TD 1

Exercice 1 : Commutation de circuits vs commutation de paquets

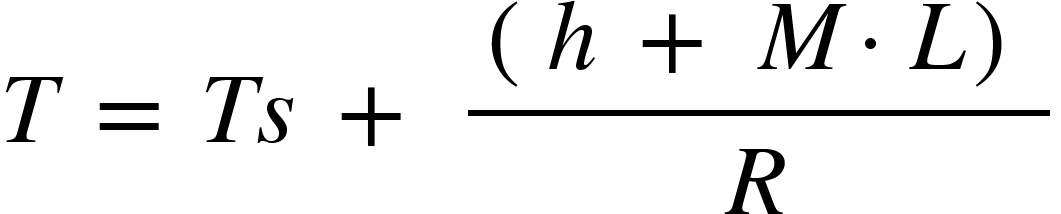
1)

**Mode datagramme par rapport à un réseau à commutation de circuits :**

* **Avantages**
* sans connexion : plus rapide à transmettre
* plus flexible / résilient en cas de panne
* plus simple à mettre en place (pas de gestion / réservation de ressources)
* plus efficace pour partager les ressources
* **Inconvénients**
* pas de qualité de service garantie (mode best effort)

2)

1. Temps pour un réseau à commutation de circuits



**h** : *en-tête protocolaire (header) pour l’ensemble des données*

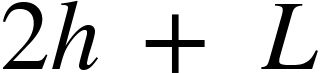
**M** : *nombre de paquets*

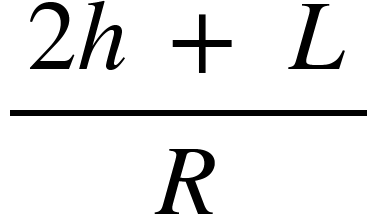
**L** : *taille d’un paquet*

**R** : *débit*

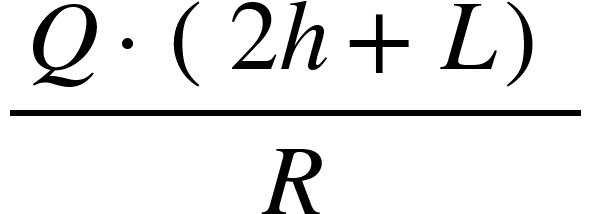
**Ts** : *temps d’établissement du circuit*

1. Temps pour un réseau à commutation de paquets mode datagramme

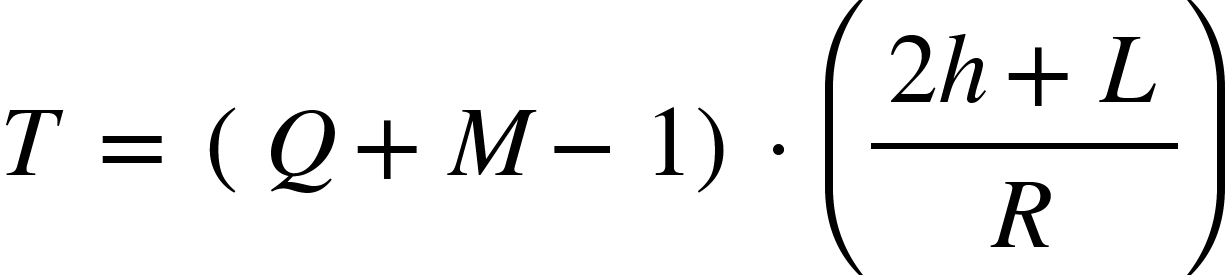
taille d’un paquet =

temps pour transmettre 1 paquet sur une liaison = 

temps pour émettre le premier des M paquets jusqu’au destinataire =



Chaque (2\*h + L) / R seconde un nouveau paquet est transmis (Store et forward)



**Q :** *nombre de liaisons*

**h** : *en-tête protocolaire (header) pour l’ensemble des données*

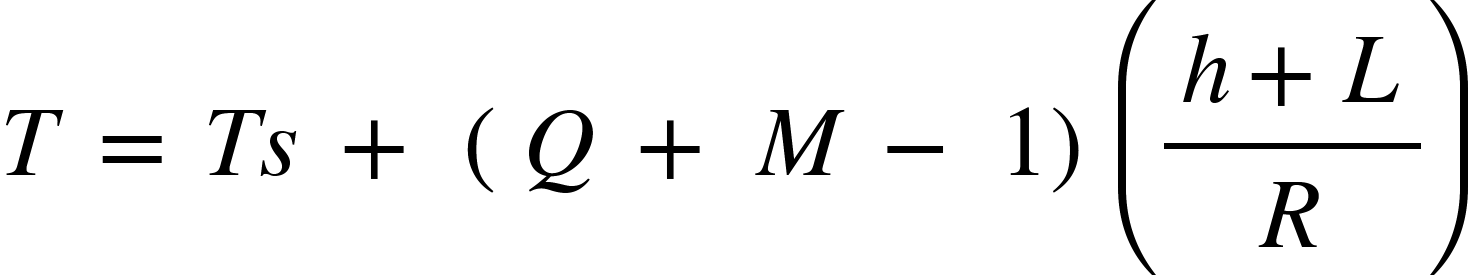
**M** : *nombre de paquets*

**L** : *taille d’un paquet*

**R** : *débit*

**Ts** : *temps d’établissement du circuit*

1. Temps pour un réseau à commutation de paquets avec des circuits virtuels (VC)

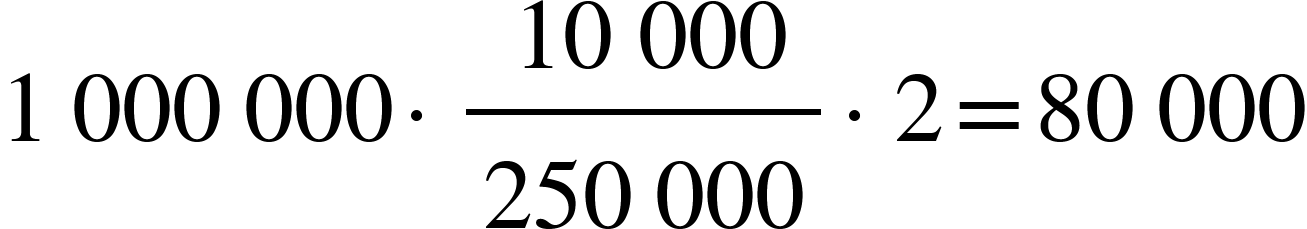


Exercice 2 : Calcul de débit et de délai

4 principaux délais :

* délais de transmission
* délais de propagation
* délais d’attente dans les buffers (files d’attente) ⇒ **variables selon les contraintes du réseau**
* délais de traitement du paquet ⇒ **variables selon les contraintes du réseau**

Temps de latence = 

Bdp (Bandwidth Delay Product) =  où 10 000/250 000 correspond au RTT (Road Trip Time)

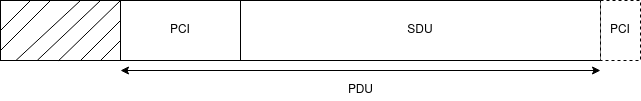
1. 233ms → 3ms + 200ms + 10ms + 20ms

1Kio (kibioctet) = 1024 octets (2^10)

1Ko (kilooctet) = 1000 octets (10^3)

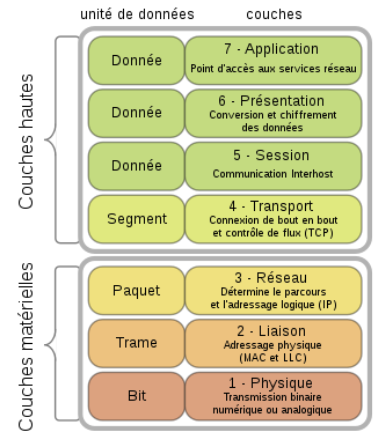
Exercice 3 :

PDU (Protocol Data Unit) pour une couche N, est composé de la SDU (Service Data Unit) et data PCI (Protocol Control Information)



Si n = dernière couche SDU = données de l’utilisateur

Sinon SDU = PDU de la dernière couche n+1



Données > Segments > Paquets > Trames > Bits

Les données de couches supérieurs (7,6,5) qui vont être intégrées dans des segments de couche 4 qui eux même sont incorporées dans des paquets de couche 3 dans des trames de couche 2 et des bits de couche 1

TCP/IP | Application > Transport > Internet > Accès Réseau

TCP/UDP IP/IPv6 Ethernet/Wifi

Exercice 4 :

L3 = couche ip

— Combien de liens physiques sont mis en œuvre dans ce réseau ?

3 liens (liaison radio, fibre optique, paire téléphonique)

— Toutes les entités L2 sont-elles homologues ?

Non, protocole L2 différents. Ils sont homologues 2 à 2 AB (10 octets) et CD (2 octets)

— Quelles sont, dans cette liste, les entités homologues ?

— L2 en A et L2 en D : NON

— L1 et L3 en B : NON (pas même niveau et au dessus de couches physiques différentes)

— L2 en A et L2 en B : OUI

— L3 en B et L3 en C : OUI

— IP en A, B, C, D : OUI

— UDP en A et UDP en D : OUI (obligée de fonctionner de bout en bout pour que cela fonctionne)

— L2 en B et L2 en C : NON

— Quelle est la taille, en nombre d’octets, du PDU dans les entités RTP ?

12(En Tête RTP) + 64(SDU) = 76

— Quelle est la taille, en nombre d’octets, du PDU sur la couche physique entre A et B ?

Somme de la colonne 64 (données applicatives énoncé) +82(A) = 146

— Quel est le rendement sur le lien A - B (en pourcentage exprimé) ?

Calculer la quantité donnée utile/quantité donnée transmise = 64/146\*100 = 43,84

— Quelle est la taille, en nombre d’octets, du PDU dans l’entité IP en B ?

12 + 8 + 20 + 64(Données applicatives) = 104

- Quelle est la taille, en nombre d’octets, du SDU de l’entité IP en C ?

12 + 8 + 64 = 84

- Quelle est la taille, en nombre d’octets, du PDU sur la couche physique entre C et D ?

Somme de D + données applicatives : 64 + 12 + 8 + 20 + 2 + 7 = 113

Exercice 5

Routeur : Couche réseau (3)

Concentrateur/Hub: Couche physique (1)

Commutateur/Switch: Couche liaison (2)

Pont/Bridge: Couche liaison (2)

Exercice 6

Adresse physique/ Adresse mac,

un hub concentrateur ne gère pas d’adresse

un routeur gère des adresses ip

Exercice 7

Encapsulation des segments en paquets, **Réseau**

Réception d’un flux de bits et formatage en une trame pour la couche supérieure : **Couche physique**

Contrôle de bout en bout : **Couche transport**

Détermination du chemin dans le réseau : **Couche réseau**

Exercice 8

Physique et liaison de donnée dans le modèle OSI

Exercice 9

Au niveau de la couche d’accès réseau : **Adresse physique/mac**

Au niveau de la couche réseau IP :  **adresse ip**

Au niveau de la couche transport TCP/UDP: **numéro de port**

Exercice 10

**Couche réseau,** permet de s’adapter au MTU (Maximum Transmission Unit) de la couche liaison

Exercice 11

Le protocole **FTP** pour les fichiers

Le protocole **SMTP** pour les mails

Exercice 12

**Domaine de collision :** communication de deux équipements en même temps => conflit

Le routeur au centre indique qu’il y deux réseaux (un de chaque côté) => 2 domaines de diffusion

Le hub compte pour 1 domaine de collision, 2 pour le premier switch et 4 pour le second => 7 domaines de collision

Exercice 13:

|  | **Mac S** | **Mac D** | **IP S** | **IP D** | **Port S** | **Port D** |
| --- | --- | --- | --- | --- | --- | --- |
| **R1** | 02:60:8c:5c:05.02 | 00:00:0C:00:0A:DD | 152.81.51.203 | 152.81.53.239 | Port aléatoire entre 1000 et 65535 | 21 |
| **R2** | 00:00:0c:00:10:BE | 00:00:0c:00:07:BA | // | // | // | // |
| **R3** | 00:00:0c:00:09:BD | @Mac B | // | // | // | // |

TD 2

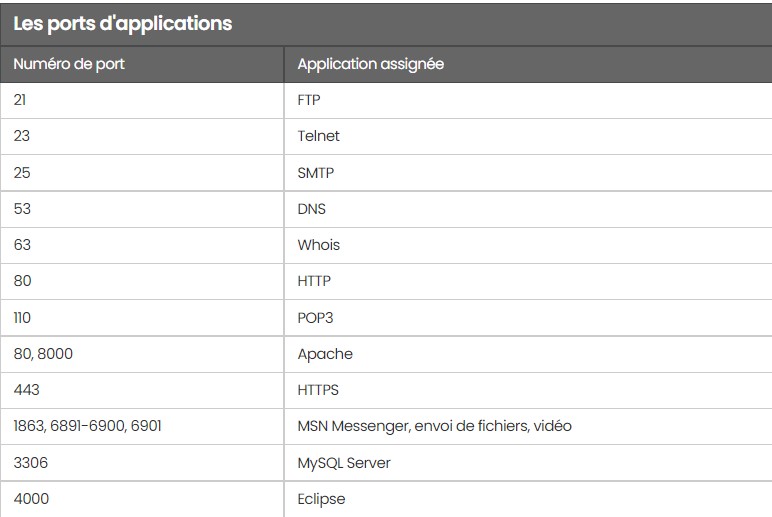
Exercice 1 :

1. C1 → S : Port source : X > 1024 - Port destination : 80
2. C2 → S : Port source : Y > 1024 - Port destination : 80
3. S → C1 : Port source : 80 - Port destination : X
4. S → C2 : Port source : 80 - Port destination : Y

System ports : 0 - 1023 (assigned by IANA)

User ports : 1024 - 49151

Dynamic ports : 49152 - 65535



Exercice 2 :

1. Vrai
2. Faux : une requête par ressource donc 3 requêtes
3. Vrai : même DNS (je crois à confirmer) cad ?

Exercice 3 :

**1er cas :**

Résolution DNS : RTT1 + RTT2 + … + RTTn

Établissement connexion TCP : RTT0

Envoi GET + réception HTML : RTT0

=> Somme de 1 à n (RTTi) + 2RTT0

**2e cas (subdivisé en plusieurs cas t’a capté) :**

**Connexion non persistante :**

Résolution DNS : RTT1 + RTT2 + … + RTTn

Établissement connexion TCP : RTT0

Envoi GET + réception HTML : RTT0

Pour chaque objet(x8) :

* Connexion TCP : RTT0
* GET : RTT0

Au total => Somme de 1 à n (RTTi) + 2RTT0 + 8\*2RTT0

**Connexion non persistante avec le navigateur configuré avec 5 connexions TCP en parallèle :**

Résolution DNS : RTT1 + RTT2 + … + RTTn

Établissement connexion TCP : RTT0

Envoi GET + réception HTML : RTT0

Connexion TCP x5 : RTT0

GET x5 : RTT0

Connexion TCP x3 : RTT0

GET x3 : RTT0

Au total => Somme de 1 à n (RTTi) + 6RTT0

**Connexion persistante sans pipelining :**

Résolution DNS : RTT1 + RTT2 + … + RTTn

Établissement connexion TCP : RTT0

Envoi GET + réception HTML : RTT0

Pour chaque objet(x8) : RTT0

Au total => Somme de 1 à n (RTTi) + 2RTT0 + 8RTT0

**Connexion persistante avec pipelining :**

Résolution DNS : RTT1 + RTT2 + … + RTTn

Établissement connexion TCP : RTT0

Envoi GET : RTT0

Pour tous les objets RTT0

Au total => Somme de 1 à n (RTTi) + 3RTT0

Exercice 4 :

1. DNS et UDP puis

HTTP et TCP

1. URL : http://www.univ-lorraine.fr/content/faire-dialoger-les-savoirs-cest-innover

Version : 1.1

Persistance : (oui) keep-alive

IP Serveur : idk : pas visible au niveau http, seulement le nom.

Type navigateur : Mozilla 5.0 sur MAC (beurk mac)

Exercice 5 :

DNS sur UDP ou HTTPS mais plutôt HTTPS car sécurisé

FTP

TCP

TFTP

UDP

port 21

port client >= 1024

RFC 959 (TCP)

<https://datatracker.ietf.org/doc/html/rfc959> 4.1 FTP Commands

login : USER

password PASS

**Mode actif : serveur établit la connexion**

port serveur 20

port client >=1024

problème si le client derrière NAT/firewall

**Mode passif : client établit la connexion**

port serveur >= 1024 (donné par le serveur commande PASSIV)

port client >= 1024

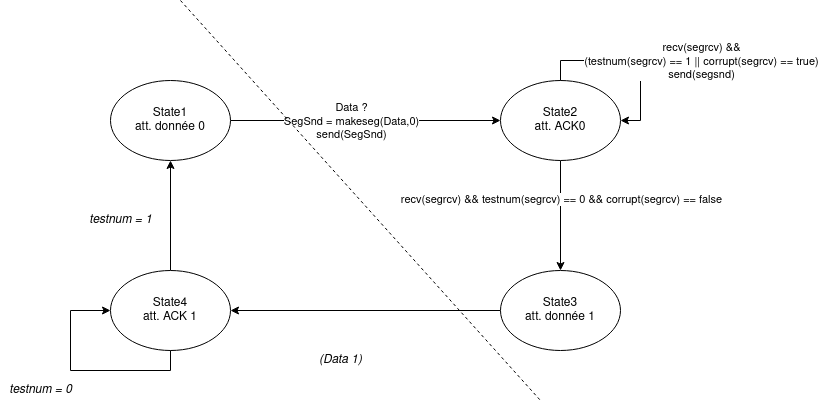
Exercice 6 :

TD 3

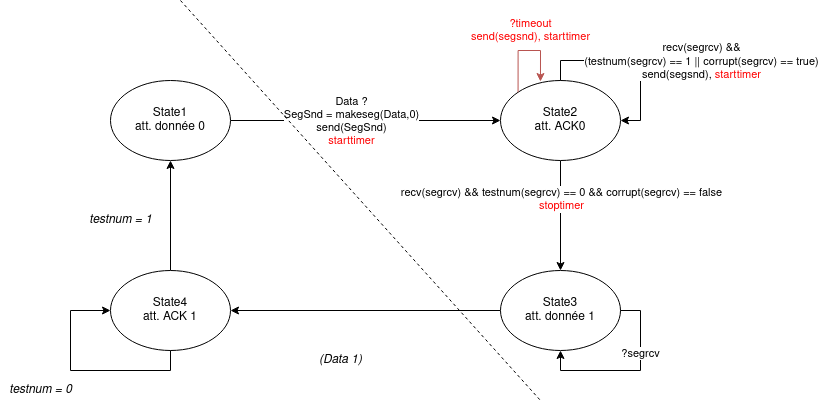
1. **Protocole de transport simple : STOP-AND-WAIT**

Exercice 1 :

Q2)



Q3)



Exercice 2 :

T = Ttrans + Tprop + Tack + T prop + Tproc

On néglige Tack, Tproc

T = Ttrans + 2xTprop = Ttrans + RTT (Round Trip Time)

U = Ttrans / Ttotal = Ttrans / (Ttrans + 2xTprop)

si a = Tprop / Ttrans, U = 1 / (1+2a)

2) application numérique liaison satellite : 4000 bit, 1 Mbit/s, 300 000 km/s

Distance Terre - Satellite : 36 000km

Ttrans = 4 000 / 1 000 000 = 1 / 250 = 4ms

Tprop = 2 x 36 000 / 300 000 = 240ms

a = 0,24\*230 = 60

U = 1 / 1 + 2\*60 = 0,00826 = 0,826 % (très inefficace, besoin d’utiliser un protocole à fenêtre)

3) application numérique coaxial : 4000 bit, 1 Mbit/s, 200 000 km/s

Distance : 10km

Tprop = 10 / 200 000 = 0,05 ms

Ttrans = 4 000 / 1 000 000 = 4ms

a = 0,0125

U = 1 / (1+0,025) = 97,5 %

Exercice 3

A FAUX

B VRAI

C VRAI

D FAUX

Exercice 4

a) Port d’origine : NON

b) Numéro de séquence : OUI (que TCP)

c) Taille des fenêtres : OUI (que TCP)

d) Données : NON

e) Port de destination : NON

f) Numéro d’accusé de réception : OUI (que TCP)

Exercice 5

1er numéro de séquence aléatoire puis numérotation séquentielle suivant les octets transmis dans le payload (charge utile)

TD 4

**Exercice 1**

152.81.50.31 : 1001 1000.0101 0001. 0011 0010.0001 1111

00101001 10000100 00110101 01010001 : 41.132.53.81

**Exercice 2**

152.81.5.4/20 => 152.81.0.0/20, 152.81.15.255

152.81.5.4 et netmask 255.255.192.0 => 152.81.0.0, 152.81.63.255

152.81.165.4/17 => 152.81.128.0/17, 152.81.255.255

**Exercice 3**

| Nb machines | Nb total adresses | Bloc adresses | Nb bit requis | Longueur préfixe | Masque réseau |
| --- | --- | --- | --- | --- | --- |
| 2 | 4 (2+2) | 4 | 2 | /30 | 255.255.255.252 |
| 5 | 7 (5+2) | 8 | 3 | /29 | .248 |
| 7 | 9 | 16 | 4 | /28 | .240 |
| 10 | 12 | 16 | 4 | /28 | .240 |
| 18 | etc | 32 | 5 | /27 | .224 |
| 31 |  | 64 | 6 | /26 | .192 |
| 80 |  | 128 | 7 | /25 | .128 |
| 140 |  | 256 | 8 | /24 | .0 |
| 502 |  | 512 | 9 | /23 | .254.0 |
| 1000 |  | 1024 | 10 | /22 | .252.0 |
| 1500 |  | 2048 | 11 | /21 | .248.0 |
| 2200 |  | 4096 | 12 | /20 | .240.0 |

**Exercice 4**

a)

192.164.**95**.5

192.164.**92**.10

192.164.**83**.1

=> 192.164.80.0/20

b)

=> 128.0.0.0/1

**Exercice 5**

Il rest 6 bits pour coder la partie hôte (/26)

=> nb adresses ip valides = 2⁶ - 2 = 62

Il reste 26 - 8 = 18 bits pour créer des sous réseaux => 2^18 réseaux possibles

(les adresses de sous réseaux “tous les bits à 0 ou à 1 peuvent être utilisées)

**Exercice 6**

192.1.7.0/24 ; adresse du 4ème hôte du 8ème sous-réseaux, étant donnée 12 hôtes par sous-réseaux.

Il faut 4 bits pour la partie hôte (car 2^4 = 16)

192.1.7.rrrr | hhhh

1er ss-res : 192.1.7.0 / 28

2eme ss-res : 192.1.7.16 / 28

3eme ss-res : 192.1.7.32/28

4eme ss-res : 192.1.7.48/28

[...]

8eme ss-res : 192.1.7.112/28

4eme hote : 192.1.7.116/28

**Exercice 7**

152.16.0.0/16

Nombre de sous-réseaux à créer : 314 (9 bits pour coder la partie réseau, il reste 7 bits (16-9) pour coder la partie hôte)

Adresse IP 3ème hôte et 4ème sous réseau ?

1er ss-res : 152.16.0.0/25

2eme ss-res : 152.16.0.128/25

3eme ss-res : 152.16.1.0/25

4eme ss-res : 152.16.1.128/25

3eme hôte : 152.16.1.131

**Exercice 8**

a)c) deja dans le /16

e) pas valdie premier octet dépasse

b)d) bonnes

**Exercice 9**

195.10.21.0 (classe c) /24

255.255.255.0

25 machines => 5 bits nécessaires

4 sous-réseaux => 3 bits nécessaires

195.10.21.rrrh hhhh /27

255.255.255.224

1er: 195.10.21.0 / 27 broadcast : 195.10.21.31

2eme: .32/27 .63

3eme: .64/27 .95

4eme: .96/27 .127

5eme: .128/27 .159

6eme: .160/27 .191

7eme: .192/27 .223

8eme: .224/27 .255

**Exercice 10**

* Extraire l’adresse IP de destination du paquet (DestAddr)
* Recherche l’adresse du réseau de destination = trouver dans la table de routage le réseau ayant le plus long préfixe commun avec DestAddr

**If** DestAddr appartient à réseau directement connecté **then**

* Envoyer le paquet IP vers la destination sur ce réseau

(Si destAdress n’est pas dans la table ARP, envoyer une requête ARP)

* Encapsuler le paquet dans une trame avec l’@ MAC du destinataire et l’envoyer sur l’interface où est connecté le sous réseau de destination

**Else if** DestAdrr appartient à un réseau autre que 0.0.0.0 **then**

* envoyer vers la passerelle (prochain routeur) indiquée dans la table de routage

(Si adresse MAC passerelle inconnue, faire résolution ARP)

**Else if** route par défaut (0.0.0.0) **then**

* Envoyer un paquet vers la passerelle par défaut

**Else**

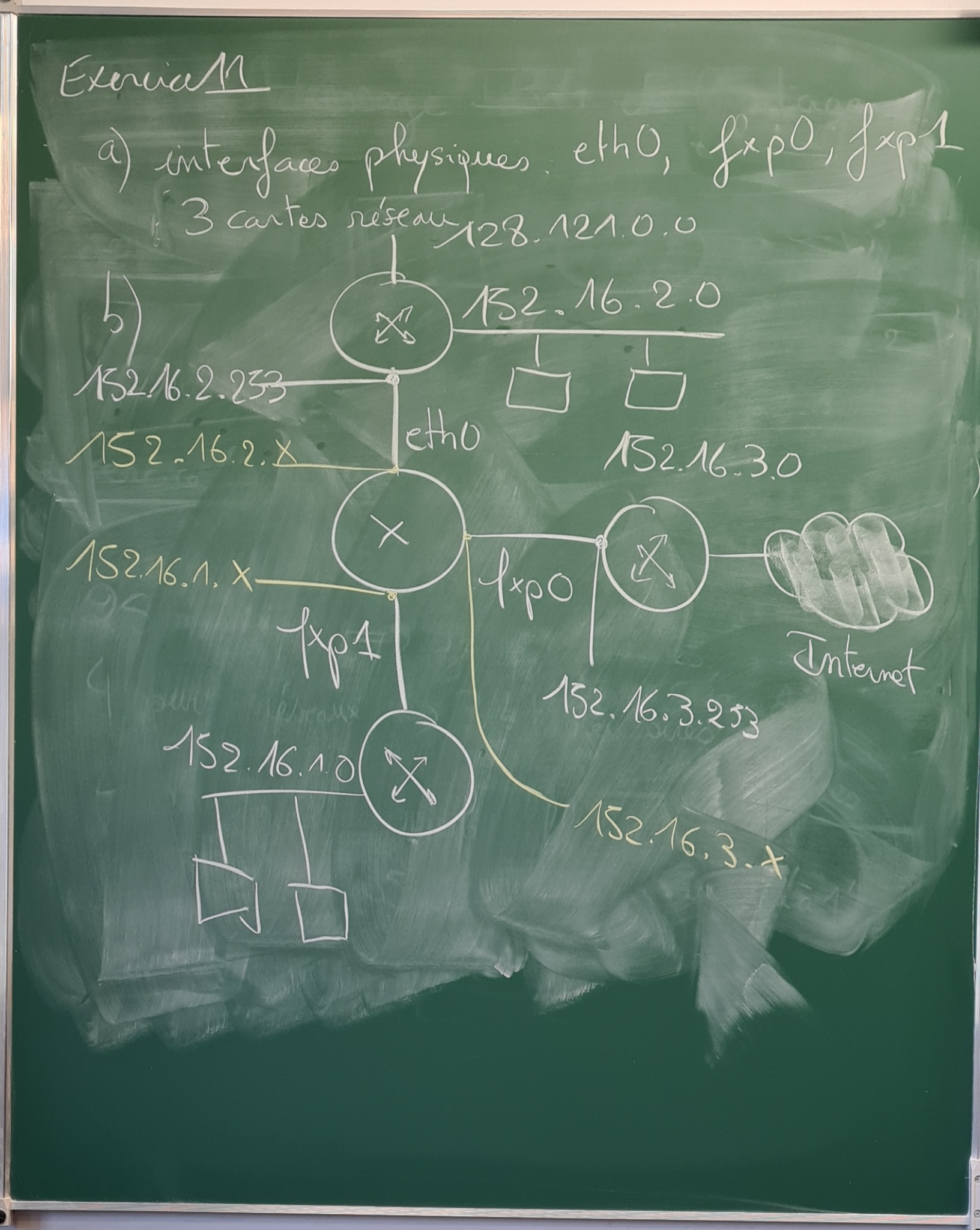
* jeter paquet

**Exercice 11**

1. interfaces physiques eth0, fxp0, fxp1 3 cartes réseau , 4 si on prend l’adresse de loopback

Une adresse IP par interface.

1. voir photo



Magnifique photo prise par Lolo 1

**Exercice 12**

****

Magnifique photo prise par Lolo 2

Adresse eth0-R2 : 195.10.21.62

|  | Destination | Gateway | Mask | Flags | Interface |
| --- | --- | --- | --- | --- | --- |
| R1 | 127.0.0.0 195.10.21.0  195.10.21.32  0.0.0.0 | 0.0.0.0  0.0.0.0  0.0.0.0  195.10.21.62 | 255.0.0.0  255.255.255.255  255.255.255.224  0.0.0.0 | U  U  U  UG | lo  eth0  eth1  eth1 |
| R2 | 127.0.0.0  195.10.21.0  195.10.21.32  195.10.21.96  0.0.0.0  195.10.21.64 | 0.0.0.0  195.10.21.61  0.0.0.0  0.0.0.0  195.10.21.126  0.0.0.0 | 255.0.0.0  255.255.255.224  255.255.255.224  255.255.255.224  0.0.0.0  255.255.255.224 | U  UG  U  U  UG  U | lo  eth0  eth0  eth1  eth1  eth2 |