Exercices de formation

Formation : M2I

Exercice 1 : La recherche d'information, un monde vaste..

Sujet de l'exercice :

Pour constater si une machine est potentiellement vulnérable nous pouvons utiliser différents outils.

Objectif de l'exercice :

Réaliser un scan Nmap de la machine metasploitable.

Que fait :

- nmap -sS [ip-machine]
- nmap -p1-65535 -sS [ip-machine]
- nmap -sX [ip-machine]
- nmap -sS -p- -sV [ip-machine]
- nmap -o [ip-machine]

Exercice 2 : Un moteur de recherche pas comme les autres.

Sujet de l'exercice :

Nous utiliserons dans cet exercice l'outil Shodan.

Objectif de l'exercice :

Trouver des machines Windows et Linux, ainsi que des caméras.

Il est possible de trouver des webcams disponibles sans login à cause d'un service bien particulier.

Trouver des potentiels "exploit".

Exercice 3 : Une bibliothèque particulière

Sujet de l'exercice :

Vous avez remarqué que la machine metasploitable possède de nombreux ports ouverts.

Objectif de l'exercice :

Vous devez trouver les CVE associées aux services de metasploitable.

Exercice 4 : Réalisation d'une machine sécurisé linux.

Sujet de l'exercice :

Linux devrait être une machine sécurisée pour pouvoir l'utiliser sans problème c'est ce que nous réaliserons avec une machine Debian avec 10G de stockage et 3G de RAM.

Objectif de l'exercice :

a. Préparer la machine avec un nom, un mot de passe root/administrateur, un nouvel utilisateur avec son mot de passe.
b. Partitionnement du disque:
    i. un volume de boot primaire 771,8MB f ext4 /boot
    ii. un volume primaire de 10G utiliser comme volume physique pour chiffrement, on utilisera du dm-crypt avec chiffrement aes-256
c. Configuration des volumes chiffrés:
    i. création des volumes sur le /dev/sda2 chiffré
    ii. donner une phrase secrète sécurisé
    iii. configurer les volume logique (LVM)
    iv. création d'un groupe de volumes:

Groupe de volume	Nom du volume logique	taille du volume
VGCRYPT	lv_swap	488MB
VGCRYPT	lv_home	952MB
VGCRYPT	lv_tmp	952MB

VGCRYPT	lv_root	Espace restant

d. configuration des nouvelles participations provenant des volumes logiques:

Volumes logiques	Utiliser comme	Point de montage	Etiquette
lv_swap	espace d'échange ("swap")		
lv_home	système de fichiers journalisé XFS	/home	HOME
lv_tmp	système de fichiers journalisé ext4	/tmp	TMP
lv_root	système de fichiers journalisé ext4	/	RACINE

e. refus de l'analyse du CD
f. sélection du pays miroir de l'archive Debian ainsi qu le miroir
g. sélection minimale des logiciels, serveur SSH
h. Installation du GRUB
i. on choisit la partition de boot.

Exercice 5 : Des scripts auto

Sujet de l'exercice :

De nombreux scripts bash sont possibles pour faciliter la vie d'un utilisateur ou pour accentuer la sécurité, c'est pourquoi nous allons créer un script bash.

Objectif de l'exercice :

Faire un script qui affiche l'adresse ip de la machine, son nom ainsi que le système d'exploitation au démarrage de la machine, vous pouvez ajouter une date si vous le désirez et d'autres options. (intéressez vous à "motd").

Exercice 6 : Décryptage

Sujet de l'exercice :

Nous avons récupéré des données avec l'outil secretdump.

Objectif de l'exercice :

Décrypter différents hash du document crackingMDP.txt

- hash NT
- hash DCC
- hash DCC2

Il y a aussi ce hash: 7ecc19e1a0be36ba2c6f05d06b5d3058 qu'il faudrait retrouver en clair.

Exercice 7 : SSH sécurisé

Sujet de l'exercice :

L'utilisation du service ssh est parfois utile pour accéder à votre machine à distance.

Objectif de l'exercice :

Il vous faut 2 machines, une machine serveur et une machine client.

sécurisation côté client avec avec la connexion ssh à partir d'une clé.

Exercice 8 : Manipulations SELinux

Sujet de l'exercice :

Suivre le TP et les explications fournies afin de comprendre le fonctionnement des contextes établis par SELinux sur les fichiers et dossiers.

Objectif de l'exercice :

Prérequis :

- Installer une machine CentOS 8 64 bits
- Ajouter l'utilisateur créé lors de l'installation dans les sudoers
- Mettre le clavier en AZERTY si besoin
- Laisser la langue en en_US.utf8 (important pour l'analyse de log par la suite)
- Vérifier avec la commande 'sestatus' que SELinux est activé en mode enforced

SELinux dans tous ses états

SELinux propose **trois modes** différents.

- Dans le mode **strict** (Enforcing), les accès sont restreints en fonction des règles SELinux en vigueur sur la machine.
- Le mode **permissif** (Permissive) peut être considéré comme un mode de débogage. Les règles SELinux sont interrogées, les erreurs d'accès sont enregistrées dans les logs, mais l'accès ne sera pas bloqué.
- Lorsque SELinux est **désactivé** (Disabled), l'accès n'est pas restreint, et rien n'est enregistré dans les logs.

La commande getenforce vous informe sur le mode en vigueur sur votre machine.

```
$ getenforce
Enforcing
```

La commande setenforce permet de basculer temporairement – jusqu'au prochain redémarrage – entre les modes strict (Enforcing) et permissif (Permissive).

```
$ getenforce
Enforcing
$ sudo setenforce 0
$ getenforce
Permissive
$ sudo setenforce 1
$ getenforce
Enforcing
```

Le mode SELinux par défaut est défini dans le fichier /etc/selinux/config.

```
# /etc/selinux/config
SELINUX=enforcing
SELINUXTYPE=targeted
```

Ici, SELINUX prendra une des trois valeurs enforcing, permissive ou disabled. Quant à SELINUXTYPE, on gardera la valeur par défaut targeted, qui garantit la surveillance des principaux services réseau.

Lorsque SELinux est activé - autrement dit, lorsqu'on passe du mode disabled à permissive ou enforcing, il faut impérativement songer à réétiqueter l'ensemble des fichiers du système. Pour ce faire, il suffit de créer un fichier vide .autorelabel à la racine du système de fichiers avant de redémarrer.

```
$ sudo touch /.autorelabel
$ sudo reboot
```

Le réétiquetage du système de fichiers peut prendre un certain temps, en fonction de la taille de votre installation. Pour une première utilisation, je vous conseille d'opter pour le

mode permissif par défaut. Cela évite de se retrouver avec un système qui ne démarre plus en mode strict.

- Sur une installation fraîche de RHEL, CentOS ou Fedora, SELinux est activé en mode strict (Enforcing) par défaut.
- Sur un serveur dédié sous CentOS chez Online, SELinux est désactivé (Disabled) par défaut.

Exemple pratique n° 1

Maintenant que nous disposons du minimum syndical de bagage théorique, passons à la pratique. Pour ce premier exemple, je passe en mode strict.

```
$ getenforce
Enforcing
```

Dans sa configuration par défaut, Apache est censé servir les pages web rangées dans /var/www/html. Je copie la page web par défaut à cet endroit.

```
$ cd /var/www/html/
$ sudo chown microlinux:microlinux .
$ cp -R /usr/share/httpd/noindex/* .
```

J'édite la page index.html pour la différencier de la page fournie par défaut, je démarre Apache et j'obtiens ceci.

Jusqu'ici, tout va bien. J'ai réussi à configurer un hébergement web avec SELinux en mode strict, et je me sens un peu comme Monsieur Jourdain dans *Le Bourgeois Gentilhomme*, qui fait de la prose sans le savoir.

Notez ici que dans la configuration par défaut, Apache sert les pages web rangées dans `/usr/share/httpd/noindex` en l'absence de page d'index. Cette fonctionnalité peut s'avérer quelque peu déroutante pour la suite de nos manipulations. Il vaut mieux la désactiver, ce qui peut se faire simplement en commentant la directive correspondante dans le fichier `/etc/httpd/conf.d/welcome.conf`.

```
# <LocationMatch "^/+$">
#     Options -Indexes
#     ErrorDocument 403 /.noindex.html
# </LocationMatch>
```

Le contexte de sécurité SELinux

Le contenu de mon répertoire /var/www/html ressemble à ceci.

```
$ cd /var/www/html/
$ ls -l
total 16
drwxr-xr-x. 3 microlinux microlinux 4096 30 janv. 15:24 css
drwxr-xr-x. 2 microlinux microlinux 4096 30 janv. 15:24 images
-rw-r--r--. 1 microlinux microlinux 4899 30 janv. 15:26 index.html
```

L'option -Z me permet d'afficher le contexte de sécurité de ces fichiers.

```
$ ls -Z
drwxr-xr-x. microlinux microlinux
unconfined_u:object_r:httpd_sys_content_t:s0 css
drwxr-xr-x. microlinux microlinux
unconfined_u:object_r:httpd_sys_content_t:s0 images
-rw-r--r--. microlinux microlinux
unconfined_u:object_r:httpd_sys_content_t:s0 index.html
```

Cette même option -Z peut s'utiliser avec la commande ps pour afficher le contexte de sécurité d'un processus.

```
# ps axZ | grep httpd
system_u:system_r:httpd_t:s0    3948 ?    Ss   0:00 /usr/sbin/httpd -DFOREGROUND
system_u:system_r:httpd_t:s0    3949 ?    S    0:00 /usr/sbin/httpd -DFOREGROUND
system_u:system_r:httpd_t:s0    3950 ?    S    0:00 /usr/sbin/httpd -DFOREGROUND
system_u:system_r:httpd_t:s0    3951 ?    S    0:00 /usr/sbin/httpd -DFOREGROUND
...
```

Un contexte SELinux est présenté sous la forme utilisateur:rôle:type:niveau. Laissons de côté les utilisateurs SELinux, les rôles et les niveaux, et concentrons-nous sur le type, qui correspond au troisième champ dans le contexte SELinux.

- unconfined_u:object_r:**httpd_sys_content_t**:s0
- system_u:system_r:**httpd_t**:s0

Nous avons vu que sur le serveur web en cours d'exécution, le processus httpd est étiqueté httpd_t. Les fichiers servis par Apache semblent tous munis de l'étiquette httpd_sys_content_t. Si nous affichons le contexte de sécurité des fichiers appartenant à Apache, nous constatons qu'ils semblent tous appartenir à une même famille en termes d'étiquettes.

- Les fichiers de configuration sont étiquetés httpd_config_t.
- Les fichiers logs sont tous de type httpd_log_t.

Essayons maintenant de cerner le principe de fonctionnement de SELinux en partant de cet exemple.

- Il est probablement logique qu'un processus étiqueté httpd_t puisse interagir avec des fichiers étiquetés httpd_sys_content_t, httpd_config_t, httpd_log_t, etc.
- En revanche, il n'est probablement pas logique qu'un processus étiqueté httpd_t puisse interagir avec un fichier comme /etc/shadow, qui porte l'étiquette shadow_t.

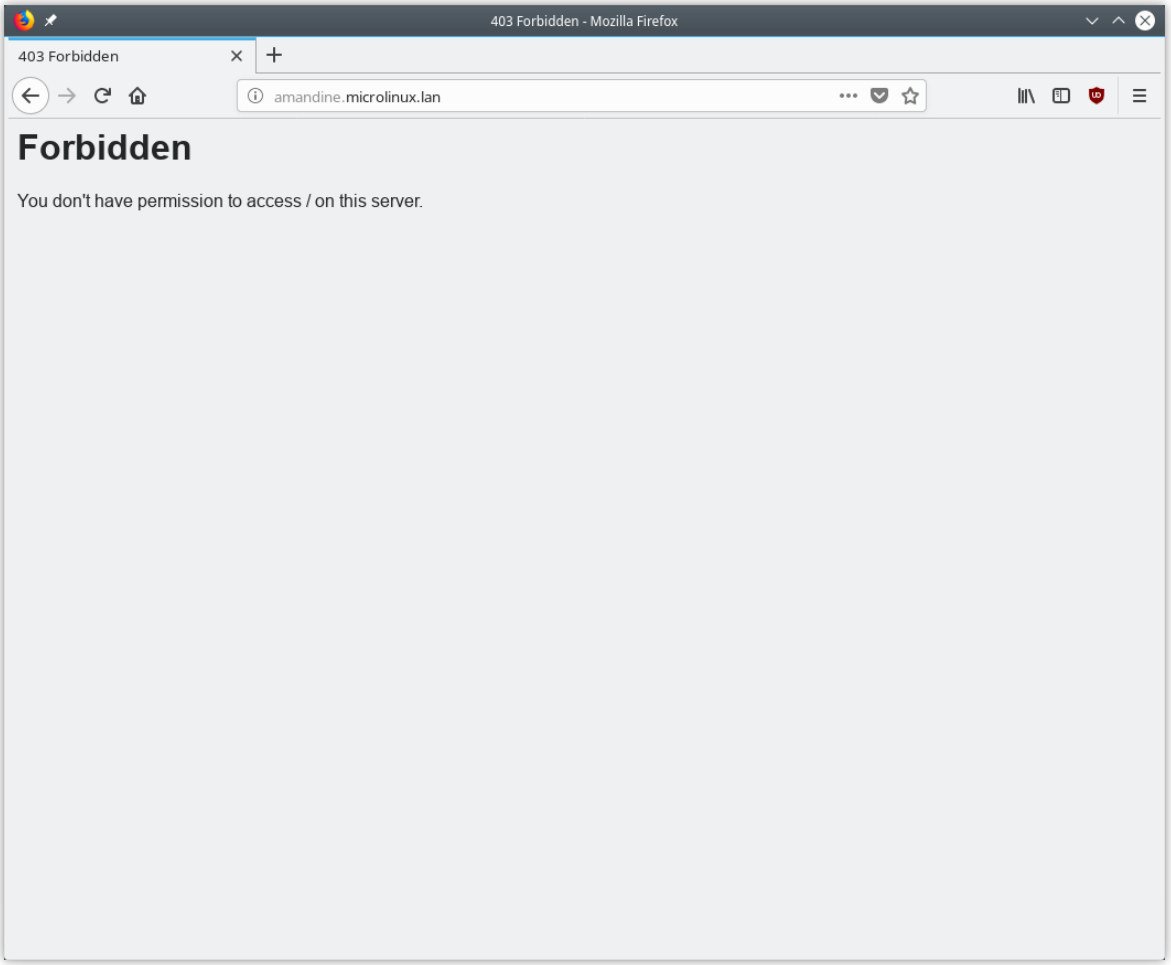
Faisons une petite expérience pour mieux comprendre ce principe.

Exemple pratique n° 2

Dans ce deuxième exemple, je vais copier les fichiers servis par Apache vers un répertoire en-dessous de /srv, effacer le contenu original, puis déplacer les fichiers copiés vers l'emplacement original (remplacez 'microlinux' par le nom de votre utilisateur).

```
$ cd /var/www/html/
$ sudo mkdir /srv/html
$ sudo chown -R microlinux:microlinux /srv/html/
$ cp -R * /srv/html/
$ rm -rf *
$ mv /srv/html/* .
```

Certains parmi vous auront le vague sentiment de tourner en rond pour revenir à la case départ. Ce n'est pas tout à fait le cas, comme nous allons le voir tout de suite. Rafraîchissez la page web qui s'affiche dans le navigateur, et voici ce que vous obtenez.



Jetons un œil dans les logs d'Apache pour voir ce qui se passe.

```
$ sudo cat /var/log/httpd/error_log
...
[Thu Jan 31 08:25:15.894012 2019] [core:error] [pid 4435]
(13)Permission denied: [client 192.168.2.2:58016] AH00035:
access to /index.html denied (filesystem path '/var/www/html/index.html')
because search permissions are missing on a component of the path
```

Et là, nous restons quelque peu perplexes, parce que nous avons vérifié à deux fois les droits d'accès de nos fichiers, et qu'ils ont l'air corrects. Mystère et boules de gomme.

Retournons maintenant dans /var/www/html et jetons un oeil sur le contexte SELinux des fichiers.

```
$ cd /var/www/html/
$ ls -Z
drwxr-xr-x. microlinux microlinux unconfined_u:object_r:var_t:s0   css
drwxr-xr-x. microlinux microlinux unconfined_u:object_r:var_t:s0   images
-rw-r--r--. microlinux microlinux unconfined_u:object_r:var_t:s0   index.html
```

L'administrateur perplexe souhaite sans doute en savoir un peu plus sur le pourquoi du comment de ce refus d'obtempérer. Les logs de SELinux ne sont pas ce qu'il y a de plus lisible, mais fort heureusement, il existe un outil pratique pour nous faciliter la tâche. Installez le paquet setroubleshoot-server et invoquez la commande suivante.

```
$ sudo sealert -a /var/log/audit/audit.log | less
100% done
found 1 alerts in /var/log/audit/audit.log
--------------------------------------------------------------
SELinux is preventing /usr/sbin/httpd from getattr access
on the file /var/www/html/index.html.
***** Plugin restorecon (94.8 confidence) suggests *****
If you want to fix the label.
/var/www/html/index.html default label should be httpd_sys_content_t.
Then you can run restorecon. The access attempt may have been stopped
due to insufficient permissions to access a parent directory in which
case try to change the following command accordingly. Do
# /sbin/restorecon -v /var/www/html/index.html
...
```

 Notez au passage qu'ici, j'ai invoqué la commande sur un serveur configuré en anglais (LANG=en_US.utf8). Sur un système configuré en français, on obtient un mélange de français et d'anglais assez folklorique.

Quoi qu'il en soit, retenons le contenu de ce message d'erreur.

- SELinux empêche Apache d'accéder au fichier index.html.
- L'étiquette par défaut devrait être httpd_sys_content_t.
- La commande restorecon permet de rétablir l'étiquette correcte.

Je vais suivre les recommandations sans toutefois les prendre au pied de la lettre, car je vais utiliser restorecon pour réétiqueter correctement tout le contenu de mon répertoire /var/www/html.

```
$ sudo restorecon -R -v /var/www/html/
```

Je recharge la page web, et je constate qu'elle s'affiche à nouveau correctement.

Modifier le contexte SELinux

Les fichiers que nous avons créés dans /var/www/html étaient tous étiquetés httpd_sys_content_t, alors que leurs copies correspondantes dans /srv/html étaient de type var_t.

En mode strict aussi bien qu'en mode permissif, chaque fichier nouvellement créé à un certain endroit du système de fichiers sera étiqueté de manière appropriée par le système. La commande matchpathcon pourra nous renseigner sur l'étiquette utilisée en fonction du répertoire.

```
$ matchpathcon /var/www/html/
/var/www/html   system_u:object_r:httpd_sys_content_t:s0
$ matchpathcon /var/log/httpd/
/var/log/httpd  system_u:object_r:httpd_log_t:s0
$ matchpathcon /etc/httpd/conf/
```

```
/etc/httpd/conf system_u:object_r:httpd_config_t:s0
```

La commande restorecon que nous avons utilisée un peu plus haut s'est donc chargée de restaurer le contexte par défaut des fichiers rangés dans cette arborescence.

En revanche, si nous décidons de ranger des fichiers dans un endroit un peu moins orthodoxe du système, nous devrons nous charger de définir un contexte SELinux adapté pour l'arborescence en question. Le prochain exemple nous le montrera.

Exemple pratique n° 3

Admettons que je veuille ranger les fichiers de mon serveur web dans l'arborescence /srv/web/html plutôt que dans /var/www/html.

```
$ sudo mkdir -pv /srv/web/html
mkdir: created directory '/srv/web'
mkdir: created directory '/srv/web/html'
```

Comme on peut s'y attendre, le contexte SELinux par défaut de ce répertoire n'est pas adapté à son utilisation.

```
$ matchpathcon /srv/web/html/
/srv/web/html system_u:object_r:var_t:s0
```

Pour modifier le contexte par défaut de cette arborescence, je pourrais m'y prendre de la manière suivante.

```
$ sudo semanage fcontext -a -t httpd_sys_content_t '/srv/web(/.*)?'
$ sudo restorecon -R -v /srv/web/
```

L'opération se fait donc en deux temps et mérite d'ailleurs quelques remarques.

- Comme on s'en doute, l'outil semanage avec l'option fcontext permet de gérer les contextes de sécurité SELinux.
- L'option -a (--add) permet d'ajouter une entrée.
- L'option -t (--type) spécifie le type, en l'occurrence httpd_sys_content_t.
- L'utilisation de l'expression régulière '/srv/web(/.*)?' applique la directive récursivement sur toute l'arborescence.
- Une fois que le contexte par défaut est défini, il faut l'appliquer avec restorecon.

Notons ici que la documentation de SELinux cite chcon comme premier outil pour modifier le contexte d'un fichier ou d'un répertoire. Le problème avec chcon, c'est qu'à la prochaine utilisation de restorecon sur le fichier ou ses parents, le fichier sera réétiqueté avec le contexte de son plus proche parent pour lequel un contexte spécifique est défini. Il vaut donc mieux prendre l'habitude d'utiliser semanage fcontext et restorecon pour éviter de se tirer dans le pied par la suite.

Exemple pratique n° 4

Dans ce quatrième exemple, nous mettons la pratique avant la théorie pour expliquer une autre particularité de SELinux.

Pour commencer, passez en mode permissif.

```
$ sudo setenforce 0
```

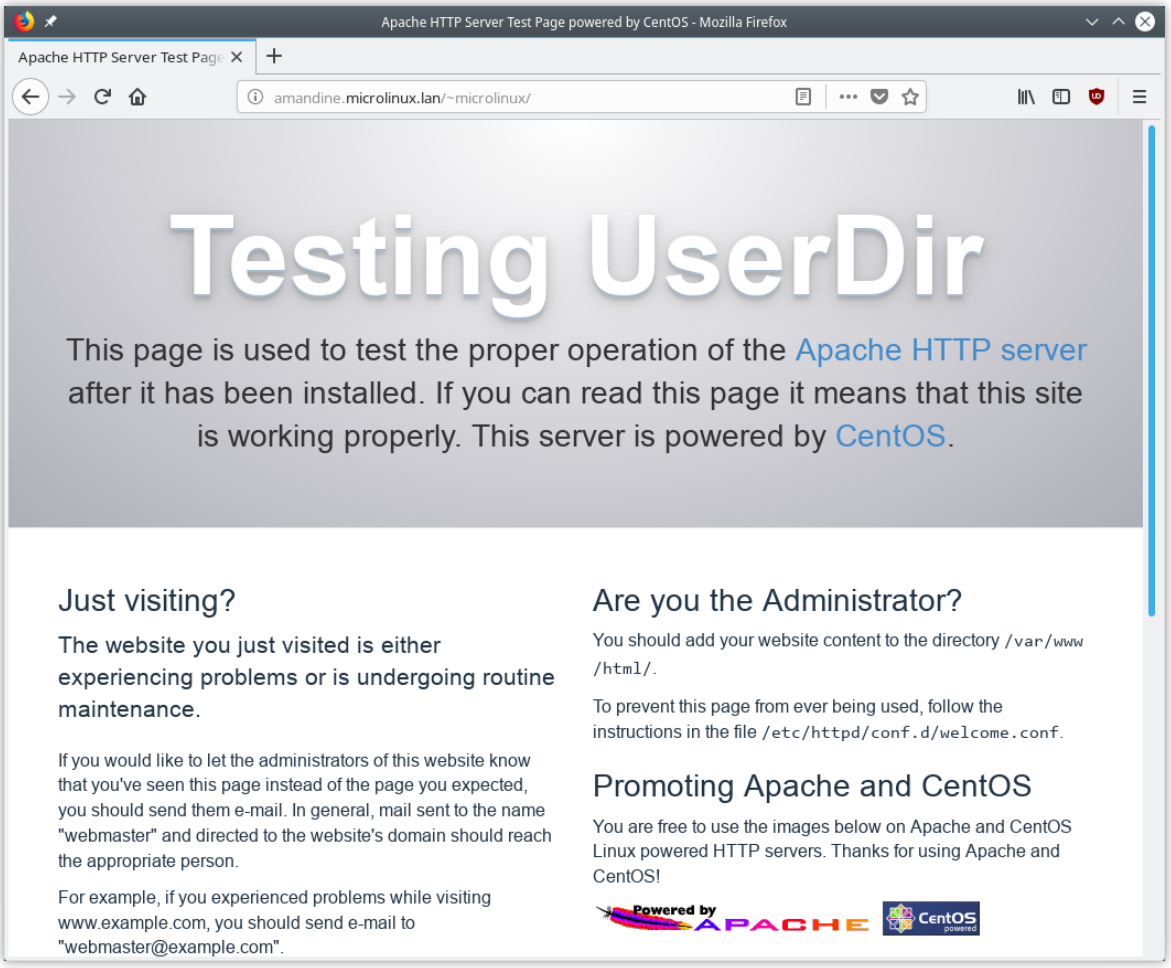
Nous allons activer l'affichage du contenu de ~/public_html pour les répertoires utilisateur. En partant de la configuration par défaut d'Apache, éditer /etc/httpd/conf.d/userdir.conf comme ceci.

```
<IfModule mod_userdir.c>
  # UserDir disabled
  UserDir public_html
</IfModule>
```

En tant que simple utilisateur, créez un répertoire ~/public_html, copiez la page web par défaut dans ce répertoire en l'éditant un tant soit peu et définissez les droits d'accès qui vont bien.

```
$ mkdir ~/public_html
$ cp -R /usr/share/httpd/noindex/* ~/public_html/
$ vim public_html/index.html (modifier le titre de la page)
$ chmod 0755 public_html
$ chmod 0711 ~
```

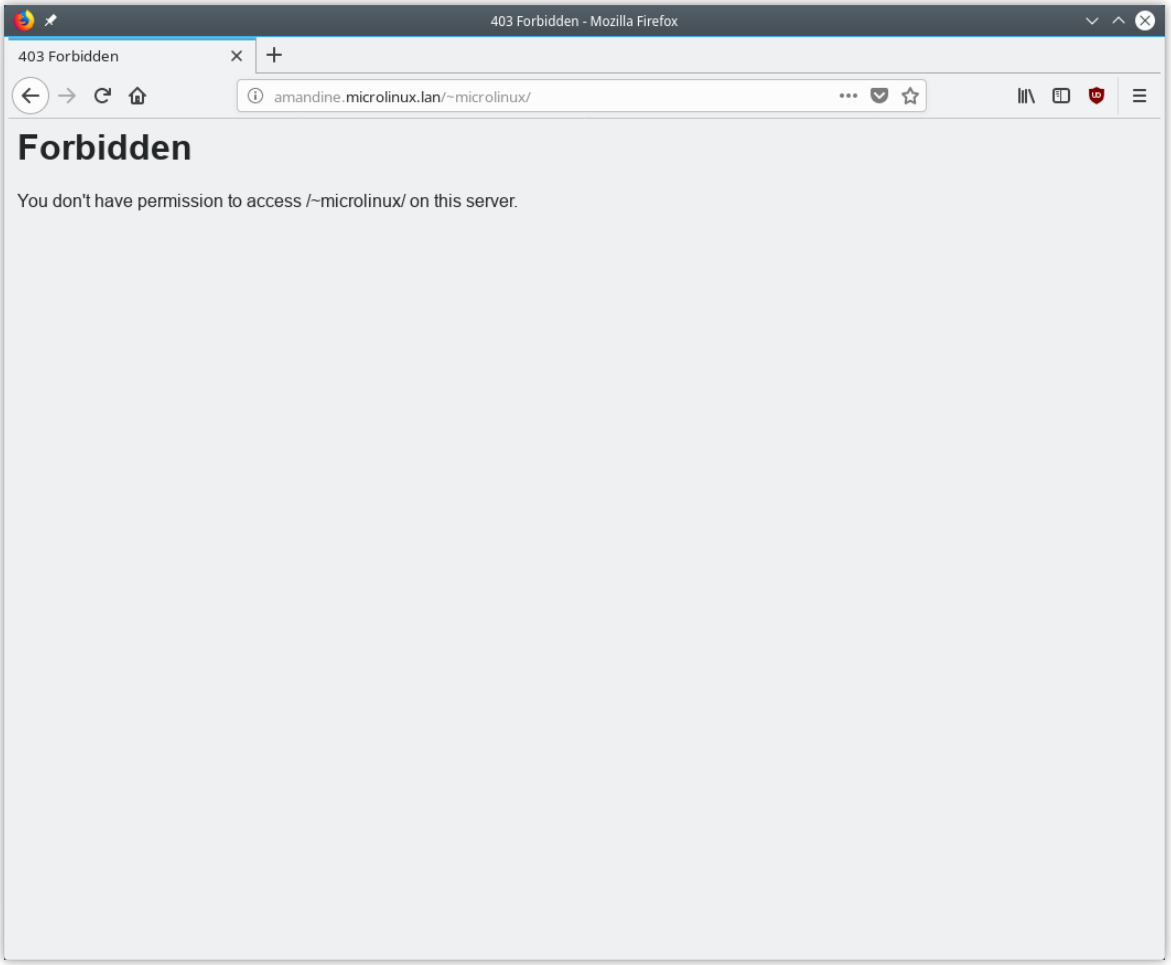
À partir de là, le site s'affiche à l'adresse http://amandine.microlinux.lan/~microlinux.



Repassons SELinux en mode strict.

```
$ sudo setenforce 1
```

Rafraichissons la page, et nous nous retrouvons avec l'erreur suivante.



Les logs dans /var/log/httpd/error_log confirment qu'il s'agit d'un problème de droits d'accès, mais ne nous en disent pas plus. Jetons un œil dans les logs de SELinux.

```
$ sudo sealert -a /var/log/audit/audit.log | less
```

La première alerte nous suggère de modifier le contexte du fichier /home/microlinux/public_html/index.html. Continuons un peu, et regardons de près la deuxième alerte.

```
*****  Plugin catchall_boolean (32.5 confidence) suggests   *****

If you want to allow httpd to enable homedirs

Then you must tell SELinux about this by enabling the

'httpd_enable_homedirs' boolean. Do

setsebool -P httpd_enable_homedirs 1
```

Gérer les booléens

Les booléens permettent de modifier une politique SELinux sans avoir à se plonger dans les arcanes de la rédaction de politiques. CentOS comporte quelques centaines de booléens, dont certains vont nous simplifier la vie.

Affichons les booléens qui concernent Apache.

```
$ getsebool -a | grep httpd
httpd_anon_write --> off
httpd_builtin_scripting --> on
httpd_can_check_spam --> off
httpd_can_connect_ftp --> off
httpd_can_connect_ldap --> off
...
```

Nous pouvons en savoir un peu plus sur le rôle de chaque booléen.

```
$ sudo semanage boolean -l | grep httpd | sort
httpd_anon_write         (off, off) Allow httpd to anon write
httpd_builtin_scripting (on, on)   Allow httpd to builtin scripting
httpd_can_check_spam     (off, off) Allow httpd to can check spam
httpd_can_connect_ftp    (off, off) Allow httpd to can connect ftp
httpd_can_connect_ldap   (off, off) Allow httpd to can connect ldap
```

Reprenons le dernier exemple pratique et tentons de corriger la politique SELinux de manière à autoriser l'affichage du contenu de ~/public_html.

```
$ sudo setsebool -P httpd_enable_homedirs on
```

Notez qu'ici l'option -P rend la directive permanente, c'est-à-dire qu'elle est conservée après un redémarrage du système.

L'ensemble des booléens personnalisés apparaît dans le fichier /var/lib/selinux/targeted/active/booleans.local. Comme le suggère l'avertissement en tête de ce fichier, cela ne sert à rien de l'éditer directement, étant donné qu'il est automatiquement regénéré à chaque invocation de setsebool.

```
# This file is auto-generated by libsemanage
# Do not edit directly.
httpd_enable_homedirs=1
```
