

SOS - Linux

Examen pratique

Nikolaos Garanis, Nathanaël Mizutani

18.04.2019

Table des matières

1	Résumé exécutif	2
1.1	Mise en place de la partition chiffrée	2
1.2	Restrictions	2
1.3	Mise en place du serveur web	2
1.4	Gestion des droits d'accès	2
2	Résumé technique	2
2.1	Mise en place de la partition chiffrée	2
2.2	Restrictions	3
2.3	Mise en place du serveur web	3
2.4	Gestion des droits d'accès	3
3	Détail des manipulations	4
3.1	Mise en place de la partition chiffrée	4
3.1.1	Création, chiffrement et formatage	4
3.1.2	Déchiffrement et montage au démarrage de la machine	4
3.2	Restrictions	5
3.2.1	Désactivation de SELinux	5
3.2.2	Masquage des processus	5
3.2.3	Demande du mot de passe pour <code>sudo -l</code>	6
3.2.4	Droits d'accès pour <code>/etc/sudoers*</code>	6
3.3	Mise en place du serveur web	6
3.3.1	Création d'un service systemd	6
3.3.2	Création d'une page web PHP	7
3.3.3	Gestion des règles iptables	7
3.4	Gestion des droits d'accès	8
3.4.1	Utilisateur devuser	8
3.4.2	Utilisateur sysuser	8
3.4.3	Utilisateur bossuser	9
4	Sources	10

1 Résumé exécutif

1.1 Mise en place de la partition chiffrée

Nous avons créé un espace mémoire chiffré sur l'ordinateur afin d'y sauvegarder les données du site web. Les données ne seront accessibles qu'au personnel autorisé à travailler sur l'ordinateur susmentionné. De cette façon les données seront protégées et leur accès contrôlé.

1.2 Restrictions

SELinux a été désactivé sur la machine. Ceci afin d'éviter des complications dans l'administration du système informatique.

Nous avons fait en sorte que chaque utilisateur ne puisse voir que son espace de travail. Il ne pourra pas voir les programmes qui ne le concernent pas. Cela limite les risques en cas de piratage ou face à un employé malveillant.

De même si un utilisateur veut effectuer des actions spéciales sur l'ordinateur, il devra introduire son mot de passe. Ceci ajoute un niveau de protection et permet d'avoir une trace de qui a fait quoi.

1.3 Mise en place du serveur web

La configuration des installations pour le site web a été fait. Un système de compartimentation a été installé afin de protéger le site web.

1.4 Gestion des droits d'accès

Un système d'autorisation par défaut afin de définir qui possède le droit de faire quoi a aussi été mis en place. Celui-ci concerne tous les utilisateurs.

Développeur web Nous avons défini les droits pour le développeur web. Il est désormais autorisé à effectuer les actions nécessaires à son travail, comme mettre en marche ou arrêter le site web. Par contre il ne pourra exécuter des actions qui sortent de son domaine de travail.

Administrateur système L'administrateur système possède désormais des droits privilégiés. Ce qui lui permettra d'effectuer toute la maintenance nécessaire sur l'ordinateur et le site web.

Management Comme demandé, le chef possède des droits nécessaires pour démarrer lui-même le site web. Il a accès en lecture aux données de celui-ci. Il peut aussi consulter le fichier `sudoers`.

2 Résumé technique

Dans cette section nous présentons, d'une manière générale, les étapes nécessaires afin d'obtenir l'environnement demandé. Nous détaillons et expliquons aussi nos choix techniques.

2.1 Mise en place de la partition chiffrée

Les fichiers du site web sont placés dans une partition séparée de celle où est installé le système d'exploitation. L'avantage est de pouvoir monter cette partition avec certaines options comme `noexec` ou `nosuid` qui empêchent l'exécution des binaires présents sur la partition. En effet, il ne devrait pas y avoir de binaire parmi les fichiers du site web. Si un attaquant réussit à en placer un, au moins il ne pourra pas l'exécuter. La partition est placée dans le deuxième disque disponible de la machine. Nous utilisons LVM (*Logical Volume Manager*) pour créer la partition.

Nous décidons aussi de chiffrer cette partition. Cela pose une protection supplémentaire dans l'éventualité où un attaquant gagne un accès physique à la machine. La partition étant chiffrée elle ne peut être lue sans le mot de passe de déchiffrement. Pour chiffrer la partition nous utilisons la spécification LUKS (*Linux Unified Key Setup*) avec l'outil `cryptsetup`.

Au démarrage de la machine, une fois le mot de passe de déchiffrement entré, la partition est déchiffrée puis montée automatiquement dans le répertoire `/var/www/html`. Ce répertoire est celui par défaut qu'Apache va utiliser pour servir les fichiers du site web.

2.2 Restrictions

L'utilisation de SELinux, activé par défaut sur CentOS 7, requiert des compétences supplémentaires qui ne sont pas demandées dans le cadre de la mise en place de notre environnement. Pour cette raison, nous désactivons SELinux afin de ne pas avoir de problèmes dû à des éléments que nous ne maîtrisons pas.

Nous restreignons aussi la visibilité des processus s'exécutant sur la machine. Par défaut, seul l'utilisateur root pourra visualiser ces processus. Il n'est en fait pas nécessaire que certains utilisateurs, par exemple le développeur web, puisse voir quels processus tournent sur la machine. Ainsi, si cet utilisateur est compromis, l'attaquant n'aura pas accès à ces informations.

Nous voulons aussi qu'à chaque fois qu'un utilisateur veuille lister ses droits sudo, il devra entrer son mot de passe. Dans le cas où un attaquant accède à un compte utilisateur sans pour autant en connaître le mot de passe, il ne pourra pas voir les droits sudo qu'il possède. C'est une protection supplémentaire contre la recherche de vulnérabilités dans les droits sudo.

2.3 Mise en place du serveur web

Le site web sera servi par un serveur Apache, lequel tournera dans un conteneur Docker. Pour que les pages dynamiques en PHP soient supportées, nous utilisons l'image Docker `php:apache` fournie par PHP. Le choix de cette image nous évite toute configuration relative à PHP et Apache. L'utilisation d'un conteneur Docker permet de rajouter une barrière de protection supplémentaire dans l'éventualité d'une compromission du serveur web.

Afin que le conteneur soit lancé au démarrage de la machine, nous créons un service systemd. Celui-ci lancera le conteneur avec les options nécessaires pour mapper le port 80 du conteneur au port 8080 de la machine et pour monter `/var/www/html` à l'intérieur du container. Le répertoire est monté en lecture seule, ce qui empêche toutes modifications des fichiers du site web depuis le conteneur. Les options spécifient aussi le nom et *hostname* qu'aura le conteneur une fois démarré, ainsi que sa suppression automatique une fois arrêté.

Afin de tester l'installation, une page PHP de test est créée à la racine du site web (sur la machine hôte). Cette page affiche simplement le *hostname* du conteneur.

Nous voulons que les requêtes HTTP, donc arrivant sur le port 80 de la machine, soient redirigées vers son port 8080. Celui-ci étant associé au port 80 du conteneur, les requêtes seront délivrées jusqu'au serveur Apache tournant dans le conteneur. Pour ce faire, deux règles iptables sont nécessaires, une pour rediriger vers le port 8080 le trafic à destination du port 80, et la deuxième pour accepter le trafic à destination du port 8080. Ces deux règles doivent bien entendu être persistantes et donc être chargées au démarrage de la machine.

2.4 Gestion des droits d'accès

Ont accès à la machine trois utilisateurs (qui doivent être créés), chacun avec des droits distincts, adaptés à leur fonction. L'utilisateur devuser, s'occupant du développement web, a la possibilité de faire un `pull` de l'image Docker mentionnée plus haut. Il doit aussi pouvoir (re)démarrer, arrêter et activer le service systemd créé. Ces droits sont spécifiés dans un nouveau fichier `sudoers`, réservé à cet utilisateur. Nous utilisons aussi les ACL afin de donner les droits de lecture et d'écriture pour le fichier `/etc/hosts` ainsi que tous les fichiers dans `/var/www/html`.

L'utilisateur sysuser gère l'administration de la machine, il possède des privilèges plus élevés. Tout d'abord, il doit appartenir aux groupes `wheel` et `docker`. Ensuite, tout comme devuser, il peut lire et modifier tous les fichiers du site web. En plus de cela, l'utilisateur doit pouvoir exécuter n'importe quelle commande en tant que root. Ceci demande la création d'un nouveau fichier `sudoers` contenant les droits appropriés. Nous voulons aussi que l'utilisateur puisse voir tous les processus tournant sur la machine. Il nous faut pour cela utiliser l'option `gid`, qui permet de faire une exception à l'option `hidepid` que nous avons utilisé précédemment pour le montage de la partition du système de fichier `/proc`.

Quant à l'utilisateur bossuser, il possède des droits restreints. Il ne peut que lire les fichiers du site web et non pas les modifier. Comme pour les autres utilisateurs, nous gérons ces droits en utilisant les ACL. Les commandes

que peut effectuer l'utilisateur sont aussi restreintes, il ne peut que démarrer et arrêter le service du conteneur. Un nouveau fichier `sudoers` est créé pour cela. Additionnellement, nous voulons que l'utilisateur puisse lire le fichier `/etc/sudoers`. Les ACL sont utilisées pour ce faire.

3 Détail des manipulations

Nous présentons dans cette section les commandes effectuées ainsi que les fichiers de configuration nécessaires à la mise en place du site web. Nous prenons comme base la machine virtuelle fournie lors du cours de *Sécurité des Systèmes d'Exploitation* (SOS). Notre environnement possède alors les caractéristiques suivantes :

- système d'exploitation CentOS 7,
- utilisateur possédant les droits administrateurs,
- disque libre sur lequel seront mis les fichiers du site web,
- Docker installé.

3.1 Mise en place de la partition chiffrée

3.1.1 Création, chiffrement et formatage

Afin d'initialiser la partition nous utilisons LVM (*Logical Volume Manager*). Nous commençons par créer un *volume physique* sur le deuxième disque de la machine (`/dev/sdb` dans notre cas).

```
$ sudo pvcreate /dev/sdb
```

Puis, il est nécessaire de créer un *groupe virtuel* dans le volume initialisé ci-dessus. Nous le nommons `vg-site`.

```
$ sudo vgcreate vg-site /dev/sdb
```

Nous créons maintenant un *volume logique* à l'intérieur du groupe `vg-site` occupant 100% de l'espace disponible. Nous le nommons `lv-site`.

```
$ sudo lvcreate -l100%FREE vg-site -n lv-site
```

Nous utilisons ensuite LUKS (*Linux Unified Key Setup*) pour chiffrer la partition, étape à laquelle il nous est demandé d'entrer le mot de passe pour chiffrer la partition, nous utilisons `vg-site123;`.

```
$ sudo cryptsetup luksFormat /dev/vg-site/lv-site
```

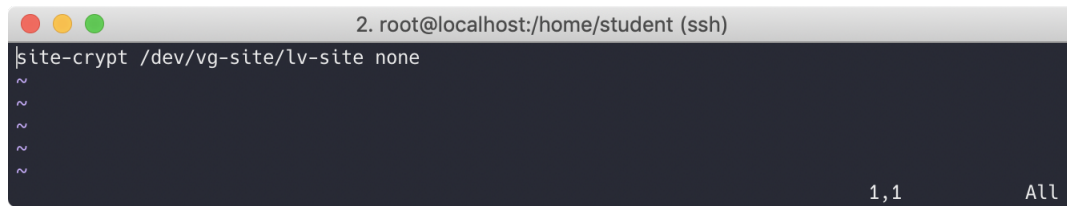
Il faut maintenant déchiffrer la partition afin de la formater en `ext4`.

```
$ sudo cryptsetup open --type luks /dev/vg-site/lv-site site-crypt
$ sudo mkfs.ext4 /dev/mapper/site-crypt
```

3.1.2 Déchiffrement et montage au démarrage de la machine

Afin que la partition soit déchiffrée au démarrage de la machine, une entrée doit être ajoutée dans le fichier `/etc/crypttab`.

```
$ echo "site-crypt /dev/vg-site/lv-site none" | sudo tee -a /etc/crypttab
```

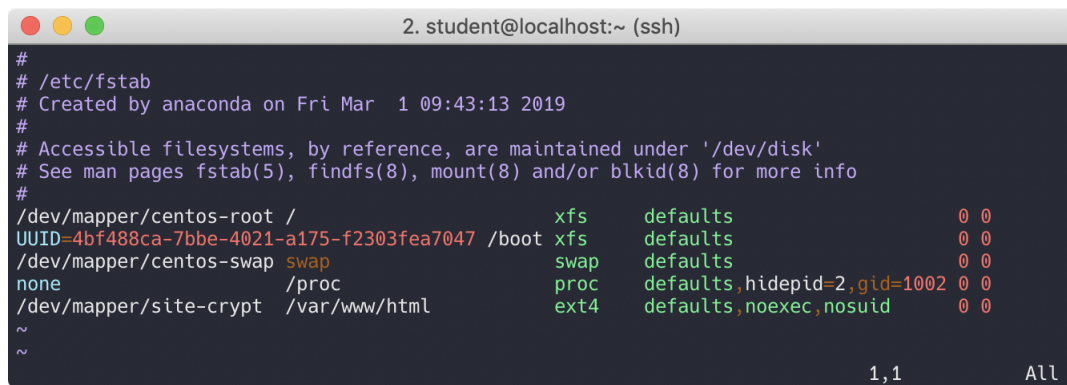


```
2. root@localhost:/home/student (ssh)
cat /etc/crypttab
/dev/vg-site/lv-site none
~
~
~
~
1,1 All
```

FIGURE 1 – Contenu du fichier crypttab

Une entrée doit aussi être ajoutée au fichier `/etc/fstab` pour que la partition déchiffrée soit montée automatiquement dans `/var/www/html` avec les options `nosuid` et `noexec`.

```
$ echo "/dev/mapper/site-crypt /var/www/html ext4 defaults,noexec,nosuid 0 0" \
| sudo tee -a /etc/fstab
```



```
2. student@localhost:~ (ssh)
cat /etc/fstab
#
# /etc/fstab
# Created by anaconda on Fri Mar 1 09:43:13 2019
#
# Accessible filesystems, by reference, are maintained under '/dev/disk'
# See man pages fstab(5), findfs(8), mount(8) and/or blkid(8) for more info
#
/dev/mapper/centos-root / xfs defaults 0 0
UUID=4bf488ca-7bbe-4021-a175-f2303fea7047 /boot xfs defaults 0 0
/dev/mapper/centos-swap swap swap defaults 0 0
none /proc proc defaults,hidepid=2,gid=1002 0 0
/dev/mapper/site-crypt /var/www/html ext4 defaults,noexec,nosuid 0 0
~
1,1 All
```

FIGURE 2 – Contenu du fichier fstab

3.2 Restrictions

3.2.1 Désactivation de SELinux

Pour désactiver SELinux il faut éditer le fichier `/etc/sysconfig/selinux` et y changer la valeur de la variable `SELINUX` à `disabled`. Afin que le changement soit pris en compte la machine doit être redémarrée.

```
$ sudo sed -i -E 's/^(SELINUX=).*$/\1disabled/' /etc/sysconfig/selinux
$ reboot
$ sestatus
SELinux status: disabled
```

3.2.2 Masquage des processus

Pour empêcher les utilisateurs de voir les processus qui tournent sur le système il nous faut modifier le fichier `/etc/fstab` afin d'y ajouter une entrée pour le système de fichier `/proc`, celui-ci doit être monté avec l'option `hidepid`. Pour que la modification prenne effet, il faut le remonter.

```
$ echo -e "none /proc proc defaults,hidepid=2 0 0" | sudo tee -a /etc/fstab
$ sudo mount -o remount /proc

# vérification
$ mount | grep hidepid=2
proc on /proc type proc (rw,nosuid,nodev,noexec,relatime,hidepid=2)
```

3.2.3 Demande du mot de passe pour `sudo -l`

Afin que le mot de passe d'un utilisateur exécutant la commande `sudo -l` lui soit toujours demandé, nous créons le fichier `/etc/sudoers.d/general`.

```
$ echo Defaults listpw=always | sudo tee /etc/sudoers.d/general
```

3.2.4 Droits d'accès pour `/etc/sudoers*`

Les droits en lecture du fichier `/etc/sudoers` ne doivent être attribués qu'à l'utilisateur `root`.

```
$ sudo chmod 400 /etc/sudoers
```

Afin de faire de même avec tous les fichiers qui seront créés dans le répertoire `/etc/sudoers.d`, nous lui attribuons des ACL par défaut.

```
$ sudo setfacl -d -m u::r,g::- ,o::- /etc/sudoers.d
```

3.3 Mise en place du serveur web

3.3.1 Création d'un service `systemd`

Nous définissons un service `systemd` qui s'occupe de lancer et d'arrêter le conteneur Docker dans lequel tourne le serveur Apache. Pour cela, nous créons le fichier `/etc/systemd/system/httpd-php-container.service` avec le contenu suivant :

```
[Unit]
Description=httpd-php-container

[Service]
Restart=always
ExecStart=/usr/bin/docker run --rm --name httpd-php-container --hostname \
httpd-php-container -p 8080:80 -v /var/www/html:/var/www/html:ro php:apache
ExecStop=/usr/bin/docker container kill httpd-php-container

[Install]
WantedBy=multi-user.target
```

```

2. root@localhost:/etc/systemd/system (ssh)
[Unit]
Description=httpd-php-container

[Service]
Restart=always
ExecStart=/usr/bin/docker run --rm --name httpd-php-container --hostname httpd-php-container -p 8080:80 -v /var/www/html:/var/www/html:ro php:apache
ExecStop=/usr/bin/docker container kill httpd-php-container

[Install]
WantedBy=multi-user.target
~
~
2,1 All

```

FIGURE 3 – Contenu du fichier `httpd-php-container.service`

Il nous faut maintenant activer le service afin que celui-ci se lance au démarrage de la machine (l'option `-now` permet en plus de le lancer immédiatement) :

```
$ sudo systemctl enable --now httpd-php-container
```

3.3.2 Création d'une page web PHP

Dans le répertoire `/var/www/html` de la machine, nous créons une page test en PHP affichant le hostname du conteneur.

```
$ echo "<?php echo 'Hello from: ' . gethostname() . \"\n\"; ?>" \
| sudo tee /var/www/html/index.php
```

3.3.3 Gestion des règles iptables

Afin de rediriger le trafic entrant sur le port 80 de la machine vers le port 8080 de celle-ci, nous ajoutons une règle iptables dans la chaîne PREROUTING de la table `nat`.

```
$ sudo iptables -t nat -A PREROUTING -p tcp --dport 80 -j REDIRECT --to-port 8080
```

Nous devons aussi accepter le trafic entrant sur le port 8080 de la machine.

```
$ sudo iptables -I INPUT -p tcp --dport 8080 -j ACCEPT
```

Afin de rendre ces règles persistantes, nous les sauvegardons dans le fichier `/etc/sysconfig/iptables`.

```
$ sudo iptables-save | sudo tee /etc/sysconfig/iptables
```

Puis, il nous faut créer le script `/sbin/ifup-local` qui sera exécuté à chaque fois qu'une interface réseau sera activée. Le fichier doit être exécutable et son contenu est le suivant :

```
$ sudo chmod +x /sbin/ifup-local
$ cat /sbin/ifup-local
#!/usr/bin/env sh
iptables-restore < /etc/sysconfig/iptables
```

3.4 Gestion des droits d'accès

3.4.1 Utilisateur devuser

Création Nous commençons par créer l'utilisateur devuser :

```
$ sudo useradd -m -s /bin/bash devuser
```

Droits sudo Afin que l'utilisateur puisse télécharger l'image Docker `php:apache` ainsi que (re)démarrer, arrêter et activer le service `httpd-php-container`, nous lui autorisons ces actions dans le fichier `/etc/sudoers.d/devuser` que nous créons avec le contenu suivant :

```
devuser ALL = /usr/bin/docker pull php:apache, \
             /usr/bin/systemctl start httpd-php-container, \
             /usr/bin/systemctl stop httpd-php-container, \
             /usr/bin/systemctl restart httpd-php-container, \
             /usr/bin/systemctl enable httpd-php-container
```

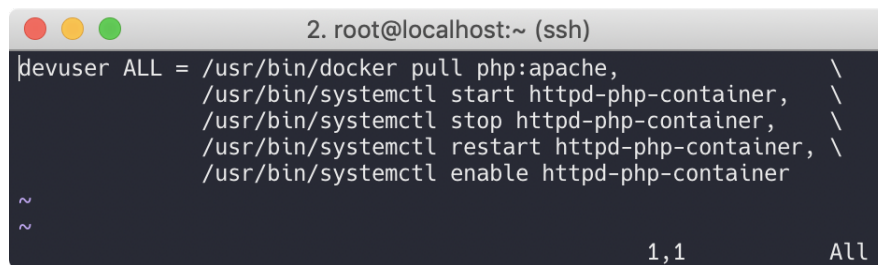


FIGURE 4 – Contenu du fichier devuser

ACL L'utilisateur doit aussi pouvoir lire et écrire tous les fichiers présents et futurs dans `/var/www/html`. Pour les fichiers déjà présents, ou pour ceux qui seront déplacés dans ce répertoire, nous appliquons les ACL de manière récursive.

```
$ sudo setfacl -R -m u:devuser:rw- /var/www/html
```

Pour les fichiers qui seront créés dans le répertoire, nous lui attribuons des ACL par défaut.

```
$ sudo setfacl -d -m u:devuser:rw- /var/www/html
```

Afin que l'utilisateur puisse modifier le fichier `/etc/hosts`, nous appliquons les ACL suivantes :

```
$ sudo setfacl -m u:devuser:rw- /etc/hosts
```

3.4.2 Utilisateur sysuser

Création Nous créons l'utilisateur sysuser et nous l'ajoutons directement dans les groupes `wheel` et `docker` :

```
$ sudo useradd -m -s /bin/bash -G wheel,docker sysuser
```

Droits sudo Pour autoriser à l'utilisateur tous les droits sudo sans aucune restriction, nous créons le fichier `/etc/sudoers.d/sysuser`.


```
$ echo sysuser ALL = (ALL) ALL | sudo tee /etc/sudoers.d/sysuser
```

ACL Afin qu'il puisse modifier les fichiers du site web nous appliquons les mêmes ACL que pour l'utilisateur devuser :

```
$ sudo setfacl -R -m u:sysuser:rw- /var/www/html
$ sudo setfacl -d -m u:sysuser:rw- /var/www/html
```

Exception hidepid Nous avons précédemment ajouter l'option de montage `hidepid` au système de fichiers `/proc`. Afin que cette option ne s'applique pas à l'utilisateur, nous devons rajouter l'option `gid` à l'entrée `/etc/fstab` de `/proc`. La valeur à utiliser est l'id du groupe principal auquel appartient sysuser.

```
$ id -g sysuser
1002 # par exemple

# contenu de l'entrée après modification
$ cat /etc/fstab | grep /proc
none /proc proc defaults,hidepid=2,gid=1002 0 0

# appliquer le changement
$ sudo mount -o remount /proc
```

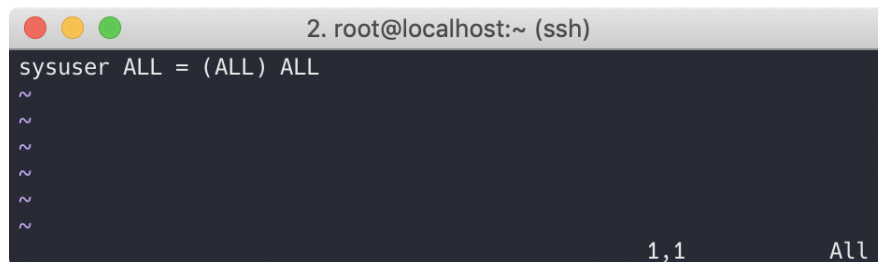


FIGURE 5 – Contenu du fichier `sysuser`

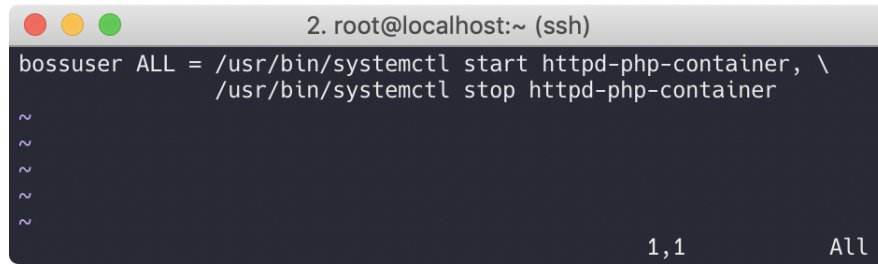
3.4.3 Utilisateur bossuser

Création Nous commençons par créer l'utilisateur bossuser :

```
$ sudo useradd -m -s /bin/bash bossuser
```

Droits sudo L'utilisateur doit pouvoir arrêter et démarrer le service `systemd` créé précédemment, pour cela nous créons le fichier `/etc/sudoers.d/bossuser` avec le contenu suivant :

```
bossuser ALL = /usr/bin/systemctl start httpd-php-container, \
               /usr/bin/systemctl stop httpd-php-container
```

A terminal window titled "2. root@localhost:~ (ssh)" displays the content of the bossuser file. The text is: "bossuser ALL = /usr/bin/systemctl start httpd-php-container, \ /usr/bin/systemctl stop httpd-php-container". Below this, there are five tilde (~) characters on the left margin. At the bottom right, it shows "1,1" and "All".

```
2. root@localhost:~ (ssh)
bossuser ALL = /usr/bin/systemctl start httpd-php-container, \
               /usr/bin/systemctl stop httpd-php-container
~
~
~
~
~
1,1 All
```

FIGURE 6 – Contenu du fichier bossuser

ACL Pour que l'utilisateur puisse seulement lire les fichiers du site web, nous utilisons les ACL par défaut sur le répertoire. Nous appliquons aussi les ACL de manière récursive pour les fichiers déjà présents.

```
$ sudo setfacl -R -m u:bossuser:r-- /var/www/html
$ sudo setfacl -d -m u:bossuser:r-- /var/www/html
```

L'utilisateur doit aussi pouvoir lire le fichier `/etc/sudoers`, nous utilisons encore une fois les ACL :

```
$ sudo setfacl -m u:bossuser:r-- /etc/sudoers
```

4 Sources

- > <https://www.vmotionhost.com/client-area/knowledgebase.php?action=displayarticle&id=163>
- > <https://www.tecmint.com/disable-selinux-temporarily-permanently-in-centos-rhel-fedora/>
- > <https://linux-audit.com/linux-system-hardening-adding-hidepid-to-proc/>
- > <https://www.sudo.ws/man/1.8.27/sudoers.man.html>
- > https://hub.docker.com/_/httpd
- > https://hub.docker.com/_/php/