



# R2.04 – LES BASES DES RÉSEAUX

**RESPONSABLE : CRISTINA ONETE**

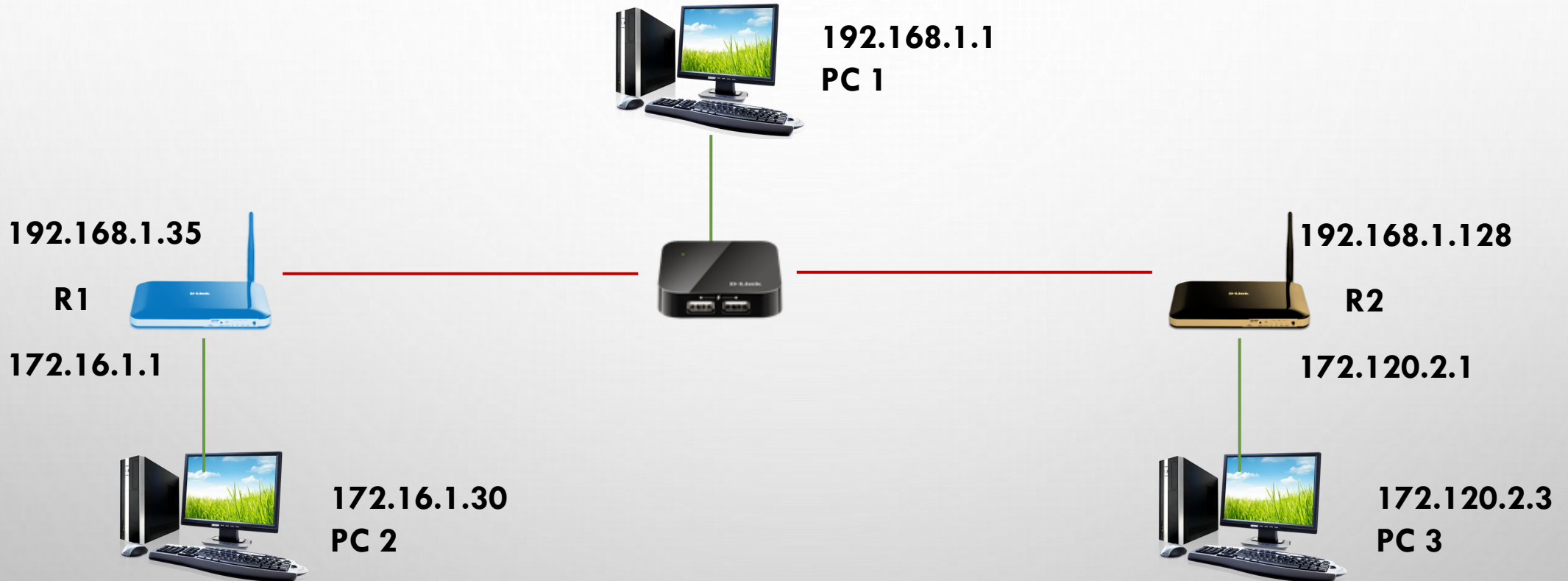
**MATERIEL : [HTTPS://WWW.ONETE.NET/TEACHING.HTML](https://www.onete.net/teaching.html)**

**EMAIL : [MARIA-CRISTINA.ONETE@UNILIM.FR](mailto:MARIA-CRISTINA.ONETE@UNILIM.FR)**



# COUCHE RÉSEAU : LE ROUTAGE

# PROBLÉMATIQUE -- LE ROUTAGE



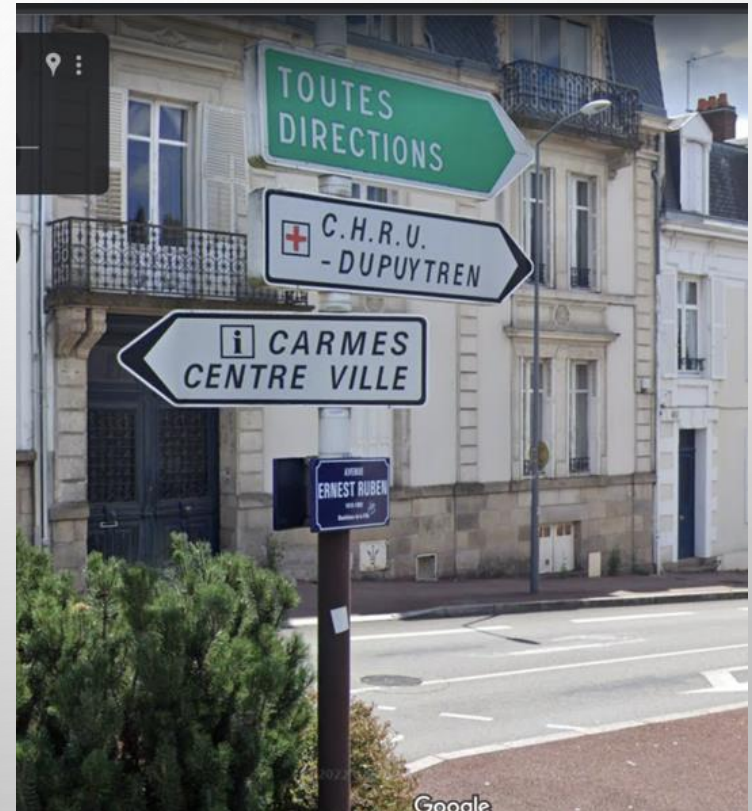
**Comment PC 1 peut-il envoyer un message à PC 2 ou PC 3 ?**

# LES TABLEAUX DE ROUTAGE



- **OBJECTIF** : Établir des chemins entre n'importe quels deux utilisateurs
- Configurés dans des nœuds de lien (routeurs)
- **ROUTAGE STATIQUE** : tableau de routage programmé à chaque nœud,
  - Écrit à la main par l'administrateur de réseau à chaque modification
  - Peut être optimisé, fonctionne bien seulement dans des petits réseaux
- **ROUTAGE DYNAMIQUE** : tableaux de routage qui s'adaptent dynamiquement
  - Les routeurs se parlent l'un à l'autre pour adapter leurs routes

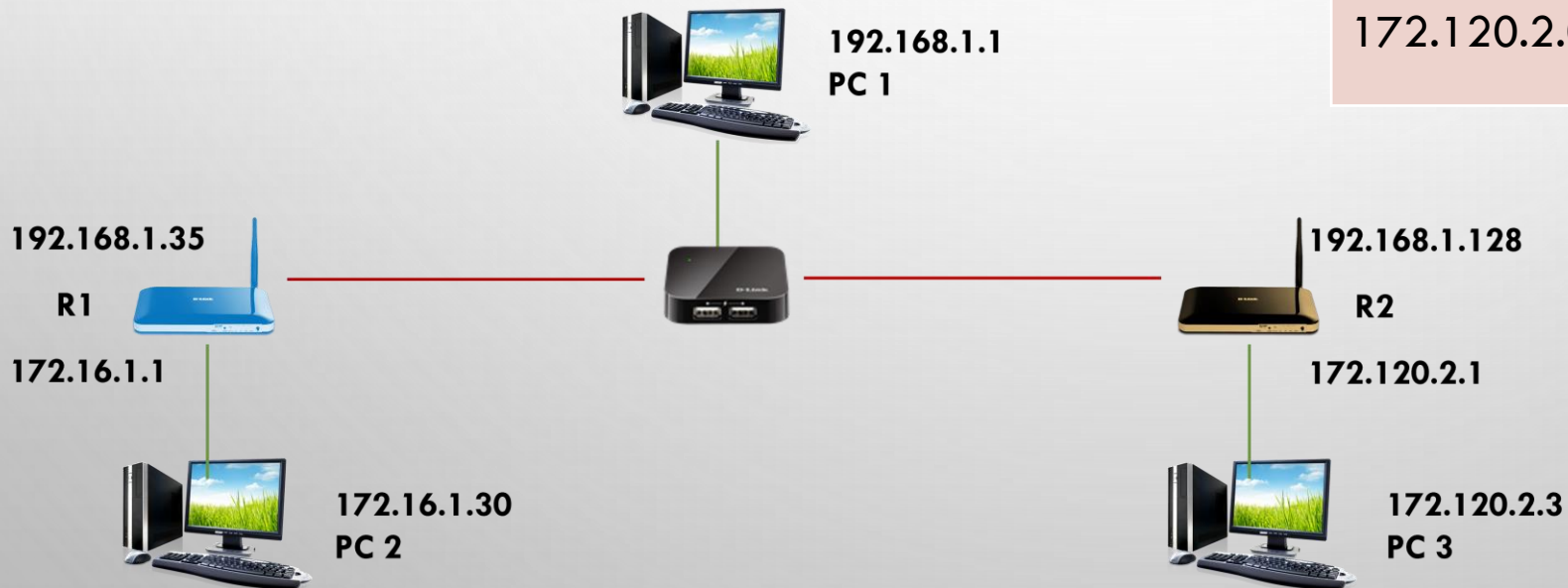
# LE ROUTAGE : UNE AUTRE VUE

- CIRCULATION : A CHAQUE POINT, UNE DIRECTION VERS LE PROCHAIN POINT
  - Au rondpoint 1 : prenez à droite
  - Plus tard : prenez à gauche
  - Au rondpoint 2 : tout droit
- ROUTAGE : CHAQUE ROUTEUR, PROCHAINE ÉTAPE
  - R1 : prochain routeur : R2
  - R2 : pour aller vers 192.168.1.0/24, R3  
pour tout autre chemin, R0
- ON N'INDIQUE JAMAIS TOUT LE CHEMIN



# ROUTAGE STATIQUE : SOLUTION 1

Destination	Router
192.168.1.0/24	none
172.16.1.0/24	192.168.1.35 
172.120.2.0/24	192.168.1.128 



# PROGRAMMER LE TABLE DE ROUTAGE

```
ip route <add ou del> <destination>/CIDR via <IP router>
```



**@dest. finale des packets**

Routage toujours nécessaire hors réseau, jamais dans le réseau

- EXEMPLE: `ip route add 172.168.1.0/24 via 192.168.1.35`

- POUR VOIR LES RÉSULTATS :

```
ip route list
```

Destination	Router
192.168.1.0/24	none
172.16.1.0/24	192.168.1.35 
172.120.2.0/24	192.168.1.128 

# UNE MEILLEURE SOLUTION

Solution précédente : OK mais non-optimale



Une meilleure solution serait de :

**Choisir routeur  
par défaut PCA**

**Ce routeur décide  
le routage plus  
loin**

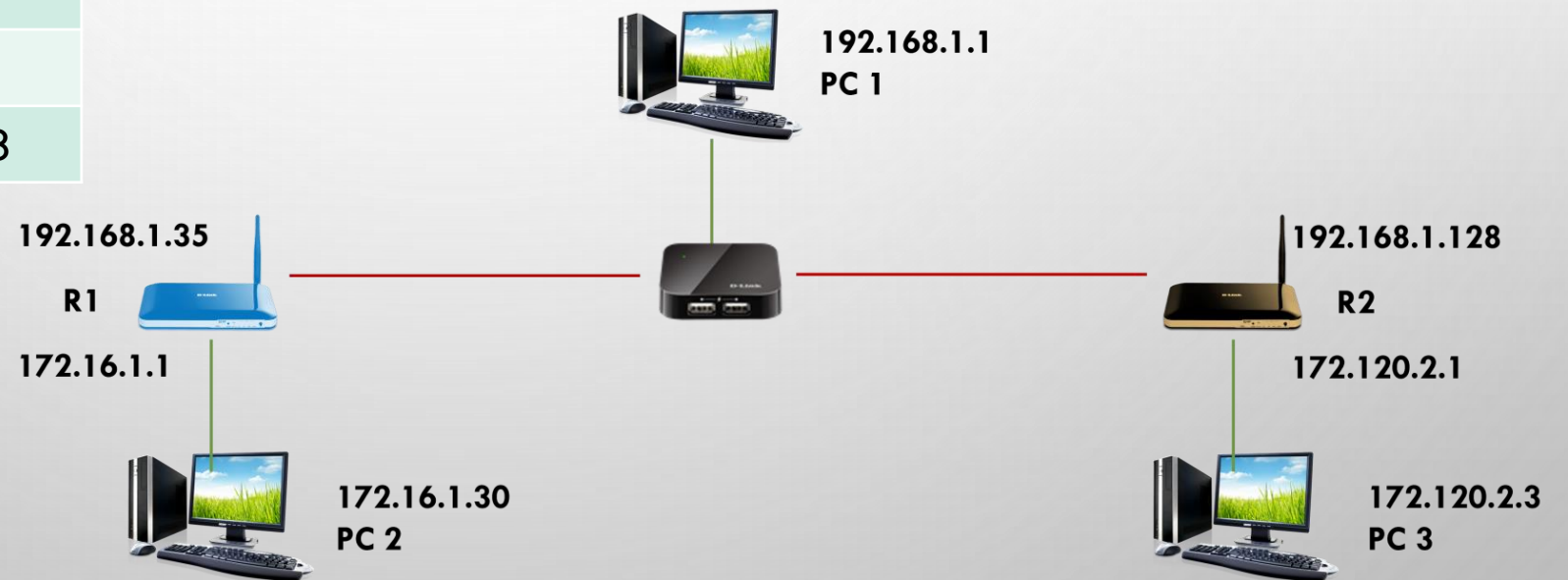
**PLUS FACILE :**  
routage plus facile  
à modifier

**PLUS FLEXIBLE :**  
routage  
modifiable selon  
besoins et routeurs



Destination	Router
192.168.1.0/24	none
172.16.1.0/24	none
172.120.2.0/24	192.168.1.128

Destination	Router
192.168.1.0/24	none
0.0.0.0/0	192.168.1.35



# IMPLEMMENTER CETTE SOLUTION

- POUR PC 1 : UN ROUTAGE PAR DÉFAUT

```
ip route add default via 192.168.1.35
```



- POUR ROUTEUR R1 :

```
ip route add 172.120.2.0/24 via 192.168.1.128
```

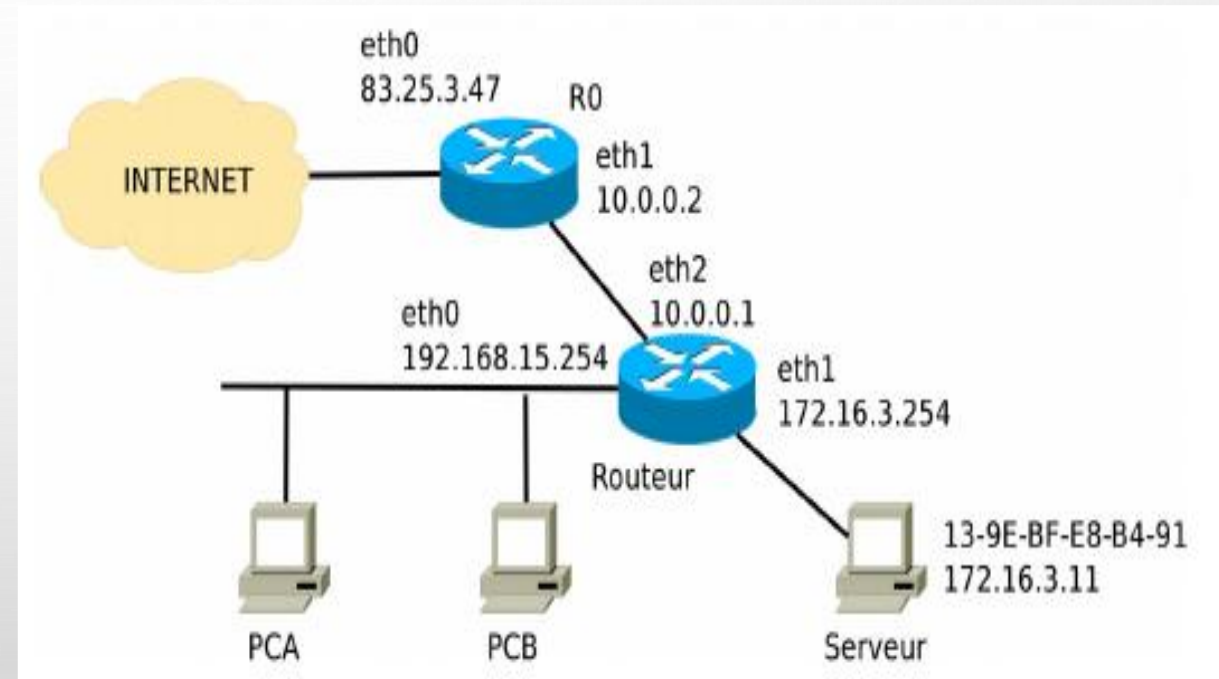


Destination	Router
192.168.1.0/24	none
0.0.0.0/0	192.168.1.35

Destination	Router
192.168.1.0/24	none
172.16.1.0/24	none
172.120.2.0/24	192.168.1.128

# ROUTE PAR DÉFAUT VS ROUTE SPÉCIFIQUE

- ROUTE PAR DÉFAUT :
  - Typique machines utilisateurs (PCA, PCB, Serveur)
  - Peut indiquer une route "principale" pour un routeur (ex. la route vers l'Internet)
- ROUTE SPÉCIFIQUE :
  - En général utilisée pour les routeurs, pour indiquer une route secondaire

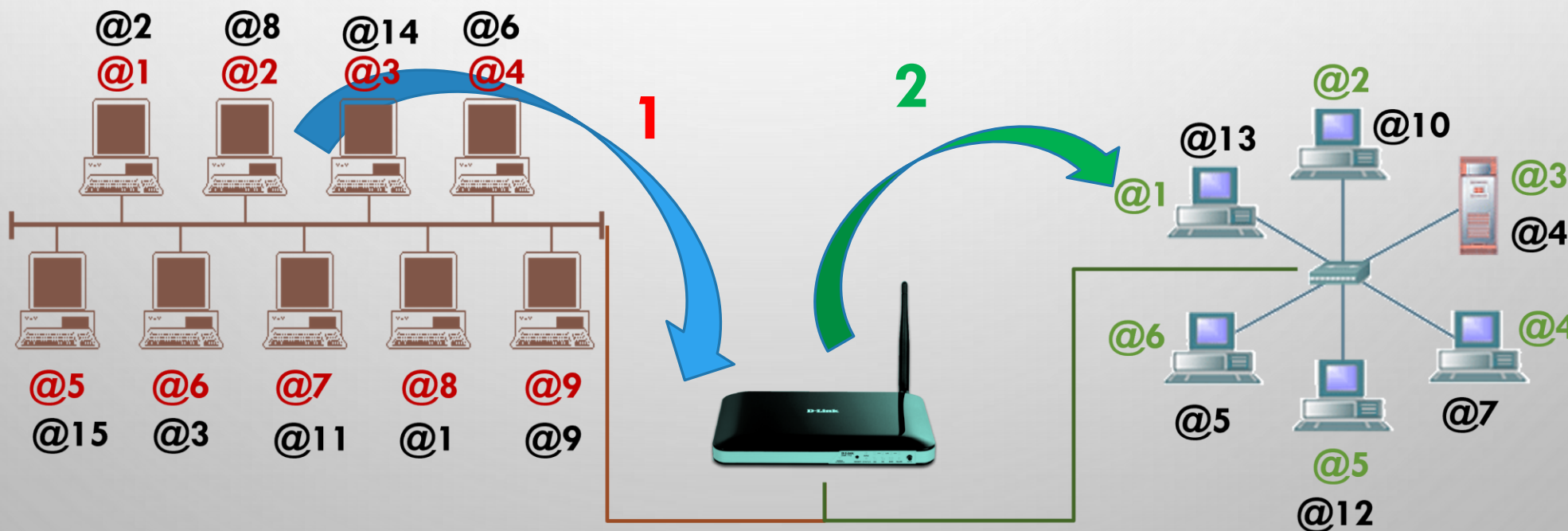




# COUCHE RÉSEAU : ENVOYER UN PACKET

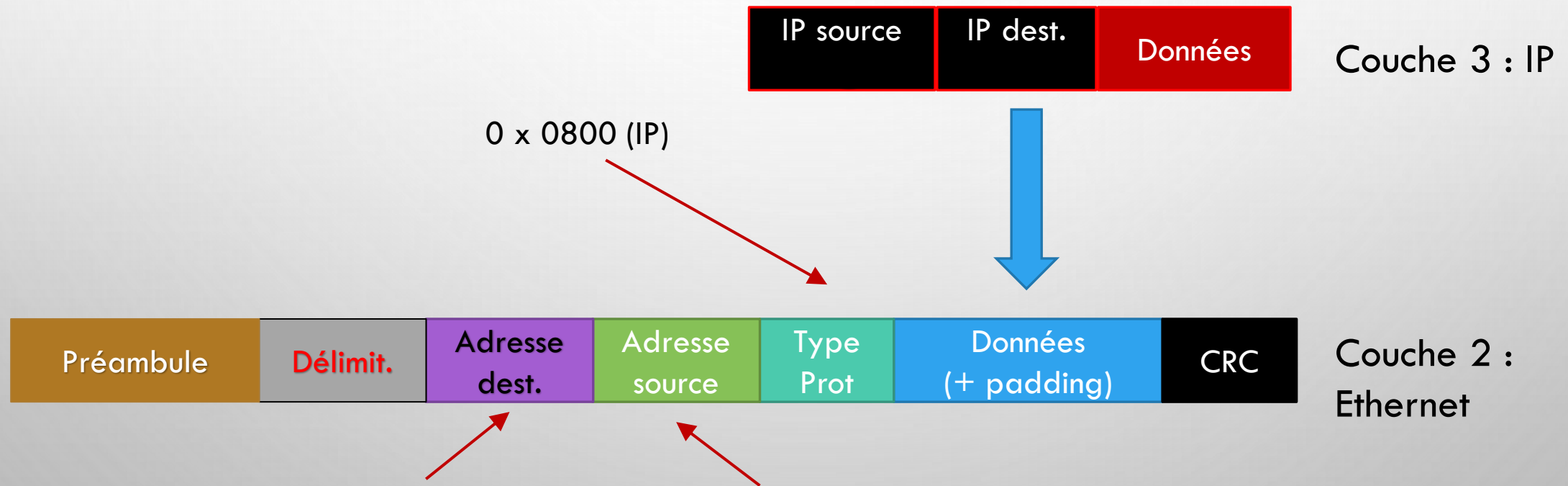
# ENVOI INTER-RÉSEAU : 2 ÉTAPES

- ▶ PROBLÉMATIQUE : comment @2 (@8) peut-il envoyer un message à @6 (@5) ?



# TRANSMISSION INTER-RÉSEAUX

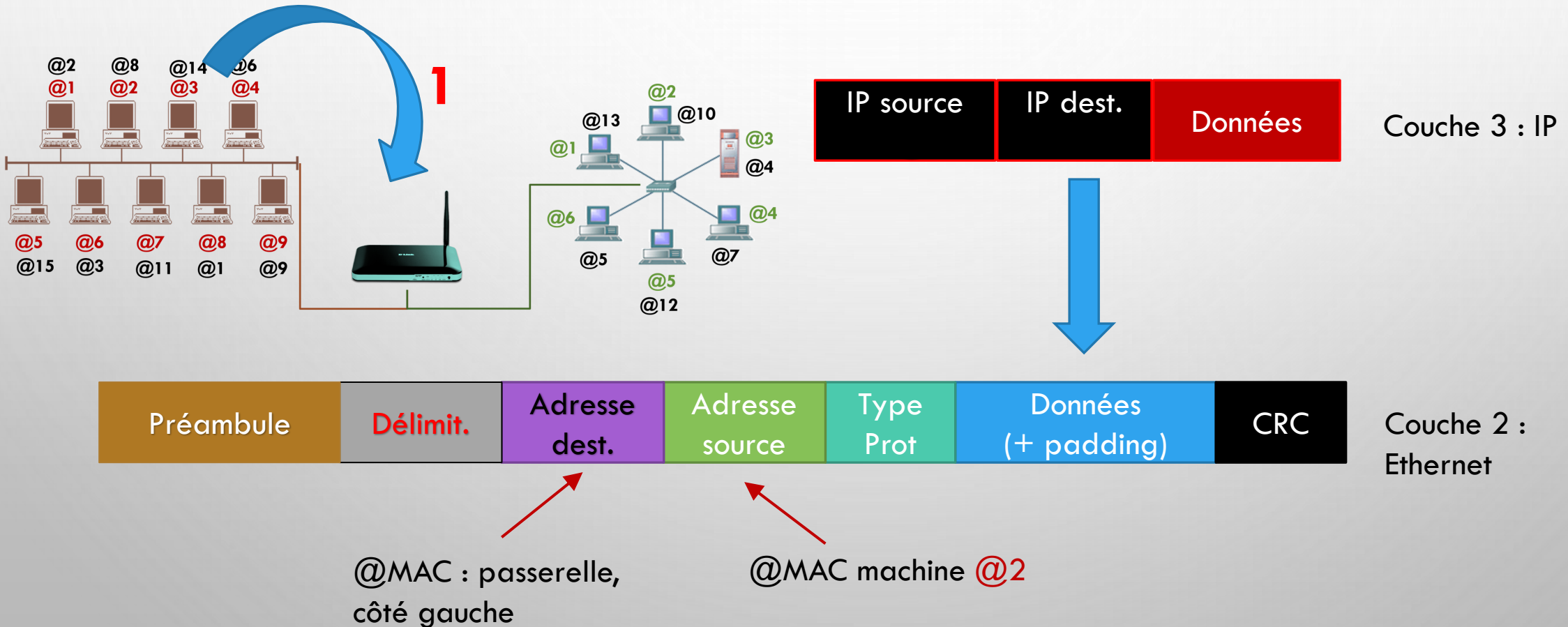
- DEUX ÉTAPES DE TRANSMISSION INTRA-RÉSEAU
- ENCAPSULATION : >> IP >> ETHERNET 2



Adresses différentes pour les 2 étapes de transmission

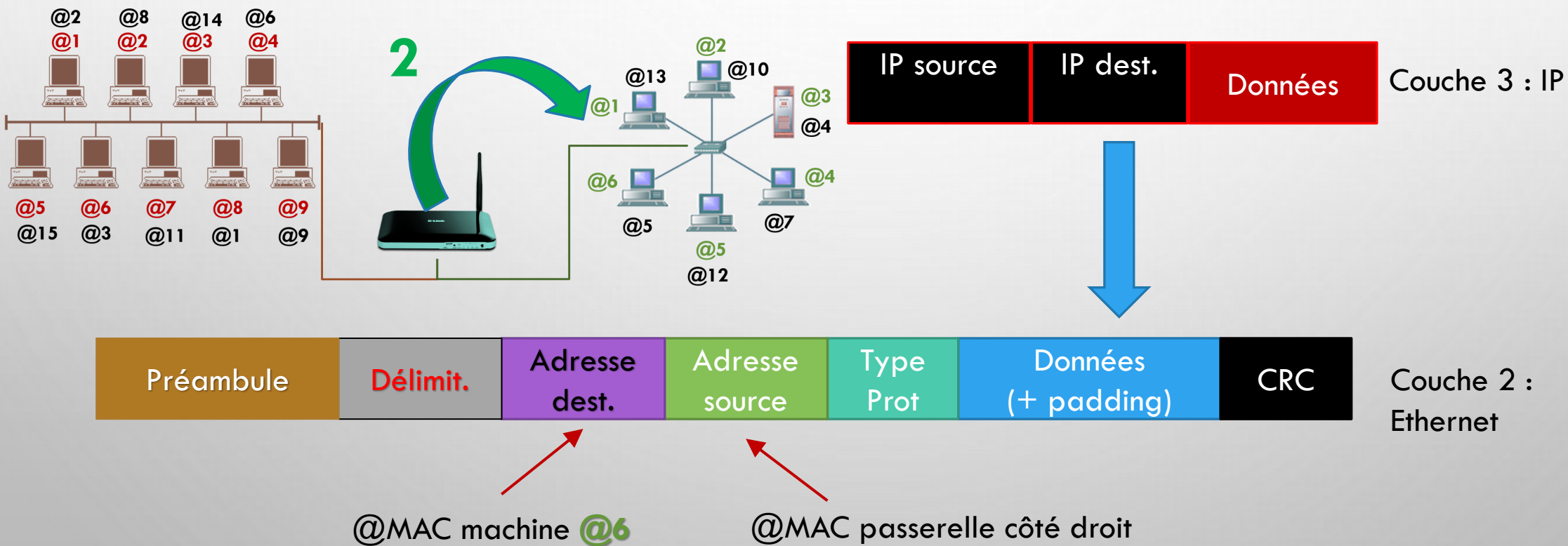
# TRANSMISSION INTER-RÉSEAUX

- PREMIÈRE ÉTAPE DE TRANSMISSION : @IP = @8, @MAC = @2 → PASSERELLE



# TRANSMISSION INTER-RÉSEAUX

- DEUXIÈME ÉTAPE : PASSERELLE → @IP = @5, @MAC = @6





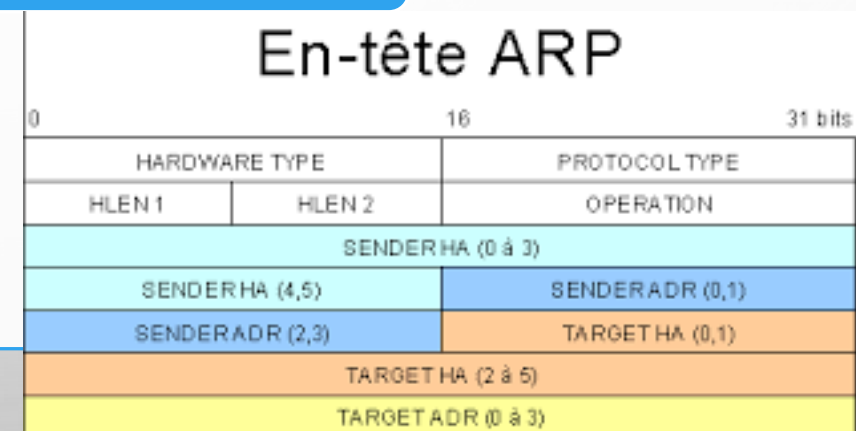
# ASSOCIATION ADRESSES MAC/IP

Comment savoir quelle adresse MAC correspond à quelle adresse IP ?

- Protocole ARP, couche 2
- Ne peut s'exécuter que dans un seul réseau (DC)
- Exécuté par une machine cherchant une @ MAC à partir d'une @IP et une machine qui connaît l'@MAC cherchée

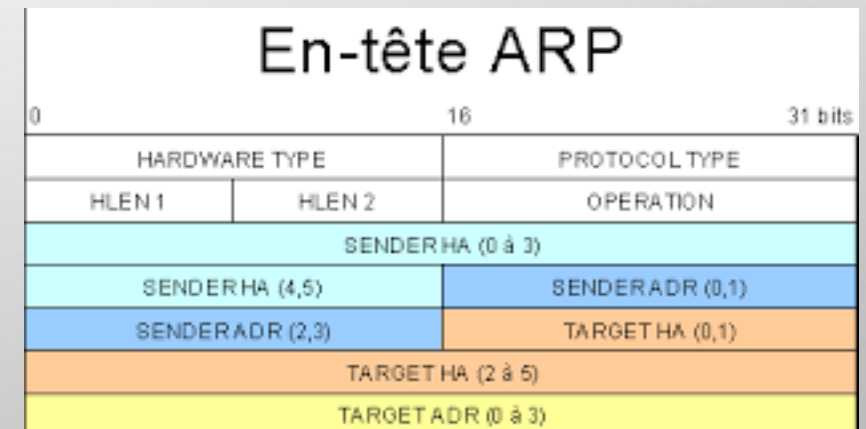
ARP : 2 types de messages : requête, réponse, encapsulées en Ethernet

- En-tête spécifique ARP
- Chaque message ARP spécifie :
  - Machine src : @MAC, @IP
  - Machine dst : @MAC
  - Machine "cible" : @MAC, @IP



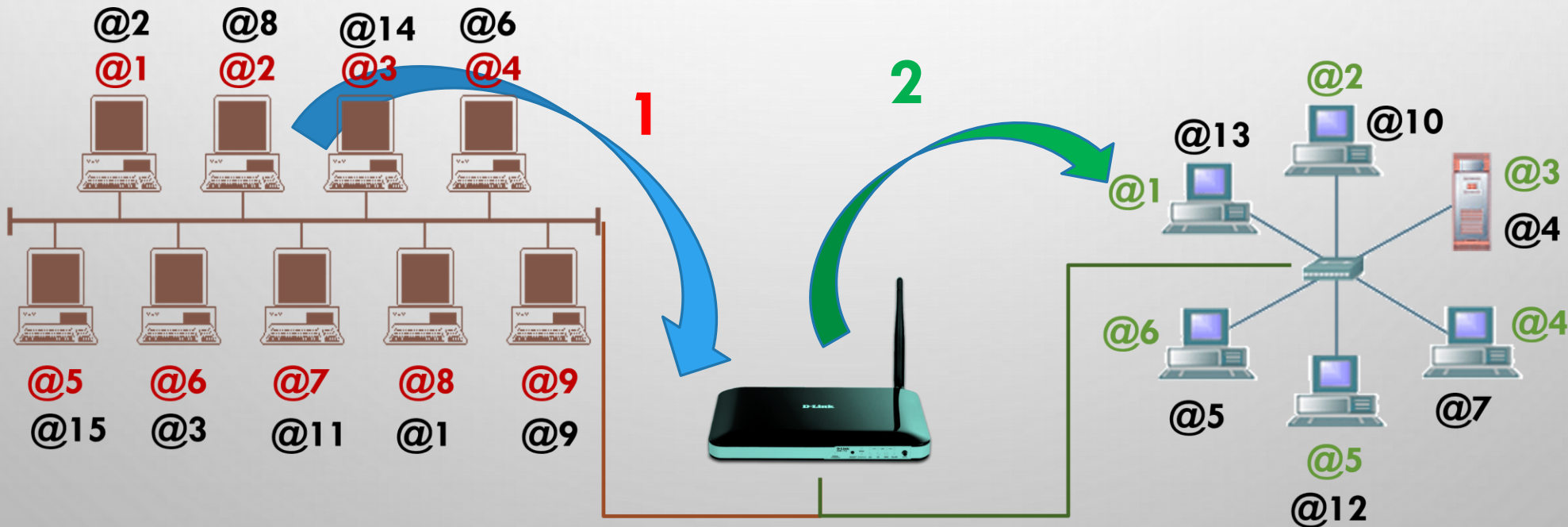
# ASSOCIATION ADRESSES MAC/IP

- ▶ Situation : Machine A (@IP  $ip_A$  @MAC  $mac_A$ ) cherche @MAC de la machine B ( $ip_B$ )
  - ▶ Machine C ( $ip_C, mac_C$ ) connaît la réponse
- ▶ ARP requête :
  - ❖ Machine src : MAC  $mac_A$ , IP  $ip_A$
  - ❖ Machine dst : MAC broadcast : FF:FF:FF:FF:FF:FF : broadcast intra-réseau !
  - ❖ Machine cible : @MAC inconnue 00:00:00:00:00:00
- ▶ ARP réponse :
  - ❖ Machine src : MAC  $mac_C$  , IP  $ip_C$
  - ❖ Machine dst : MAC  $mac_A$
  - ❖ Machine cible : @MAC  $mac_B$  , IP  $ip_B$



# ICMP : DÉBOGAGE À LA COUCHE 3

- SUPPOSONS UNE ERREUR DE TRANSMISSION À L'ÉTAPE 2
- UN MESSAGE D'ERREUR EST ENVOYÉ PAR LA PASSERELLE À L'EXPÉDITEUR (MESSAGE ICMP)



# UN MESSAGE ICMP

```
Frame 224: 102 bytes on wire (816 bits), 102 bytes captured (816 bits)
Ethernet II, Src: AsustekC_b2:d0:11 (54:04:a6:b2:d0:11), Dst: CadmusCo_76:53:17 (08:00:27:76:53:17)
  Destination: CadmusCo_76:53:17 (08:00:27:76:53:17)
  Source: AsustekC_b2:d0:11 (54:04:a6:b2:d0:11)
  Type: IP (0x0800)
Internet Protocol Version 4, Src: 172.16.2.232 (172.16.2.232), Dst: 172.16.2.234 (172.16.2.234)
  Version: 4
  Header length: 20 bytes
  Differentiated Services Field: 0xc0 (DSCP 0x30: Class Selector 6; ECN: 0x00: Not-ECT (Not ECN-CE))
  Total Length: 88
  Identification: 0xe491 (58513)
  Flags: 0x00
  Fragment offset: 0
  Time to live: 64
  Protocol: ICMP (1)
  Header checksum: 0x3761 [correct]
  Source: 172.16.2.232 (172.16.2.232)
  Destination: 172.16.2.234 (172.16.2.234)
  [Source GeoIP: Unknown]
  [Destination GeoIP: Unknown]
Internet Control Message Protocol
  Type: (Destination unreachable)
  Code: 0 (Network unreachable)
  Checksum: 0xfcff [correct]
Internet Protocol Version 4, Src: 172.16.2.234 (172.16.2.234), Dst: 172.20.1.1 (172.20.1.1)
Internet Control Message Protocol
  Type: 8 (Echo (ping) request)
  Code: 0
  Checksum: 0xc25b
```

▶ Source/destination du message ICMP

▶ Source/destination du message encapsulé par ICMP

**Pourquoi une différence ?**

# UN MESSAGE ICMP

```
Frame 224: 102 bytes on wire (816 bits), 102 bytes captured (816 bits) on interface 0
Ethernet II, Src: AsustekC_b2:d0:11 (54:04:a6:b2:d0:11), Dst: CadmusCo_76:53:17 (08:00:27:76:53:17)
  Destination: CadmusCo_76:53:17 (08:00:27:76:53:17)
  Source: AsustekC_b2:d0:11 (54:04:a6:b2:d0:11)
  Type: IP (0x0800)
Internet Protocol Version 4, Src: 172.16.2.232 (172.16.2.232), Dst: 172.16.2.234 (172.16.2.234)
  Version: 4
  Header length: 20 bytes
  Differentiated Services Field: 0xc0 (DSCP 0x30: Class Selector 6; ECN: 0x00: Not-ECT (Not ECN))
  Total Length: 88
  Identification: 0xe491 (58513)
  Flags: 0x00
  Fragment offset: 0
  Time to live: 64
  Protocol: ICMP (1)
  Header checksum: 0x3761 [correct]
  Source: 172.16.2.232 (172.16.2.232)
  Destination: 172.16.2.234 (172.16.2.234)
  [Source GeoIP: Unknown]
  [Destination GeoIP: Unknown]
Internet Control Message Protocol
  Type: 3 (Destination unreachable)
  Code: 0 (Network unreachable)
  Checksum: 0xfcff [correct]
Internet Protocol Version 4, Src: 172.16.2.234 (172.16.2.234), Dst: 172.20.1.1 (172.20.1.1)
Internet Control Message Protocol
  Type: 8 (Echo (ping) request)
  Code: 0
  Checksum: 0xc25b
```

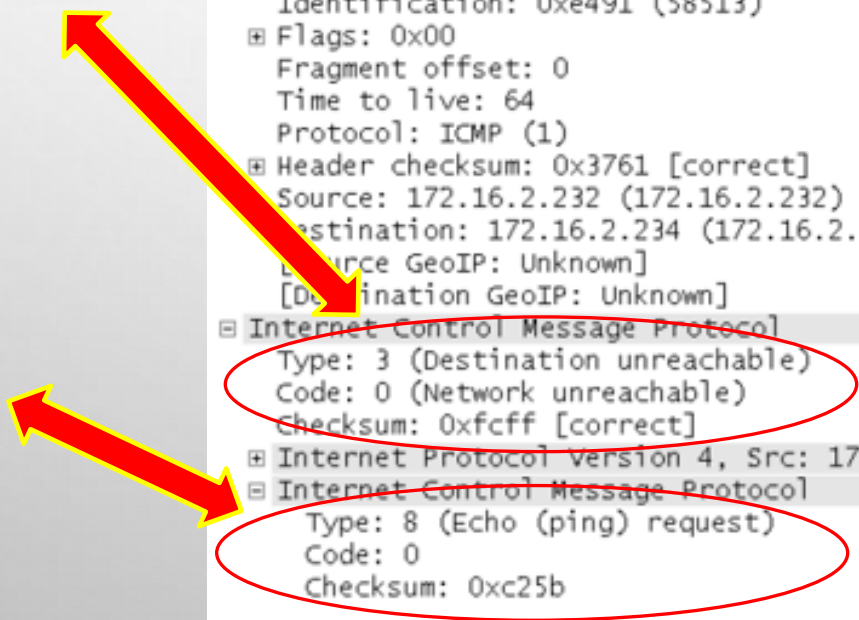
- ▶ 172.16.2.234 veut envoyer un message (ping) à 172.20.1.1
- ▶ 172.20.1.1 est dans un autre réseau
- ▶ 172.16.2.232 est alors une passerelle

# UN MESSAGE ICMP

```
Frame 224: 102 bytes on wire (816 bits), 102 bytes captured (816 bits)
Ethernet II, Src: AsustekC_b2:d0:11 (54:04:a6:b2:d0:11), Dst: CadmusCo_76:53:17 (08:00:27:76:53:17)
  Destination: CadmusCo_76:53:17 (08:00:27:76:53:17)
  Source: AsustekC_b2:d0:11 (54:04:a6:b2:d0:11)
  Type: IP (0x0800)
Internet Protocol Version 4, Src: 172.16.2.232 (172.16.2.232), Dst: 172.16.2.234 (172.16.2.234)
  Version: 4
  Header length: 20 bytes
  Differentiated Services Field: 0xc0 (DSCP 0x30: Class Selector 6; ECN: 0x00: Not-ECT (Not ECN-CE))
  Total Length: 88
  Identification: 0xe491 (58513)
  Flags: 0x00
  Fragment offset: 0
  Time to live: 64
  Protocol: ICMP (1)
  Header checksum: 0x3761 [correct]
  Source: 172.16.2.232 (172.16.2.232)
  Destination: 172.16.2.234 (172.16.2.234)
  [Source GeoIP: Unknown]
  [Destination GeoIP: Unknown]
Internet Control Message Protocol
  Type: 3 (Destination unreachable)
  Code: 0 (Network unreachable)
  Checksum: 0xfcff [correct]
Internet Protocol Version 4, Src: 172.16.2.234 (172.16.2.234), Dst: 172.20.1.1 (172.20.1.1)
Internet Control Message Protocol
  Type: 8 (Echo (ping) request)
  Code: 0
  Checksum: 0xc25b
```

▶ Type et code de l'erreur signalée par ICMP

▶ Message qui a causé le problème



# UN MESSAGE ICMP

► Problème :

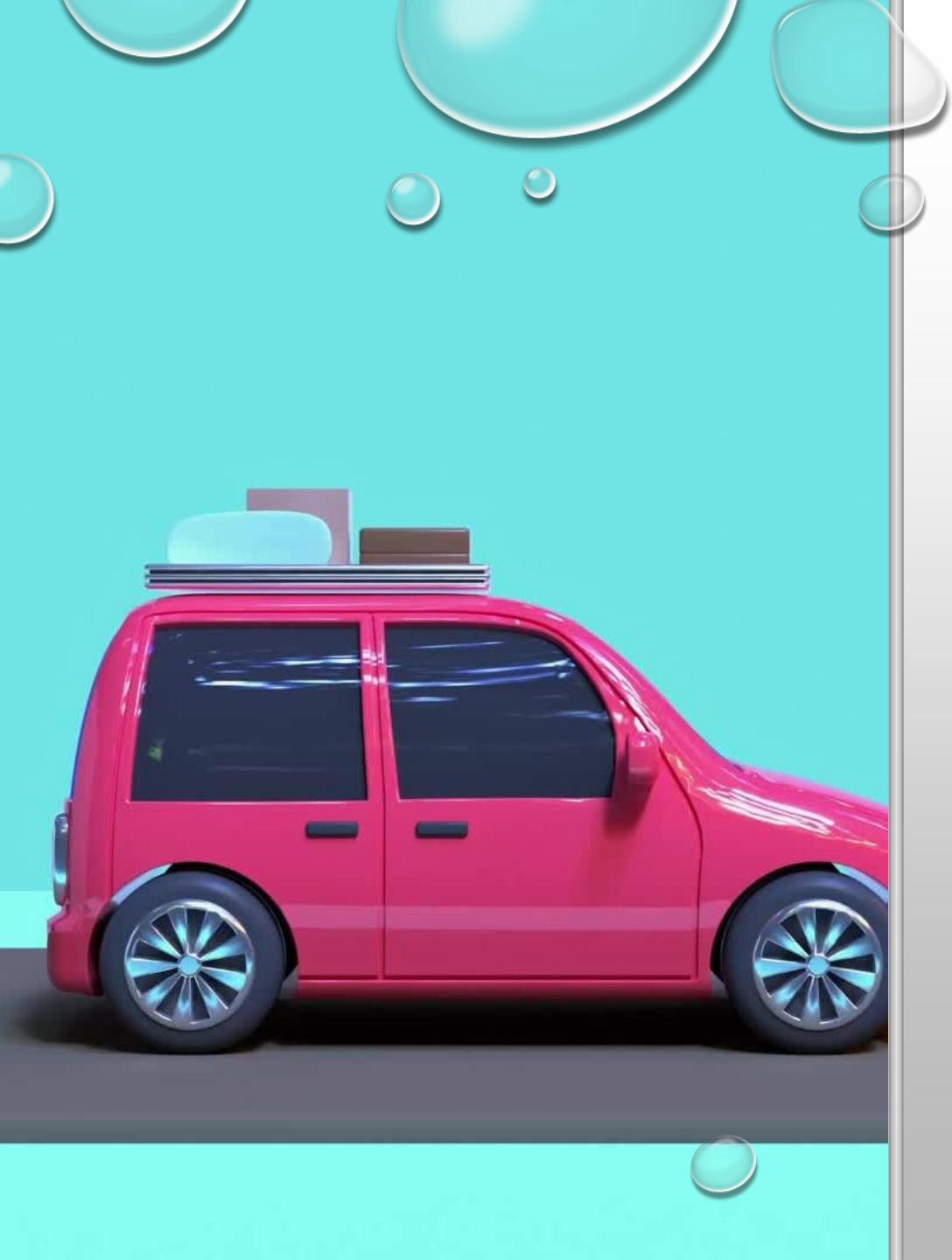
Ping envoyé par

172.16.2.234

On n'a pas pu contacter ce réseau !

```
▣ Frame 224: 102 bytes on wire (816 bits), 102 bytes captured (816 bits)
▣ Ethernet II, Src: AsustekC_b2:d0:11 (54:04:a6:b2:d0:11), Dst: CadmusCo_76:53:17 (08:00:27:76:53:17)
  ▣ Destination: CadmusCo_76:53:17 (08:00:27:76:53:17)
  ▣ Source: AsustekC_b2:d0:11 (54:04:a6:b2:d0:11)
  Type: IP (0x0800)
▣ Internet Protocol Version 4, Src: 172.16.2.232 (172.16.2.232), Dst: 172.16.2.234 (172.16.2.234)
  Version: 4
  Header length: 20 bytes
  ▣ Differentiated Services Field: 0xc0 (DSCP 0x30: Class Selector 6; ECN: 0x00: Not-ECT (Not ECN-CE))
  Total Length: 88
  Identification: 0xe491 (58513)
  ▣ Flags: 0x00
  Fragment offset: 0
  Time to live: 64
  Protocol: ICMP (1)
  ▣ Header checksum: 0x3761 [correct]
  Source: 172.16.2.232 (172.16.2.232)
  Destination: 172.16.2.234 (172.16.2.234)
  [Source GeoIP: Unknown]
  [Destination GeoIP: Unknown]
▣ Internet Control Message Protocol
  Type: 3 (Destination unreachable)
  Code: 0 (Network unreachable)
  Checksum: 0xfcff [correct]
▣ Internet Protocol Version 4, Src: 172.16.2.234 (172.16.2.234), Dst: 172.20.1.1 (172.20.1.1)
▣ Internet Control Message Protocol
  Type: 8 (Echo (ping) request)
  Code: 0
  Checksum: 0xc25b
```

A red arrow points from the text 'On n'a pas pu contacter ce réseau !' to the ICMP Echo request packet in the capture.



# LA COUCHE TRANSPORT





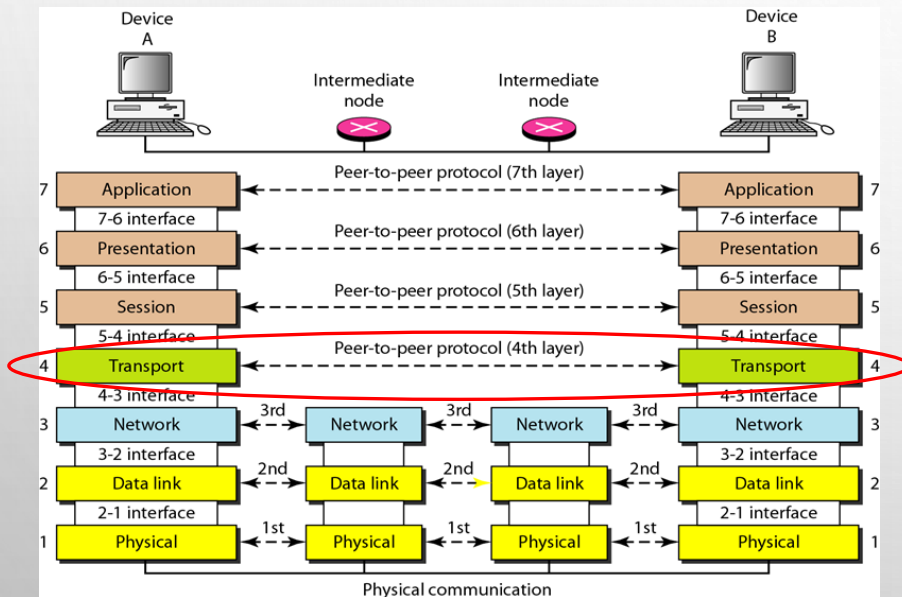
# LA COUCHE TRANSPORT

## La couche transport

**Objectif** : la fiabilité des communications

Couche transport : des sessions

- ▶ Sessions de communication : flow bidirectionnel
- ▶ Fiabilité : accusé de réception/renvoi
- ▶ Notion de service et protocole client/serveur



**À cette couche : transmissions fiables, ports, protocoles UDP, TCP**

# LA PROBLÉMATIQUE DES PORTS

- DEUX ORDINATEURS PEUVENT ÉCHANGER DES MESSAGES CONCERNANT PLUSIEURS SERVICES



- UN SERVEUR PEUT HÉBERGER PLUSIEURS SERVICES
- ON VEUT QUE LE BON MESSAGE SOIT TRAITÉ PAR LE BON SERVICE

# SOLUTION : LES PORTS

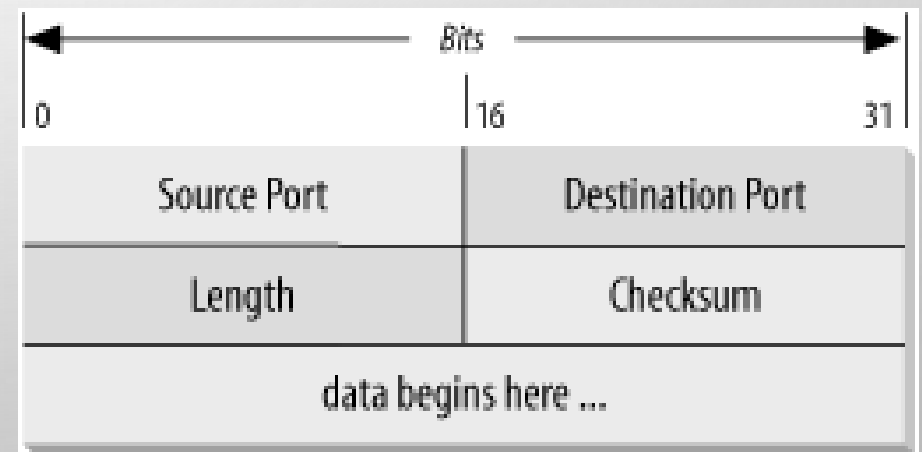
- CHAQUE SERVICE = UN NOUVEAU "PORT"



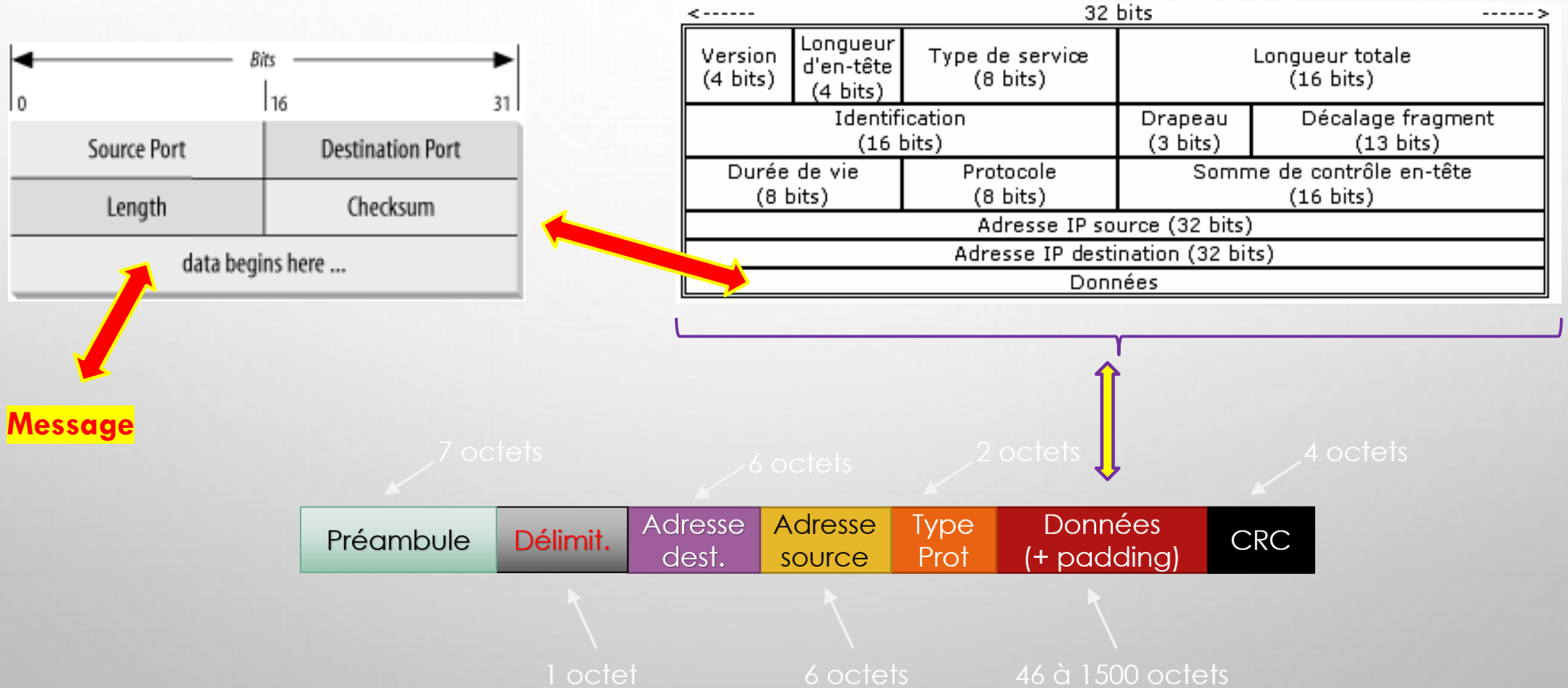
- 1 PORT = 16 BITS
- IL Y A DES PORTS RÉSERVÉS
- 1 APPLICATION  $\leftarrow \rightarrow$  (ADRESSE IP, PROTOCOLE, PORT)

# LE PROTOCOLE UDP

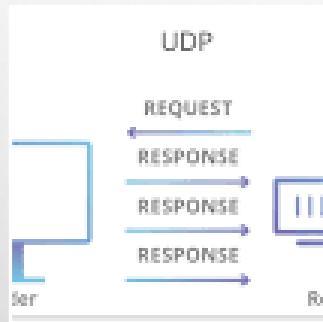
- Protocole basique de la couche transport
- Messages = datagrammes (d'où UDP = User Datagram Protocol)
- Spécifie le port source/destination
  - Peu de fiabilité garantie en dehors d'une checksum



# ENCAPSULATION

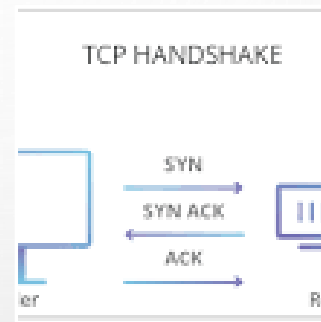


# LA TRANSMISSION FIABLE : TCP



## UDP permet :

- Séparation messages par service : ports
- Messages encapsulés : IP, puis Ethernet
- Pas de transmission fiable

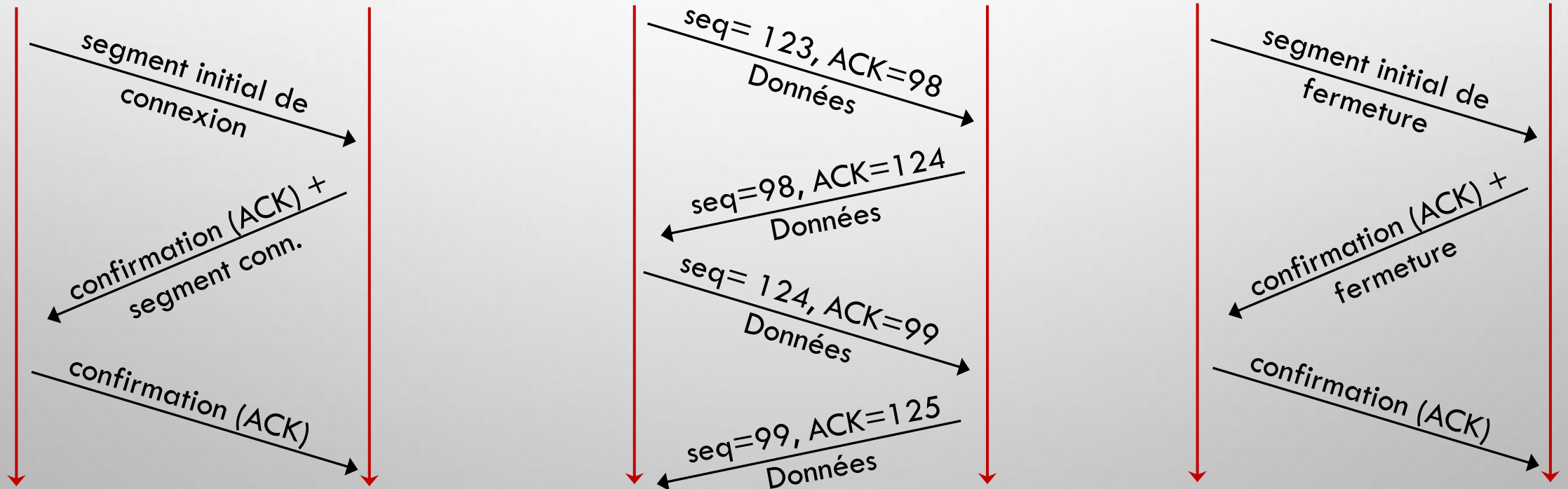


## TCP permet :

- Transmission ordonnée des flux
- Correction d'erreurs
- Transmission confirmée de messages

# CONNEXION TCP

- LA CONNEXION S'OUVRE AVEC UNE SUITE DE MESSAGES
  - PUIS LES DEUX MACHINE ÉCHANGENT DES MESSAGES ET CONFIRMENT LEUR RÉCEPTION





# PROTOCOLES CLIENT-SERVEUR

(COUCHE 7)



# CLIENT, SERVEUR

- PROTOCOLE CLIENT-SERVEUR :
  - ❖ Les messages passent entre une machine client et une machine serveur
- MACHINE SERVEUR :
  - ❖ Toujours à l'écoute, souvent sur un port standard
  - ❖ Peut se connecter à un client
  - ❖ Peut jouer également le rôle d'un client
- MACHINE CLIENT :
  - ❖ Contacte le serveur (typiquement port non-standard)

# LA COMMANDE SS

Les connexions d'une machine :  
la commande ss

- Connexions liées à des protocoles différents (applications différentes)
- Statut : à l'écoute, en cours, en cours de finir...

Paramétrisation

- ss -ln ports écoute (en format numérique)
- ss -uan toute connexion sur udp, format numérique
- ss -tn connexions sur TCP, format numérique

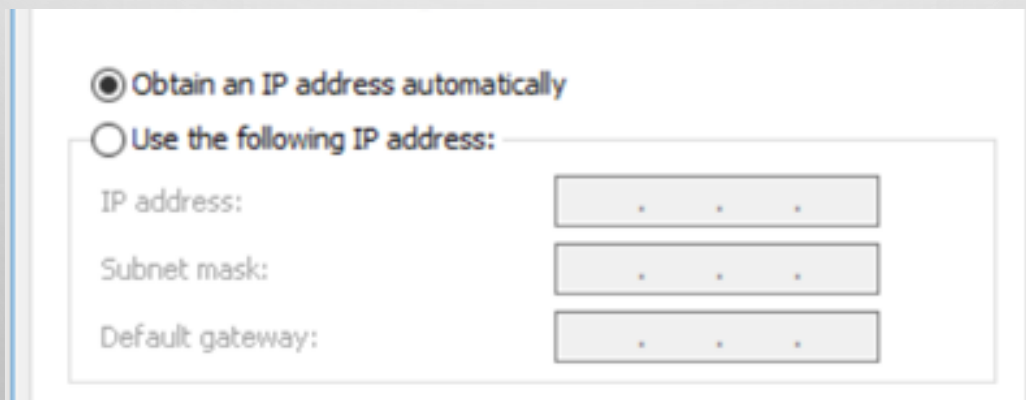


# LE PROTOCOLE DHCP



# PROBLÉMATIQUE

- On arrive dans un nouvel environnement (campus, hôtel, maison d'un ami...)
- On veut se connecter à son réseau
  - Une fois connectés, nous allons faire partie de son réseau
  - L'adresse physique ne change pas, mais on a une nouvelle adresse IP
- Pour la plupart on choisit de le faire automatiquement



The image shows a network configuration dialog box with two main options:

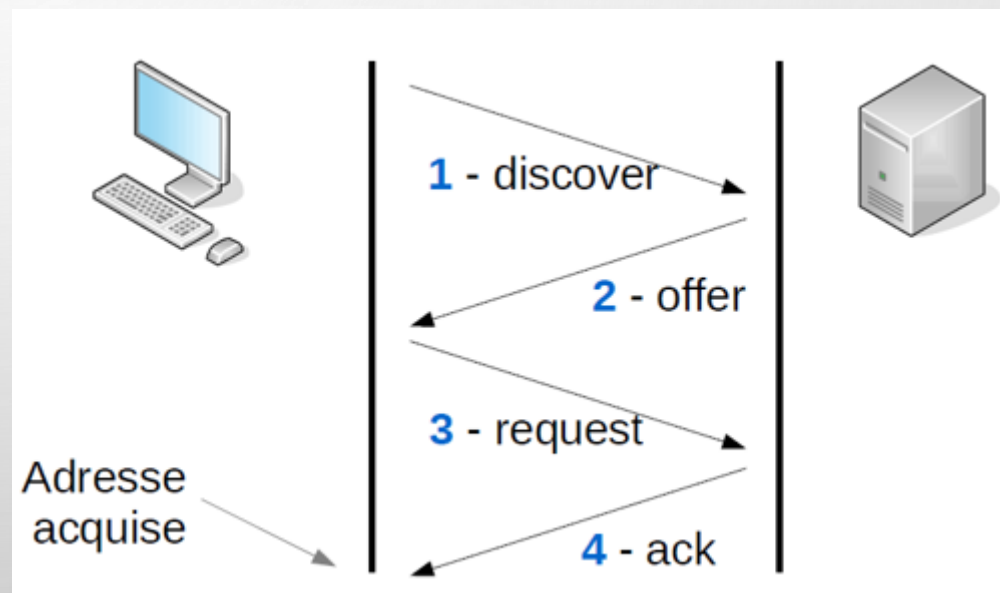
- Obtain an IP address automatically
- Use the following IP address:

Under the second option, there are three input fields for manual configuration:

- IP address: [ . . . ]
- Subnet mask: [ . . . ]
- Default gateway: [ . . . ]

# DHCP

- UN SERVEUR QUI DISTRIBUE AUTOMATIQUEMENT LA CONFIGURATION IP :
  - ADRESSE IP + MASQUE RÉSEAU
  - PASSERELLE PAR DÉFAUT (ROUTAGE)
  - MISE À JOUR DU SERVEUR DNS
- C'EST L'ISP QUI S'EN OCCUPE
  - GESTION CENTRALISÉE DU RÉSEAU
- 4 ÉTAPES



## ETAPE 1 : DISCOVER

- LE NOUVEAU ORDINATEUR VEUT RECEVOIR UNE ADRESSE IP
  - IL NE CONNAÎT PAS SON ADRESSE IP
  - IL FAIT UN BROADCAST

No.	Time	Source	Destination	Protocol	Length	Info
4	2.731556	0.0.0.0	255.255.255.255	DHCP	342	DHCP Discover - Transaction ID 0x9fe73d47
7	3.658390	192.168.1.102	192.168.1.20	DHCP	342	DHCP Offer - Transaction ID 0x9fe73d47
8	3.658549	192.168.1.101	192.168.1.10	DHCP	342	DHCP Offer - Transaction ID 0x9fe73d47

▶ Frame 4: 342 bytes on wire (2736 bits), 342 bytes captured (2736 bits)
▶ Ethernet II, Src: 6e:5f:98:37:0c:07 (6e:5f:98:37:0c:07), Dst: Broadcast (ff:ff:ff:ff:ff:ff)
▶ Internet Protocol Version 4, Src: 0.0.0.0 (0.0.0.0), Dst: 255.255.255.255 (255.255.255.255)
▶ User Datagram Protocol, Src Port: bootpc (68), Dst Port: bootps (67)
▶ Bootstrap Protocol

## ETAPE 2 : OFFER

- LE SERVEUR DHCP A REÇU SON MESSAGE ET LUI PROPOSE UNE ADRESSE IP
- CETTE OFFERTE EST LIMITÉE POUR UN NOMBRE DE MINUTES INITIELLEMENT

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000			DHCP	342	DHCP Discover - Transaction ID 0x9fe73d47
2	0.926834	192.168.1.102	192.168.1.20	DHCP	342	DHCP Offer - Transaction ID 0x9fe73d47
3	0.926993	192.168.1.101	192.168.1.10	DHCP	342	DHCP Offer - Transaction ID 0x9fe73d47

> Frame 2: 342 bytes on wire (2736 bits), 342 bytes captured (2736 bits)

> Ethernet II, Src: aa:be:53:3b:a5:42 (aa:be:53:3b:a5:42), Dst: 6e:5f:98:37:0c:07 (6e:5f:98:37:0c:07)

> Internet Protocol Version 4, Src: 192.168.1.102 (192.168.1.102), Dst: 192.168.1.20 (192.168.1.20)

> User Datagram Protocol, Src Port: bootps (67), Dst Port: bootpc (68)

▼ Bootstrap Protocol

Your (client) IP address: 192.168.1.20 (192.168.1.20)

- > Option: (t=53,l=1) DHCP Message Type = DHCP Offer
- > Option: (t=54,l=4) DHCP Server Identifier = 192.168.1.102
- > Option: (t=51,l=4) IP Address Lease Time = 10 minutes
- > Option: (t=1,l=4) Subnet Mask = 255.255.255.0
- > Option: (t=3,l=4) Router = 192.168.1.254
- > Option: (t=15,l=11) Domain Name = "domain1.net"
- > Option: (t=6,l=4) Domain Name Server = 8.8.8.8

## ETAPE 3 : REQUEST

- LE NOUVEL ORDINATEUR CHOISIT DE DEMANDER L'ADRESSE QU'ON LUI A DONNÉ
  - L'ADRESSE NE VA PLUS EXPIRER

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	0.0.0.0	255.255.255.255	DHCP	342	DHCP Discover - Transaction ID 0x9fe73d47
2	0.926834	192.168.1.102	192.168.1.20	DHCP	342	DHCP Offer - Transaction ID 0x9fe73d47
3	0.926993	192.168.1.101	192.168.1.10	DHCP	342	DHCP Offer - Transaction ID 0x9fe73d47
4	0.927374	0.0.0.0	255.255.255.255	DHCP	342	DHCP Request - Transaction ID 0x9fe73d47
5	0.928931	192.168.1.102	192.168.1.20	DHCP	342	DHCP ACK - Transaction ID 0x9fe73d47

Frame 4: 342 bytes on wire (2736 bits), 342 bytes captured (2736 bits)

Ethernet II, Src: 6e:5f:98:37:0c:07 (6e:5f:98:37:0c:07), Dst: Broadcast (ff:ff:ff:ff:ff:ff)

Internet Protocol Version 4, Src: 0.0.0.0 (0.0.0.0), Dst: 255.255.255.255 (255.255.255.255)

User Datagram Protocol, Src Port: bootpc (68), Dst Port: bootps (67)

Bootstrap Protocol

- ▷ Option: (t=53,l=1) DHCP Message Type = DHCP Request
- ▷ Option: (t=54,l=4) DHCP Server Identifier = 192.168.1.102
- ▷ Option: (t=50,l=4) Requested IP Address = 192.168.1.20
- ▷ Option: (t=55,l=12) Parameter Request List



## ETAPE 4 : ACKNOWLEDGEMENT

- LE SERVEUR CONFIRME L'ALLOCATION DE LA NOUVELLE CONFIG. IP

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	0.0.0.0	255.255.255.255	DHCP	342	DHCP Discover - Transaction ID 0x9fe73d47
2	0.926834	192.168.1.102	192.168.1.20	DHCP	342	DHCP Offer - Transaction ID 0x9fe73d47
3	0.926993	192.168.1.101	192.168.1.10	DHCP	342	DHCP Offer - Transaction ID 0x9fe73d47
4	0.927374	0.0.0.0	255.255.255.255	DHCP	342	DHCP Request - Transaction ID 0x9fe73d47
5	0.928931	192.168.1.102	192.168.1.20	DHCP	342	DHCP ACK - Transaction ID 0x9fe73d47

▶ Frame 5: 342 bytes on wire (2736 bits), 342 bytes captured (2736 bits)
▶ Ethernet II, Src: aa:be:53:3b:a5:42 (aa:be:53:3b:a5:42), Dst: 6e:5f:98:37:0c:07 (6e:5f:98:37:0c:07)
▶ Internet Protocol Version 4, Src: 192.168.1.102 (192.168.1.102), Dst: 192.168.1.20 (192.168.1.20)
▶ User Datagram Protocol, Src Port: bootps (67), Dst Port: bootpc (68)
▼ Bootstrap Protocol

# CONFIGURATION DHCP

- CÔTÉ SERVEUR :
  - ❖ Configuration fichier dhcpd.conf
    - ❖ Spécifier sur quelle interface on fait DHCP
    - ❖ Spécifier : plage d'adresses à utiliser, optionnellement quel routeur à utiliser, DNS...
  
- CÔTÉ CLIENT :
  - ❖ Obtenir une adresse dynamiquement : dhclient eth0
  - ❖ Configuration perenne : fichier /etc/network/interfaces

# FICHER INTERFACES

- CONFIGURATION PERENNE DES INTERFACES D'UNE MACHINE

```
auto lo
iface lo inet loopback
```

← Loopback, interface lo

```
auto eth0
iface eth0 inet static
address 192.168.56.11
netmask 255.255.255.0
gateway 192.168.56.1
```

← Config. statique, eth0

```
auto eth1
iface eth1 inet dhcp
```

← Config. par DHCP

# DIFFÉRENCES DHCP, INTERFACES

- LE FICHIER /ETC/NETWORK/INTERFACES INDIQUE :
  - ❖ L'adresse IP + routeur de la machine elle-même
  - ❖ Pas de possibilité de configuration DNS
  
- LE FICHIER /ETC/DHCP/DHCPD.CONF INDIQUE :
  - ❖ Une plage d'adresses IP qui seront données à une machine cherchant une configuration dynamique
  - ❖ Un routeur que les machines prenant une config. dynamique peuvent utiliser
  - ❖ Un serveur DNS pour ces machines aussi ...

The left half of the image features a solid black background. Scattered across this background are numerous water droplets of various sizes and shapes. Some are perfectly spherical, while others are more elongated or flattened. Each droplet has a bright white highlight on its upper-left edge, giving it a three-dimensional appearance. The droplets are concentrated more towards the top and bottom edges of the black area.

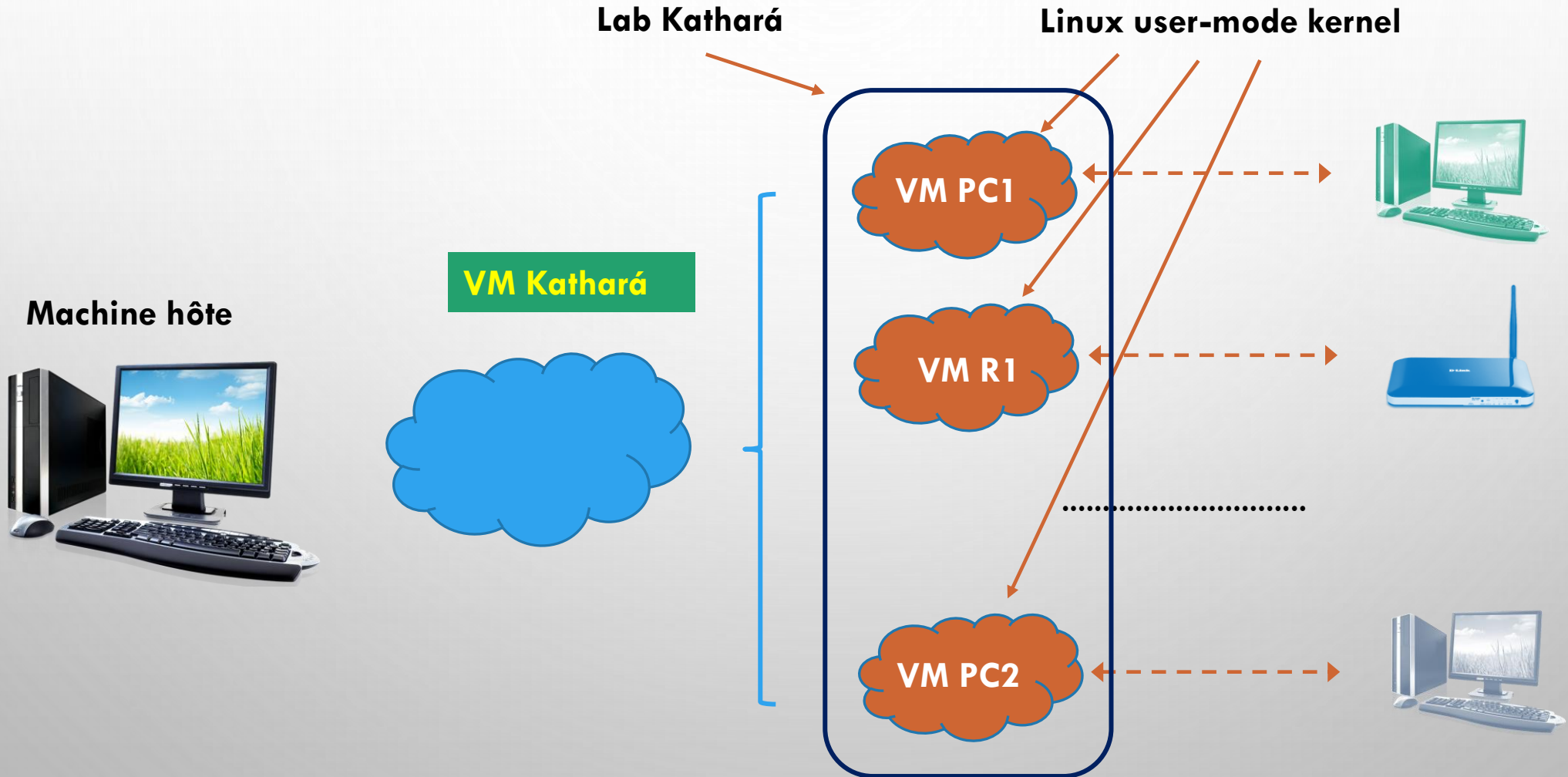
# INTRODUCTION À KATHARÁ



# L'OUTIL KATHARÁ

- PERMET DE SIMULER LE FONCTIONNEMENT DE TOUT UN RESEAU SUR UNE MACHINE
- UN LOGICIEL QUI PERMET DE CRÉER DES CONTAINERS SUR UNE **MACHINE HÔTE**
  - Chaque conteneur est une VM Linux
  - Chaque VM joue le rôle d'un élément d'un réseau : un PC, un routeur, un serveur...
  - Toutes ces VMs sont dans un seul VM dans lequel on exécute Kathará
  - On peut manipuler VM par VM, ou faire toute la simulation fonctionner à partir de Kathará
- CECI NOUS PERMET DE SIMULER LE FONCTIONNEMENT D'UN RÉSEAU EN TOUTE SÉCURITÉ

# L'INFRASTRUCTURE AVEC KATHARÁ



# LES LABS KATHARÁ

- 1 LAB = 1 REPERTOIRE (CRÉÉ EN DÉBUT DU TP)
- LAB KATHARÁ : TOPOLOGIE SPÉCIFIÉE DANS UN FICHIER LAB.CONF
  - Lab.conf indique l'interconnexion physique entre les machines
  - Les machines partageant un réseau/médium physique forment 1 domaine de collision
- 1 MACHINE = 1 MACHINE VIRTUELLE
- DES FICHIERS POUR DÉCRIRE LE COMPORTEMENT DES MACHINES AU START ET À L'ARRÊT



# LA CONFIGURATION D'UN LAB : LAB.CONF

**<nom\_machine>[<#>]** : interface eth<#> de la machine <nom\_machine>

**net0, net1** : deux domaines de collision distincts, domain0, domain1

**Dessinez la topologie indiquée par ce script**

Démarrage du lab une fois lab.conf fini :

kathara lstart

```
>> sudo nano lab.conf
```

```
pca[0]=net0
```

```
pcb[0]=net0
```

```
pcb[1]=net1
```

```
pcc[0]=net1
```

# LES FICHIERS .STARTUP

- CHAQUE MACHINE PEUT AVOIR UN FICHIER <NOM MACHINE>.STARTUP

- UNE INSTRUCTION TYPIQUE DANS UN TEL FICHIER POURAIT ÊTRE

**IP ADDRESS ADD 192.168.1.2/24 DEV ETH1**

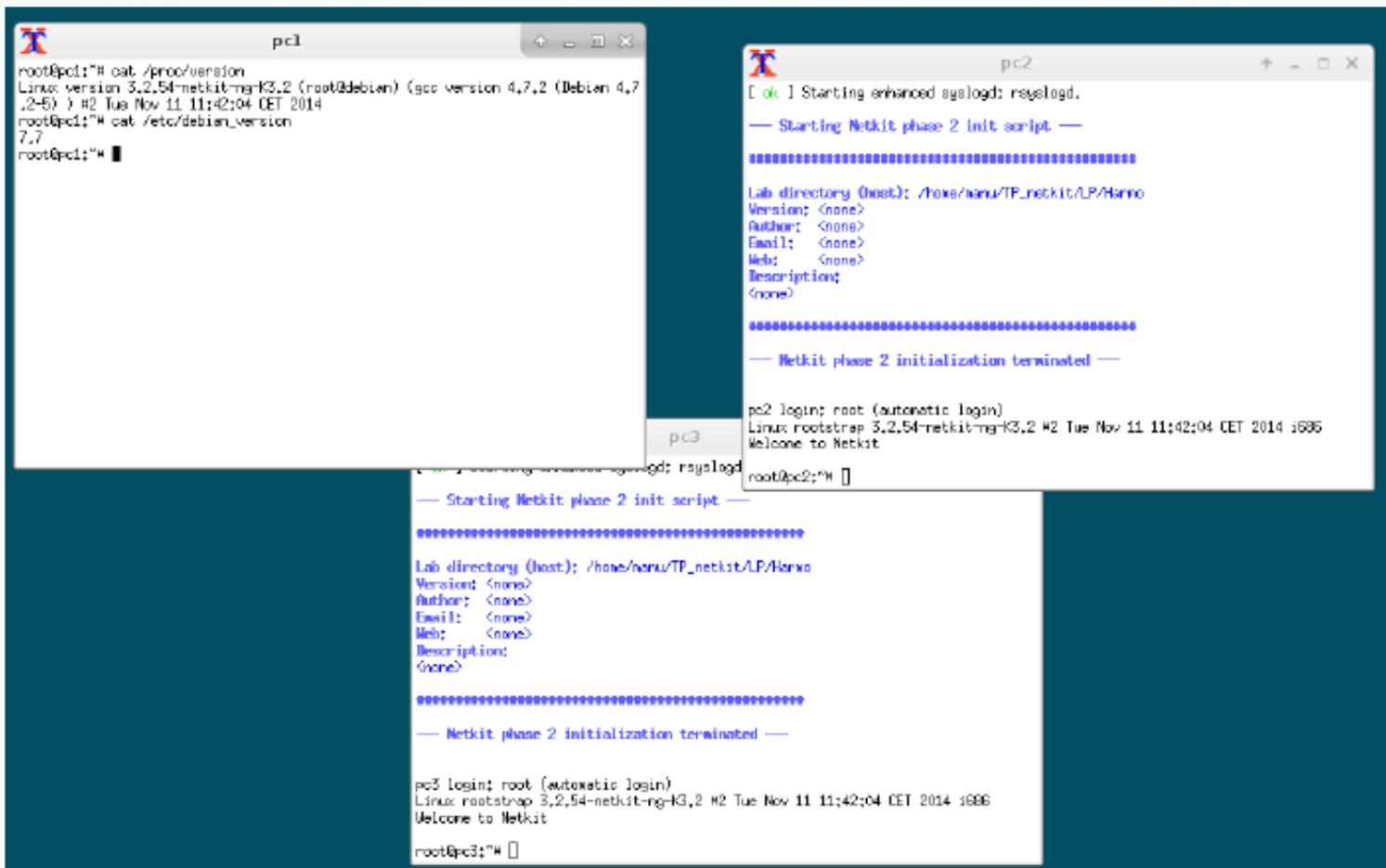
**IP LINK SET DEV ETH1 UP**

- ARRÊT PROPRE D'UN LAB :

kathara.lclean

kathara.wipe

# LES TROIS VMS INITIALISÉES



```
root@pc1:~# cat /proc/version
Linux version 3.2.54-netkit-ng-K3.2 (root@debian) (gcc version 4.7.2 (Debian 4.7
.2-5) ) #2 Tue Nov 11 11:42:04 CET 2014
root@pc1:~# cat /etc/debian_version
7.7
root@pc1:~# █

[ ok ] Starting enhanced syslogd: syslogd.
--- Starting Netkit phase 2 init script ---
=====
Lab directory (host): /home/naru/TP_netkit/UP/Hanno
Version: <none>
Author: <none>
Email: <none>
Web: <none>
Description:
<none>
=====
--- Netkit phase 2 initialization terminated ---

pc2 login: root (automatic login)
Linux rootstrap 3.2.54-netkit-ng-K3.2 #2 Tue Nov 11 11:42:04 CET 2014 :686
Welcome to Netkit
root@pc2:~# █

[ ok ] Starting enhanced syslogd: syslogd.
--- Starting Netkit phase 2 init script ---
=====
Lab directory (host): /home/naru/TP_netkit/UP/Hanno
Version: <none>
Author: <none>
Email: <none>
Web: <none>
Description:
<none>
=====
--- Netkit phase 2 initialization terminated ---

pc3 login: root (automatic login)
Linux rootstrap 3.2.54-netkit-ng-K3.2 #2 Tue Nov 11 11:42:04 CET 2014 :686
Welcome to Netkit
root@pc3:~# █
```

# UNE CONNEXION INTERNET

- Le logiciel Kathará sert à simuler des configurations de réseau avant de les mettre en pratique
  - Pour anticiper des crashes, des difficultés concernant le routage...
- Parfois, même pour une simulation il nous faut une connexion Internet
- Faisable en Kathará en utilisant des bridges :
  - Le pont crée sur PCC une nouvelle interface (eth1) avec une adresse de classe B, connectée à une nouvelle interface créée sur la machine hôte

```
>> sudo nano lab.conf
```

```
pca[0]=net0
```

```
pcb[0]=net0
```

```
pcb[1]=net1
```

```
pcc[0]=net1
```

```
pcc[bridged]=true
```