Travaux pratiques

1. Contrôle des fichiers

But : contrôler les fichiers et les droits associés.

1. Pour chaque fichier il est possible d'établir une somme de contrôle, ou checksum. La commande **md5sum** calcule la somme de contrôle d'un fichier au format MD5. Calculez le checksum du fichier /etc/passwd :

```
# md5sum /etc/passwd
8737073e74b3035af61911a6a4323b53  /etc/passwd
```

2. Modifiez une information, même simple, par exemple un commentaire, dans /etc/passwd, et recalculez le MD5. Ici un « é » a été remplacé par un « e » :

```
$ md5sum /etc/passwd
a7450bcc205cbee80c429d9522c43f53  /etc/passwd
```

La somme est totalement différente, ce qui montre que le fichier a été modifié. Si vous conservez dans un fichier MD5SUM les sommes de contrôle d'origine, vous pourrez ainsi détecter les modifications de manière bien plus fiable qu'avec la date.

3. Lors de la recherche suivante :

```
# find ! -user root -perm -4000
```

Vous tombez sur un programme /usr/bin/lppasswd particulier :

```
-rwsr-xr-x   1 lp       sys         14112 avr 18 00:55
./bin/lppasswd
```

Le droit SUID est présent alors que le propriétaire n'est pas root. Pourquoi ?
Le propriétaire des fichiers manipulés par lppasswd est l'utilisateur lp. L'usage du SUID-bit n'est pas conditionné par le propriétaire root. Dans ce cas, la commande **lppasswd** sera exécutée en tant que utilisateur lp.

2. Sécurité des utilisateurs

But : gérer la politique de sécurité des mots de passe les utilisateurs.

1. Il n'y a actuellement aucune politique de sécurité des utilisateurs, ils sont tous libres de modifier leur mot de passe à leur guise. Vous avez utilisé la commande **chage** pour modifier la politique de sécurité pour les utilisateurs existants. Vous allez maintenant modifier cette politique pour tous les utilisateurs futurs. Modifiez le fichier **`/etc/login.defs`** pour définir un changement de mot de passe tous les 40 jours, l'interdiction de changer de mot de passe avant 7 jours et que l'utilisateur soit averti 10 jours avant :
PASS_MAX_DAYS 40
PASS_MIN_DAYS 7
PASS_WARN_AGE 10

2. Vous voulez mettre en place, en tant que root, une modification à la volée des mots de passe de tous les utilisateurs, via un script. Le mot de passe de 8 caractères est généré par pwck. Les UID des utilisateurs commencent à 1000. Construisez d'abord une boucle qui va lire le fichier **`/etc/passwd`** :

```
while read line
```

```
do
...
done </etc/passwd
```

3. Pour chaque ligne, isolez le login et l'UID :

```
user=$(echo $ligne| cut -d: -f1)
uid=$(echo $ligne| cut -d: -f3)
```

4. Générez un mot de passe sécurisé sur 8 caractères et placez celui-ci dans une variable intitulée `pass` :

```
pass=$(pwgen -s -1)
```

5. Modifiez le mot de passe de l'utilisateur :

```
echo $pass | passwd --stdin $user
```

Le script complet est :

```
while read line
do
user=$(echo $ligne| cut -d: -f1)
uid=$(echo $ligne| cut -d: -f3)
pass=$(pwgen -s -1)
echo $pass | passwd --stdin $user
done </etc/passwd
```

3. Sécurité générale du système

But : éviter les rootkits, les virus et contrôler les limites.

1. La commande **chkrootkit** permet la détection des rootkits les plus courants sur un système. Vous décidez de placer la commande en crontab. Elle sera exécutée tous les jours à 1h00 du matin. Créez un fichier cron_rootkit dans /etc/cron.d et placez-y la ligne suivante :

```
0 1 * * * /sbin/chkrootkit >/tmp/rootkit
```

2. Les résultats sont placés dans /tmp/rootkit. Ce n'est pas une bonne idée. Le mieux serait de les recevoir par mail :

```
0 1 * * * /sbin/chkrootkit | mail user@server -s "Resultats rootkit $(date)"
```

3. Effectuez une mise à jour de la base des antivirus de clamav tous les deux jours à 2h00 du matin, selon le même principe :

```
0 2 */2 * * freshclam >/dev/null 2>&1
```

4. Vos utilisateurs ont tendance à consommer trop de ressources. Limitez les personnes du groupe users à 256 processus. Modifiez le fichier **`/etc/security/limits.conf`** en rajoutant la ligne suivante :

```
@users        hard    nproc           256
```

5. Quelle que soit la distribution, efforcez-vous de mettre à jour aussi souvent que possible vos packages pour une question de sécurité :
Debian et Ubuntu : `apt-get upgrade`

Red Hat et Fedora : `yum update`

openSUSE : `zypper update`

4. Sécurité réseau

But : vérifier les ports, la configuration du pare-feu et des TCP Wrappers.

1. Lancez nmap sur votre propre machine. Comparez les résultats avec ceux de netstat -A inet -a. Quelle est la principale différence ?
   Nmap scanne les ports ouverts sur votre machine, pas les connexions sortantes. netstat donne la liste des ports locaux et distants ouverts, et les processus associés.
   Mais surtout nmap fournit quand il peut le déterminer le nom réel et les versions des services et du système d'exploitation testés. Nmap est un outil de sécurité et aussi un outil de hacking.

2. Un produit comme Wireshark permet de « sniffer » un réseau : il est à l'écoute de tout le trafic, peut l'enregistrer, le filtrer, etc. Deux services tournent sur un serveur : telnetd et sshd. Si vous écoutez le trafic de et vers ce serveur, vous devriez remarquer quelque chose. Quoi ?
   Le trafic à destination et depuis le port 22 est illisible : il est crypté. Il n'y a pas de moyen simple de récupérer ou d'analyser le contenu. Par contre le trafic du port 23 est totalement lisible : la communication n'est pas sécurisée. Tout passe en clair, dont les mots de passe. Que devez-vous faire ?
   Désactivez le service telnetd.

3. Vous avez le choix entre les TCP Wrappers et mettre en place une protection de type pare-feu. Quel principe doit guider votre solution ?
   Le pare-feu netfilter travaille au niveau des protocoles : il filtre les adresses, les ports, les protocoles. Le contrôle d'accès se fait au niveau du noyau.
   Les TCP Wrappers sont orientés, comme leur nom l'indique, TCP, donc ils agissent sur la couche de transport, et les services binaires : le contrôle d'accès a lieu au niveau du service.

4. Le fichier **`/etc/hosts.allow`** contient :

   ```
   sshd : ALL
   ```

   Le fichier **`/etc/hosts.deny`** contient :

   ```
   sshd : ALL EXCEPT 192.168.1.25
   ```

   Vous remarquez que tout le monde se connecte via ssh. De toute évidence il y a une erreur dans la configuration : seul 192.168.1.25 devrait y parvenir. Le premier fichier est interprété en premier. Corrigez.
   Vous avez plusieurs solutions :
   - Supprimer la ligne de hosts.allow et laisser hosts.deny intact.
   - Modifier hosts.allow :

     ```
     sshd : 192.1.68.1.25
     ```

   - Et modifier hosts.deny ainsi :

     ```
     sshd : ALL
     ```

5. Créez des règles netfilter qui interdisent aux utilisateurs toute connexion tcp sur le port 23, sauf pour les machines du sous-réseau 192.168.1.0/24 :

   ```
   # iptables -A INPUT -p tcp --dport 22 -s ! 192.168.1.0/24 -j DROP
   ```