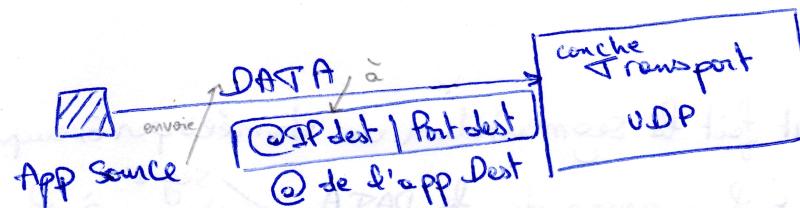


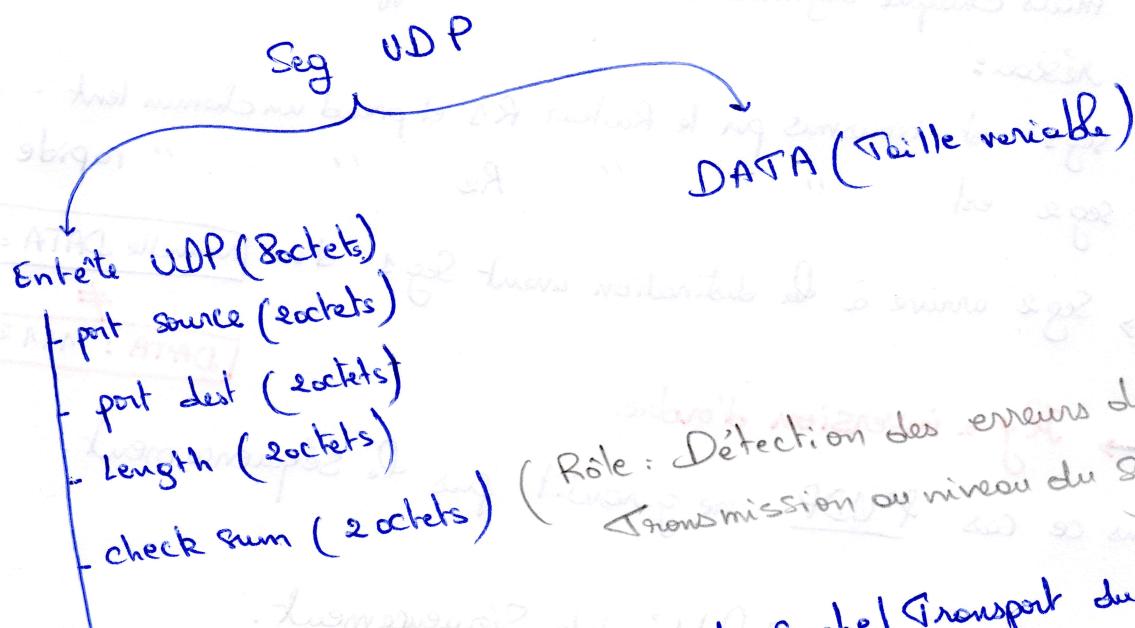
# Protocole UDP

- + La couche transport fait l'identification des applications par un numéro de port.

- + Elle utilise le protocole UDP pour faire communiquer des applications

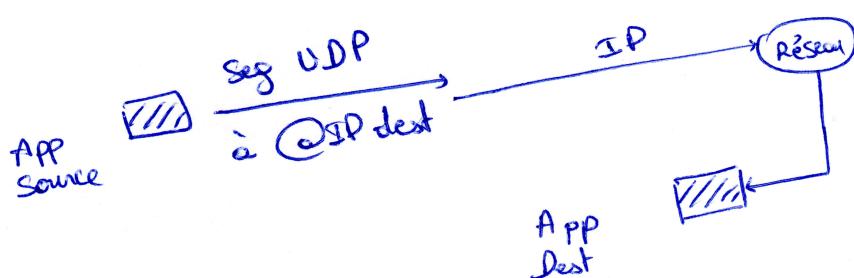


- + La couche Transport ajoute une En-tête (Encapsulation) qui est le segment



## IP dest ?

- : le segment à envoyer à la couche Transport du récepteur nécessite un passage par la couche IP.
- + IP a comme rôle livrer le segment UDP dans le réseau.



- ✓ L'UDP est un bon choix pour la couche Transport si on a une DATA de petite taille car il offre la rapidité.

## Problèmes du UDP.

### \* La fiabilité

Au niveau du réseau, si les routeurs sont chargés,

il y a risque que le DATA n'atteint pas la destination.

Il y a risque de perte de données.

### \* Le Séquencement

\* La couche Transport fait la segmentation des données par exemple

DATA :  $\boxed{111} \boxed{222}$  et il y a envoie de

seg<sup>1</sup> seg<sup>2</sup>

seg<sup>1</sup>  
et seg<sup>2</sup> à la source.

mais chaque segment suit une route différente de l'autre dans le même réseau:

Seg<sup>1</sup> est transmis par le Routeur R<sub>1</sub> et prend un chemin lent.

Seg<sup>2</sup> est " " R<sub>2</sub> " " rapide.

→ Seg<sup>2</sup> arrive à la destination avant Seg<sup>1</sup> d'où Nouvelle DATA = 222111  
DATA:  $\boxed{111} \boxed{222}$  !!!

→ Il y a inversion d'ordre

Dans ce cas l'UDP ne garantit pas le séquencement.

Réq:

Q TCP garantit la fiabilité et le séquencement.

Un autre protocole de la couche Transport



APAC une sono se dégarnit alors il n'y a pas mal de 90%

Stibzak il n'y a pas mal de 90% effectif dégarnit

# Protocole DHCP

## Client / Serveur

- Un service : un processus lancé en arrière plan et tourne en permanence  
(une application serveur)  
(pour un service Réseau)

de Rôle Traitement des requêtes reçues à partir d'une application et retourne une réponse à une app (App Client)  
(pour un service Réseau)

- Un service Réseau : Communication entre app serveur et app Client  
sur une machine M<sub>1</sub>      sur une machine M<sub>2</sub>

et on utilise le réseau pour la communication.

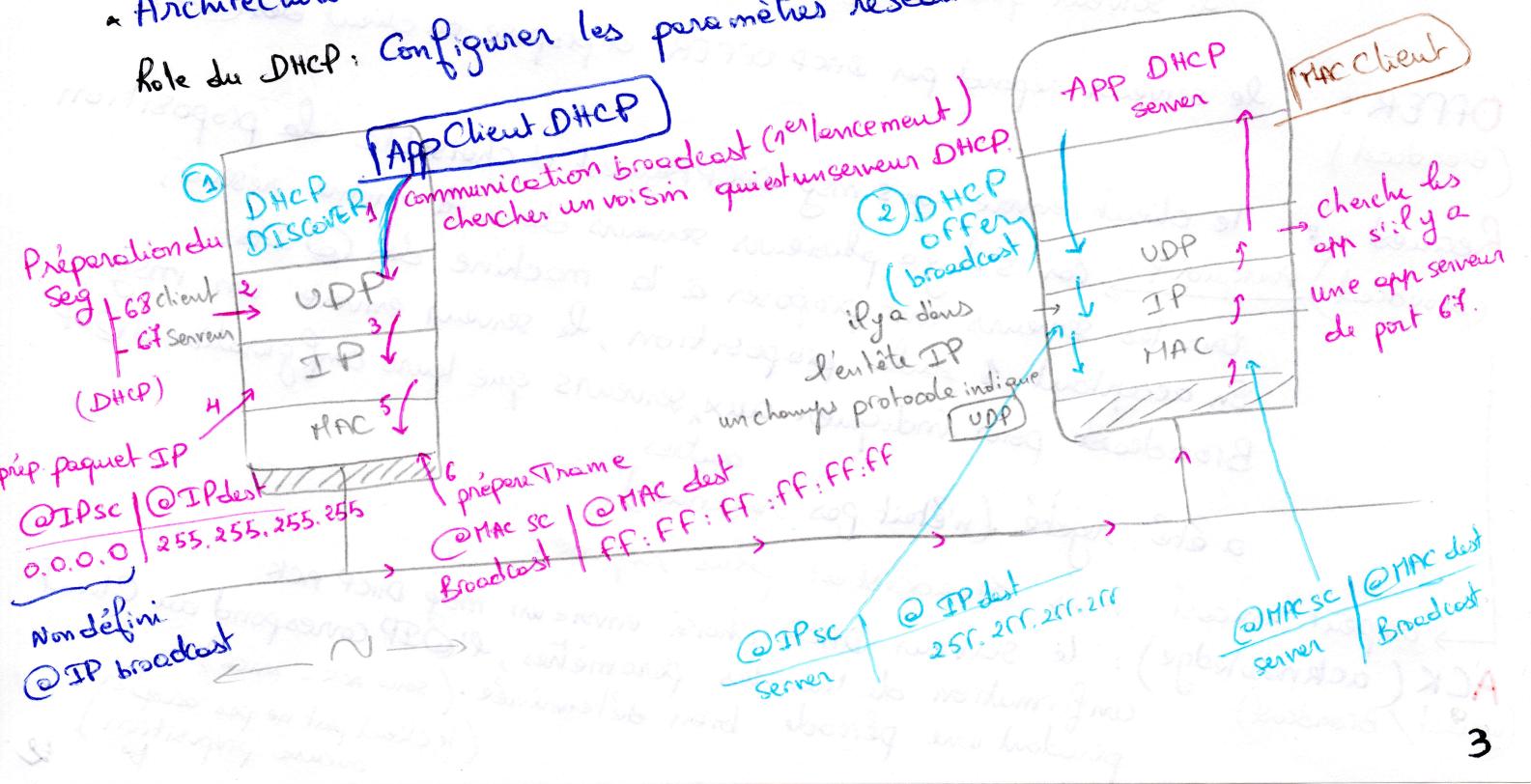
→ Donc il nous faut un protocole Applicatif (pour la couche application)  
un protocole de Communication qui est défini par

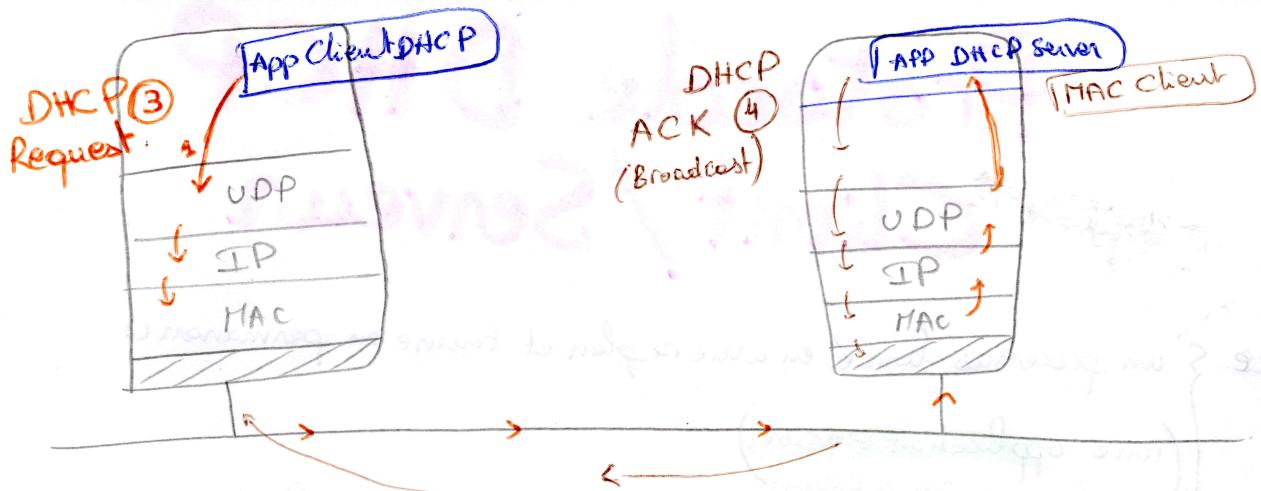
le séquencement des msg  
la structure des msg

## Protocole DHCP : Dynamic Host Configuration Protocol

Architecture de la Communication : Client / Serveur

Rôle du DHCP : Configurer les paramètres réseau d'une machine.





### MSG DHCP OFFER

partie fixe

champs Client MAC (seul le client peut le changer)

- Your IP (peut être changé seulement par le serveur)

partie variable  
options

server ID

Mask

Router (Gateway par défaut proposée)

DNS.

**DORA:** Processus d'attribution d'@IP via DHCP.

**Discover :** le client DHCP cherche un serveur DHCP dans le réseau (broadcast) et il envoie le msg DHCP DISCOVER avec l'@MAC du client (Ainsi si il y a plusieurs machine M1 et M2 qui cherchent une @IP le serveur peut distinguer entre M1 et M2).

**Offer :** le serveur répond par DHCP OFFER et propose au client une @IP. (Broadcast)

**Request :** le client envoie un msg DHCP Request et accepte la proposition (Broadcast) Pourquoi? car s'il y a plusieurs serveurs dans le même réseau, tous les serveurs vont proposer à la machine des @IPs. En acceptant 1 seule proposition, le client envoie un msg Broadcast pour indiquer aux serveurs que leurs configuration d'@IP a été rejeté (nouvelle retenue).

on peut unicast mais Broadcast est plus rapide.

**ACK (acknowledge) :** le serveur DHCP choisi envoie un msg DHCP ACK confirmation de tous les paramètres, l'@IP correspond au Client pendant une période bien déterminée. (sans ACK - NACK) (le client peut ne pas accepter aucune proposition)

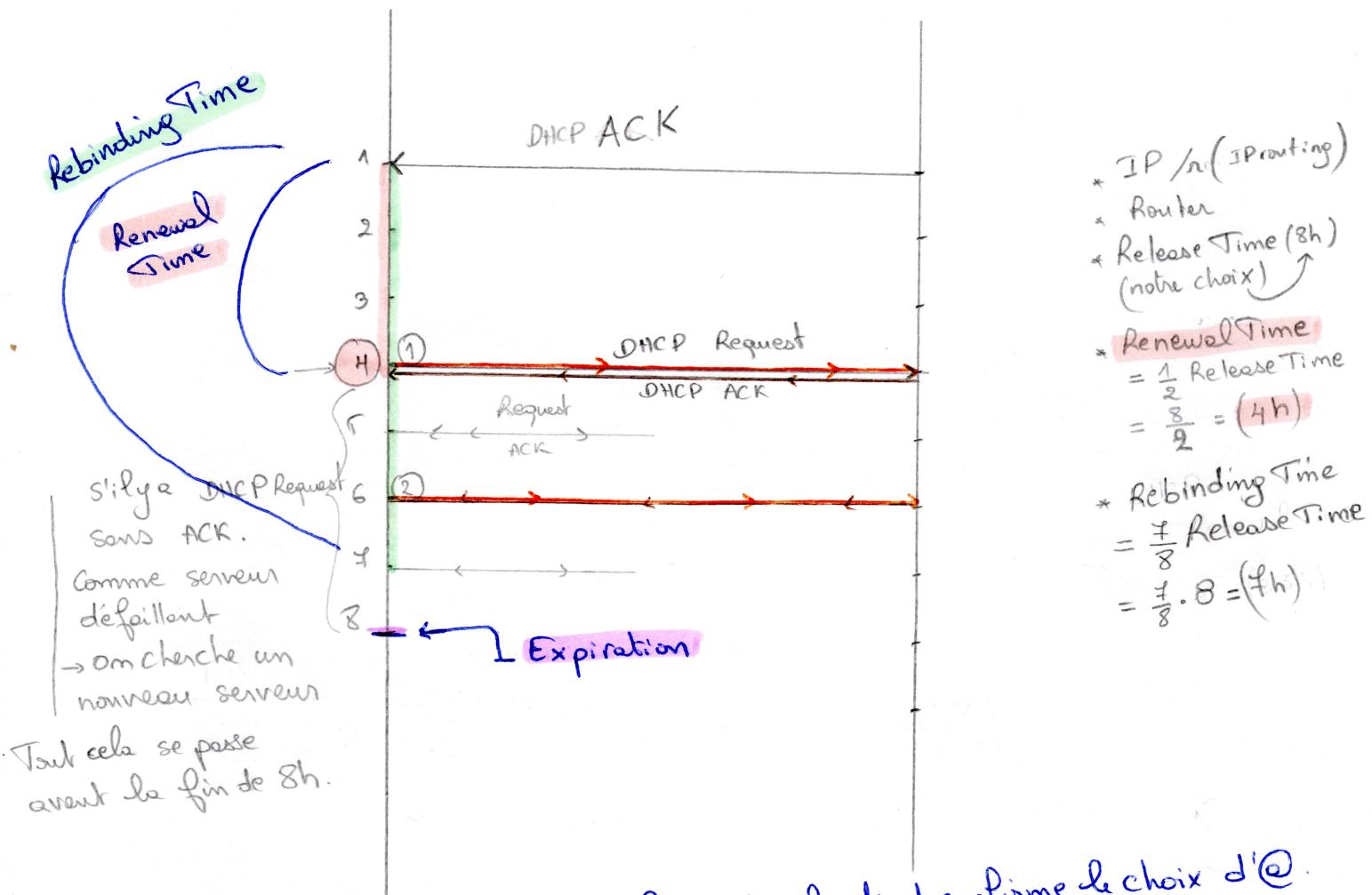
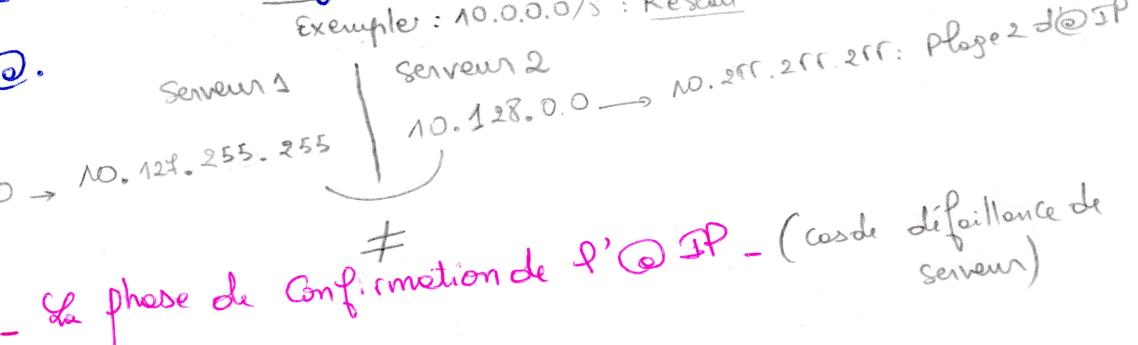
# Est ce qu'on peut avoir plusieurs Serveurs dans un Réseau ?

→ Oui, mais à condition qu'on définit à chaque serveur une plage d'adresses du même réseau.

On configure la plage d'@ dans un fichier de configuration spécifique à chaque serveur.

- Ces plages d'@ doivent être disjointes, sinon il peut y avoir un risque de conflit d'@.

Exemple : 10.0.0.0/8 : Réseau



- \* IP /n (IP routing)
- \* Router
- \* Release Time (8h) (notre choix)
- \* Renewal Time
 
$$= \frac{1}{2} \text{ Release Time}$$

$$= \frac{1}{2} \cdot 8 = (4h)$$

- \* Rebinding Time
 
$$= \frac{7}{8} \text{ Release Time}$$

$$= \frac{7}{8} \cdot 8 = (7h)$$

- \* Dans le cas d'un serveur normal à 4h le client confirme le choix d'@. mais en cas de défaillance il va chercher d'autres serveurs il considère la durée restante (4h → 8h) comme la nouvelle Release Time, divisé par 2 pour obtenir la nouvelle Renewal Time... et il expire à 8h s'il ne trouve aucun serveur qui lui permette de garder son @IP actuelle. (D'où il est obligé de renouveler son @IP).