M2PCCI

Cheick CISSOKO

Gaëtan LAGIER

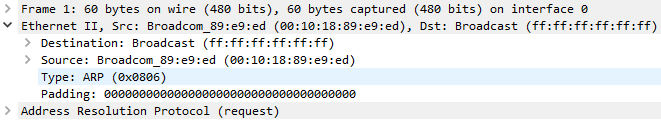
Eric THIERRY

**CR réseau TP2**

**2016/2017**

**LE PROTOCOLE ARP**

1) Ce qui permet d’identifier les paquets comme étant de type ARP est le champ “type” de la trame Ethernet. En effet lors de la capture de paquet avec l’application Wireshark, il est indiqué Type : ARP dans la trame Ethernet.



2) Dans la trame Ethernet, le paquet ARP se situe dans le champ Data.

3) Format de la trame ARP :

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Hardware  Type  2bytes | Protocol  Type  2bytes | Hardware  Size  1byte | Protocol  Size  1byte | Opcode  2bytes | @ Mac Source  6bytes | @ IP  Source  4bytes | @ Mac Destination  6bytes | @ IP  Destination  4bytes |

* Les champs Hardware (type t1 et size s1) et Protocole (type t2 et size s2) représentent respectivement le type t1 de taille s1 du protocole dont on veut la résolution en un autre type t2 de taille s2. Ces champs ne sont plus réellement utilisés aujourd’hui. En effet, le protocole ARP est exclusivement utilisé pour résoudre des adresses Ethernet en adresse IP, mais pour être universel, il a été créé ainsi.
* Le champ Opcode permet de préciser si le paquet est une requête ou une réponse : Request (x01), Reply (x02).
* Les champs restants sont respectivement l’adresse Ethernet et l’adresse IP de la machine qui envoie le paquet, puis l’adresse Ethernet et l’adresse IP de la machine qui est le destinataire de ce paquet.

4) Le niveau Ethernet ne connaît pas la taille des paquets. Il ne peut donc pas en déterminer la fin.

RMQ : Les octets de bourrage correspondent à la parie padding située après le protocole ARP.

5) A la réception du paquet, toute la partie data est transmise à la couche ARP par la couche Ethernet. Comme le protocole Ethernet ne connaît pas la taille du paquet, il transmet la partie data + le padding qu’il considère comme faisant partie des données. Il n’a donc pas connaissance de ce bourrage.

6) **Algorithme du protocole ARP**:

Tant que (vrai) faire :

Attendre (évènement) :

* **Si** évènement est « Question sur adresse Internet » (requête interne de IP vers ARP), consulter si la table ARP contient l’adresse Ethernet associée à l’adresse IP ciblée. Si la table contient l’adresse, celle ci est transmise au protocole IP, sinon, un paquet ARP request est envoyé sur le reseau pour trouver l’adresse demandée par IP. **Finsi**
* **Si** évènement est expiration du timer d’effacement associé à une entrée, un paquet ARP request est envoyé sur le reseau pour trouver l’association de IP et Ethernet. **Finsi**
* **Si** événement est réception requête, envoi d’un paquet ARP Reply avec l’adresse Ethernet correspondant à l’adresse IP de la machine. **Finsi**
* **Si** événement est réception requête qui demande l’adresse Ethernet d’une adresse IP présente dans la table ARP en « Published », envoie d’un paquet ARP avec l’association demandée. **Finsi**
* **Si** événement est réception d’un paquet ARP gratuit request et que l’adresse IP source de ce paquet est inconnue. Aucune action n’est entreprise. **Finsi**
* **Si** événement est réception d’un paquet ARP gratuit request et que l’adresse IP source de ce paquet correspond à l’adresse IP de la machine courante. Un message avertissant un conflit IP est affiché sur la console. Un paquet ARP gratuit reply est envoyé pour informer l’autre machine du conflit IP. **Finsi**
* **Si** événement est réception d’un paquet ARP gratuit request et que l’adresse IP est connue, dans ce cas l’adresse Ethernet correspondante est mise à jour dans la table ARP. **Finsi**

Fin Tant que.

7) Lors d’un premier ping de A vers B, A envoi un ARP request à B pour connaître son adresse Ethernet. B répond. Cependant, B profite de cet ARP request pour enregistrer l’adresse Ethernet de A avec son adresse IP, économisant une requête ARP de sa part pour un éventuel envoi futur.

8) Un paquet ICMP est envoyé par A vers B. Aucun paquet ARP request n’est envoyé par A car cette machine possède les informations liées à B dans sa table ARP suite au ping précédent. A la réception de ce paquet ICMP par B, un ARP request est envoyé vers A. A son tour, la machine A envoie un ARP reply vers B. En effet, B ne connaît plus A car il y a eu suppression de sa table ARP. Ainsi, pour connaître l’adresse de destination du paquet ICMP reply, B doit envoyer un ARP request vers A.

9) C’est le timer de ping (1 seconde) qui re-déclenche l’émission d’un paquet ARP request. Le protocole ARP envoi juste un paquet pour demander l’adresse. Si il n’a pas de réponse, il n’y a aucune réémission.

Il est possible de tester cela avec l’envoi d’un seul paquet par le ping : *ping –c 1*. Nous constatons qu’il y a alors **un seul** paquet ARP envoyé.

RMQ : timer ARP = 1200 secondes soit 20 minutes.

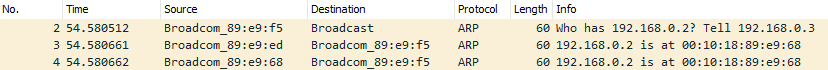
10) Lors de la configuration d’une interface, un paquet ARP gratuit est envoyé sur le réseau. Un paquet ARP gratuit est un paquet envoyé en broadcast (Ethernet) avec l’adresse IP de la machine ajoutée sur le réseau. Si une machine possède déjà cette adresse IP, elle va répondre à cet ARP gratuit request par un ARP gratuit reply pour annoncer que l’adresse IP associée à cette interface est déjà utilisée. Si il n’y a aucune réponse, pas de conflit IP.

Cet ARP gratuit permet alors de prévenir d’un conflit IP sur le reseau. Les deux machines concernées affiche un message d’alerte en console. De plus, si l’adresse IP est connue par une autre machine dans sa table ARP, l’adresse Ethernet correspondante est alors mise à jour lors de la réception de ce paquet ARP gratuit.

11) Deux paquet ARP reply sont envoyé. En effet lors d’un ping de C vers B, la machine C envoit un ARP request en broadcast (premier paquet sur la photo). B recoit ce paquet et répond avec un ARP reply (troisième paquet sur la photo). Cependant, la machine A possède la correspondance adresse IP et adresse Ethernet de B en published dans sa table. La clause published autorise alors A a répondre au ARP request envoyé par C (deuxième paquet sur la photo). Il y a donc à l’issu de ce ping deux paquets ARP reply :

1. Paquet envoyé par Machine B : source ethernet Machine B, source ARP Machine B
2. Paquet envoyé par Machine A : source ethernet Machine A, source ARP Machine B

Les adresse Ethernet source de ces deux paquets correspondent à l’adresse de la machine émétrice, cependant l’adresse source au niveau de la trame ARP est dans les deux cas l’adresse de la machine B, celle visée initiallement par le ARP request.



12) Il est possible de déclarer en published une association IP Ethernet/IP dans la

table ARP afin de récuperer des paquets qui ne nous sont pas destinés. En effet, lors de l’envoi d’un paquet, une machine quelquonque envoie un paquet ARP request, et la machine destination va répondre avec un ARP reply. Ensuite (il faut que se soit après le paquet ARP reply de la machine destination pour écraser l’association dans la table ARP) la machine pirate va repondre au ARP request en associant l’adresse IP de destination avec sa propre adresse Ethernet. Cela écrase donc la première association dans la table ARP de la machine source. Au prochain envoie d’un paquet, l’adresse Ethernet associée à l’adresse IP de destination sera donc l’adresse de la machine pirate.

13) Lors de la configuration d’une machine avec une adresse IP correspondant à une adresse IP déjà utilisée sur le reseau, un message d’alerte est affiché sur les deux machines correspondantes. Ce message indique que l’adresse IP est déjà utilisé ou que une machine tiers vient d’utiliser cette adresse pour configurer une de ses interface. Il correspond à un conflit d’IP sur le reseau :

Lors de la configuration de l’interface, un ARP gratuit est envoyé sur le reseau. Si une machine possède déjà cette adresse IP, un message d’alerte s’affiche sur sa console et elle envoie en réponse un ARP gratuit à destination de la machine « voleuse » pour la prévenir et llui faire afficher le message d’alerte.

14) La machine emetrice du paquet ICMP va envoyer une requete ARP, les deux machines avec la même adresse IP vont répondre en donnant leur adresse ethernet. La premiere qui repond se voit retourner un paquet icmp (ping) mais le reply de la deuxieme machine ecrase dans la table arp de la machine qui ping l’adresse ethernet de la premiere machine. Ainsi, les paquet icmp du ping suivant seront tous redirigés vers la deuxieme machine.

Au bout de 20 minutes, le timer à zero va reemettre une requete ARP, ainsi, on relance les dès pour savoir qui va recevoir les paquets icmp. En effet c’est toujours la machine qui répond en dernier qui « gagne ».

**Protocole ICMP :**

15) Ils sont identiques entre request et reply, et correspondent bien à ceux envoyés (dans le fichier texte).

Pour un ping, ils sont identiques entre une request et un reply pour cette request. Pour un même ping, identifier reste le même pour toutes les requetes et sequence number s’incrémente à chaque requète, identifiant ainsi le numèros de la requete pour le ping (processus) donné.

16) Pour calculer les deux octets du checksum :

* Additionner par mot de 16 bits l’ensemble des données du paquet ICMP (sauf les deux octets du checksum)
* Réaliser le complément à 1 de ce résultat

Par exemple dans notre cas, voici les données de deux paquets ICMP :

1er paquet request : 0x 08 00 **60 8B** 12 32 85 42 00 00 00 … (suivi d’une multitude de zéro). Les octets 60 8B sont les octets du checksum.

Il est donc nécessaire d’additionner (0x) 08 00 + 12 32 + 85 42 = 0x 9F74 (additionner les zéros suivants est bien évidemment inutile).

Puis le complément à 1 de 0x 9F74 = 0x 608B soit les octets du checksum.

2éme paquet reply : 0x 00 00 **68 8B** 12 32 85 42 … .

Addition (0x) : 00 00 + 12 32 + 85 42 = 0x 9774.

Complément à 1 = 0x 68 8B. Correct !

**Protocole DHCP :**

17) Un paquet DHCP request est envoyé par notre machine. En brodcast 255…… Un paquet DHCP ACK est alors envoyé par le routeur par deffaut avec son adresse IP et notre adresse IP a utiliser sur ce reseau et eventuellement le nom du reseau (pas dans notre cas).

Dans notre cas, il n’y a pas de difference de l’interface apres un dhclient em0 car il y a un adressage par defffaut avec une adresse.

Fragmentation

18) Dans le cas de l’envoi d’un paquet de 4600 bytes :

Envoi de quatre paquets :

Un premier paquet IP avec un identifiant n , une taille 1500(data+enteteIP) et un flag more fragment et un fragment offset à 0.

Un deuxieme paquet IP avec un identifiant n , une taille 1500 et un flag more fragment et un fragment offset à 1480.

Un troisieme paquet IP avec un identifiant n , une taille 1500 et un flag more fragment(x01) et un fragment offset à 2960.

Un paquet UDP, avec en IP un identifiant n , une taille 188(data+entete IP + entete UDP) et un flag x00 fragment et un fragment offset à 4440. Dans la trame udp, il y a la taille total du paquet non framenté de 4608 (data + entete UDP)

UDP sur le dernier paquet car c’est en réalité un seul gros paquet UDP. IP rassemble du premier au dernier avec entete.