SUDO

I.Introduction

Il y a parfois nécessité, dans le monde unix, d'avoir des droits spéciaux à un utlisateur ou un groupe d'utilisateurs pour exécuter un programme donné sans pour autant accorder des droits excessifs. Ainsi, il existe quelques façons pour donner des droits à certains utilisateurs.Parmi ces moyens, on trouve:

- 'su <nom_commande>' qui nécessite la connaissance du mot de passe root (ou alors disposer d'un fichier .pw avec le password) donc solution peut sécurisée
- l'activation du 'suid-bit' qui permet à des utilisateurs, ayant les bons droits d'accès, d'utiliser la commande voulue, seulement ce système se révèle fastidieux pour l'attribution et la gestion d'accès à plusieurs fichiers.

Reste la solution 'sudo' (superuser do) qui autorise l'exécution d'une commande avec le droits d'un autre utilisateur. C'est cette solution que nous allons décrire dans la suite.

II.Principes

Sudo est disponible sur beaucoup de distributions de type unix telles que : FreeBSD, MacOS, Solaris, AIX, HP-UX, IRIX ...
Lorsqu'il n'est pas fournit de base on peut quand même l'installer en le téléchargeant via ftp://ftp.sudo.ws/pub/sudo/ ou en faisant appel à une commande d'update plus spécifique à l'OS (telle que 'apt-get install sudo' pour debian, ou la commande 'rpm -Uvh [fichier_sudo].rpm' pour une RedHat).
Une fois assuré d'avoir "sudo" à disposition, il est maintenant possible de :

- restreindre les commandes qu'un utilisateur peut utiliser dans un environnement multi-utilisateurs.
- disposer de logs indiquant qui a lancé quoi comme commande avec 'sudo' (utilisé avec syslogd sudo peut logger les commandes vers un hôte spécifique aussi bien qu'en local). On peut ainsi répertorier les tentatives réussies et échouées par syslog (par défaut), dans un fichier, ou les deux. Ceci est configurable dans le fichier "sudoers".
- avoir un système de timeout qui, à chaque fois qu'un utilisateur se sert de sudo et entre son password, il dispose d'un "ticket" de 5 minutes (modifiable au moment de la compilation). Chaque nouvelle commande renouvelle ce ticket pour 5 autres minutes. Concernant la fin de session l'on peut également disposer d'un fichier .logout dans lequel sera défini certaines tâches à exécuter à l'expiration du ticket.
- utiliser le même fichier "sudoers" d'une machine (dans lequel sont définies les autorisations concernant les utlisateurs) sur plusieurs machines.

Comme nous allons le voir juste après, "sudo" est lié à la manipulation de droits et est aidé en celà par la présence :

➔ d'un répertoire d'estampilles (timestamps) '/var/run/sudo' pour la gestion des tickets. Celui-ci appartient à root et cette appartenance est vérifiée par 'sudo' pour des raisons de sécurité. De même, tous les timestamps avec une date supérieure à "current_time + 2 * `TIMEOUT` " sont ignorés et loggés.
➔ d'un répertoire /etc/sudoers qui contient la liste des utilisateurs et de ce qu'ils ont le droit de faire.

III.Utilisation

Cette partie a pour but de donner le meilleur aperçu possible (les détails de configuration et d'utilisation ne manquant pas) concernant l'utilisation de "sudo".
Tout d'abord, cette commande dispose de beaucoup d'options et un petit tour dans le "man" donne la syntaxe suivante :
sudo -K | **-L** | **-V** | **-h** | **-k** | **-l** | **-v**

sudo [**-HPSb**] [**-a** *auth_type*] [**-c** *class*|-] [**-p** *prompt*] [**-u** *username*|*#uid*]
{**-e** file [...] | **-i** | **-s** | *command*}

Je ne détaillerais pas toutes les options (l'utilisation du "man" est nécessaire), mais les plus utiles permettent de lister les commandes autorisées (exemple : sudo -l), de lancer une commande avec les droits d'un autre utilisateur (exemple : sudo -u toto ls ~toto) ou encore de lancer la commande en tâche de fond (exemple : sudo -b open ~toto/sudo.sxw). Il est également possible d'éditer des fichiers avec
sudoedit [**-S**] [**-a** *auth_type*] [**-p** *prompt*] [**-u** *username*|*#uid*] file [...]

équivalent de sudo -e, qui crée une copie temporaire d'un fichier avec les droits du propriétaire et le rend accessible à celui qui a invoqué la commande.

Outre la commande 'sudo' en elle-même, la commande 'visudo' (en tant que root) permet d'éditer le fichier 'sudoers'. Cette commande sera détaillée dans la suite.

A noter que dans des versions récentes de 'sudo', on peut ajouter à 'sudoers' ces lignes :

```
Defaults        logfile=/var/adm/sudo.log
Defaults:paul     logfile=/usr/local/log/sudo.paul
Defaults:jean     logfile=/usr/local/log/sudo.jean
```

et ceci aura pour effet de logger les évènements de paul et de jean dans des fichiers différents. Il faudra également pour celà, s'assurer de ne pas logger par syslog.

1.visudo

Cette commande permet d'éditer le fichier '/etc/sudoers' d'une manière plus sure qu'une édition 'à la main' (ce qui est toujours possible mais mal, sous root, en changeant les droits du fichier). En effet, visudo permet de bloquer une multiple édition du fichier, et effectue une vérification basique de la syntaxe.
Parmi les options de 'visudo', on retrouve :

- l'option 'visudo -c'
  qui active le mode "check-only". Le fichier 'sudoers' courant est ainsi vérifié au niveau syntaxique, et le détail de la vérification est affiché.

- l'option ' visudo -f '
  qui permet de modifier l'emplacement du fichier 'sudoers' ou de l'éditer à un autre emplacement que celui par défaut.

- l'option ' visudo -q '
  active le mode silence, ainsi le détail des erreurs de syntaxe n'est pas affiché. Généralement utilisé avec l'option ' -c ', elle permet d'avoir un rapport d'erreur plus court.

- l'option ' visudo -s '
  qui active une vérification plus fine de la syntaxe. Ainsi, il y aura une erreur si un alias est utilisé avant même d'être défini.

- l'option ' visudo -V '
  qui affiche la version actuelle de ' visudo '.

2.sudoers

On peut trouver par défaut ce fichier dans "/etc/sudoers " ou "/usr/local/etc/sudoers".
Voici ce que donne un fichier sudoers type une fois édité :

```
# sudoers file.
#
# This file MUST be edited with the 'visudo' command as root.
#
# See the man page for details on how to write a sudoers file.
#

# Host alias specification
Host_Alias                  MACHINE_TEST=localhost
Host_Alias                  RESEAU_TEST=192.168.0.4, 192.168.0.5

# User alias specification
User_Alias                  ADMIN_LOCAL=user1
User_Alias                  ITINERANT=toto, titi

# Cmnd alias specification
Cmnd_Alias    SHUTDOWN=/sbin/shutdown
Cmnd_Alias    SYSLOGD=/etc/syslogd

# User privilege specification
root                        ALL=(ALL) ALL
ADMIN_LOCAL                 MACHINE_TEST=NOPASSWD:SYSLOGD
ADMIN_LOCAL                 RESEAU_TEST = (ITINERANT) SHUTDOWN
ITINERANT   MACHINE_TEST=SHUTDOWN
ITINERANT   ALL = !/usr/bin/su, !/usr/bin/su root
```

On peut remarquer divers champs, avec à chaque fois une syntaxe spéciale.
En premier lieu il y a les "définitions" à l'aide d'alias.On en distingue 4 types : `User_Alias`, `Runas_Alias`, `Host_Alias` et `Cmnd_Alias`.
Celles-ci permettent d'associer un nom générique à une entité précise (utilisateur, machine, commande...).

`Host_Alias` permet de définir des alias pour les machines, ainsi

```
Host_Alias                  MACHINE_TEST=localhost
Host_Alias                  RESEAU_TEST=192.168.0.4, 192.168.0.5
```

Associe à l'alias "machine_test" à la machine locale et "reseau_test" à deux autres machines distantes.

`User_Alias` permet de définir des groupes d'utilisateurs, ainsi

```
User_Alias                  ADMIN_LOCAL=user1
User_Alias                  ITINERANT=toto, titi
```

Définit deux groupes d'utilisateurs "admin_local" (contenant "user1") et un groupe "itinérant" (contenant "toto" et "titi").

`Cmnd_Alias` permet de définir des groupes de commandes, ainsi

```
Cmnd_Alias    SHUTDOWN=/sbin/shutdown
Cmnd_Alias    SYSLOGD=/etc/syslogd
```

Permet d'associer à "shutdown" la commande pour arrêter la machine, et à "syslogd" la commande d'accès au démon syslog.

`Runas_Alias` permet de créer un groupe contenant les utilisateurs pour lesquels on souhaite se faire passer.
On peut également utiliser `Runas_Alias` pour déclarer les utilisateurs. Son avantage par rapport à `User_Alias` est que l'on peut inclure un alias existant dans une déclaration.
Sa syntaxe est : Runas_Alias GROUPE = utilisateur1, utilisateur2

Il ne reste plus qu'à préciser les droits sur les commandes (par défaut celles-ci sont lancées en tant que root)

```
root                        ALL=(ALL) ALL
ADMIN_LOCAL                 MACHINE_TEST=NOPASSWD:SYSLOGD
ADMIN_LOCAL                 RESEAU_TEST = (ITINERANT) SHUTDOWN
ITINERANT   MACHINE_TEST=SHUTDOWN
ITINERANT   ALL = !/bin/su, !/bin/su root
```

Ainsi:
- root, sur toutes les machines, a tous les droits
- le groupe "admin_local", sur "machine_test", peut se servir de "syslogd" sans avoir à rentrer de password
- le groupe "admin_local", sur le "reseau_test", en tant qu' "itinerant", peut se servir de "shutdown"
- le groupe "itinerant", sur "machine-test", peut se servir de "shutdown"
- le groupe "itinerant", sur toutes les machines, ne peut pas lancer la commande "su", ni "su root".

Un utilisateur du groupe 'itinerant' pourra donc faire un "sudo shutdown" sans problème (en entrant son mot de passe), mais un "sudo su" ne sera pas autorisé.

3.outils annexes

Des outils ont annexes ont été créé comme 'sudolog-usage', 'sudoers-lint', 'sudoscript' ou encore 'sudosh'.
Sinon, d'autres sudo-like existent : super, runas, priv, calife, osh, ssu, su1, op, suSub, Power Broker, ksu (dans un environnement Kerberos).

4.liens de documentation

une documentation fournie (en anglais) http://www.sudo.ws/sudo/
deux documentations en français cette fois :
http://guide.andesi.org/html/ksudo.html
http://lea-linux.org/admin/admin_env/sudo.html