|  |  |
| --- | --- |
|  | Sécuriser un serveur nginx avec AppArmor |

# Présentation

Linux possède plusieurs outils permettant de sécuriser les applications.   
Toutefois, ces outils demandent parfois une configuration manuelle ou sont souvent méconnus.

Les modules de sécurité Linux (Linux Security Modules ou LSM) font partie de ces outils.  
Leur intégration dans le noyau de Linux leur donne ainsi une vue globale de tout le système.   
Il existe plusieurs modules de sécurité Linux, SELinux et AppArmor étant les plus utilisés.

**Linux Security Modules**

SELinux est un système de sécurité pour Linux permettant de limiter les actions possibles en fonction d’une politique de sécurité. Il permet d’appliquer notamment différentes méthodes de contrôle d’accès obligatoire.   
SELinux possède un fonctionnement différent de celui d’Apparmor, mais il est également bien plus complet (et également plus complexe).   
Il se base sur des étiquettes attribuées à des ressources en plus de politiques.   
Celui-ci est activé par défaut sur Red Hat Enterprise Linux (et dérivés) et il est fortement recommandé de le laisser activé.

Apparmor est un autre module de sécurité et de contrôle d’accès obligatoire pour Linux.   
Celui-ci repose principalement sur des chemins de fichiers.   
Cependant, il peut limiter l’accès à de nombreuses autres ressources et fonctionalités.   
Sa configuration est uniquement textuelle et sous forme de profils par application.   
Combiné avec ses outils d’administration, il est simple et rapide à mettre en place.   
Ces deux outils sont capables de fonctionner en mode alerte uniquement (complain/permissive), ce qui permet de ne pas perturber le bon fonctionnement des services.   
Les messages générés signalent toute infraction et fournissent une trace auditable.   
C’est pourquoi il est vivement conseillé d’activer ces modules même s’ils ne seront jamais en mode strict/défense. (enforce)

# Mise en place

Apparmor est administré avec différents outils de gestion mais ceux-ci ne sont pas toujours installés par défaut.

Pour installer les différents outils AppArmor …  
**>> sudo apt install apparmor-easyprof apparmor-profiles apparmor-profiles-extra apparmor-utils nginx**

Cette commande a également installé le serveur nginx( s’il n’était pas déjà installé).

**Rappel** …  
Historiquement, l’utilisateur root possédait tous les droits sur un système, sans aucune restriction.   
C’est pourquoi le système de capacités (*capabilities*) a été introduit.

Toutes les permissions de l’utilisateur root sont maintenant séparées en capacités.   
Par exemple, la capacité **net\_bind\_service** permet à un programme de se positionner en écoute sur un port privilégié (en dessous de 1024) sans toutefois octroyer aucune autre permission.   
Il est donc possible d’être administrateur (root) avec aucune capacités et donc sans les pleins pouvoirs.

Inversement, un programme avec un utilisateur non privilégié peut obtenir des capacités supplémentaires, obtenant une partie du pouvoir de l’utilisateur root.

# Création d’un profil Apparmor

La manière la plus simple de créer un profil Apparmor est tout simplement de prendre un profil de base ou vide et d’activer le mode Alerte (complain).   
Par défaut tout est refusé, c’est donc en utilisant les messages d’infraction de ce mode qu’il est donc possible de créer un profil par apprentissage.  
Il est essentiel que le comportement de l’application pendant ce temps d’apprentissage soit normal et correct, et que toutes les fonctionnalités de l’application soient exploitées.   
Dans le cas contraire, le nouveau profil ne serait pas complet. Une fonctionnalité non explorée lors de l’apprentissage pourrait solliciter une ressource qui n’a pas été autorisée dans le profil AppArmor.

Les profils sont stockés dans le répertoire /etc/apparmor.d, avec comme nom le chemin de l’exécutable à limiter avec les oblique (/) remplacés par des points (.).

Un utilitaire existe pour créer un profil initial …  
**>> sudo aa-autodep /usr/sbin/nginx  
Writing updated profile for /usr/sbin/nginx.**

On active ensuite le profil en mode Alerte (complain) …  
**>> sudo aa-complain nginx  
Setting /usr/sbin/nginx to complain mode.**

Comme dernière étape, on redémarre le service nginx …  
**>> sudo systemctl restart nginx.service**

À l’aide d’un fureteur (*browser*), on ouvre la page par défaut du serveur nginx …  
**>> http://<Adresse Ip du serveur>  
>> http://<Adresse Ip du serveur>/test**

Cette étape permet de créer une empreinte du comportement normal de l’application.

Une fois l’apprentissage terminé, il suffit d’examiner les infractions et valider celles qui sont normales, afin de les ajouter en tant qu’exception dans lke profil Apparmor.

Pour ce faire, on utilise l’utilitaire aa-logprof …  
**>> sudo aa-logprof  
Reading log entries from /var/log/audit/audit.log.  
Updating AppArmor profiles in /etc/apparmor.d.  
Complain-mode changes:  
  
Profil: /usr/sbin/nginx  
Mode compatibilité: dac\_override  
Gravité: 9  
  
[1 - capability dac\_override,]  
(A)llow / [(D)eny] / (I)gnore / Audi(t) / Abo(r)t / (F)inish  
  
<…>**

# Gestion des infractions

Ce programme va parcourir tous les messages d’infraction et demander, une par une quelle action est désirée.   
La décision sera retranscrite dans le profil.

**Profil: /usr/sbin/nginx  
Mode compatibilité: dac\_override  
Gravité: 9  
  
[1 - capability dac\_override,]  
(A)llow / [(D)eny] / (I)gnore / Audi(t) / Abo(r)t / (F)inish**

Ce menu permet de choisir si l’infraction est erronée et doit être autorisée et ajoutée au profil.   
Celui-ci sera répété pour chaque infraction.

Pour cet exemple, le mode compatibilité (capability) dac\_override permet d’ignorer le contrôle d’accès sur les fichiers. De la même manière que les permissions sur les fichiers ne s’appliquent pas à l’utilisateur root (par défaut).

Cette permission est nécessaire si le fichier de journalisaiton (log) de nginx n’appartient pas à un groupe ou utilisateur dont nginx fait partie.   
Ceci est le cas par défaut.   
Si on souhaite modifier le propriétaire du fichier (utilisateur de nginx – www-data) on doit sélectionner **D** ou **I** afin de refuser cet accès.   
Sinon dans la majorité des cas, on sélectionne **A** pour autoriser.

**Remarque** …  
Le mode compatibilité (*capability*) demeure limité par Apparmor, même en la possédant il reste impossible d’accéder aux fichiers auxquels Apparmor bloque l’accès.

**Profil: /usr/sbin/nginx  
Chemin d'accès: /var/log/nginx/error.log  
Nouveau mode: w  
Gravité: 8  
  
[1 - /var/log/nginx/error.log w,]  
(A)llow / [(D)eny] / (I)gnore / Audi(t) / Abo(r)t / (F)inish**

Cette infraction est une tentative d’accès en écriture (w) à un fichier.  
Pour cet exemple, il s’agit du fichier de journalisation d’erreur (error log).  
On autorise en en sélectionnant l’option **A**.   
On fait de même pour /var/log/nginx/access.log lorsque demandé.

**Profil: /usr/sbin/nginx  
Chemin d'accès: /etc/ssl/openssl.cnf  
Nouveau mode: owner r  
Gravité: 2  
  
[1 - #include <abstractions/openssl>]  
 2 - #include <abstractions/ssl\_keys>  
 3 - owner /etc/ssl/openssl.cnf r,  
(A)llow / [(D)eny] / (I)gnore / (G)lob / Glob with (E)xtension / (N)ew / Audi(t) / (O)wner permissions off / Abo(r)t / (F)inish**

Pour cet exemple, il existe de multiples actions possibles.   
Celles-ci sont numérotées de 1 à 3.   
Les crochets [ et ] identifie la ligne et l’action actuellement sélectionnée.   
Pour choisir l’option désirée, on sélectionne le numéro correspondant ou on utilise les touches fléchées du clavier.

Les lignes contenant un #include indiquent qu’Apparmor a détecté un profil de base contenant les permissions nécessaires pour autoriser cette infraction.   
Il faut décider si la suggestion est judicieuse.

Dans ce cas, le profil /abstractions/openssl correspond parfaitement.  
On le garde sélectionné et on appuie sur A afin de l’autoriser.

**Remarque** …  
L’accès à la configuration est bloqué.  
Ce cas est similaire au cas de la journalisation (logs), mais dans ce cas l’accès est en lecture uniquement (r).   
On sélectionne **A**.

**Profil: /usr/sbin/nginx  
Chemin d'accès: /etc/nginx/modules-enabled/  
Nouveau mode: owner r  
Gravité: unknown  
  
[1 - #include <abstractions/totem>]  
 2 - owner /etc/nginx/modules-enabled/ r,  
(A)llow / [(D)eny] / (I)gnore / (G)lob / Glob with (E)xtension / (N)ew / Audi(t) / (O)wner permissions off / Abo(r)t / (F)inish**

**Profil: /usr/sbin/nginx  
Chemin d'accès: /etc/nginx/modules-enabled/  
Nouveau mode: owner r  
Gravité: unknown  
  
[1 - #include <abstractions/totem>]  
 2 - owner /etc/nginx/modules-enabled/ r,  
(A)llow / [(D)eny] / (I)gnore / (G)lob / Glob with (E)xtension / (N)ew / Audi(t) / (O)wner permissions off / Abo(r)t / (F)inish**

Cette modification proposée introduit deux nouveautés.  
Ton constate qu’il s’agit d’une permission sur un répertoire, en lecture (r).  
Ensuite, le profil de base détecté n’est pas correct.

Totem est un lecteur multimédia qui n’est pas utilisé avec le site web par défaut de nginx.   
Utiliser ce profil accorderait bien trop de permissions inutiles à nginx.

On peut utiliser les flèches ou sélectionner l’option 2.   
Cette action (owner /etc/nginx/modules-enabled/ r) signifie que nginx doit être propriétaire du fichier/répertoire concerné.

**Profil: /usr/sbin/nginx  
Chemin d'accès: /usr/share/nginx/modules-available/mod-http-geotp.conf  
Nouveau mode: owner r  
Gravité: unknown  
  
[1 - #include <abstractions/totem>]  
 2 - owner /usr/share/nginx/modules-available/mod-http-geotp.conf r,  
(A)llow / [(D)eny] / (I)gnore / (G)lob / Glob with (E)xtension / (N)ew / Audi(t) / (O)wner permissions off / Abo(r)t / (F)inish**

Même problématique pour cet exemple. Le profil totem n’est pas le plus adapté.  
 Cependant autoriser uniquement le fichier mod-http-geoip.conf semble trop restrictif car il parait plus logique d’autoriser nginx à lire tous les fichiers de configuration des modules.   
Pour modifier cet état, on sélectionne **G** ou **E**.

E est plus restrictif donc le plus adapté.   
G autorise tous les fichiers du dossier et E tous les fichiers avec l’extension indiquée (.conf).   
Le tout dans le monde indiqué : lecture (r).

Ne pas oublier vérifier que la nouvelle option est bien sélectionnée après avoir sélectionné G ou E et avant de confirmer avec A.

Ce fichier contient le numéro de processus.   
On peut lui donner la permission en lecture et écriture (rw).   
On sélectionne A.

**Profil: /usr/sbin/nginx  
Chemin d'accès: /var/www/html/index.nginx-debian.html  
Nouveau mode: r  
Gravité: unknown  
  
[1 - #include <abstractions/web-data>]  
 2 - /var/www/html/index.nginx-debian.html r,  
(A)llow / [(D)eny] / (I)gnore / (G)lob / Glob with (E)xtension / (N)ew / Audi(t) / (O)wner permissions off / Abo(r)t / (F)inish**

Il s’agit de donner l’accès aux fichiers des pages web.   
En principe, il est nécessaire de donner accès à tous les fichiers récursivement dans ce répertoire du site web.   
Toutefois, le profil présélectionné web-data s’en charge déjà.   
On valide avec l’option A.

De la même manière, on peut également autoriser …

* A – /etc/nginx/mime.types – r
* **A** – [2 – owner /etc/nginx/conf.d/ r,] – r
* **A** – [2 – owner /etc/nginx/sites-enabled/ r,] – r
* **G** – /etc/nginx/sites-available/default   
  et ensuite
* **A** – [2 – owner /etc/nginx/sites-available/\* r,]

**Profil: /usr/sbin/nginx  
Chemin d'accès: /etc/nsswitch.conf  
Nouveau mode: owner r  
Gravité: unknown  
  
[1 - #include <abstractions/nameservice>]  
 2 - 2 - owner /etc/nsswitch.conf r,  
(A)llow / [(D)eny] / (I)gnore / (G)lob / Glob with (E)xtension / (N)ew / Audi(t) / (O)wner permissions off / Abo(r)t / (F)inish**

Pour terminer, le profil nameservice est correct.   
On confirme avec l’option A.

# Sauvegarde des changements au profil

Une fois tous les messages d’infraction corrigé, l’utilitaire aa-logprof demande une confirmation afin de sauvegarder ces changements dans le fichier du profil.

**= Changed Local Profiles =  
The following local profiles were changed. Would you like to save them?  
 [1 - /usr/sbin/nginx]  
(S)ave Changes / Save Selec(t)ed Profile / [(V)iew Changes] / View Changes b/w (C)lean profiles / Abo(r)t  
Writing updated profile for /usr/sbin/nginx.**

On appuie sur **S** pour les sauvegarder.

Une fois le profil sauvegardé, une permission nécessaire n’est pas automatiquement assignée dans le profil.   
Il faut donc l’ajouter manuellement.

En effet, nginx fonctionne à l‘aide d’un processus parent et de processus enfants qui effectue le travail.   
Le processus parent nécessite la permission d’exécuter les processus enfants en tant qu’utilisateur moins privilégié.   
Il a besoin donc des capacités (*capabilities*) setgid et setuid .

Il faut également ajouter deux autres capacités afin d’autoriser l’écoute sur un port privilégié. (80)

Pour y arriver il faut modifier le fichier contenant le profil …  
**>> sudo nano /etc/apparmor.d/usr.sbin.nginx  
# Last Modified: Tue May 31 13:13:48 2022  
#include <tunables/global>  
  
/usr/sbin/nginx flags=(complain) {  
 #include <abstractions/base>  
 #include <abstractions/dovecot-common>  
<…>  
 #include <abstractions/web-data>  
  
 capability dac\_override,  
 capability dac\_read\_search;   
 capability net\_bind\_service;   
 capability setgid,   
 capability setuid,   
  
 /usr/sbin/nginx mr,  
 /var/log/nginx/access.log w,  
 /var/log/nginx/error.log w,  
 owner /etc/nginx/mime.types r,** owner /etc/nginx/modules-enabled/\*.conf r,   
 **owner /etc/nginx/nginx.conf r,   
 owner /etc/nginx/sites-available/\* r,   
 owner /etc/nginx/sites-enabled/\* r,   
 owner /etc/nginx/nginx.conf r,  
 owner /etc/nginx/sites-available/default r,  
 owner /proc/sys/kernel/random/boot\_id r,**  
 **owner /run/nginx.pid rw,**

On peut assouplir certaines permissions qui n’ont pas pu être détectées par aa-logprof, car l’installation par défaut du service nginx sous Debian n’est pas très complexe.

Une fois que l’on a inspecté les règles pour valider leur cohérence, on sauvegarde et on quitte.

Pour recharger le profil …  
**>> sudo apparmor\_parser --replace /etc/apparmor.d/usr.sbin.nginx**

**Rappel** …  
Pour recharger toutes les règles …  
**>> sudo systemctl reload-or-restart apparmor.service**

# Activation du mode strict (enforce) pour le profil

Afin de basculerr le profil en mode bloquant/strict …  
**>> sudo aa-enforce nginx   
Setting /usr/sbin/nginx to enforce mode.**

Comme dernière étape, il suffit de redémarrer nginx…  
 **>> sudo systemctl reload-or-restart nginx.service**

La commande aa-status devrait afficher les processus comme confined …  
 **>> sudo aa-status  
apparmor module is loaded.  
31 profiles are loaded.  
5 profiles are in enforce mode.  
 /usr/bin/pidgin//sanitized\_helper  
 /usr/bin/totem//sanitized\_helper  
 /usr/bin/uname  
 /usr/sbin/nginx  
 syslog-ng  
<…>**

Dans un fureteur, la page s’affiche normalement, sans problème.

Toutefois, si l’on configure Nginx pour servir un dossier en dehors de /var/www/html, l’accès sera refusé au programme et celui-ci retournera une erreur HTTP 403.

En cas de compromission du processus du service nginx, celui-ci n’aura accès à aucun fichier du périphérique de stockage et ses permissions limitées rendront l’élévation de privilège très compliquée.

Dans le cas où un accès global à tous les sous-répertoire est nécessaire, il est possible de doubler le \*.   
Par exemple, owner /etc/nginx/\*\* r, autorise l’accès à tous les fichiers, répertoires et sous-répertoires (récursivement) possédés par l’utilisateur de nginx dans le répertoire/etc/nginx.

# Référence

<https://blog.devensys.com/apparmor-un-outil-simple-et-gratuit-pour-securiser-vos-serveurs-linux/>