ARP

Objectif

L'objectif de ce TP est de manipuler le protocole ARP afin de réaliser une attaque de type homme du milieu (*man-in-the-middle*).

1 Introduction

Le protocole nommé *An Ethernet Address Resolution Protocol* dit ARP, est défini par la RFC 826 par David C. Plummer depuis 1982. À cette époque, l'utilisation de l'Ethernet 10Mbit/s commençait à se répandre car les fabriquants avaient accès aux spécifications matériels définies par DEC, Intel et Xerox. Toutefois, chaque fournisseur de ce type de réseau inventait sa propre méthode de résolution d'adresses ce qui mena à des incompatibilités notoires lors des tentatives d'interconnexions de réseaux originaires de différents constructeurs. Le protocole ARP propose de résoudre ces problèmes de compatibilité en définissant, une fois pour toute, un protocole standard de résolution d'adresses.

2 Rappels sur ARP

Chaque carte connectée à un réseau possède un identifiant de 48 bits fixé à la fabrication. Cet identifiant est appelé Media Access Control (MAC) ou adresse physique.

Lorsqu'un périphérique est branché à un réseau Ethernet, il tente de communiquer avec un autre périphérique dont il connaît l'adresse IP. Ce dernier regarde dans sa table de correspondance MAC/IP, appelée table ou cache ARP, s'il connaît l'adresse MAC correspondant à l'IP à joindre. Si la relation est connue, il peut joindre directement la machine cible. Par contre, s'il ne connaît pas l'adresse MAC, le périphérique émetteur envoie une première requête ARP à tous les autres périphériques du sous-réseau (*broadcast*). Il est à noter que les routeurs ne laissent pas passer ces requêtes. Cette requête est nommée `who-has` et prend la valeur 1 dans le champ «opération» de l'entête du message ARP. En contre-partie, le périphérique dont l'adresse IP correspond à celle demandée répond directement à l'émetteur en *unicast*. L'opération de réponse est nommée `is-at` et prend la valeur 2. L'émetteur reçoit la réponse et met à jour sa table ARP.

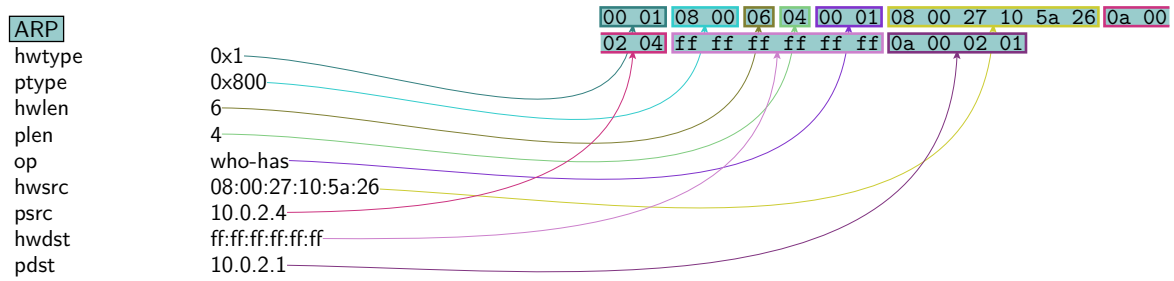

FIGURE 1 – Requête ARP (sans Ethernet) de type `who-has` de `08:00:27:10:5a:26` cherchant à joindre le périphérique à l'adresse IP : `10.0.2.1`.

3 Faiblesse du protocole

À l'époque de la conception du protocole ARP, la sécurité n'était pas une question centrale. En effet, lors d'une requête de type `who-has` n'importe quel périphérique du sous-réseau peut prétendre posséder l'adresse MAC correspondant à l'adresse IP demandée.

Exercice 1
Essayez d'identifier une attaque profitant de ce mécanisme et proposez une solution simple.

4 Étude du Cache ARP

Préparation : Utilisez deux postes capables de communiquer.

Exercice 2

1. *Lancez dans un terminal de l'une des deux machines la commande* `arp -a`. *Normalement, la table affichée devrait être vide.*
2. *Faites un* `ping` *vers l'autre machine puis affichez de nouveau le contenu du cache ARP. Normalement une entrée faisant la correspondance entre l'adresse IP de la machine et son adresse MAC doit apparaître.*
3. *Utilisez la commande* `arp -d` *suivie de l'adresse IP nouvellement inscrite dans le cache pour l'effacer.*

   Vous remarquerez que l'entrée ne disparaît pas, elle passe à l'état FAILED. Le noyaux Linux fait tout ce qu'il peut pour ne pas effacer les informations ARP car elles sont considérées précieuses. L'entrée ne disparaîtra complètement qu'au bout de quelques minutes (une dizaine tout au plus).
4. *Faites un* `ping` *entre les deux machines puis affichez la table ARP. Dé-configurez l'interface puis re-configurez là. Affichez la table ARP. Qu'en déduisez vous ?*

Il est également possible d'afficher la table ARP par la commande `ip neigh show`. Cette commande donne en plus l'état de l'entrée (REACHABLE, DELAY, STALE, INCOMPLETE, PROBE).

— INCOMPLETE - On est en attente d'une résolution d'adresse classique qui a été initiée mais pas complétée.
— REACHABLE - L'information dans la table est d'actualité, le voisin en question a été atteint il y a au plus quelques dizaines de secondes).
— STALE - L'entrée est obsolète on ne tentera pas de résoudre l'adresse tant que du trafic ne sera pas a envoyer à nouveau.
— DELAY - L'entrée est obsolète mais du trafic a été envoyé récemment au voisin et on est en attente de voir si le noyau arrive à voir que les messages sont reçus correctement (auquel cas on passera à l'état REACHABLE) ou pas (auquel cas on enverra une requête ARP en *unicast* et on passera à l'état PROBE).
— PROBE - L'entrée est obsolète et des requêtes ARP en *unicast* ont été envoyées. Si on a une réponse on passe en REACHABLE et sinon on envoie des requêtes ARP en *broadcast* et passe en INCOMPLETE.

5 Empoisonnement du cache ARP

Exercice 3
À partir de l'une des deux machines, essayez de fausser le cache de la seconde en utilisant Scapy. Montrez le résultat de votre attaque en utilisant la commande `arp` *sur la cible.*

Rappels :

— Scapy doit être lancé en `root` pour envoyer des messages ARP car le protocole fait le pont entre la couche 2 et la couche 3 ;
— Pour créer un paquet ARP : `a = ARP()` ;
— Utilisez `send(a)` pour envoyer.

6 L'attaque de l'homme du milieu

Préparation : Démarrez une machine supplémentaire.

L'exercice précédent nous a permis de montrer qu'il était possible d'empoisonner et de tromper le cache ARP d'un périphérique réseau. Nous allons mettre en œuvre une attaque plus sophistiquée nommée : l'attaque de l'homme du milieu [1].

Le principe de l'attaque est le suivant : l'attaquant souhaite intercepter le trafique entre deux machines du réseaux de manière transparente.

Exercice 4

Sans lire la suite du TP, réfléchissez à comment utiliser la technique d'empoisonnement de cache afin de réaliser une attaque de l'homme du milieu.

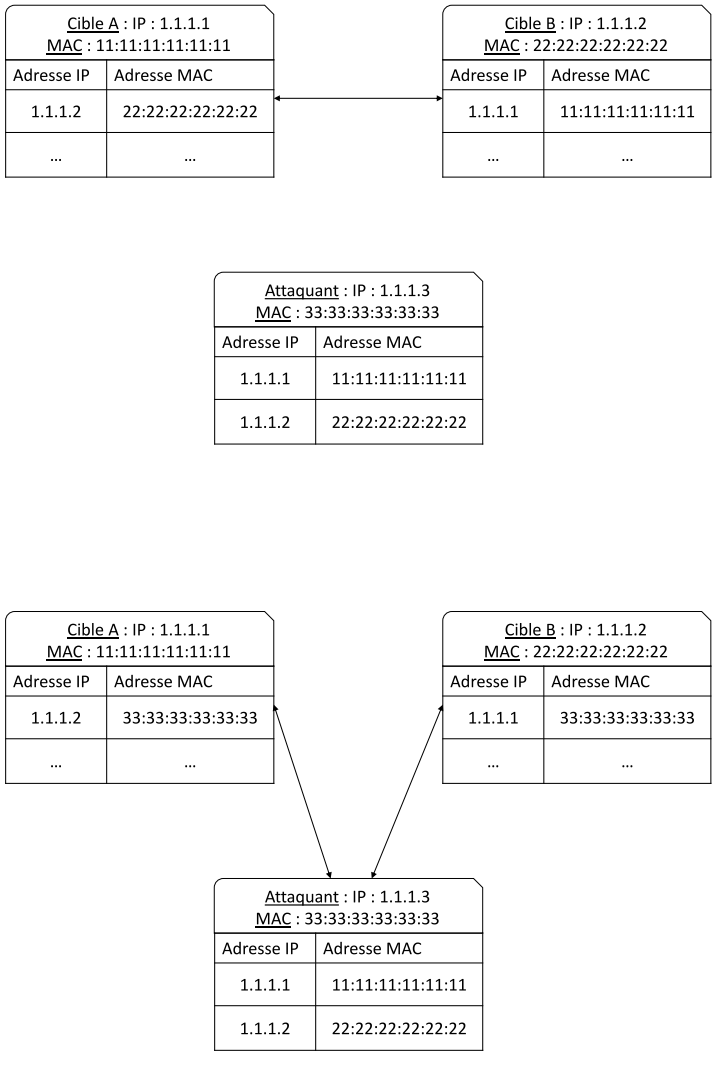

FIGURE 2 – La première partie de la figure représente une communication normale entre Cible A et Cible B. La seconde partie de la figure représente la même communication mais interceptée par Attaquant.

1. En V.O : *The Man-In-The-Middle attack*.

Exercice 5
Faites un `ping`*, ou ouvrez une socket entre deux machines afin de vérifier qu'elles peuvent communiquer.*

Avec la troisième machine, qui servira d'attaquant, détournez le trafic entre les deux premières à l'aide de Scapy. Modifiez leur cache ARP, en vous inspirant de l'illustration sur la page suivante.

Si l'attaque réussi vous pourrez espionner le trafic entre les deux machines cibles sur l'attaquant avec Wireshark par exemple. N'oubliez pas que la machine de l'attaquant doit rediriger le trafic entre les deux cibles avec la commande : `sysctl net.ipv4.ip_forward=1`.

Amélioration de l'attaque :

À ce point du TP, l'attaque ne fonctionne qu'à moitié, en effet, la commande `ping` lève des alertes de redirection. Ces dernières apparaissent régulièrement au début puis de moins en moins souvent. Elles sont dues à des messages ICMP redirect qui demande à la cible1 de ne pas passer par l'attaquant pour atteindre la cible2.

Exercice 6
Suite à un `ping` *entre les deux cibles, observez le trafic au niveau de la cible1. Qui est à l'origine de ces messages ICMP ? Pourquoi ces messages sont envoyés ? Regardez comment varie l'adresse MAC destination des* `ping request` *de la cible1 en fonction des messages ICMP et des messages ARP reçus.*

Ces messages sont très pénalisants pour notre attaque. On peut bloquer leur envoi sur une interface en ajoutant une règle à iptables, ou en mettant à zéro `sysctl net.ipv4.conf.{ethX,all}.send_redirects` , ethX étant l'interface par laquelle on ne veut pas qu'ils soient envoyés.

Exercice 7
Modifiez la configuration de l'attaquant pour que les envois de messages ICMP redirect cessent. Refaites un empoisonnement et un `ping`*. Il ne devrait plus y avoir d'alerte.*

Exercice 8
Faites un `traceroute` *entre les cibles lorsque l'empoisonnement n'a pas lieu puis lorsqu'il a lieu. Qu'est-ce que vous en déduisez ?*

7 Complément - Empoisonnement par création

De manière générale, on ne connaît pas les entrées présentes dans la table ARP d'une cible. Par conséquent, on débute par l'envoi d'une requête ARP `who-has`. Si on respecte le protocole, cette requête est envoyée en diffusion. Si on ne respecte pas le protocole et que l'on envoie la même requête mais en *unicast*, on diminue l'étendue de la visibilité, mais on envoie tout de même un message plutôt suspect car on ne respecte pas le protocole. Ainsi, pour maintenir l'attaque active, on envoie un premier message de type `who-has` en *unicast* au cas où l'entrée n'existerait pas, puis on envoie régulièrement des paquets de type réponse ARP `is-at` pour maintenir l'entrée dans le cache ARP de la cible.