

Dernière mise-à-jour : 2020/01/30 03:27

# 109.1 - Fondamentaux de protocoles Internet (4/60)

## Mieux Comprendre TCP Version 4

### En-tête TCP

L'en-tête TCP est codée sur 4 octets soit 32 bits :

| 1er octet   | 2ème octet | 3ème octet            | 4 ème octet |
|-------------|------------|-----------------------|-------------|
| Port source |            | Port destination      |             |
|             |            | Numéro de séquence    |             |
|             |            | Numéro d'acquittement |             |
| Offset      | Flags      |                       | Fenêtre     |
| Checksum    |            | Pointeur Urgent       |             |
| Options     |            | Padding               |             |
|             |            | Données               |             |

Vous noterez que les numéros de ports sont codés sur 16 bits. Cette information nous permet de calculer le nombres de ports maximum en IPv4, soit  $2^{16}$  ports ou 65 535.

L'**Offset** contient la taille de l'en-tête.

Les **Flags** sont :

- URG - Si la valeur est 1 le pointeur urgent est utilisé. Le numéro de séquence et le pointeur urgent indique un octet spécifique.
- ACK - Si la valeur est 1, le paquet est un accusé de réception
- PSH - Si la valeur est 1, les données sont immédiatement présentées à l'application
- RST - Si la valeur est 1, la communication comporte un problème et la connexion est réinitialisée

- SYN - Si la valeur est 1, le paquet est un paquet de synchronisation
- FIN - Si la valeur est 1, le paquet indique la fin de la connexion

La **Fenêtre** est codée sur 16 bits. La Fenêtre est une donnée liée au fonctionnement d'expédition de données appelé le **sliding window** ou la **fenêtre glissante**. Puisque il serait impossible, pour des raisons de performance, d'attendre l'accusé de réception de chaque paquet envoyé, l'expéditeur envoie des paquets par groupe. La taille de cette groupe s'appelle la Fenêtre. Dans le cas d'un problème de réception d'une partie de la Fenêtre, toute la Fenêtre est ré-expédiée.

Le **Checksum** est une façon de calculer si le paquet est complet.

Le **Padding** est un champ pouvant être rempli de valeurs nulles de façon à ce que la taille de l'en-tête soit un multiple de 32

## En-tête UDP

L'en-tête UDP est codée sur 4 octets soit 32 bits :

| 1er octet   | 2ème octet | 3ème octet       | 4 ème octet |
|-------------|------------|------------------|-------------|
| Port source |            | Port destination |             |
| longueur    |            | Checksum         |             |
| Données     |            |                  |             |

L'en-tête UDP a une longueur de 8 octets.

## Fragmentation et Ré-encapsulation

La taille limite d'un paquet TCP, l'en-tête comprise, ne peut pas dépasser **65 535 octets**. Cependant chaque réseau est qualifié par son MTU ( Maximum Tranfer Unit ). Cette valeur est la taille maximum d'un paquet autorisée. L'unité est en **octets**. Pour un réseau Ethernet sa valeur est de 1 500. Quand un paquet doit être expédié sur un réseau ayant un MTU inférieur à sa propre taille, le paquet doit être **fractionné**. A la sortie du réseau, le paquet est reconstitué. Cette reconstitution s'appelle **ré-encapsulation**.

## Adressage

L'adressage IP requiert que chaque périphérique sur le réseau possède une adresse IP unique de 4 octets, soit 32 bits au format XXX.XXX.XXX.XXX. De cette façon le nombre total d'adresses est de  $2^{32} = 4.3$  Milliards.

Les adresses IP sont divisées en 5 classes, de A à E. Les 4 octets des classes A à C sont divisés en deux, une partie qui s'appelle le **Net ID** qui identifie le réseau et une partie qui s'appelle le **Host ID** qui identifie le hôte :

|   | <b>1er octet</b> | <b>2ème octet</b> | <b>3ème octet</b> | <b>4 ème octet</b> |
|---|------------------|-------------------|-------------------|--------------------|
| A | Net ID           |                   | Host ID           |                    |
| B | Net ID           |                   | Host ID           |                    |
| C |                  | Net ID            |                   | Host ID            |
| D |                  | Multicast         |                   |                    |
| E |                  | Réservé           |                   |                    |

L'attribution d'une classe dépend du nombre de hôtes à connecter. Chaque classe est identifiée par un **Class ID** composé de 1 à 3 bits :

| <b>Classe</b> | <b>Bits ID Classe</b> | <b>Valeur ID Classe</b> | <b>Bits ID Réseau</b> | <b>Nb. de Réseaux</b> | <b>Bits ID hôtes</b> | <b>Nb. d'adresses</b> | <b>Octet de Départ</b> |
|---------------|-----------------------|-------------------------|-----------------------|-----------------------|----------------------|-----------------------|------------------------|
| A             | 1                     | 0                       | 7                     | $2^7=128$             | 24                   | $2^{24}=16\ 777\ 216$ | 1 - 126                |
| B             | 2                     | 10                      | 14                    | $2^{14}=16\ 834$      | 16                   | $2^{16}=65\ 535$      | 128 - 191              |
| C             | 3                     | 110                     | 21                    | $2^{21}=2\ 097\ 152$  | 8                    | $2^8=256$             | 192 - 223              |

Dans chaque classe, certaines adresses sont réservées pour un usage privé :

| <b>Classe</b> | <b>IP de Départ</b> | <b>IP de Fin</b> |
|---------------|---------------------|------------------|
| A             | 10.0.0.0            | 10.255.255.255   |
| B             | 172.16.0.0          | 172.31.255.255   |
| C             | 192.168.0.0         | 192.168.255.255  |

Il existe des adresses particulières ne pouvant pas être utilisées pour identifier un hôte :

| Adresse Particulière          | Description                                  |
|-------------------------------|--|
| 169.254.0.0 à 169.254.255.255 | Automatic Private IP Addressing de Microsoft |
| Hôte du réseau courant        | Tous les bits du Net ID sont à 0             |
| Adresse de réseau             | Tous les bits du Host ID sont à 0            |
| Adresse de diffusion          | Tous les bits du Host ID sont à 1            |

L'adresse de réseau identifie le **segment** du réseau entier tandis que l'adresse de diffusion identifie tous les hôtes sur le segment de réseau.

Afin de mieux comprendre l'adresse de réseau et l'adresse de diffusion, prenons le cas de l'adresse 192.168.10.1 en classe C :

|                                  | 1er octet | 2ème octet | 3ème octet | 4 ème octet     |
|----------------------------------|-----------|------------|------------|-----------------|
|                                  | Net ID    |            |            | Host ID         |
| Adresse IP                       | 192       | 168        | 10         | 1               |
| Binaire                          | 11000000  | 10101000   | 000001010  | 00000001        |
| Calcul de l'adresse de réseau    |           |            |            |                 |
| Binaire                          | 11000000  | 10101000   | 000001010  | <b>00000000</b> |
| Adresse réseau                   | 192       | 168        | 10         | 0               |
| Calcul de l'adresse de diffusion |           |            |            |                 |
| Binaire                          | 11000000  | 10101000   | 000001010  | <b>11111111</b> |
| Adresse de diffusion             | 192       | 168        | 10         | 255             |

## Masques de sous-réseaux

Tout comme l'adresse IP, le masque de sous-réseau compte 4 octets ou 32 bits. Les masques de sous-réseaux permettent d'identifier le Net ID et le Host ID :

| Classe | Masque        | Notation CIDR |
|--------|---------------|---------------|
| A      | 255.0.0.0     | /8            |
| B      | 255.255.0.0   | /16           |
| C      | 255.255.255.0 | /24           |

Le terme **CIDR** veut dire **Classless InterDomain Routing**. Le terme Notation CIDR correspond au nombre de bits d'une valeur de 1 dans le masque

de sous-réseau.

Quand un hôte souhaite émettre il procède d'abord à l'identification de sa propre adresse réseau par un calcul AND (ET) appliqué à sa propre adresse et son masque de sous-réseau qui stipule :

- $1 \times 1 = 1$
- $0 \times 1 = 0$
- $1 \times 0 = 0$
- $0 \times 0 = 0$

Prenons le cas de l'adresse IP 192.168.10.1 ayant un masque de 255.255.255.0 :

|                       | <b>1er octet</b> | <b>2ème octet</b> | <b>3ème octet</b> | <b>4 ème octet</b> |
|-----------------------|------------------|-------------------|-------------------|--------------------|
| Adresse IP            | 192              | 168               | 10                | 1                  |
| Binaire               | 11000000         | 10101000          | 00001010          | 00000001           |
| Masque de sous-réseau |                  |                   |                   |                    |
| Binaire               | 11111111         | 11111111          | 11111111          | 00000000           |
| Calcul AND            | 11000000         | 10101000          | 00001010          | 00000000           |
| Adresse réseau        | 192              | 168               | 10                | 0                  |

Cet hôte essaie de communiquer avec un hôte ayant une adresse IP de 192.168.10.10. Il procède donc au même calcul en appliquant **son propre masque de sous-réseau** à l'adresse IP de l'hôte destinataire :

|                       | <b>1er octet</b> | <b>2ème octet</b> | <b>3ème octet</b> | <b>4 ème octet</b> |
|-----------------------|------------------|-------------------|-------------------|--------------------|
| Adresse IP            | 192              | 168               | 10                | 10                 |
| Binaire               | 11000000         | 10101000          | 00001010          | 00001010           |
| Masque de sous-réseau |                  |                   |                   |                    |
| Binaire               | 11111111         | 11111111          | 11111111          | 00000000           |
| Calcul AND            | 11000000         | 10101000          | 00001010          | 00000000           |
| Adresse réseau        | 192              | 168               | 10                | 0                  |

Puisque l'adresse réseau est identique dans les deux cas, l'hôte émetteur présume que l'hôte de destination se trouve sur son réseau et envoie les paquets directement sur le réseau sans s'adresser à sa passerelle par défaut.

L'hôte émetteur essaie maintenant de communiquer avec un hôte ayant une adresse IP de 192.168.2.1. Il procède donc au même calcul en appliquant **son propre masque de sous-réseau** à l'adresse IP de l'hôte destinataire :

|                       | <b>1er octet</b> | <b>2ème octet</b> | <b>3ème octet</b> | <b>4 ème octet</b> |
|-----------------------|------------------|-------------------|-------------------|--------------------|
| Adresse IP            | 192              | 168               | 2                 | 1                  |
| Binaire               | 11000000         | 10101000          | 00000010          | 00000001           |
| Masque de sous-réseau |                  |                   |                   |                    |
| Binaire               | 11111111         | 11111111          | 11111111          | 00000000           |
| Calcul AND            | 11000000         | 10101000          | 00000010          | 00000000           |
| Adresse réseau        | 192              | 168               | 2                 | 0                  |

Dans ce cas, l'hôte émetteur constate que le réseau de destination 192.168.2.0 n'est pas identique à son propre réseau 192.168.10.0. Il adresse donc les paquets à la passerelle par défaut.

## VLSM

Puisque le stock de réseaux disponibles sous IPv4 est presque épuisé, une solution a été trouvée pour créer des sous-réseaux en attendant l'introduction de l'IPv6. Cette solution s'appelle le VLSM ou Variable Length Subnet Masks. Le VLSM exprime les masques de sous-réseaux au format CIDR.

Son principe est simple. Afin de créer des réseaux différents à partir d'une adresse réseau d'une classe donnée, il convient de réduire le nombre d'hôtes. De cette façon les bits 'libérés' du Host ID peuvent être utilisés pour identifier les sous-réseaux.

Pour illustrer ceci, prenons l'exemple d'un réseau 192.168.1.0. Sur ce réseau, nous pouvons mettre  $2^8 - 2$  soit 254 hôtes entre 192.168.1.1 au 192.168.1.254.

Supposons que nous souhaiterions diviser notre réseau en 2 sous-réseaux. Pour coder 2 sous-réseaux, il faut que l'on libère 2 bits du Host ID. Les deux bits libérés auront les valeurs binaires suivantes :

- 00
- 01
- 10
- 11

Les valeurs binaires du quatrième octet de nos adresses de sous-réseaux seront donc :

- 192.168.1.00XXXXXX
- 192.168.1.01XXXXXX
- 192.168.1.10XXXXXX
- 192.168.1.11XXXXXX

où les XXXXXX représentent les bits que nous réservons pour décrire les hôtes dans chacun des sous-réseaux.

Nous ne pouvons pas utiliser les deux sous-réseaux suivants :

- 192.168.1.00XXXXXX
- 192.168.1.11XXXXXX

car ceux-ci correspondent aux débuts de l'adresse réseau 192.168.1.0 et de l'adresse de diffusion 192.168.1.255.

Nous pouvons utiliser les deux sous-réseaux suivants :

- 192.168.1.01XXXXXX
- 192.168.1.10XXXXXX

Pour le premier sous-réseau l'adresse réseau et l'adresse de diffusion sont :

|                                  |          |          |          |                  |
|----------------------------------|----------|----------|----------|------------------|
| Sous-réseau #1                   | 192      | 168      | 1        | 01XXXXXX         |
| Calcul de l'adresse de réseau    |          |          |          |                  |
| Binaire                          | 11000000 | 10101000 | 00000001 | 01 <b>000000</b> |
| Adresse réseau                   | 192      | 168      | 1        | <b>64</b>        |
| Calcul de l'adresse de diffusion |          |          |          |                  |
| Binaire                          | 11000000 | 10101000 | 00000001 | 01 <b>111111</b> |
| Adresse de diffusion             | 192      | 168      | 1        | <b>127</b>       |

- L'adresse CIDR du réseau est donc 192.168.1.64/26 car le Net ID est codé sur 24+2 bits.
- Le masque de sous-réseau est donc le 11111111.11111111.11111111.11000000 ou le 255.255.255.192
- Nous pouvons avoir  $2^6 - 2$  soit 62 hôtes.

- La plage valide d'adresses IP est de 192.168.1.65 à 192.168.1.126

Pour le deuxième sous-réseau l'adresse réseau et l'adresse de diffusion sont :

|                                  |          |          |          |          |
|----------------------------------|----------|----------|----------|----------|
| Sous-réseau #2                   | 192      | 168      | 1        | 10XXXXXX |
| Calcul de l'adresse de réseau    |          |          |          |          |
| Binaire                          | 11000000 | 10101000 | 00000001 | 10000000 |
| Adresse réseau                   | 192      | 168      | 1        | 128      |
| Calcul de l'adresse de diffusion |          |          |          |          |
| Binaire                          | 11000000 | 10101000 | 00000001 | 10111111 |
| Adresse de diffusion             | 192      | 168      | 1        | 191      |

- L'adresse CIDR du réseau est donc 192.168.1.128/26 car le Net ID est codé sur 24+2 bits.
- Le masque de sous-réseau est donc le 11111111.11111111.11111111.11000000 ou le 255.255.255.192
- Nous pouvons avoir  $2^6-2$  soit 62 hôtes.
- La plage valide d'adresses IP est de 192.168.1.129 à 192.168.1.190

La valeur qui sépare les sous-réseaux est 64. Cette valeur comporte le nom **incrément**.

## Ports et sockets

Afin que les données arrivent aux applications que les attendent, TCP utilise des numéros de ports sur la couche transport. Le numéros de ports sont divisés en trois groupes :

- **Well Known Ports**
  - De 1 à 1023
- **Registered Ports**
  - De 1024 à 49151
- **Dynamic et/ou Private Ports**
  - De 49152 à 65535

Le couple **numéro IP:numéro de port** s'appelle un **socket**.

## /etc/services

Les ports les plus utilisés sont détaillés dans le fichier **/etc/services** :

```
[root@centos7 ~]# more /etc/services
# /etc/services:
# $Id: services,v 1.55 2013/04/14 ovasik Exp $
#
# Network services, Internet style
# IANA services version: last updated 2013-04-10
#
# Note that it is presently the policy of IANA to assign a single well-known
# port number for both TCP and UDP; hence, most entries here have two entries
# even if the protocol doesn't support UDP operations.
# Updated from RFC 1700, ``Assigned Numbers'' (October 1994). Not all ports
# are included, only the more common ones.
#
# The latest IANA port assignments can be gotten from
#      http://www.iana.org/assignments/port-numbers
# The Well Known Ports are those from 0 through 1023.
# The Registered Ports are those from 1024 through 49151
# The Dynamic and/or Private Ports are those from 49152 through 65535
#
# Each line describes one service, and is of the form:
#
# service-name  port/protocol  [aliases ...]  [# comment]

tcpmux          1/tcp           # TCP port service multiplexer
tcpmux          1/udp           # TCP port service multiplexer
rje             5/tcp           # Remote Job Entry
rje             5/udp           # Remote Job Entry
```

```

echo      7/tcp
echo      7/udp
discard   9/tcp      sink null
discard   9/udp      sink null
systat    11/tcp     users
systat    11/udp     users
daytime   13/tcp
--More-- (0%)

```

Notez que les ports sont listés par deux :

- le port TCP
- le port UDP

La liste la plus complète peut être consultée sur le site Internet [www.iana.org](http://www.iana.org).

Pour connaître la liste des sockets ouverts sur l'ordinateur, saisissez la commande suivante :

```

[root@centos7 ~]# netstat -an | more
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address          Foreign Address        State
tcp      0      0 0.0.0.0:22              0.0.0.0:*
                                         LISTEN
tcp      0      0 127.0.0.1:7127            0.0.0.0:*
                                         LISTEN
tcp      0      0 127.0.0.1:631             0.0.0.0:*
                                         LISTEN
tcp      0      0 127.0.0.1:25              0.0.0.0:*
                                         LISTEN
tcp      0      0 127.0.0.1:52284            0.0.0.0:*
                                         LISTEN
tcp      0      0 127.0.0.1:49669            0.0.0.0:*
                                         LISTEN
tcp      0      0 127.0.0.1:52284            127.0.0.1:46641        ESTABLISHED
tcp      0      0 10.0.2.15:22              10.0.2.2:47261         ESTABLISHED
tcp      0      0 127.0.0.1:46641            127.0.0.1:52284        ESTABLISHED
tcp6     0      0 :::22                  :::*
                                         LISTEN
tcp6     0      0 :::631                 :::*
                                         LISTEN
udp      0      0 10.0.2.15:49309            10.0.2.3:53           ESTABLISHED
udp      0      0 0.0.0.0:42155             0.0.0.0:*

```

|      |   |   |                 |                    |             |
|------|---|---|-----------------|--------------------|-------------|
| udp  | 0 | 0 | 0.0.0.0:5353    | 0.0.0.0:*          |             |
| udp  | 0 | 0 | 127.0.0.1:323   | 0.0.0.0:*          |             |
| udp  | 0 | 0 | 0.0.0.0:68      | 0.0.0.0:*          |             |
| udp  | 0 | 0 | 0.0.0.0:14451   | 0.0.0.0:*          |             |
| udp  | 0 | 0 | 10.0.2.15:37244 | 212.83.184.186:123 | ESTABLISHED |
| udp6 | 0 | 0 | ::1:323         | :::*               |             |
| udp6 | 0 | 0 | :::35912        | :::*               |             |
| raw6 | 0 | 0 | :::58           | :::*               | 7           |

#### Active UNIX domain sockets (servers and established)

| Proto | RefCnt | Flags   | Type   | State     | I-Node | Path                       |
|-------|--------|---------|--------|-----------|--------|----------------------------|
| unix  | 2      | [ ACC ] | STREAM | LISTENING | 20224  | public/pickup              |
| unix  | 2      | [ ACC ] | STREAM | LISTENING | 20228  | public/cleanup             |
| unix  | 2      | [ ACC ] | STREAM | LISTENING | 20231  | public/qmgr                |
| unix  | 2      | [ ACC ] | STREAM | LISTENING | 11278  | /run/lvm/lvmpolld.socket   |
| unix  | 2      | [ ACC ] | STREAM | LISTENING | 13838  | /var/run/dbus/system_bus_s |
| ocket |        |         |        |           |        |                            |
| unix  | 2      | [ ACC ] | STREAM | LISTENING | 20253  | public/flush               |
| unix  | 2      | [ ACC ] | STREAM | LISTENING | 20268  | public/showq               |

--More--

Pour connaître la liste des applications ayant ouvert un port sur l'ordinateur, saisissez la commande suivante :

```
[root@centos7 ~]# netstat -anp | more
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address          Foreign Address        State      PID/Program name
tcp    0      0 0.0.0.0:22                0.0.0.0:*
                                              LISTEN
                                              855/sshd
tcp    0      0 127.0.0.1:7127             0.0.0.0:*
                                              LISTEN
                                              3275/Remote Access
tcp    0      0 127.0.0.1:631              0.0.0.0:*
                                              LISTEN
                                              854/cupsd
tcp    0      0 127.0.0.1:25               0.0.0.0:*
                                              LISTEN
                                              2214/master
tcp    0      0 127.0.0.1:52284             0.0.0.0:*
                                              LISTEN
                                              3389/Remote Access
tcp    0      0 127.0.0.1:49669             0.0.0.0:*
                                              LISTEN
                                              3275/Remote Access
tcp    0      0 127.0.0.1:52284             127.0.0.1:46641
                                              ESTABLISHED
                                              3389/Remote Access
tcp    0      0 10.0.2.15:22               10.0.2.2:47261
                                              ESTABLISHED
                                              4557/sshd: trainee
tcp    0      1 10.0.2.15:55144             86.241.135.118:443
                                              SYN_SENT
                                              3275/Remote Access
```

|      |   |   |                 |                    |             |                     |
|------|---|---|-----------------|--------------------|-------------|---------------------|
| tcp  | 0 | 0 | 127.0.0.1:46641 | 127.0.0.1:52284    | ESTABLISHED | 3275/Remote Access  |
| tcp6 | 0 | 0 | :::22           | :::*               | LISTEN      | 855/sshd            |
| tcp6 | 0 | 0 | ::1:631         | :::*               | LISTEN      | 854/cupsd           |
| udp  | 0 | 0 | 0.0.0.0:42155   | 0.0.0.0:*          |             | 525/avahi-daemon: r |
| udp  | 0 | 0 | 0.0.0.0:5353    | 0.0.0.0:*          |             | 525/avahi-daemon: r |
| udp  | 0 | 0 | 127.0.0.1:323   | 0.0.0.0:*          |             | 556/chronyrd        |
| udp  | 0 | 0 | 0.0.0.0:68      | 0.0.0.0:*          |             | 4501/dhclient       |
| udp  | 0 | 0 | 0.0.0.0:14451   | 0.0.0.0:*          |             | 4501/dhclient       |
| udp  | 0 | 0 | 10.0.2.15:37244 | 212.83.184.186:123 | ESTABLISHED | 556/chronyrd        |
| udp6 | 0 | 0 | ::1:323         | :::*               |             | 556/chronyrd        |
| udp6 | 0 | 0 | :::35912        | :::*               |             | 4501/dhclient       |
| raw6 | 0 | 0 | :::58           | :::*               | 7           | 653/NetworkManager  |

#### Active UNIX domain sockets (servers and established)

| Proto | RefCnt | Flags   | Type   | State     | I-Node | PID/Program name | Path                            |
|-------|--------|---------|--------|-----------|--------|------------------|---------------------------------|
| unix  | 2      | [ ACC ] | STREAM | LISTENING | 20224  | 2214/master      | public/pickup                   |
| unix  | 2      | [ ACC ] | STREAM | LISTENING | 20228  | 2214/master      | public/cleanup                  |
| unix  | 2      | [ ACC ] | STREAM | LISTENING | 20231  | 2214/master      | public/qmgr                     |
| unix  | 2      | [ ACC ] | STREAM | LISTENING | 11278  | 1/systemd        | /run/lvm/lvmpolld.socket        |
| unix  | 2      | [ ACC ] | STREAM | LISTENING | 13838  | 1/systemd        | /var/run/dbus/system_bus_socket |
| unix  | 2      | [ ACC ] | STREAM | LISTENING | 20253  | 2214/master      | public/flush                    |
| unix  | 2      | [ ACC ] | STREAM | LISTENING | 20268  | 2214/master      | public/showq                    |
| unix  | 2      | [ ACC ] | STREAM | LISTENING | 13859  | 1/systemd        | /var/run/rpcbind.sock           |

--More--

## Résolution d'adresses Ethernet

Chaque protocole peut être encapsulé dans une **trame** Ethernet. Lorsque la trame doit être transportée de l'expéditeur au destinataire, ce premier doit connaître l'adresse Ethernet du dernier. L'adresse Ethernet est aussi appelée l'**adresse Physique** ou l'**adresse MAC**.

Pour connaître l'adresse Ethernet du destinataire, l'expéditeur fait appel au protocol **ARP**. Les informations reçues sont stockées dans une table. Pour visualiser ces informations, il convient d'utiliser la commande suivante :

```
[root@centos7 ~]# arp -a
```

```
? (10.0.2.3) at 52:54:00:12:35:03 [ether] on enp0s3  
gateway (10.0.2.2) at 52:54:00:12:35:02 [ether] on enp0s3
```

## Options de la commande

Les options de cette commande sont :

```
[root@centos7 ~]# arp --help  
Usage:  
arp [-vn]  [<HW>] [-i <if>] [-a] [<hostname>]           <-Display ARP cache  
arp [-v]      [-i <if>] -d  <host> [pub]                  <-Delete ARP entry  
arp [-vnD]  [<HW>] [-i <if>] -f  [<filename>]            <-Add entry from file  
arp [-v]  [<HW>] [-i <if>] -s  <host> <hwaddr> [temp]       <-Add entry  
arp [-v]  [<HW>] [-i <if>] -Ds <host> <if> [netmask <nmt>] pub    <--'-'  
  
-a                      display (all) hosts in alternative (BSD) style  
-e                      display (all) hosts in default (Linux) style  
-s, --set                set a new ARP entry  
-d, --delete              delete a specified entry  
-v, --verbose             be verbose  
-n, --numeric             don't resolve names  
-i, --device              specify network interface (e.g. eth0)  
-D, --use-device          read <hwaddr> from given device  
-A, -p, --protocol        specify protocol family  
-f, --file                read new entries from file or from /etc/ethers  
  
<HW>=Use '-H <hw>' to specify hardware address type. Default: ether  
List of possible hardware types (which support ARP):  
ash (Ash) ether (Ethernet) ax25 (AMPR AX.25)  
netrom (AMPR NET/ROM) rose (AMPR ROSE) arcnet (ARCnet)  
dlci (Frame Relay DLCI) fddi (Fiber Distributed Data Interface) hippi (HIPPI)  
irda (IrLAP) x25 (generic X.25) infiniband (InfiniBand)
```

## eui64 (Generic EUI-64)

<html>

Copyright © 2004-2017 Hugh Norris.<br><br> <a rel="license" href="http://creativecommons.org/licenses/by-nc-nd/3.0/fr/"></a><br />Ce(tte) oeuvre est mise à disposition selon les termes de la <a rel="license" href="http://creativecommons.org/licenses/by-nc-nd/3.0/fr/">Licence Creative Commons Attribution - Pas d'Utilisation Commerciale - Pas de Modification 3.0 France</a>.

</html>

From:

<https://www.ittraining.team/> - **www.ittraining.team**



Permanent link:

<https://www.ittraining.team/doku.php?id=elearning:workbooks:french:14:102:l113>

Last update: **2020/01/30 03:27**