

TP ARP

TP U6 SISR

SPINELLI Dylan
BTS SIO SISR | PARIS YNOV CAMPUS

Table des matières

1-	Introduction au protocole ARP	2
2-	Réalisation de la maquette sur Cisco	3
3-	Visualisation des requêtes ARP	4
4-	Test avec un routeur	5
5-	Observation des requêtes ARP sur Wireshark	6

1-Introduction au protocole ARP

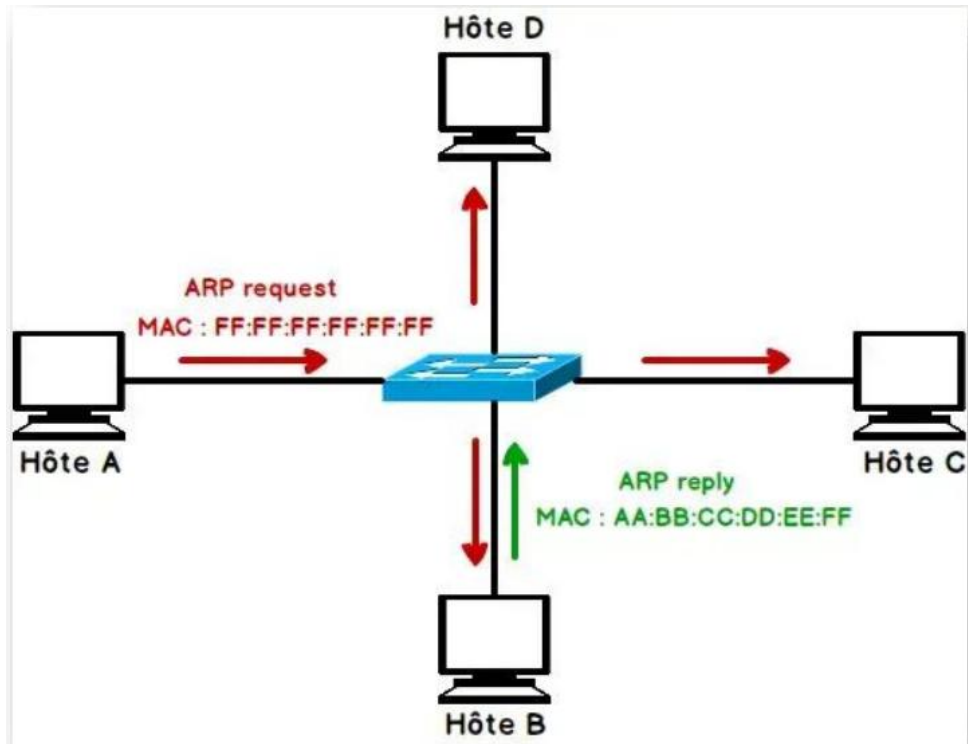
Le protocole **ARP** (*Address Resolution Protocol*) est un élément fondamental des réseaux informatiques. Son rôle principal est d'assurer la traduction entre une adresse logique de niveau 3 (l'adresse IP) et une adresse physique de niveau 2 (l'adresse MAC) du modèle OSI.

Concrètement, lorsqu'un équipement souhaite envoyer des données à une autre machine sur son réseau local, il connaît généralement son adresse IP, mais il a impérativement besoin de son adresse MAC pour construire la trame Ethernet. Le protocole ARP permet d'interroger le réseau via un message de diffusion (Broadcast) pour découvrir cette adresse MAC manquante.

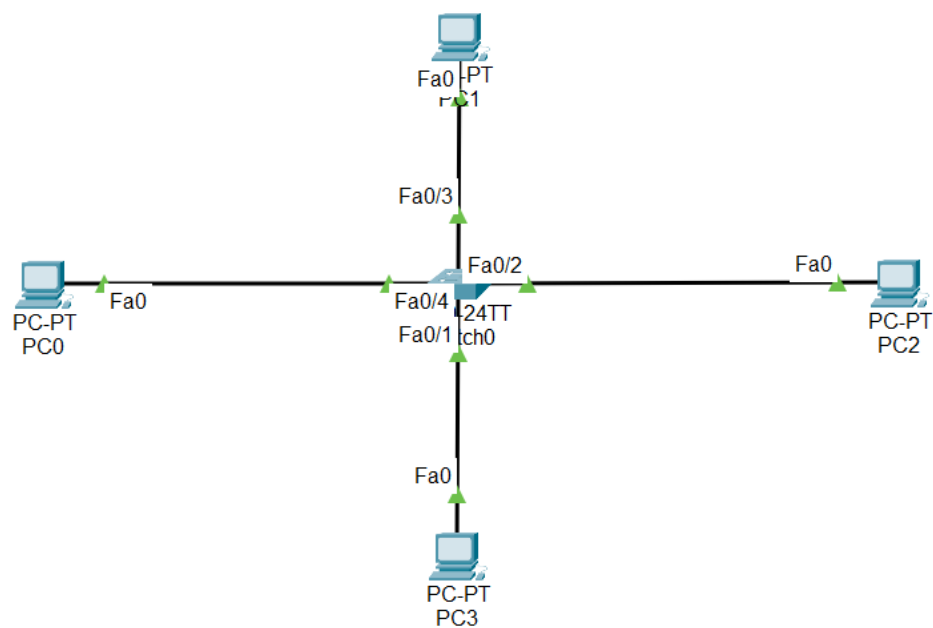
Ce compte-rendu a pour objectif d'illustrer le fonctionnement de ce protocole, d'abord à travers une simulation sur **Cisco Packet Tracer**, puis par l'analyse de trames réelles sur une machine virtuelle Debian à l'aide de l'outil **Wireshark**.

2- Réalisation de la maquette sur Cisco

Nous allons réaliser la maquette suivante sur cisco :



Voici la maquette réalisée dans Cisco :



3-Visualisation des requêtes ARP

Faisons un ping entre PC1 et PC2 pour visualiser les requêtes ARP :

The screenshot shows the Cisco Packet Tracer interface. The network topology consists of a central 24TT Switch0 connected to four PCs (PC0, PC1, PC2, PC3) via their respective Fa0/24 ports. PC1 is connected to Fa0/24, PC2 to Fa0/23, PC3 to Fa0/22, and PC0 to Fa0/21. The Event List panel on the right displays a list of events, including ARP requests and ICMP pings. The Event List table is as follows:

Vis	Time(sec)	Last Device	At Device	Type
	0.000	--	PC1	ICMP
	0.000	--	PC1	ARP
	0.001	PC1	Switch0	ARP
	0.002	Switch0	PC3	ARP
	0.002	Switch0	PC2	ARP
	0.002	Switch0	PC0	ARP
	0.003	PC2	Switch0	ARP
	0.004	Switch0	PC1	ARP
	0.004	--	PC1	ICMP
	0.005	PC1	Switch0	ICMP
	0.006	Switch0	PC2	ICMP
	0.007	PC2	Switch0	ICMP
	0.008	Switch0	PC1	ICMP

The bottom of the interface shows the Play Controls and a list of visible events, including ACL, Filter, ARP, BGP, Bluetooth, CAPWAP, CDP, DHCP, DHCPv6, DNS, DTP, EAPOL, EIGRP, EIGRPv6, FTP, H.323, HSRP, HSRPv6, HTTP, HTTPS, ICMP, ICMPv6, IPsec, ISAKMP, IoT, IoT TCP, LACP, LLDP, Meraki, NDP, NETFLOW, NTP, OSPF, OSPFv6, PAgP, POP3, PPP, PPPoE, PTP, RADIUS, REP, RIP, RIPng, RTP, SCCP, SMTP, SNMP, SSH, STP, SYSLOG, TACACS, TGP, TFTP, Telnet, UDP, USB, VTP.

On constate que les requêtes ARP sont bien envoyées à tous les PC du réseau.

4-Test avec un routeur

Faisons maintenant le test avec un routeur. On teste un ping de PC0 vers le routeur :

The screenshot shows the Cisco Packet Tracer interface. The main workspace displays a network topology with a central switch (Switch0) connected to four PCs (PC0, PC1, PC2, PC3) and a router (Router0). The interface includes a top menu bar, a toolbar, and a bottom status bar. On the right side, the 'Simulation Panel' is open, showing an 'Event List' table.

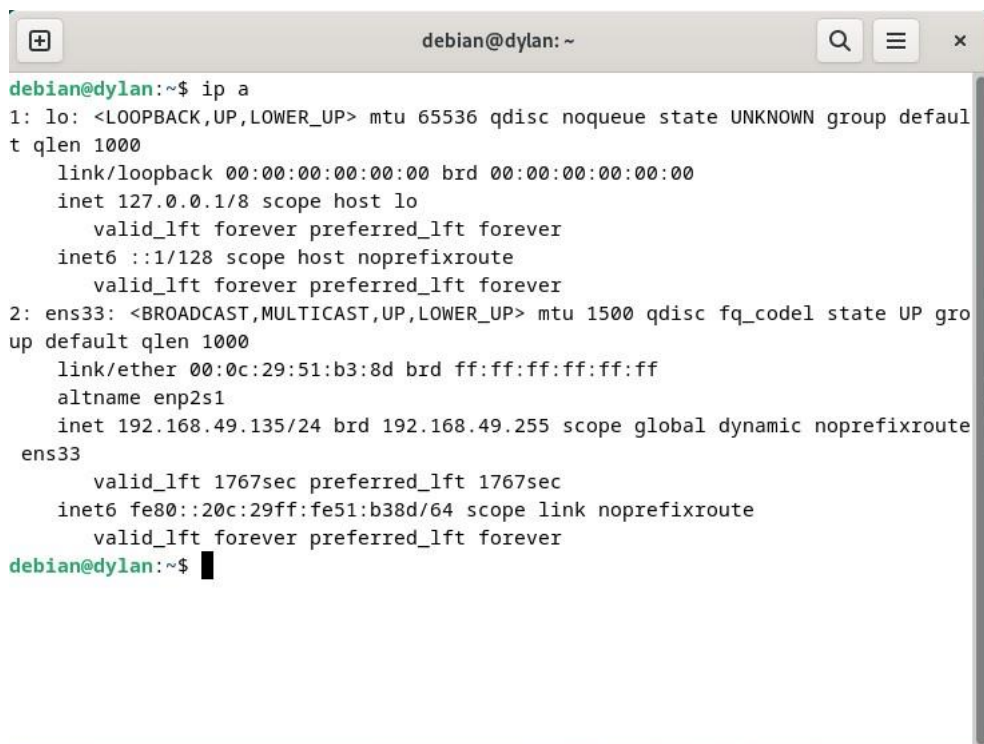
Vis.	Time(sec)	Last Device	At Device	Type
	0.000	--	PC0	ICMP
	0.000	--	PC0	ARP
	0.001	PC0	Switch0	ARP
	0.002	Switch0	PC3	ARP
	0.002	Switch0	PC2	ARP
	0.002	Switch0	PC1	ARP
	0.002	Switch0	Router0	ARP
	0.003	Router0	Switch0	ARP
	0.004	Switch0	PC0	ARP
	0.004	--	PC0	ICMP
	0.005	PC0	Switch0	ICMP

Below the event list, there are controls for 'Reset Simulation', 'Constant Delay', and 'Captured to: 0.005 s'. There are also 'Play Controls' buttons (play, pause, stop) and a section for 'Event List Filters - Visible Events' with a list of protocols and a 'Show All/None' button.

On constate que le routeur ne diffuse pas les requêtes ARP sur tous ses ports, contrairement au switch. En effet, le routeur bloque les requêtes de diffusion.

5-Observation des requêtes ARP sur Wireshark

On rappelle l'adresse IP de notre VM ainsi que celle du PC physique :



```
debian@dylan: ~  
debian@dylan:~$ ip a  
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000  
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00  
    inet 127.0.0.1/8 scope host lo  
        valid_lft forever preferred_lft forever  
    inet6 ::1/128 scope host noprefixroute  
        valid_lft forever preferred_lft forever  
2: ens33: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000  
    link/ether 00:0c:29:51:b3:8d brd ff:ff:ff:ff:ff:ff  
    altname enp2s1  
    inet 192.168.49.135/24 brd 192.168.49.255 scope global dynamic noprefixroute ens33  
        valid_lft 1767sec preferred_lft 1767sec  
    inet6 fe80::20c:29ff:fe51:b38d/64 scope link noprefixroute  
        valid_lft forever preferred_lft forever  
debian@dylan:~$
```

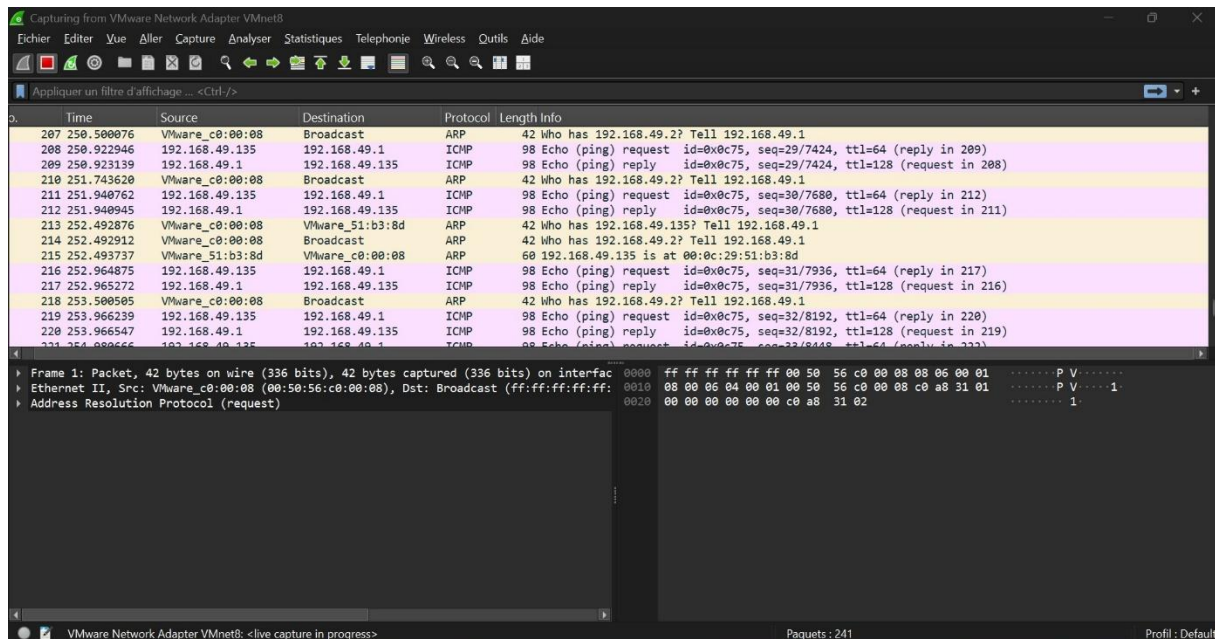
Carte Ethernet VMware Network Adapter VMnet8 :

```
Suffixe DNS propre à la connexion. . . . :  
Adresse IPv6 de liaison locale. . . . . : fe80::3d1b:6907:7874:124c%6  
Adresse IPv4. . . . . : 192.168.49.1  
Masque de sous-réseau. . . . . : 255.255.255.0  
Passerelle par défaut. . . . . :
```

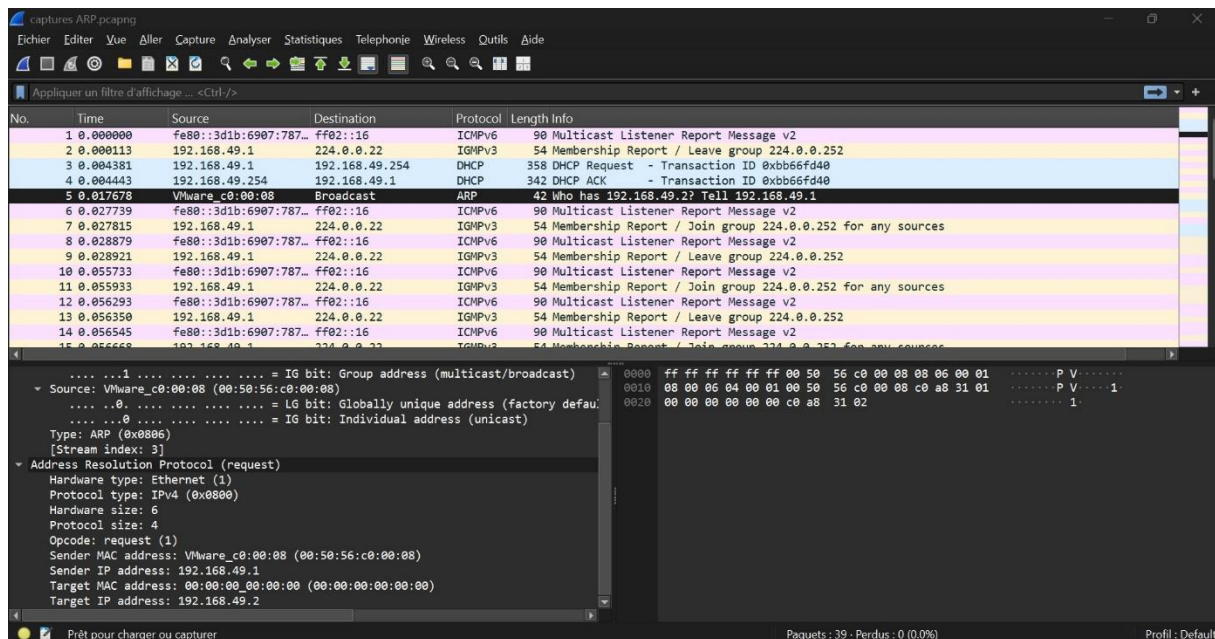
Voici la gateway de la VM :

```
debian@dylan:~$ ping 192.168.49.1
PING 192.168.49.1 (192.168.49.1) 56(84) bytes of data.
64 bytes from 192.168.49.1: icmp_seq=1 ttl=128 time=0.617 ms
64 bytes from 192.168.49.1: icmp_seq=2 ttl=128 time=0.959 ms
64 bytes from 192.168.49.1: icmp_seq=3 ttl=128 time=0.778 ms
64 bytes from 192.168.49.1: icmp_seq=4 ttl=128 time=1.01 ms
64 bytes from 192.168.49.1: icmp_seq=5 ttl=128 time=1.10 ms
64 bytes from 192.168.49.1: icmp_seq=6 ttl=128 time=1.29 ms
64 bytes from 192.168.49.1: icmp_seq=7 ttl=128 time=1.19 ms
^C
--- 192.168.49.1 ping statistics ---
7 packets transmitted, 7 received, 0% packet loss, time 6032ms
rtt min/avg/max/mdev = 0.617/0.994/1.294/0.217 ms
debian@dylan:~$ ip route show
default via 192.168.49.2 dev ens33 proto dhcp src 192.168.49.135 metric 100
192.168.49.0/24 dev ens33 proto kernel scope link src 192.168.49.135 metric 100
debian@dylan:~$ nano /etc/network/interfaces
debian@dylan:~$
```

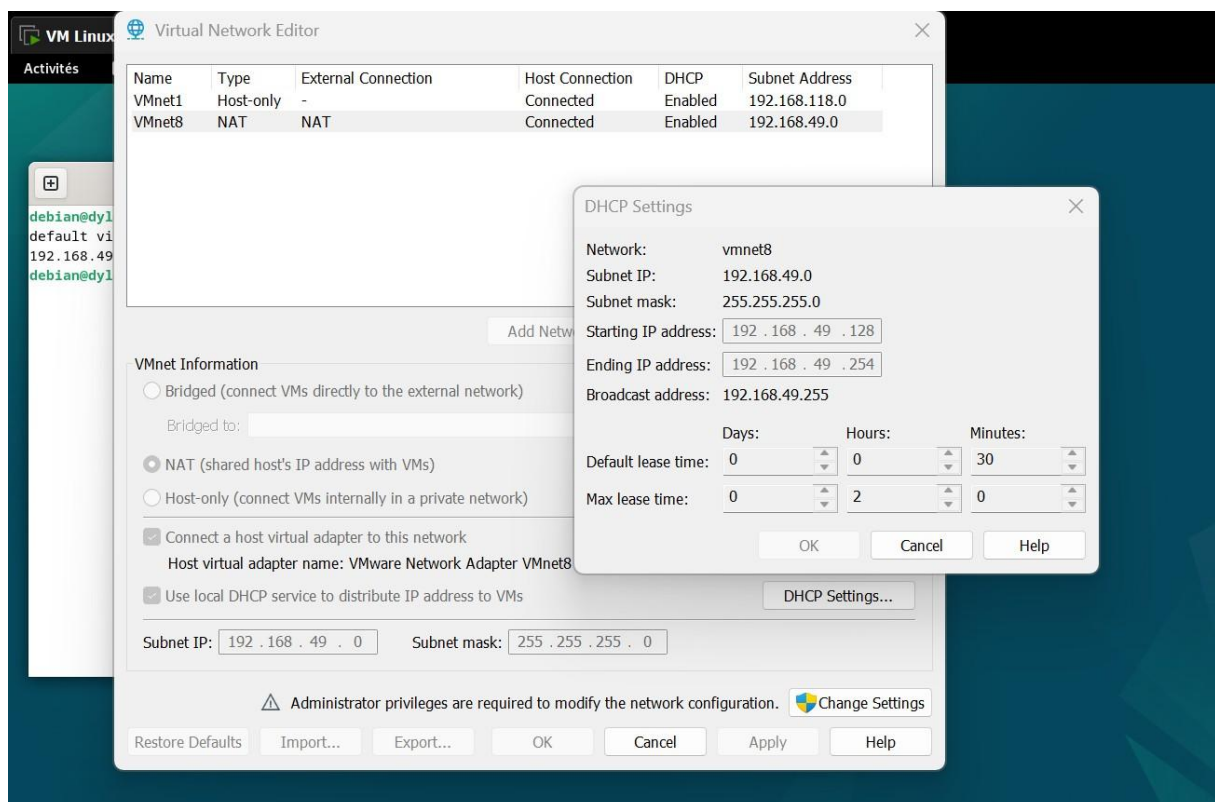
On peut observer les requêtes ARP sur wireshark :



- 1. La requête (ARP Request - Ligne jaune) :** On observe la trame « Who has 192.168.49.2? Tell 192.168.49.1 ». La machine hôte (.1) cherche l'adresse MAC de la passerelle NAT de VMware (.2). Dans les détails de la trame (en bas), on voit que l'adresse MAC de destination est ff:ff:ff:ff:ff:ff. C'est l'adresse de **Broadcast** (diffusion) : le message est envoyé à tout le réseau.
- 2. Les pings (ICMP - Lignes roses) :** On observe ensuite des échanges "Echo request / Echo reply" entre notre VM (192.168.49.135) et l'hôte (192.168.49.1), prouvant que la communication bidirectionnelle fonctionne une fois les adresses MAC connues.
- 3. La réponse (ARP Reply) :** Plus bas, on peut voir la VM informer le réseau de son adresse : « 192.168.49.135 is at 00:0c:29:51:b3:8d ».



Vérification de la passerelle (Gateway) : > La commande `ip route show` tapée sur la VM Debian confirme que la route par défaut (default via) passe par 192.168.49.2. C'est l'adresse de la passerelle virtuelle créée par le service NAT de VMware.



Cette configuration est confirmée par le *Virtual Network Editor* de VMware, qui montre que le sous-réseau 192.168.49.0/24 est bien géré en NAT, ce qui permet à la VM de communiquer avec l'extérieur en utilisant l'adresse IP de la machine physique.