902						
42092.158	511		969403		2813	4544
6897						
42094.110	511		969914		1047	3857
6876						
42095.187	1022		970936		3796	3577
6871						
42096.059	1533		972469		7028	3521
6871						
42096.982	1533		974002		6646	3389
6870						
42098.032	1533		975535		5840	3398
6869						
42098.917	1533		977068		6931	3848
6869						
42100.126	1022		978090		3379	3663
6861						

La variation du nombre de collisions reste limité qui oscille entre 700 et 850.

A mon avis, cette relative stabilité est expliqué par un résonnement statistique (tirage aléatoire).

2.3,7:

Les captures d'écran ne me fournit pas assez d'information pour répondre à cette question.

Cependant, on trouve la réponse dans le cours : « Plus les paquets sont grands moins il y a de collisions » et c'est plutôt logique sachant qu'un paquet plus gros occupe le support plus longtemps pour un même lapse de temps.

2.3,8:

$T_{prop} = \text{Longueur du support} / \text{vitesse de transmission dans le support.}$

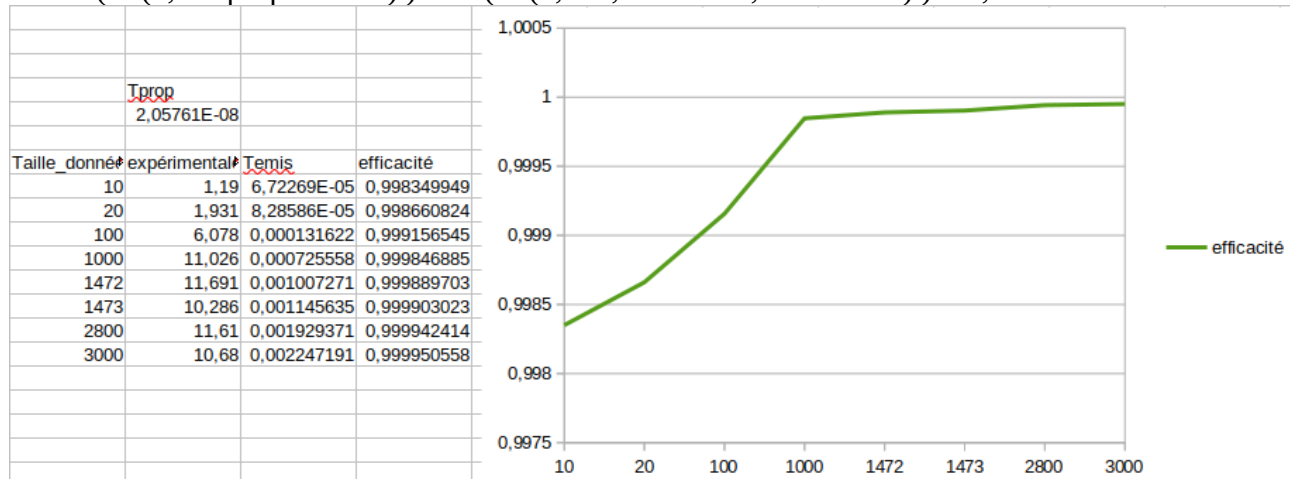
$T_{prop} = 5 / (0,81 * 3 \times 10^8) = 0,000000021 = 2,1 * 10^{-8}$

$\text{Temis} = \text{Taille_paquet} / \text{débit}$

$\text{débit} = 11691 \text{ kbit/s} \rightarrow 1461,375 \text{ ko/s} \rightarrow 1461375 \text{ o/s}$

$\text{Temis} = 1472 / 1461375 = 0,001007271$

$$E = 1/(1+(5,4 * T_{prop} / T_{emis})) = 1/(1+(5,4 * 2,1 * 10^{-8} / 0,001007271)) = 0,999887431$$



2.3,9 :

C'est le protocole CSMA/CD qui est responsable du partage « équitable », c'est une conséquence du tirage aléatoire (tantôt plus rapide sur la machine 1, tantôt plus rapide sur la machine 3) et de la priorité donnée au paquet avec un grand nombre de tentative (en augmentant la probabilité de tiré un temps court) qui évite ainsi la monopolisation d'une des 2 machines émettrice.

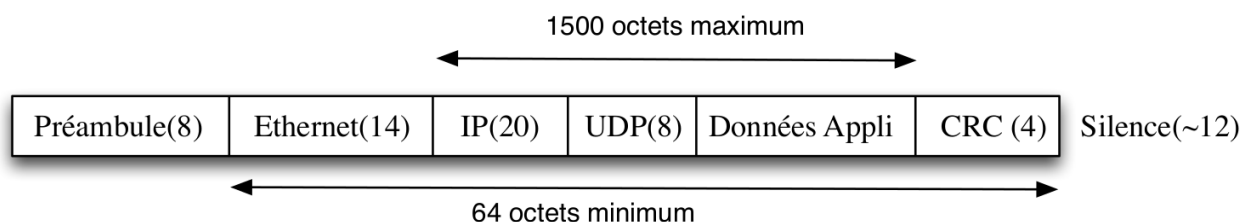
2.4,10 :

Je vais prendre des mesures réaliser pour le G10 qui a fait le TP en présentiel et donc sur des machines réelles plutôt que simulé.

Taille_données	kbit/s
10	1190
20	2342
100	6078
1000	11026
1472	11691
1473	10286
2800	11610
3000	10680

2.4,11 :

On admet que le débit physique est de 10 Mbit/s
Une trame typique UDP en réseau local :



Ainsi, on remarque que quelque soit la taille des données , les en-têtes prennent de la place : 8 pour le préambule , 14 (+ 4 de CRC) pour ETHERNET , 20 pour IP , 8 pour UDP. A cela, on peut rajouter 12 octets de silence inter-trame (pour un débit de 10Mbit/s (=1,25Mo/s)) . De plus, Ethernet

impose une taille minimum de 64 octets minimum donc on en déduit que la taille minimale d'une trame (avec son silence) est de $64 + 8 + 12 = 84$. Donc pour une taille des données comprise entre 0 et 18 ($64 - (14 + 20 + 8 + 4) = 18$), la trame fait la même taille et donc le même débit.

On pose :

$$\text{Débit_applicatif} / \text{Taille_totale} = \text{Débit_physique} / \text{Taille_données}$$

Donc :

$$\text{Débit_applicatif} = \text{Débit_physique} * \text{Taille_données} / \text{Taille_totale} (= \text{proportion utile de donnée})$$

Remarque :

- Pour des données allant de 0 à 18 octets, Taille totale ne bouge pas et reste à 84. Seulement Taille_données varie de 0 à 18 :

$$\text{Débit_applicatif} = \text{Débit_physique} * \text{Taille_données} / 84$$

- Pour des données allant de 19 à 1472 (le max pour 1 seul paquet), on a :

$$\text{Débit_applicatif} = \text{Débit_physique} * \text{Taille_données} / (66 + \text{Taille_données}) \quad (66 = \text{somme des en-têtes} + \text{CRC} + \text{silence})$$

Ensuite, à partir des multiples de 1472 + 1, 2 paquets sont envoyés : 1 plein avec 1472 octets et 1 avec 1 octets de donnée.

- Pour des données allant de 1473 à 1490 (, on a :

$$\text{Débit_applicatif} = \text{Débit_physique} * \text{Taille_données} / (1538 + 84) \quad (1538 = 1472 + \text{somme des en-têtes} + \text{CRC} + \text{silence})$$

- Pour des données allant de 1473 à 1490 (, on a :

$$\text{Débit_applicatif} = \text{Débit_physique} * \text{Taille_données} / (1538 + 66 + \text{Taille_données}) \quad (1538 = 1472 + \text{somme des en-têtes} + \text{CRC} + \text{silence})$$

Ainsi de suite...

On peut négliger le bourrage entre 0 et 18 et donner une formule général :

$$\text{Débit_applicatif} = \text{Débit_physique} * \text{Taille_données} / (\text{Taille_données} + (66 * ((\text{Taille_données} - 1) // 1472) + 1))$$

Le graphique suivante compare les débits expérimentaux avec les débits calculés par tranche (1 formule par cas cité au dessus (tient compte du silence))(courbe Jaune) et avec les débits théoriques calculés avec la formule générale (courbe rouge) :

