DS: Sécurité et cryptographie**, Classes:** 2ing GTR**, A.U:** 2013/2014
Enseignant : Hdhili M. H
Documents non autorisés

Exercice 1 [12pts]:

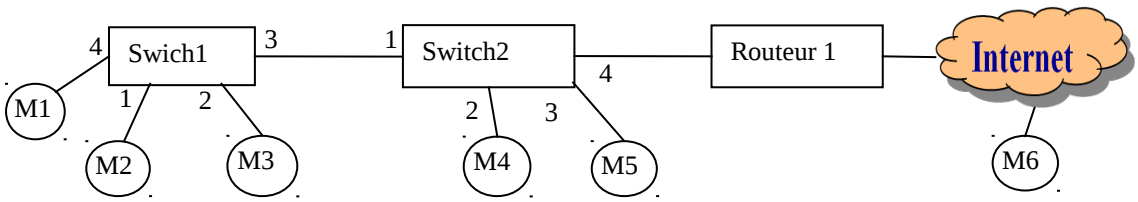
Soit les messages suivants décrivant le fonctionnement du protocole DHCP. “Hw Dest addr” et “Hw Src Address” représentent l'adresse MAC destination et l'adresse MAC source, respectivement. On donne seulement quelques détails du troisième message.

Filter: bootp Expression... Clear Apply Save

No.	Time	Hw Dest Addr	Hw Src Addr	Source	Destination	Protocol	Length	Info
1	0.000000	Broadcast	Giga-Byt_c9:28:31	0.0.0.0	255.255.255.255	DHCP	348	DHCP Discover - Transaction ID 0x7e2c562a
2	0.017351	Giga-Byt_c9:28:31	Comtrend_8a:5c:b9	192.168.1.1	192.168.1.2	DHCP	316	DHCP Offer - Transaction ID 0x7e2c562a
3	0.017722	Broadcast	Giga-Byt_c9:28:31	0.0.0.0	255.255.255.255	DHCP	373	DHCP Request - Transaction ID 0x7e2c562a
4	0.052182	Giga-Byt_c9:28:31	Comtrend_8a:5c:b9	192.168.1.1	192.168.1.2	DHCP	316	DHCP ACK - Transaction ID 0x7e2c562a

```
▼Bootstrap Protocol
  Message type: Boot Request (1)
  Hardware type: Ethernet (0x01)
  Hardware address length: 6
  Hops: 0
  Transaction ID: 0x7e2c562a
  Seconds elapsed: 0
▶Bootp flags: 0x0000 (Unicast)
  Client IP address: 0.0.0.0 (0.0.0.0)
  Your (client) IP address: 0.0.0.0 (0.0.0.0)
  Next server IP address: 0.0.0.0 (0.0.0.0)
  Relay agent IP address: 0.0.0.0 (0.0.0.0)
  Client MAC address: Giga-Byt_c9:28:31 (00:0d:61:c9:28:31)
  Client hardware address padding: 00000000000000000000
  Server host name not given
  Boot file name not given
  Magic cookie: DHCP
▶Option: (53) DHCP Message Type
▶Option: (61) Client identifier
▶Option: (50) Requested IP Address
▼Option: (54) DHCP Server Identifier
    Length: 4
    DHCP Server Identifier: 192.168.1.1 (192.168.1.1)
```

Soit le réseau Ethernet câblé suivant où les cercles sont des stations de travail. M5 implémente un serveur DHCP. Une description des attaques est donnée en annexe.



1) On suppose que M2 vient d'être configué par le serveur DHCP.
   a. Quels sont les noeuds qui ont reçu le DHCP discover? Expliquer.
   b. Quels sont les noeuds qui ont reçu le DHCP offer? Expliquer.
   c. Quels sont les noeud qui ont reçu le DHCP request? Expliquer.
   d. Quels sont les noeuds qui ont reçu le DHCP ack? Expliquer.
2) Supposons que le serveur DHCP sauvegarde les adresses MAC des neouds légitimes dans un fichier local. Un noeud ne peut être servi que si son adresse existe dans ce fichier.
   a. En se basant sur le sniffing passif, monter comment un attaquant lié à l'un des switch pourrait déterminer l'adresse IP du serveur DHCP et les adresses MAC des noeuds configurés par ce dernier?
   b. En déduire que cet attaquant pourrait bloquer la configuration, par DHCP, des noeuds du réseau. Expliquer comment doit-il procéder?
3) Donner une spécification des messages envoyés par un attaquant d'adresse MAC « MAC_ATTAQUANT » pour exécuter l'attaque « DHCP starvation » en se limitant aux champs

suivants : adresse MAC source (niveau liaison de donnée), adresse MAC source placé dans l'entête DHCP (niveau Application). Expliquer votre raisonnement?

4) Montrer comment peut-on utiliser un sniffer passif au niveau du serveur DHCP pour détecter qu'il y a des requêtes DHCP qui ont été lancées par un même nœud en spécifiant différentes adresses MAC.

5) Comment l'attaquant pourrait-il utiliser différentes adresses MAC (niveau liaison) pour réussir son attaque « DHCP starvation » ? (N. B : les trames sont commutées selon l'adresse MAC).

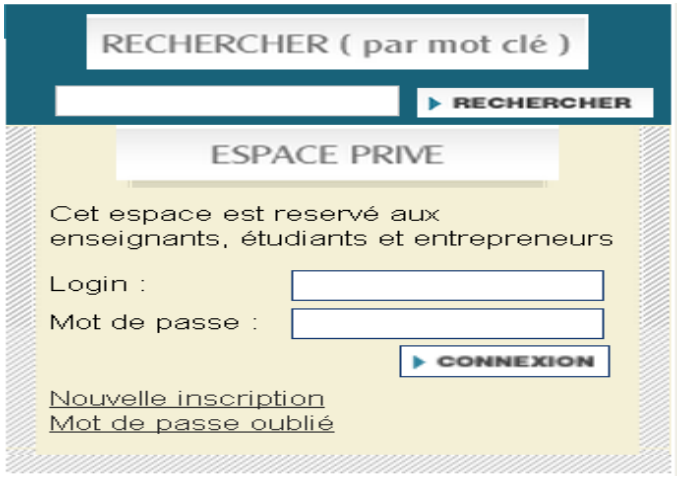
6) On exécute sur la machine victime (root@victime d'adresse IP 10.0.0.171) et sur la machine pirate (root@pirate d'adresse IP 10.0.0.227) les commandes décrite dans la figure 1. La commande *traceroute* permet d'afficher les nœuds traversés jusqu'au nœud destinataire donnée en paramètre de la commande. Le nœud 10.0.0.1 est une passerelle.

   a. Donner le nombre de nœuds intermédiaires entre la victime et la passerelle avant l'exécution de l'attaque. Expliquer ?
   b. Donner l'adresse MAC de la passerelle sauvegardé par la victime avant l'exécution de l'attaque.
   c. Donner l'adresse MAC de la passerelle sauvegardé par la victime après l'exécution de l'attaque.
   d. Donner le nombre de nœuds intermédiaires entre la victime et la passerelle après l'exécution de l'attaque. Expliquer ?
   e. En déduire l'attaque qui a été exécutée par le pirate. Expliquer.

```
root@victime [~] traceroute 10.0.0.1
traceroute to 10.0.0.1 (10.0.0.1), 30 hops max, 40 byte packets
 1  10.0.0.1  (10.0.0.1)  1.218 ms  1.061 ms  0.849 ms

root@victime [~] arp
Address          HWtype HWAddress               Flags Mask      Iface
10.0.0.1         ether   00:b0:c2:88:de:65       C                eth0
10.0.0.227       ether   00:00:86:35:c9:3f       C                eth0

root@pirate [~] Exécution d'une attaque
 0:0:86:35:c9:3f 0:60:8:de:64:f0 0806 42: arp reply 10.0.0.1 is-at 0:0:86:35:c9:3f
0:0:86:35:c9:3f 0:60:8:de:64:f0 0806 42: arp reply 10.0.0.1 is-at 0:0:86:35:c9:3f

root@victime [~] arp
Address          HWtype  HWAddress               Flags Mask      Iface
10.0.0.1         ether   00:00:86:35:c9:3f       C                eth0
10.0.0.227       ether   00:00:86:35:c9:3f       C                eth0

root@victime [~] traceroute 10.0.0.1
traceroute to 10.0.0.1 (10.0.0.1), 30 hops max, 40 byte packets
1 10.0.0.227 (10.0.0.227) 1.712 ms 1.465 ms 1.501 ms
2 10.0.0.1 (10.0.0.1) 2.238 ms 2.121 ms 2.169 ms
```

Figure 1 : commandes exécutées sur les machines pirate et victime

7) Donner trois attaques possibles exploitant la rubrique suivante d'une page web. Préciser comment chaque attaque va être lancée et donner une contre mesure



Exercice 2 [4pts]:

Soit **M** un message divisé en blocs $\{x_2,x_3,\ldots x_p\}$chacun de taille **n** bits et soit **K** une clé de même taille que les blocs (n bits). Soit $\{c_2,c_3,\ldots c_p\}$les cryptogrammes des blocs obtenus en appliquant la clé K aux blocs. Le chiffrement des blocs se fait selon le schéma suivant:

C_0= IV0 (valeur initiale) ; C_1= IV2 (valeur initiale) ;pour j de 2 à p, $c_j=E_K(C_{j-2} \oplus C_{j-1} \oplus x_j)$

La fonction E_K est inversible et son inverse est D_K c'est-à-dire que $D_K(E_K(x))=x$

1) Montrer que ce procédé de chifrement est symétrique?
2) Montrer qu'un même bloc clair n'est pas chiffré de la même façon s'il est dans deux blocs clairs différents
3) Supposons que le destinataire a reçu tous les blocs chiffrés mais il ne sait pas qu'il y a eu une erreur de transmission dans un seul bloc. Déterminer le nombre de blocs qui seront mal déchiffrés?
4) Prenons le cas où $E_K(x)=D_K(x) =K \oplus x$. Montrer que si un attaquant arrive à déchiffrer un seul bloc (par cryptanalyse), il pourra retrouver la clé de chiffrement.

Exercice 3 [4pts]:

1) Soit p=23, q=29, e=493 et n=p.q. Déterminer d (sachant que $\phi=(p-1)(q-1)$ et $e.d \equiv 1 \bmod \phi$) puis décrypter le cryptogramme 12.
2) Un Professeur **P** envoie, par mail, les notes de ses étudiants au service examen **SE** de l'école. Les clés publique de **P** et **SE** sont (e_P=3,n_P=55) et (e_{SE}=3,n_{SE}=33) respectivement.
   a. Déterminer la clé privée (d_P) de **P** et celle de **SE** (d_{SE}).
   b. Afin d'assurer la confidentialité, **P** chiffre chaque note avec la clé du **SE**. Donner le message chiffré correspondant à la note 12 ?
   c. Pour assurer l'authenticité et la confidentialité, P signe chaque note puis il la chiffre avec la clé du SE. SE reçoit le message 23. Donner la note correspondante ?

Algorithm Key generation for RSA public-key encryption

SUMMARY: each entity creates an RSA public key and a corresponding private key. Each entity A should do the following:

1. Generate two large random (and distinct) primes p and q, each roughly the same size.
2. Compute $n = pq$ and $\phi = (p-1)(q-1)$. (See Note 8.5.)
3. Select a random integer e, $1 < e < \phi$, such that $\gcd(e, \phi) = 1$.
4. Use the extended Euclidean algorithm (Algorithm 2.107) to compute the unique integer d, $1 < d < \phi$, such that $ed \equiv 1 \pmod{\phi}$.
5. A's public key is (n, e); A's private key is d.

Algorithm RSA public-key encryption

SUMMARY: B encrypts a message m for A, which A decrypts.

1. *Encryption.* B should do the following:
   (a) Obtain A's authentic public key (n, e).
   (b) Represent the message as an integer m in the interval $[0, n-1]$.
   (c) Compute $c = m^e \bmod n$ (e.g., using Algorithm 2.143).
   (d) Send the ciphertext c to A.
2. *Decryption.* To recover plaintext m from c, A should do the following:
   (a) Use the private key d to recover $m = c^d \bmod n$.

Annexe

ARP spoofing :
Cette attaque corrompt le cache de la machine victime MV. Le pirate envoie des paquets ARP réponse à MV indiquant que la nouvelle adresse MAC correspondant à l'adresse IP d'une autre machine destinataire MD est la sienne. La machine du pirate recevra donc tout le trafic à destination de MD, il lui suffira alors d'écouter passivement le trafic (et/ou le modifier). Il routera ensuite les paquets vers la véritable destination.

DHCP starvation :
L'attaquant consomme (réserve) toute la plage d'adresse IP du serveur DHCP pour lui. Pour ce faire, il lance plusieurs requêtes DHCP (requêtes diffusées dans le réseau local) en changeant à chaque fois son adresse MAC pour que le serveur DHCP le considère comme un nouveau client et le sert.

TCP syn flooding
Nous avons vu qu'une connexion TCP s'établie en trois phases (TCP Three Way Handshake). Le SYN Flooding exploite ce mécanisme d'établissement en trois phases. Les trois étapes sont l'envoi d'un SYN, la réception d'un SYN-ACK et l'envoi d'un ACK. Le principe est de laisser sur la machine cible un nombre important de connexions TCP en attentes. Pour cela, le pirate envoie un très grand nombre de demandes de connexion (flag SYN à 1), la machine cible renvoie les SYN-ACK en réponse au SYN reçus. Le pirate ne répondra jamais avec un ACK, et donc pour chaque SYN reçu la cible aura une connexion TCP en attente. Etant donné que ces connexions semi-ouvertes consomment des ressources mémoires au bout d'un certain temps la machine est saturée et ne peut plus accepter de connexion. Ce type de déni de service n'affecte que la machine cible.

Faux seveurs DHCP
L'attaquant prend le rôle d'un serveur DCHP. Il répond avec un DHCPOFFER en donnant de faux paramètres IP à l'utilisateur. L'attaquant peut effectuer un déni de service sur le serveur légitime afin qu'il n'interfère pas avec cette attaque. Pour limiter l'impact de cette attaque, le DHCP snooping est utilisée dans certains types de commutateurs. Ce mécanisme permet la mise en place d'une liste de ports sur le commutateur sur lequel se trouvent les "trusted dhcp server".

ARP flooding
L'attaquant inonde le réseau avec des trames ARP « who is » en changeant à chaque fois son adresse MAC. Les commutateurs ajoutent dans leurs tables de commutation les adresses MAC observé en correspondance avec leurs ports d'entrée.

Tiny fragments :
L'attaque consiste à fragmenter sur deux paquets IP une demande de connexion TCP. Le premier paquet IP de 68 octets ne contient comme données que les 8 premiers octets de l'en-tête TCP (ports source et destination ainsi que le numéro de séquence). Les données du second paquet IP renferment alors la demande de connexion TCP (flag SYN à 1 et flag ACK à 0).
Or, les filtres IP appliquent la même règle de filtrage à tous les fragments d'un paquet. Le filtrage du premier fragment (Fragment Offset égal à 0) déterminant cette règle elle s'applique donc aux autres (Fragment Offset égal à 1) sans aucune autre forme de vérification. Ainsi, lors de la défragmentation au niveau IP de la machine cible, le paquet de demande de connexion est reconstitué et passé à la couche TCP. La connexion s'établit alors malgré le filtre IP.

Fragment overlapping :
Toujours d'après la RFC 791 (IP), si deux fragments IP se superposent, le deuxième écrase le premier. L'attaque consiste à forger deux fragments d'un paquet IP. Le filtre IP accepte le premier de 68 octets (voir Tiny Fragments) car il ne contient aucune demande de connexion TCP (flag SYN = 0 et flag ACK = 0). Cette règle d'acceptation s'applique, là encore, aux autres fragments du paquet. Le deuxième (avec un Fragment Offset égale à 1) contenant les véritables données de connexion est alors accepté par le filtre IP. Ainsi, lors de la défragmentation les données du deuxième fragment écrasent celles du premier à partir de la fin du 8ème octet (car le fragment offset est égal à 1). Le paquet réassemblé constitue donc une demande de connexion valide pour la machine cible. La connexion s'établit malgré le filtre IP.