Labo SELinux Durée = 60 min

1	Objectifs	sudo ./c 2

Découvrir et comprendre les principaux mécanismes de sécurité offerts par SELinux
Connaître les outils utiles (CLI & GUI) à l'administration
Comprendre les messages d'erreur produits lors d'un **blocage** et les résoudre
Identifier la configuration SELinux d'une distribution Fedora16

2	Postes de travail (image = Win7)

Session Ouvrir une session administrateur sous Windows 7 : compte=**albert** username=**admin**

Action Copier le dossier partagé **\\10.2.1.1\doclabo\RSX\7_SELinux** sur le bureau
Lancer **VirtualBox** (raccourci bureau)
Fichier – Importer **Fedora16.ova**

Info L'image Fedora 16 - 64bit a été modifiée comme suit :

- Création de 4 comptes utilisateurs

User	Pass	Description
root	rootroot	
admin	adminadmin	admin système
paul	paulpaul	utilisateur avec droit minimum Voir user_u dans Annexe 1
jean	jeanjean	admin web

- Installation d'outils d'administration
  SELinux Management
  SELinux Audit Log Analysis
  SELinux Policy Generation Tool
  APOL

- Installation des services utiles pour les scénarios
  httpd
  mcstrans

Ces informations utiles se trouvent dans la rubrique **Description** de cette machine virtuelle (VM)

Action Démarrer cette VM qui ouvre automatiquement une session **admin**

Contrôler en haut à droite que la session est bien admin

Identifier les raccourcis bureau

Ouvrir un terminal avec le raccourci 

3 Type enforcement

But 3.1 **Afficher le contexte de sécurité de l'utilisateur courant**

Action Exécuter la commande `id -Z`

Q_3.1a Relever le contexte de sécurité (user:role:type) retourné
unconfined_u:unconfined_r:unconfined_t

But 3.2 **Afficher le contexte de sécurité de chaque fichier**

Action Dans le dossier courrant `/home/admin/` exécuter la commande `ls -Z`

Q_3.2a Relever les contextes de sécurité retournés
unconfined_u:object_r:user_home_t
unconfined_u:object_r:audio_home_t

Action
```
cd Documents
echo bonjour > hello.txt
ls -Z
```

Q_3.2b Relever le contexte de sécurité du fichier hello.txt. Comment ce contexte a-t-il été défini ?
unconfined_u:object_r:user_home_t
Ce contexte a été fixé par héritage du dossier parent

But 3.3 **Afficher le contexte de sécurité d'un processus**

Action `ps -eZ`

Q_3.3a Relever les 2 domaines de sécurité les plus présents dans les processus
system_u:system_r:kernel_t
unconfined_u:unconfined_r:unconfined_t

Remarque Observer les nombreux domaines : init_t, kernel_t, udev_t, auditd_t, NetworkManager_t, abrt_t, abrt_dump_oops_t, avahi_t, crond_t, keyboardd_t, systemd_logind_t, mcelog_t, audisp_t, syslogd_t, system_dbusd_t, policykit_t, modemmanager_t, lldpad_t, dhcpc_t, fcoemon_t, sendmail_t, xdm_t, xserver_t, consolekit_t, accountsd_t, unconfined_t, rtkit_daemon_t, devicekit_power_t, cupsd_t, rpm_t, devicekit_disk_t, colord_t, gnomeclock_t

Action
```
less hello.txt
```
Ouvrir un nouveau terminal (terminal 2) et exécuter `ps -eZ | grep less`

Q_3.3b Relever le domaine de sécurité de la commande less
unconfined_u:unconfined_r:unconfined_t

Action Sortir de la commande less avec <q>
`passwd` (ne pas entrer de mot de passe)
Afficher le contexte de sécurité dans terminal 2

Q_3.3c Relever le domaine de sécurité de la commande passwd
unconfined_u:unconfined_r:passwd_t

Q_3.3d Quel est le nom de l'opération qui a permis à la commande passwd de se lancer dans le domaine passwd_t depuis le domaine unconfined_t ?
Une transition de domaine

Action Fermer terminal 2

But 3.4 **Trouver les règles de transition et d'autorisation de la commande passwd**

Utiliser la commande **sesearch** (man sesearch) pour retrouver les règles ci-dessous :

Domain transition (Labo 3.4)

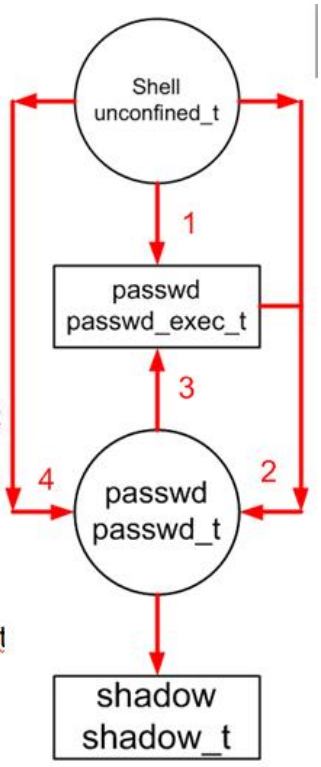
- **Domain** = type of a process = kernel_t, …, unconfined_t, passwd_t

1 Autoriser le domaine unconfined_t à exécuter un fichier de type passwd_exec_t
allow unconfined_t passwd_exec_t : file {execute}

2 Si 1 → autoriser la transition vers dom passwd_t
type_transition unconfined_t passwd_exec_t : process passwd_t

3 Autoriser le domaine passwd_t à utiliser un fichier de type passwd_exec_t comme point d'entrée
allow passwd_t passwd_exec_t : file {entrypoint}

4 Autoriser le dom unconfined_t à transiter au dom passwd_t
allow unconfined_t passwd_t : process transition



Q_3.4a Quel est le contexte de sécurité du fichier **/usr/bin/passwd** ?
`ls -Z /usr/bin/passwd → system_u:object_r:passwd_exec_t`

Q_3.4b Compléter `sesearch -A -s ..... -t .....` pour retrouver la règle 1
`sesearch -A -s unconfined_t -t passwd_exec_t`

Remarque Trop de résultats sont retournés par la commande car la recherche est trop générale.
Dans le cas présent, il suffit de rechercher les occurrences directes
`sesearch -A -s unconfined_t -t passwd_exec_t -d`

Q_3.4c Compléter `sesearch -T -s ..... –t .....` pour la règle 2
`sesearch -T -s unconfined_t -t passwd_exec_t`

Q_3.4d Traduire cette règle en français
Une transition a lieu vers le domaine passwd_t si une application du domaine unconfined_t (source) lance un exécutable de type passwd_exec_t (target)

Q_3.4e Compléter `sesearch -A -s ..... -t .....` pour la règle 3
`sesearch -A -s passwd_t -t passwd_exec_t`

Q_3.4f Compléter `sesearch -A -s ..... -t .....` pour la règle 4
`sesearch -A -s unconfined_t -t passwd_t -d`

Q_3.4g Traduire cette règle en français
Le domaine unconfined_t est autorisé à transiter vers le domaine passwd_exec_t

Remarque Dans les règles allow, il est intéressant d'analyser les permissions que possède un domaine sur les différents types.

Q_3.4h Rechercher pour le domaine passwd_t, les types sur lesquels il a la permission d'écriture.
Compléter `sesearch -T -s ..... –p ..... -d`
`sesearch -T -s passwd_t –p write -d`

Q_3.4i Est-ce que passwd_t à la permission d'écriture sur le type shadow_t de la classe fichier ?
`Oui, sesearch -A -s passwd_t -t shadow_t -c file –p write`

But 3.5 **Explorer le contexte de sécurité associé à chaque port**
Scénario : Rechercher sur quel port tcp le serveur apache peut écouter.

Action `sesearch -A -s httpd_t -c tcp_socket -p name_bind -d`

Q_3.5a Relever les types de sockets qui peuvent être écoutées par le domaine httpd_t

```
allow httpd_t ntop_port_t : tcp_socket name_bind ;
allow httpd_t http_cache_port_t : tcp_socket name_bind ;
allow httpd_t http_port_t : tcp_socket name_bind ;
allow httpd_t puppet_port_t : tcp_socket name_bind ;
allow httpd_t jboss_management_port_t : tcp_socket name_bind ;
allow httpd_t ephemeral_port_type : tcp_socket name_bind ;
allow httpd_t ftp_port_t : tcp_socket name_bind ;
```

Action Démarrer l'application **GUI SELinux Management** grâce au raccourci

Attendre (~30 sec) le chargement de l'application
Utiliser la rubrique (Select) **Network Port** pour identifier les numéros de port autorisé

Filter http_port_t

SELinux Port Type	Protocol	MLS/MCS Level	Port
http_port_t	tcp	s0	8443
http_port_t	tcp	s0	8009
http_port_t	tcp	s0	8008
http_port_t	tcp	s0	488
http_port_t	tcp	s0	443
http_port_t	tcp	s0	80

4 Utilisateurs

But 4.1 **Utilisateurs SELinux prédéfinis**

Action Dans **SELinux Management – SELinux User**, identifier les 9 utilisateurs prédéfinis de Fedora16
La commande **`seinfo -u`** donne les mêmes résultats

Info **unconfined_u** est l'environnement où tous les utilisateurs Linux sont liés par défaut.
Cet utilisateur donne accès au domaine **unconfined_t** qui a accès à la plupart des domaines présents.

user_u est utilisateur ordinaire non privilégié qui a le droit d'établir une session et d'utiliser les logiciels qui ne servent pas à l'administration du système (ex. calculatrice, web browser,…).
Cet utilisateur n'a pas accès au réseau en écoute et ne peut pas démarrer un service

staff_u est identique à user_u et permet l'accès aux programmes setuid - getgid et à d'autres fonctions du système (état des processus,…).

Voir aussi Annexe 1

But 4.2 **Utilisateur par défaut**

Q_4.2a Quel est le contexte de sécurité de l'utilisateur courant ?
`id -Z` → **unconfined_u:unconfined_r:unconfined_t**

Action Dans **SELinux Management - User Mapping**

Remarque **semanage** est l'outil d'administration en ligne de commande

Action Depuis un bash **root** :
`semanage login -l`

Info L'utilisateur SELinux permet de définir le contexte de sécurité du shell obtenu après authentification.
L'entrée __default__ permet de lier (mapper) les comptes Linux à un utilisateur SELinux.

Le fichier **`/etc/selinux/targeted/contexts/users/unconfined_u`** contient cette équivalence

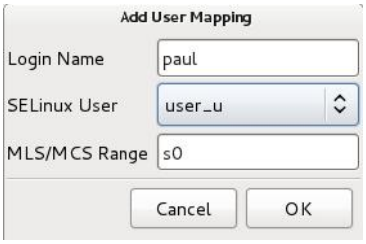
Action Afficher ce fichier (utiliser la touche <tab> pour le remplissage automatique)
`cat /etc/selinux/targeted/contexts/users/unconfined_u`

Q_4.2b Quel est le contexte de sécurité obtenu après un local login ?
unconfined_r:unconfined_t

But 4.3 **Restreindre l'utilisateur paul avec les droits user_u**

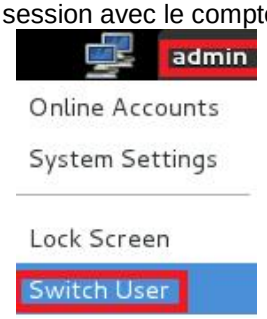
Remarque Afin d'améliorer la sécurité du système, le compte **root** est le seul compte qui devrait être lié à **unconfined_u**.
Dans un système bien configuré, le compte **root** est un compte à n'utiliser qu'en cas d'urgence.

Action Dans **SELinux Management - User Mapping - Add**



Info L'entrée spécifique créée pour l'utilisateur paul va permettre de le lier à user_u.
Le champ MLS/MCS Range sera expliqué au §7.

Action **Switch User**, puis établir une session avec le compte **paul**



Q_4.3a Pouvez-vous démarrer SELinux Management ? **Non**
Avez-vous accès à la commande su ? **Non**
Avez accès à la commande pwd ? **Oui**
Quel est le contexte du bash de paul : **id –Z → user_u:user_r:user_t**

Action `ls -l /var/`

Q_4.3b **Quelle est la raison de ces points d'interrogations ?**
user_u n'a pas les permissions nécessaires pour afficher les attributs de ces dossiers (permission getattr manquante)

Action Revenir à la **session 1 (admin)** en typant **<Ctrl>+<Alt>+<F1>**
Info Vous pouvez revenir à la **session 2 (paul)** en typant **<Ctrl>+<Alt>+<F2>**

But 4.4 **Comparer le domaine user_t avec le domaine unconfined_t**

Q_4.4a Combien de règles de transition de domaine la commande retourne pour unconfined_t ?
Action `sesearch -T -s unconfined_t -c process | less`
185

Q_4.4b Combien de règles de transition de domaine la commande retourne pour user_t ?
Action `sesearch -T -s user_t -c process | less`
55

5	Blocages

But 5.1 **Utilisation d'un booléen**
Interdire à user_u de lancer un exécutable depuis son répertoire personnel

Remarque Dans le scénario suivant, considérer l'exécutable pwd2 comme un binaire quelconque que l'utilisateur paul a dans son dossier /home/paul

Action Avec le compte paul en typant **<Ctrl>+<Alt>+<F2>**

Q_5.1a Relever le contexte du binaire /bin/pwd ?
```
ls -Z /bin/pwd
```
system_u:objet_r:bin_t

Action
```
cp /bin/pwd ~/pwd2
```
copier le binaire dans le dossier personnel
```
cd ~
```

Q_5.1b Relever le contexte du binaire ~/pwd2 ?
```
ls -Z ~/pwd2
```
user_u:objet_r:user_home_t

Action
```
./pwd2
```
lancer l'exécutable depuis le dossier courant en ajoutant **./**

Q_5.1c Est-ce que vous avez accès à la commande pwd2 ? **Oui**

Action Dans **SELinux Management** (compte admin)
Menu **Boolean**, décocher la valeur **`allow_user_exec_content`**
Parcourir la liste pour découvrir les modules installés
L'élément se trouve dans le module unknown

Remarque Penser à l'avenir à utiliser le filtre pour faciliter la recherche

Q_5.1d Avez-vous accès à la commande pwd2 avec le compte paul ? **Non**

Remarques La commande `semanage boolean -l` affiche ces booléens.

But 5.2 **Utilisation des logs**
Scénario : paul n'a pas pu exécuter un binaire depuis son dossier personnel (But 5.1).

Remarque Dans SELinux, les logs sont très importants. Ils permettent même d'être utilisés pour générer des petits modules patch pour débloquer une situation.

Action Dans la session **admin**, lancer l'application GUI **SELinux Audit Log Analysis**

Message	Date	Source Type	Target Type	Object Class	Permission	Executable	Command
Denied	Oct 12 14:57:12	user_t	user_home_t	file	execute		bash

Action Dans un bash **root**, chercher l'entrée log correspondante aux informations ci-dessus

```
ausearch -m AVC -c bash -ts today
```

Remarque Nous recherchons dans les logs les entrées faites par AVC = Access Vector Cache
Nous filtrons sur la commande qui a causé le log, ici le bash (domaine user_t) qui s'est fait refuser le droit d'exécuter pwd.
Finalement nous filtrons sur les logs d'aujourd'hui

Q_5.2a Remplir le tableau à partir des informations du log :

```
time->Wed Oct 24 15:11:05 2012
type=SYSCALL msg=audit(1351084265.713:91): arch=c000003e syscall=59 success=no exit=-13 a0=105ad40 a1=e68810 a2
=10555f0 a3=8 items=0 ppid=1977 pid=2106 auid=1003 uid=1003 gid=1003 euid=1003 suid=1003 fsuid=1003 egid=1003 s
gid=1003 fsgid=1003 tty=pts0 ses=4 comm="bash" exe="/bin/bash" subj=user_u:user_r:user_t:s0 key=(null)
type=AVC msg=audit(1351084265.713:91): avc:  denied  { execute } for  pid=2106 comm="bash" name="pwd2" dev=dm-1
 ino=159490 scontext=user_u:user_r:user_t:s0 tcontext=user_u:object_r:user_home_t:s0 tclass=file
```

Temps :	**Date et heure**
Type log :	**AVC et SYSCALL**
Succès :	**Non**
Commande :	**bash**
Contexte source :	**user_u:user_r:user_t**
Nom de la cible :	**pwd2**
Contexte cible :	**user_u:object_r:user_home_t**
Classe de la cible :	**file**
Permission(s) manquante(s) :	**execute**

Action Utiliser la commande audit2why pour obtenir une explication textuelle du blocage

```
ausearch -m AVC -c bash -ts today | audit2why
```

Q_5.2b Qu'avez-vous appris ?
The boolean allow_user_exec_content was set incorrectly

Action Utiliser la commande audit2allow pour trouver une solution

```
ausearch -m AVC -c bash -ts today | audit2allow
```

Q_5.2c Quelle est la solution ?
allow user_t user_home t:file execute;

Remarque La commande audit2allow, nous a proposé deux solutions.
La première consiste à modifier l'état d'un booléen.
La deuxième propose d'ajouter une règle qui autorise le domaine user_t à exécuter un fichier de type user_home_t.
C'est à l'administrateur de décider s'il désire autoriser l'accès ou non et quelle solution choisir.

But 5.3 **Résoudre un blocage**
Scénario : Autoriser **paul** à lancer netcat qui écoute le port 1234
Générer un module avec la commande audit2allow

Action Avec le compte **paul**, contrôler l'accès à la commande netcat (nc) dans un terminal
```
printf "GET / HTTP/1.0\r\n\r\n" | nc www.tdeig.ch 80
```

Q_5.3a Qu'avez-vous obtenu ? **Le code HTML de la page web**

Action Lancer un netcat en écoute sur le port 1234 : `nc -l 1234`

Q_5.3b Que s'est-il passé ? **Action refusée.**

Q_5.3c Quelle est l'explication fournie par audit2why (compte **root**) à la commande
```
ausearch -m AVC -c nc -ts today | audit2why
```
user_t n'a pas la permission name_bind sur le contexte du port 1234

Action Avec le compte **root**, générer un module qui ajoute la règle manquante pour autoriser user_t à écouter sur le port 1234.
```
ausearch -m AVC -c nc -ts today | audit2allow –M toto
```

Action `ls –l`
Info Le dossier courant contient le fichier source généré (.te) ainsi que la version compilée (.pp)

Installer le module à l'aide de la commande :
```
semodule -i toto.pp
```

Tester à nouveau la commande dans la session de **paul** :
```
nc -l 1234
```

Ouvrir un deuxième terminal :
```
nc localhost 1234
bonjour<ENTER>
```
Vérifier si le message bonjour a bien été affiché dans le terminal 1

But 5.4 **Résoudre un blocage, en changeant le type d'un dossier**
Scénario : L'admin souhaite faire une sauvegarde des personnalisations apportées à son système (mappage utilisateur, booléen, changement de contexte, …) dans un dossier présent dans son répertoire personnel.
La commande **semanage** permet de faire ce backup, mais elle n'a pas l'autorisation d'accès et d'écriture au dossier personnel de l'admin.

Remarque L'aide en ligne `man semanage` montre les possibilités qui sont supérieures à l'outil GUI :

Action Depuis un bash **root**, créer le dossier **conf**
```
mkdir ~/conf
```

Exporter la configuration de SELinux vers ce dossier :
```
semanage -o ~/conf/data.txt
```

Q_5.4a Que s'est-il passé ? **Permission denied → Action refusée**

Q_5.4b Chercher la raison du blocage avec la commande ausearch et audit2why ?
```
ausearch -m AVC -ts today -c semanage | audit2why
```

```
[root@SELinux admin]# ausearch --message AVC -ts today -c semanage | audit2why
type=AVC msg=audit(1351082794.330:65): avc:  denied  { write } for  pid=1469 comm="semanag
e" name="conf" dev=dm-1 ino=159157 scontext=unconfined_u:unconfined_r:semanage_t:s0-s0:c0.
c1023 tcontext=unconfined_u:object_r:admin_home_t:s0 tclass=dir

        Was caused by:
                Missing type enforcement (TE) allow rule.

                You can use audit2allow to generate a loadable module to allow this access
```

La commande semanage est exécutée dans le domaine semanage_t.
Le domaine semanage_t n'a pas la permission d'écriture sur un dossier de type admin_home_t.

Q_5.4c Chercher pour le domaine semanage_t un type défini sur lequel semanage_t à la permission d'écriture pour une classe répertoire (dir).
```
sesearch -A -s .......... -c ... -p ..... -d
sesearch -A -s semanage_t -c dir -p write -d
```

```
[root@SELinux admin]# sesearch -A -s semanage_t -c dir -p write -d
Found 7 semantic av rules:
   allow semanage_t semanage_tmp_t : dir { ioctl read write create getattr setattr lock unlink link rename a
dd_name remove_name reparent search rmdir open } ;
   allow semanage_t file_context_t : dir { ioctl read write getattr lock add_name remove_name search open }
;
   allow semanage_t selinux_config_t : dir { ioctl read write create getattr setattr lock unlink link rename
 add_name remove_name reparent search rmdir open } ;
   allow semanage_t selinux_var_lib_t : dir { ioctl read write create getattr setattr lock unlink link renam
e add_name remove_name reparent search rmdir open } ;
   allow semanage_t default_context_t : dir { ioctl read write getattr lock add_name remove_name search open
 } ;
   allow semanage_t tmp_t : dir { ioctl read write getattr lock add_name remove_name search open } ;
   allow semanage_t semanage_store_t : dir { ioctl read write create getattr setattr lock unlink link rename
 add_name remove_name reparent search rmdir open } ;
```

Remarque Parmi les 7 types de destination, nous choisissons semanage_store_t car il est spécifique à semanage et non général comme file_context_t.

Action Modifier le type du dossier conf vers semanage_store_t
```
chcon ~/conf/ -t semanage_store_t
semanage -o ~/conf/data.txt
cat ~/conf/data.txt                    Affiche le contenu de la sauvegarde
```

Remarque La résolution d'un blocage par modification de type (relabeling dossier, fichier, socket, etc.) est une solution très utilisée par les administrateurs.
L'avantage est de résoudre le blocage sans autoriser le domaine semanage à accéder à tout le contenu du dossier personnel mais uniquement au dossier conf ; contrairement à l'ajout de la règle **`allow semanage_t user_home_t :dir write ;`**

But 5.5 **Lancer le serveur apache dans le bon domaine**
Comprendre l'importance des transitions de domaine

Action Dans un bash **root**, lancer le serveur apache :
`httpd`

Q_5.5a Quel est le domaine du serveur apache ?
`ps -eZ | grep httpd` → **unconfined_u:unconfined_r:unconfined_t**

Action Tuer le processus : `killall httpd`
Démarrer le serveur apache en tant que service : `service httpd start`

Q_5.5b Quel est le domaine du serveur apache ?
`ps -eZ | grep httpd` → **system_u:system_r:httpd_t**

Q_5.5c Quelle méthode de lancement du serveur apache est plus sécuritaire ? Pourquoi ?
La deuxième méthode qui le restreint dans le domaine httpd_t.
Ce domaine a été conçu pour permettre au serveur d'apache de fonctionner avec le principe du moindre privilège.

6 Role Based Access Control

But 6.1 **Afficher les rôles SELinux présents dans le système**

Q_6.1a Combien de rôles sont définis ?

Action Depuis le compte admin, exécuter la commande **seinfo**

```
Statistics for policy file: /etc/selinux/targeted/pol
Policy Version & Type: v.26 (binary, mls)

   Classes:            82    Permissions:       241
   Sensitivities:       1    Categories:       1024
   Types:            3602    Attributes:        290
   Users:              10    Roles:              13
   Booleans:          204    Cond. Expr.:       241
   Allow:           90266    Neverallow:          0
   Auditallow:         97    Dontaudit:        6944
   Type_trans:      13374    Type_change:        62
   Type_member:        46    Role allow:         23
```

Action Afficher l'identifiant des rôles présents : **seinfo -r**

Remarque Remarquer la présence de rôles tels que :
logadm_r = administrateur des logs
webadm_r = administrateur web

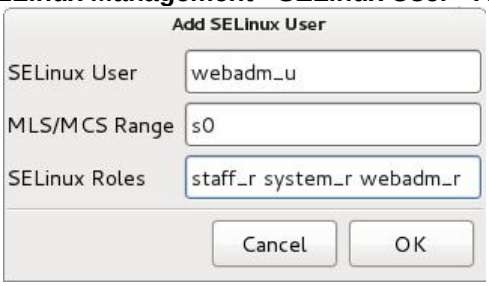
Info Le rôle webadm_r n'autorise pas l'établissement d'une session.

But 6.2 **Créer un utilisateur SELinux autorisé à démarrer le service apache et à administrer le contenu du serveur**

Choix Ce nouvel utilisateur **webadm_u** doit posséder les rôles suivants :

- staff_r autorisant l'établissement d'une session
- webadm_r autorisant l'administration du web serveur
- system_r autorisant le lancement d'un service

Action Dans **SELinux Management - SELinux User - Add**



Remarque Il est préférable de créer un nouvel utilisateur SELinux pour ne pas modifier les utilisateurs SELinux prédéfinis.
Le champ MLS/MCS Range sera expliqué au §7.

Action Dans **User Mapping**, ajouter (add) une entrée qui lie **jean** à **webadm_u**
Si **webadm_u** n'est pas présent, fermer l'outil puis l'ouvrir (bug)
Enregistrer et essayer d'établir une connexion avec le compte **jean**

Action La commande **semanage user -l** indique la relation **SELinux User – SELinux Role(s)**

Q_6.2a L'ouverture de session a-t-elle réussi ? **non**

Remarque **webadm_u** n'a pas encore un fichier de configuration qui indique à quel contexte de sécurité le rattacher après le login. (voir **Q_4.2a**)

Action Dans un bash **root**, copier le fichier de **staff_u** et le renommer en **webadm_u**
```
cd /etc/selinux/targeted/contexts/users/
cp staff_u webadm_u
```

Q_6.2b L'ouverture de session a-t-elle réussi ? **oui**

Q_6.2c Relever le contexte de sécurité du shell
`id –Z → webadm_u:staff_r:staff_t`

But 6.3 **Autoriser les actions de jean dans le fichier sudoers**

Action Dans un bash **root** (session **admin**), modifier le contenu du fichier sudoers :
`visudo`
Typer **i** pour passer en mode insertion

Ajouter les deux lignes suivantes à la fin du fichier :
```
jean   ALL=(ALL)   ROLE=webadm_r TYPE=webadm_t /sbin/service
jean   ALL=(jean)  ROLE=webadm_r TYPE=webadm_t ALL
```

Typer **ESC** puis **:wq** pour sauvegarder les modification et sortir de l'éditeur.

Remarque Un service ne peut être démarré que par **root**.
La première ligne permet à jean de lancer un service en tant que **root** mais avec l'obligation de transiter vers le rôle webadm_r et le domaine webadm_t.
La deuxième ligne permet à jean de pouvoir lancer n'importe quelle commande autorisée pour le rôle webadm_r.

Action Dans une session **jean**, ouvrir un terminal
`service httpd start`

Q_6.3a Le service a-t-il démarré ? **non**

Action `sudo service httpd start`

Q_6.3b Le service a-t-il démarré ? **oui**

Remarque Avec **jean ALL=(ALL) ROLE=webadm_r TYPE=webadm_t /sbin/service**
jean est autorisé à lancer un service en tant que **root** mais avec l'obligation d'utiliser le rôle webadm_r.
Cette obligation l'empêche de lancer ou d'arrêter un autre service que apache.

Action Essayer de démarrer un autre service pour tester
`sudo service mcstrans start` mcstrans = service de traduction des catégories SELinux

Action `ls /var/www` afficher le dossier des pages web

Q_6.3c Le contenu du dossier s'est-il affiché ? **non**

Action `sudo -u jean ls /var/www`

Remarque La commande sudo demande par défaut à exécuter une commande en tant que **root**.
L'option -u jean permet à jean d'exécuter une commande en transitant vers le rôle **webadm_r** selon
jean ALL=(jean) ROLE=webadm_r TYPE=webadm_t ALL

Q_6.3d Le contenu du dossier s'est-il affiché ? **oui**

Action Vérifier la transition vers le rôle **webadm_r** :
```
sudo -u jean sh    ouvrir un shell
id -Z
```

Q_6.3e Relever le contexte de sécurité du shell : **webadm_u:webadm_r:webadm_t**

Action Quitter le shell actif : `exit`

7 Multi-Category Security

But 7.1 **Le dossier A est réservée à jean ; paul et jean ont accès au dossier B → créer 2 catégories**

Action Dans une session **root**,
`vi /etc/selinux/targeted/setrans.conf` éditer le fichier
Typer **i** pour passer en mode insertion
Ajouter à la fin du fichier :

s0:c0=A	Identifiant A = catégorie c0
s0:c1=B	Identifiant B = catégorie c1
s0:c0.c1=AB	Identifiant AB = union des catégories c0 et c1

Typer **ESC** puis **:wq**

But 7.2 **Assigner une(des) catégorie(s) aux utilisateurs**

Action Dans **SELinux Management - User Mapping**, double cliquer sur **jean**, modifier la ligne MLS/MCS Range avec **s0-s0:c0.c1**

Remarque Le 1er élément **s0** définit le niveau par défaut de l'utilisateur (création d'un fichier)
Le 2ème élément **s0:c0.c1** donne accès aux catégories c0 et c1 (intervalle c0 c1)

Q_7.2a Que se passe-t-il ?

```
libsemanage.validate_handler: MLS range s0-s0:c0.c1 for
Unix user jean exceeds allowed range s0 for SELinux user
webadm_u (No such file or directory).
```

L'intervalle alloué à jean dépasse celui de webadm_u

Action Dans **SELinux User,** mettre **s0-s0:c0.c1023** à **webadm_u** pour autoriser toutes les catégories
Dans **User Mapping**, mettre **s0-s0:c0.c1** à jean
Dans **SELinux User,** mettre **s0-s0:c0.c1023** à **user_u**
Dans **User Mapping**, mettre **s0-s0:c1** à paul

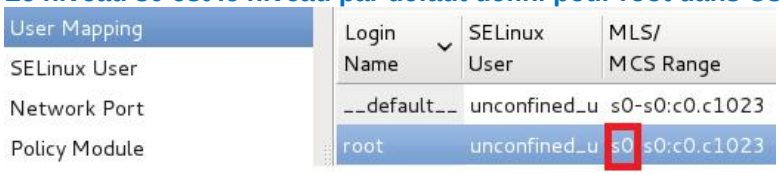
But 7.3 **Changer un dossier de catégorie**

Action Dans une session **root**

```
cd /tmp/
mkdir A
mkdir B
```

Q_7.3a Quel est le contexte de sécurité de ces 2 dossiers ? Pourquoi ?
`ls -Z → unconfined_u:object_r:user_tmp_t:s0`

Le niveau s0 est le niveau par défaut défini pour root dans User Mapping :



Action Configurer les catégories :

```
chcat +s0:c0 A      change SELinux categoty
chcat +s0:c1 B
```

Q_7.3b Quel est le contexte de sécurité de ces 2 dossiers ? Pourquoi ?
`ls -Z → unconfined_u:object_r:user_tmp_t:s0:c0 A`
`ls -Z → unconfined_u:object_r:user_tmp_t:s0:c1 B`

Action

Ajouter les droits RWX (X est nécessaire pour permettre d'entrer dans le dossier)

```
chmod a=rwx A
chmod a=rwx B
```

Fermer les sessions jean et paul en cours afin d'obtenir la nouvelle configuration lors de la prochaine authentification

Tester avec le compte **jean**

```
cd /tmp
cd A
cd ..
cd B
```

Tester avec le compte **paul**

```
cd /tmp
cd A
cd ..
cd B
```

Annexe 1 : Utilisateurs SELinux disponibles dans Fedora 16

The guest_u SELinux user:

This profile is used for users that need to be tightly controlled. The guest_u SELinux user can only log in using OpenSSH. Guest users have no access to network resources, setuid, setgid programs.

The xguest_u SELinux user:

This profile is identical to that of guest_u. The exception is that Xguest users can only log in to Xwindows and cannot log in using OpenSSH. Another exception of Xguest users is that this partical user can access HTTP port using a SELinux restricted instance of Mozilla Firefox.

The user_u SELinux user:

The user_u SELinux user resembles a ordinary unprivileged SELinux confined user. This user can log in using Xwindows and OpenSSH, has access to network resources, but cannot use setuid and setgid programs.

The staff_u SELinux user:

This SELinux user is identical to user_u except that staff_u can access setuid and getgid programs. The staff_u SELinux user can also stat all process on the system amongst other minor extra privileges compared to user_u.

The sysadm_u SELinux user:

This user is designed for SELinux restricted root login, which is not recommended. This SELinux user is used in a Multi-Level Security Environment where there is no unconfined_u.

Réf : **http://selinux-mac.blogspot.ch/2009/06/selinux-lockdown-part-one-confined.html**