Confinement applicatif avec AppArmor

1. Préambule

L'objectif de ce TP est de découvrir le fonctionnement du LSM AppArmor qui met en œuvre du contrôle d'accès de type MAC orienté vers le confinement applicatif. Sa configuration s'avère très différente de celle de SELinux.

2. Prérequis

Une machine virtuelle au format VirtualBox est utilisée pour ce TP :

- SSE_MrRobot : système Ubuntu 14.04 dérivé de la VM du challenge MrRobot disponible sur VulnHub sur laquelle plusieurs changements ont été effectués :
  - passage du clavier en clavier français ;
  - modification des mots de passe de tp et root en *tp* ;
  - ajout de l'utilisateur robot dans le groupe *admin* pour pouvoir utiliser la commande sudo ;
  - ajout du paquet apparmor-utils (et ses dépendances) ;
  - modification de page d'erreur 404.php pour intégrer un reverse shell (cf. http://www.gcura.tech/vulnhub-mr-robot-1/ pour exemple) ;

3. AppArmor

Exercice 3-1

Objectif : appréhender les bases d'AppArmor

La VM à utiliser pour cet exercice est la VM MrRobot (compte robot ou root).

La commande aa-status (ou apparmor_status) permet de vérifier l'état de AppArmor

```
sudo aa-status
```

Si cet outil n'est pas installé ou que le compte utilisé ne peut pas utiliser sudo, il est possible d'obtenir des informations autrement, par exemple :

```
ls -l /sys/kernel/security/apparmor/policy/profiles/
cat /sys/kernel/security/apparmor/policy/profiles/sbin.dhclient.0/mode
```

Commande ping

Vérifier que la commande ping est fonctionnelle avant de passer à la suite, par exemple :

```
ping 127.0.0.1
```

Créer un nouveau profil `/etc/apparmor.d/bin.ping` (avec vi ou un autre éditeur) avec le contenu suivant :

```
#include <tunables/global>

/bin/ping {
  #include <abstractions/base>
  #include <abstractions/consoles>
  #include <abstractions/nameservice>

#  capability net_raw,
#  network raw,
}
```

Charger la règle avec la commande suivante :

```
cat /etc/apparmor.d/bin.ping | sudo apparmor_parser -a
```

Refaite le test de ping (`ping 127.0.0.1`). Que se passe-t-il ?

_____
_____

Chercher dans le journal du noyau des événements associés.

```
sudo tail /var/log/kern.log
```

Un message de type contenant le texte `appamor="DENIED"` doit être présent

Enlever le commentaire sur la ligne contenant « `network raw,` » dans le profil, puis recharger le profil :

```
cat /etc/apparmor.d/bin.ping | sudo apparmor_parser -r
```

Relancer la commande ping. Que se passe-t-il ? Le journal du noyau permet-il de comprendre le refus?

_____
_____
_____
_____

Activer le mode « complain » pour le profil :

```
#include <tunables/global>

/bin/ping flags=(complain) {
  #include <abstractions/base>
  #include <abstractions/consoles>
  #include <abstractions/nameservice>

#  capability net_raw,
  network raw,
}
```

et recharger le profil.

Relancer la commande ping. Que se passe-t-il ? Le journal du noyau permet-il de comprendre le précédent refus (i.e. chercher une référence de type « capability ») ?

_____

_____

Enlever le commentaire sur la ligne contenant « `capability net_raw,` » dans le profil, puis recharger le profil. Le lancement de la commande ping ne doit plus générer d'événements dans le journal du noyau.

Retirer le drapeau « complain » du profil.

```
#include <tunables/global>

/bin/ping {
  #include <abstractions/base>
  #include <abstractions/consoles>
  #include <abstractions/nameservice>

  capability net_raw,
  network raw,
}
```

et recharger le profil. Vérifier que la commande ping est toujours fonctionnelle.

Limiter l'accès à des fichiers pour la commande cat

Créer une copie de la commande cat

```
sudo cp /bin/cat /bin/cat2
```

Vérifier qu'un compte privilégié peut lire le contenu du fichier /etc/shadow avec la commande cat :

```
sudo cat /etc/shadow
```

Créer un nouveau profil `/etc/apparmor.d/bin.cat` avec le contenu suivant :

```
#include <tunables/global>

/bin/cat {
  #include <abstractions/base>
  #include <abstractions/consoles>

}
```

et charger le avec la commande suivante (attention au cat2) :

```
cat2 /etc/apparmor.d/bin.cat | sudo apparmor_parser -a
```

Pouvez-vous afficher le contenu d'un fichier avec la commande cat (par ex, /etc/hostname) ? Le journal du noyau est-il explicite ?

_____

_____

Modifier le profil :

```
#include <tunables/global>

/bin/cat {
  #include <abstractions/base>
  #include <abstractions/consoles>
  /** r,
  deny /etc/shadow r,
}
```

et recharger le :

```
cat2 /etc/apparmor.d/bin.cat | sudo apparmor_parser -r
```

Tester sa bonne application avec la commande

```
sudo cat /etc/shadow
```

Les permissions présentes sur ce fichier sont-elles la cause du refus d'accès ? Le journal du noyau contient-il un événement pour ce refus ?

_____

_____

Modifier le profil pour enregistrer les refus :

```
 #include <tunables/global>

/bin/cat {
  #include <abstractions/base>
  #include <abstractions/consoles>
  /** r,
  audit deny /etc/shadow r,
}
```

Écrire un profil qui empêche le programme vi de modifier les fichiers /etc/passwd et /etc/shadow et l'appliquer au système. Est-ce que ces deux fichiers sont désormais mieux protégés ?

_____
_____
_____
_____
_____
_____

Exercice 3-2

Objectif : Créer un profil AppArmor pour confiner PHP

Cet exercice se base sur l'article https://www.inguardians.com/2017/06/08/protecting-the-mr-robot-vuln-hub-machine-part-2-confining-wordpress-with-apparmor/. Néanmoins la VM utilisée étant celle de VulnHub et pas une VM Ubuntu 16.04, les chemins d'installation de php sont différents.

Attention, en fonction de votre configuration réseau, les adresses IP peuvent changer. L'adresse IP de référence de la VM MrRobot est 192.168.56.101.

Avant de mettre au point le profil AppArmor, il faut d'abord tester que le reverse shell fonctionne correctement. Se connecter sur http://192.168.56.101/wp-login.php avec le compte elliot et le mot de passe ER28-0652, puis éditer la page 404.php : http://192.168.56.101/wp-admin/theme-editor.php?file=404.php

Modifier la variable $ip par l'adresse d'une machine (physique ou virtuelle) sur laquelle vous pouvez exécuter netcat (consulter http://www.gcura.tech/vulnhub-mr-robot-1/ si besoin).

Nota : cette machine peut être la VM MrRobot mais cela rend le scénario encore moins « réaliste ».

Sur cette machine, lancer la commande

```
nc -lvp 1234
```

Puis consulter la page http://192.168.56.101/404.php afin de déclencher le *reverse shell*. Au travers

de ce *shell,* changer de compte pour robot (au lieu de daemon).

```
python -c 'import pty; pty.spawn("/bin/sh")'
whoami
id
su - robot
whoami
id
```

Se connecter en robot (attention à utiliser sudo) ou en root et mettre au point un profil AppArmor pour confiner PHP et bloquer l'exécution du *reverse shell*.

Attention, certaines commandes varient en raison de la version de distribution utilisée et du mode d'installation de PHP.

```
aa-genprof /usr/sbin/php-fpm7.0 => aa-genprof /opt/bitnami/php/sbin/php-fpm
systemctl restart php7.0-fpm.service => service bitnami restart
```

A l'issue de la mise au point du profil, charger le mode « enforce » et tester si l'attaque fonctionne toujours. Que pensez-vous du mécanisme utilisé ? Pour ce cas particulier, auriez-vous préconisé l'usage d'un autre module MAC comme SELinux ?