Sophie VALENTIN, Mathieu BIVERT

Configuration et sécurisation de services réseaux
3 février 2013

Professeur : Bruno MARTIN

Table des matières

1 Topologie

2 Mise en place d'une passerelle et accès à Internet

Une passerelle (gateway) est un homme du milieu reliant deux réseaux distincts. Dans le cas présent, la machine *passerelle* doit faire communiquer les deux réseaux SLAN (192.168.1.0/24) et LAN Travaux Pratiques (192.168.2.0/24).

La passerelle doit être capable de transmettre des paquets IP :

```
(passerelle)# echo 1 > /proc/sys/net/ipv4/ip_forward
```

Afin de maintenir l'*IP forwarding* après un reboot de la machine *passerelle*, on décommente dans le fichier */etc/sysctl.conf* la ligne suivante :

```
#net.ipv4.ip_forward=1
```

L'IP Masquerade (Network Address Translation) doit être activée. Cette fonctionnalité modifie les entêtes IPs du trafic passant par *passerelle* afin de rendre invisibles, au niveau IP, les machines de LAN Travaux Pratiques depuis l'extérieur. Ici on utilise du NAT dit de source utilisant la chaîne *POSTROUTING* puisqu'on modifie les adresses sources du paquet.

```
(passerelle)# iptables -t nat -A POSTROUTING -o eth0 -s 192.168.2.0/24 -j MASQUERADE
```

Note : L'interface *eth0* est connectée au réseau SLAN.

Enfin, syslogd doit être activé afin de logger les activités d'iptables

```
(passerelle)# apt-get install inetutils-syslogd
(passerelle)# edit /etc/syslog.conf # logs dans /var/log/kernel.log
(passerelle)# services syslog
```

2.1 Tests

On choisit un client, par exemple *client-bsd*. On lui retire l'interface réseau connectée à SLAN *em0*, et on s'assure que la machine est bien connectée sur LAN Travaux Pratiques via *em1*, et qu'elle peut communiquer avec la passerelle.

```
(client-bsd)# ifconfig em0 down
(client-bsd)# ifconfig em1
(client-bsd)# ping passerelle.cs.sr
```

Puis, on configure la table de routage du client afin que la route par défaut passe par *passerelle* et on vérifie :

```
(client-bsd)# netstat -r
```

```
Internet:
Destination        Gateway            Flags    Refs      Use  Netif Expire
default            192.168.2.1        UGS         0        4    em1
localhost          link#4             UH          0      160    lo0
192.168.1.3        link#1             UHS         0        0    lo0
192.168.2.0        link#2             U           0        7    em1
obsd.localdomain   link#2             UHS         0        0    lo0
```

FIGURE 1 – Table de routage de *client-bsd*

Enfin, on s'assure qu'il est possible de contacter le serveur et d'atteindre Internet.

```
(client-bsd)# ping -c3 google.fr
```

On vérifie les logs sur la passerelle :

```
(passerelle)# tail -f /var/log/kernel.log
```

nmap ? Faire peut-être une partie "outils d'attaque et... d'audit" où on explique l'utilisation de nmap, metasploit, vas (TP de demain) et en quoi ils nous aident à sécu notre propre réseau.

3 Mise à disposition d'un accès distant sur la machine et sécurisation

Telnet, SSH, VPN Expliquer comment mettre en place TCP Wrapper avec Telnet mais pas sécurisé. essai ssh avec mitm ? (normalement il affiche un message du genre "SOMEONE MAY BE ON THE CABLE !")

3.1 Serveur Telnet et attaques de *sniffing*

On commence par installer un serveur *telnet* afin d'offrir un premier accès distant à la machine *passerelle*.

On n'utilisera pas ici le démon classique de telnet mais un démon générique nommé *inetd* permettant de minimiser le nombre de processus lancés. Quand *inetd* reçoit une demande de connexion, il lance le processus serveur adéquat.

Pour ce faire, on installe le serveur telnet.

```
(passerelle)# apt-get install telnetd
```

Puis, on modifie le fichier */etc/inetd.conf* pour y ajouter le service *telnet* dans la section des services standard.

```
(passerelle)# grep '^telnet' /etc/inetd.conf
telnet stream tcp nowait root /usr/sbin/tcpd in.telnetd
```

On va maintenant tenter d'intercepter le couple *login/passwd* lors d'une connexion *telnet* entre *client-bsd* et *passerelle*.

Pour opérer, on utilise l'outil *ettercap* depuis la machine BackTrack. Il faut le lancer ainsi :

```
(backtrack)# ettercap -G
```

Dans le menu, on clique tout d'abord sur *Sniff* et on sélectionne *Unified sniffing* avant de choisir l'interface *eth1* qui est sur LAN Travaux Pratiques.

Dans le menu *Hosts*, on lance un scan du réseau local puis depuis le même menu, on affiche la liste des hôtes.

On reconnaît les adresses IP de *passerelle* et *client-bsd*. Il suffit donc de spécifier que ce sont nos deux cibles avec les boutons *TARGET 1* et *TARGET 2*.

Comme il s'agit d'un réseau commuté, on peut empoisonner les tables ARP (*ARP poisoning*) avant de lancer le *sniffing* pour récupérer le traffic entre les deux cibles.

Dans le menu *Mitm*, on sélectionne donc *ARP Poisoning* avant de sélectionner dans le menu *Start* le choix *Start Sniffing*.

La machine *client-bsd* se connecte avec *telnet*.

Et on constate qu'il est extrêmement facile d'obtenir le couple *login/password* en clair :

On ne veut donc pas proposer un service d'accès à distance non sécurisé, c'est-à-dire basé sur des communications transmises en clair sur le réseau.

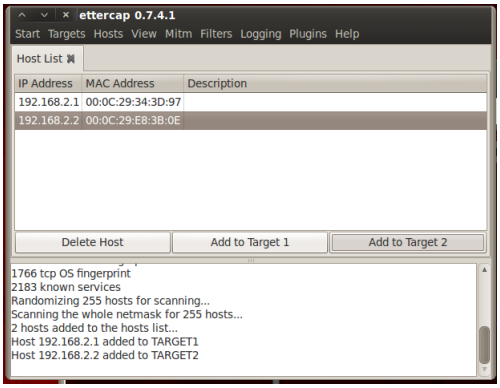

FIGURE 2 – Hôtes détectés par le scan

FIGURE 3 – Connexion à *passerelle*

FIGURE 4 – Couple en clair

3.2 Serveur SSH et optimisations

SSH (Secure SHell) joue un rôle similaire à telnet, mais contrairement à ce dernier, il intègre des mécanismes de sécurité, comme l'encryption des communications.

On installe puis démarre un serveur ssh sur la plateforme :

```
(passerelle)# apt-get install openssh-server
(passerelle)# service ssh start
```

On teste une connexion avec mot de passe (l'option *-v* est utilisé pour montrer la différence avec une connexion à base de clefs, voir plus bas)

```
(client-bsd)$ ssh -v cssr@passerelle.cs.sr
<METTRE L'OUTPUT ICI>
```

Afin d'éviter d'avoir à entrer le mot de passe à chaque connexion, on peut utiliser un système à base de clefs. On génère un couple clef privée/publique de type RSA sur le client :

```
(client-bsd)$ ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/home/cssr/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/cssr/.ssh/id_rsa
Your public key has been saved in /home/cssr/.ssh/id_rsa
The key fingerprint is:
f4:3e:54:9f:df:eb:72:cc:7a:da:29:a5:79:58:0c:44 cssr@client-bsd
The key's randomart image is:
+--[ RSA 2048]----+
|            .E   |
```

```
|             .   |
|        .   o    |
|       . . . o . |
|        S o   =  |
|         o     =.|
|          o   O o|
|            . *.*o|
|              oX= |
+-----------------+
```

Deux fichiers sont alors générés :

id_rsa.pub Clé publique devant être connue du serveur ;

id_rsa Clé privée ne devant pas être dévoilée.

Côté serveur, on doit maintenant ajouter la clef publique sur le serveur. Ssh cherche dans le répertoire *$HOME/.ssh/* les fichiers dontle nom commence par *authorized_keys*. Ceux-ci doivent contenir une clef par ligne.

On transmet ainsi la clé publique sur le serveur, par exemple avec *scp* :

```
(client-bsd)$ scp .ssh/id_rsa.pub cssr@passerelle.cs.sr:~/.ssh/authorized_keys2
```

On prend soin d'ajuster les permissions du fichier :

```
(passerelle)# chmod 700 authorized_keys2
```

Enfin, on s'assure que l'authentification par clefs marche bien, et qu'il est difficile de récupérer les informations sur la connexion avec une attaque mitm.

4 Configuration du serveur web

4.1 Tests

5 Configuration serveur (smtp+imaps) et client (gpg) email

La figure 5 donne un exemple simple de routage d'email, faisant intervenir 3 pièces logicielles :

MTA Mail Transfer Agent (eg. serveur SMTP), qui route les emails de domaines en domaines jusqu'à arriver à bonne destination ;

MDA Mail Delivery Agent (eg. serveur IMAP(s)), qui délivre les emails aux MUAs qui lui demandent ;

MUA Mail User Agent c'est le client email, dont le rôle principal est de récupérer les emails depuis un MDA, et d'en envoyer à un MTA ;

D'autres agents optionnels peuvent venir s'y greffer.

En pratique, on installe et configure postfix sur *passerelle*.

5.1 Tests

6 Configuration VPN

6.1 Tests

7 OpenVAS & Metasploit

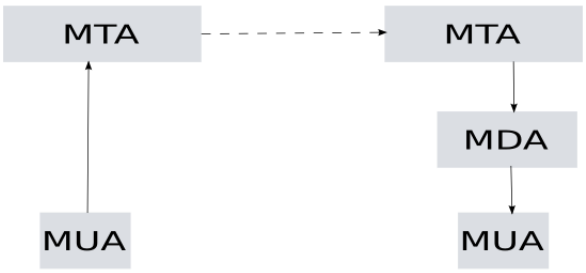

FIGURE 5 – Un MUA envoie un email à un MTA, qui le forward jusqu'à un MTA final, qui le transmet à un MDA. Enfin, le MUA du destinataire récupère l'email depuis ce MDA