

Gestion des utilisateurs

Ivan Kurzweg

5 mai 2007

Gestion des utilisateurs

by Ivan Kurzweg

Copyright © 2002 - 2006 Ivan Kurzweg , Pascal Picard

Permission to use, copy, modify, and distribute this documentation for any purpose with or without fee is here by granted, provided that the above copyright notice and this permission notice appear in all copies.

Ce cours est largement inspiré du cours de Pascal Picard disponible sur <https://seth.homeunix.net/corto>, et des documentations des systèmes *BSD et Linux.

Table des matières

1	Commandes de gestion des utilisateurs	2
1.1	Introduction	2
1.1.1	Le compte super-utilisateur	3
1.1.2	Les comptes systèmes	4
1.1.3	Les comptes utilisateurs	4
1.2	Ajouter/Enlever un utilisateur	4
1.3	Analyse du fichier <code>/etc/passwd</code>	5
1.4	Analyse du fichier <code>/etc/group</code>	6
1.5	Sécurisation des comptes	7
1.5.1	<i>GNU/Linux</i>	7
1.5.2	<i>xBSD</i>	8
1.6	Commandes de gestion des utilisateurs	9
1.6.1	Gérer les utilisateurs avec <code>user{add,mod,del}</code> sous <i>GNU/Linux</i>	9
1.6.2	Gérer les groupes avec <code>group{add,mod,del}</code> sous <i>GNU/Linux</i>	10
1.6.3	Cas particulier de <i>FreeBSD</i>	10
1.6.4	Vérification	13
1.6.5	Edition en ligne	13
1.7	Programmes de gestion des utilisateurs sous <i>FreeBSD</i>	13
1.7.1	<code>adduser</code>	13
1.7.2	<code>rmuser</code>	14
1.7.3	<code>chpass</code>	15
1.8	Autres commandes	15
2	Exercices	16
2.1	Exercices d'applications	16
2.1.1	Exercice 1	16
2.1.2	Exercice 2	16
2.1.3	Exercice 3	16
2.1.4	Exercice 4	16
3	Exercices d'évaluation	16
3.1	Gestion des utilisateurs	16
3.2	Droits	16
3.2.1	Commandes	16
3.2.2	Exercice 1	17
3.2.3	Exercices 2	17
3.3	Scripts	17
3.3.1	Listage de répertoire	17
3.3.2	Affichage des droits	17

Liste des tableaux

1	Autres commandes	15
---	------------------	----

1 Commandes de gestion des utilisateurs

Nous abordons dans cette partie la gestion des comptes d'utilisateurs (création, modification et suppression) qui est une des premières tâches de l'administrateur système.

1.1 Introduction

Tout accès au système est effectué par l'intermédiaire de comptes, et tous les processus sont exécutés par des utilisateurs, la gestion des comptes et des utilisateurs est donc capitale. Chaque compte est associé avec un certain nombre d'informations utilisé pour identifier le compte.

- **User name** - *nom d'utilisateur* : c'est le nom tapé à l'invite `login`. Les noms d'utilisateur *doivent être uniques* sur le système; vous ne pouvez pas avoir deux utilisateurs avec le même nom d'utilisateur. Il y a un certain nombre de règles pour la création de noms d'utilisateur valides, documentées dans `passwd(5)`;
- **Password** - *mot de passe* : Chaque compte est associé à un mot de passe. Le mot de passe peut être vide, dans ce cas aucun mot de passe ne sera requis pour accéder au système. Ceci est une très mauvaise idée; chaque compte devrait avoir un mot de passe.
- **User ID (UID)** - *identifiant utilisateur* : L'UID est un nombre compris entre 0 et 65535, utilisé pour identifier de façon unique un utilisateur sur le système. Au niveau interne, FreeBSD utilise l'UID pour identifier les utilisateurs--toute commande qui vous permet de spécifier un utilisateur convertira le nom d'utilisateur en son UID avant de le traiter.
- **Group ID (GID)** - *identifiant de groupe* : Le GID est un nombre compris entre 0 et 65535, utilisé pour identifier de façon unique le groupe principal auquel appartient l'utilisateur. Les groupes sont un mécanisme pour contrôler l'accès aux ressources qui est basé sur le GID de l'utilisateur plutôt que sur son UID. Un utilisateur peut également appartenir à plus d'un groupe.
- **Login class** - *classe de session* : Les classes de session sont une extension du mécanisme de groupe qui apporte une flexibilité supplémentaire quand on adapte le système aux différents utilisateurs.
- **Password change time** - *durée de vie d'un mot de passe* : Par défaut FreeBSD n'oblige pas les utilisateurs à changer leur mot de passe régulièrement. Vous pouvez forcer cela en fonction de l'utilisateur, en obligeant certains ou tous les utilisateurs à changer leur mot de passe après qu'une certaine période de temps se soit écoulée.
- **Account expiry time** - *date d'expiration d'un compte* : Par défaut FreeBSD ne désactive pas de comptes après une certaine période. Après la durée d'expiration écoulée le compte ne pourra plus être utilisé pour ouvrir de session sur le système, bien que les répertoires et les fichiers attachés au compte seront conservés.
- **User's full name** - *nom complet d'utilisateur* : Le nom d'utilisateur identifie uniquement le compte sur FreeBSD, mais ne reflète pas nécessairement le nom réel de l'utilisateur. Cette information peut être associée avec le compte.
- **Home directory** - *répertoire utilisateur* : Le répertoire utilisateur est le chemin complet vers un répertoire sur le système dans lequel se retrouve l'utilisateur quand il ouvre une session sur le système. Une convention commune est de mettre tous les répertoires d'utilisateurs sous `/home/username` ou `/usr/home/username`. L'utilisateur pourra stocker ses fichiers personnel dans son répertoire utilisateur et dans tout sous-répertoire qu'il pourra y créer.
- **User shell** - *interpréteur de commandes de l'utilisateur* : L'interpréteur de commandes fournit aux utilisateurs l'environnement par défaut pour communiquer avec le système. Il existe plusieurs différents types d'interpréteurs de commandes, et les utilisateurs expérimentés auront leur préférence, qui peut se refléter dans le paramétrage de leur compte.

Il y a trois principales sortes de comptes: le *super-utilisateur*, les *utilisateurs système*, et les *comptes utilisateur*. Le compte super-utilisateur, normalement appelé **root**, est utilisé pour gérer le système sans aucune limitation de privilèges. Les utilisateurs système exécutent des services. Et enfin, les comptes utilisateur sont utilisés par de véritables utilisateurs, qui ouvrent des sessions, lisent leur courrier électronique, et ainsi de suite.

Les informations concernant la définition d'un compte utilisateur sont principalement stockées dans les fichiers `/etc/passwd` et `/etc/group`, selon qu'il s'agisse d'un système GNU/Linux ou d'un système xBSD dans les fichiers `/etc/shadow` et `/etc/master.passwd`. Avant d'en présenter la structure syntaxique, intéressons nous à déterminer ce qu'implique la création et la destruction d'un compte. Enfin, pour finir nous présenterons les principales commandes de gestion des comptes.

1.1.1 Le compte super-utilisateur

Le compte super-utilisateur, habituellement appelé **root**, est préconfiguré pour simplifier l'administration système, et ne devrait pas être utilisé pour des tâches quotidiennes comme l'envoi et la réception de courrier électronique, l'exploration du système, ou la programmation.

Cela parce que le super-utilisateur, à la différence des comptes utilisateurs ordinaires, peut agir sans aucune limite, et une mauvaise utilisation du compte super-utilisateur peut être à l'origine de résultats catastrophiques. On ne peut (normalement!) pas endommager par erreur le système avec un compte

utilisateur, il est donc généralement préférable d'utiliser des comptes utilisateur ordinaires chaque fois que c'est possible, à moins d'avoir particulièrement besoin de droits supplémentaires.

Vous devriez toujours *vérifier et revérifier les commandes que vous tapez en tant que super-utilisateur*, parce qu'un espace en trop ou un caractère manquant peuvent signifier la perte définitive de données.

1.1.2 Les comptes systèmes

Les utilisateurs système sont ceux utilisés pour exécuter des services comme le DNS, le courrier électronique, les serveurs web, et ainsi de suite. La raison de cela est la sécurité; si tous les services s'exécutaient avec les droits du super-utilisateur, ils pourraient agir sans aucune restriction.

Des exemples d'utilisateurs système sont `daemon`, `operator`, `bind` (pour le serveur de noms de domaine), `news`, et `www`.

`nobody` est l'utilisateur sans privilèges générique du système. Cependant, il est important de garder à l'esprit que plus grand est le nombre de services utilisant `nobody`, plus grand sera le nombre de fichiers et de processus associés à cet utilisateur, et par conséquent plus grand sera le nombre de privilèges de cet utilisateur.

1.1.3 Les comptes utilisateurs

Les comptes utilisateur sont le principal moyen pour les véritables utilisateurs d'accéder au système, ces comptes isolent l'utilisateur du reste de l'environnement, empêchant les utilisateurs d'endommager le système et ou les comptes d'autres utilisateurs, tout en leur permettant de personnaliser leur environnement sans incidence pour les autres utilisateurs.

Chaque personne accédant à votre système ne devrait posséder que son propre et unique compte. Cela vous permet de savoir qui fait quoi, empêche un utilisateur de désorganiser l'environnement d'un autre ou de lire du courrier électronique qui ne lui est pas destiné, et ainsi de suite.

Chaque utilisateur peut configurer son propre environnement en fonction de ses besoins, pour utiliser d'autres interpréteurs de commandes, éditeurs, raccourcis de clavier, et langues.

1.2 Ajouter/Enlever un utilisateur

Créer un nouveau compte utilisateur requiert plusieurs étapes :

- Attribuer un nom de login, un `uid`, un groupe principal et un ou des groupes secondaires.
- Créer des enregistrements sur cet utilisateur dans les fichiers `/etc/passwd`, `/etc/group` et `/etc/master.passwd` ou `/etc/shadow`.
- Attribuer un (bon) mot de passe.
- Créer le répertoire de domiciliation de l'utilisateur (son `home`).
- Paramétrer le compte de l'utilisateur (durée de validité du mot de passe, date d'expiration du compte, limitation des ressources, privilèges ...).
- Copier les fichiers d'initialisation dans le répertoire de domiciliation de l'utilisateur.
- Rendre l'utilisateur propriétaire de son espace.
- Définir les paramètres supplémentaires, tels les *quotas disk*, l'impression, la messagerie...

Enlever un compte utilisateur nécessite les étapes suivantes :

- Désactiver le compte de l'utilisateur.
 - sur *GNU/Linux* en faisant `passwd -l <loginname>`
 - sur les *flavors xBSD* en préfixant le mot de passe du fichier `/etc/master.passwd` par le caractère `*` (cette technique marche aussi pour *GNU/Linux* dans le fichier `/etc/shadow`).
- Supprimer le compte de l'utilisateur.
 - Supprimer tous les processus de l'utilisateur.
 - Enlever les enregistrements concernant cet utilisateur des fichiers `/etc/passwd`, `/etc/group`, `/etc/shadow` ou `/etc/master.passwd`.
 - Retirer l'utilisateur des groupes secondaires.
 - Supprimer le fichier de courrier électronique de l'utilisateur.
 - Retirer ou rediriger les alias de courrier électronique de l'utilisateur.

- Supprimer les tâches **cron** et/ou **at**.
- Eventuellement faire une sauvegarde de l'espace de l'utilisateur puis le supprimer.
- Supprimer les fichiers et/ou répertoires temporaires de l'utilisateur.
- Mettre à zéro les quotas de cet utilisateur.

Un système Unix™ propose généralement un ensemble d'utilitaires pour faciliter la gestion des comptes. De plus, rien n'empêche de créer ces propres scripts pour automatiser au mieux cette tâche.

1.3 Analyse du fichier `/etc/passwd`

Ce fichier au format texte, référence l'ensemble des utilisateurs du système. Une ligne concerne un utilisateur et est structurée en sept champs : `loginname:password:uid:gid:gecos:homedir:shell`

- *loginname* : le nom de login de l'utilisateur.
- *password* : le mot de passe hashé, initialisé au caractère `*`.
- *uid* : identifiant utilisateur. Entier non signé codé sur quatre octets [0, 4294967296]. La valeur 0 donne les privilèges d'administration système, les valeurs inférieurs à 10 sont, par convention, réservées à des comptes systèmes. Généralement, un utilisateur standard possède un uid supérieur à 100. Pour des raisons d'interopérabilité, il convient de se limiter aux 65536 premières valeurs.
- *gid* : identifiant du groupe principal de l'utilisateur. Entier non signé codé sur quatre octets mais dont l'usage est limité, comme précédemment. Les valeurs inférieures à 10 sont réservées aux groupes systèmes.
- *gecos* : zone de commentaire. On y met généralement le nom réel de l'utilisateur. Ce champ est utilisé par les commandes **chfn**, **mail** et **finger**.
- *homedir* : répertoire de domiciliation de l'utilisateur.
- *shell* : nom de l'interpréteur de commandes. Modifiable par la commande **chfn**.

Voici un exemple de fichier `/etc/passwd` d'un système FreeBSD obtenu à partir du fichier `/etc/master.passwd`:

```
# $FreeBSD: src/etc/master.passwd,v 1.40 2005/06/06 20:19:56 brooks Exp $
#
root:*:0:0:Charlie &:/root:/bin/csh
toor:*:0:0:Bourne-again Superuser:/root:
daemon:*:1:1:Owner of many system processes:/root:/usr/sbin/nologin
operator:*:2:5:System &:/usr/sbin/nologin
bin:*:3:7:Binaries Commands and Source:/usr/sbin/nologin
tty:*:4:65533:Tty Sandbox:/usr/sbin/nologin
kmem:*:5:65533:KMem Sandbox:/usr/sbin/nologin
games:*:7:13:Games pseudo-user:/usr/games:/usr/sbin/nologin
news:*:8:8:News Subsystem:/usr/sbin/nologin
man:*:9:9:Mister Man Pages:/usr/share/man:/usr/sbin/nologin
sshd:*:22:22:Secure Shell Daemon:/var/empty:/usr/sbin/nologin
smmsp:*:25:25:Sendmail Submission User:/var/spool/clientmqueue:/usr/sbin/ ←
nologin
mailnull:*:26:26:Sendmail Default User:/var/spool/mqueue:/usr/sbin/ ←
nologin
bind:*:53:53:Bind Sandbox:/usr/sbin/nologin
proxy:*:62:62:Packet Filter pseudo-user:/nonexistent:/usr/sbin/nologin
_pflugd:*:64:64:pflugd privsep user:/var/empty:/usr/sbin/nologin
_dhcp:*:65:65:dhcp programs:/var/empty:/usr/sbin/nologin
uucp:*:66:66:UUCP pseudo-user:/var/spool/uucppublic:/usr/local/libexec/ ←
uucp/uucico
pop:*:68:6:Post Office Owner:/nonexistent:/usr/sbin/nologin
www:*:80:80:World Wide Web Owner:/nonexistent:/usr/sbin/nologin
nobody:*:65534:65534:Unprivileged user:/nonexistent:/usr/sbin/nologin

ikare①[1]:*②[2]:2323③[3]:2323④[4]:Ivan K;⑤[5]:/home/ikare⑥[6]:/bin/tcsh ←
⑦[7]
```

```
squid:*:100:100:squid caching-proxy pseudo user:/usr/local/squid:/usr/ ←
    sbin/nologin
```

- 1 `login.`
- 2 `password.`
- 3 `uid.`
- 4 `gid.`
- 5 `gecos.`
- 6 `homedir.`
- 7 `shell.`

Le fichier `/etc/passwd` stockent les mots de passe sous forme hashée. Une fonction de hashage a la propriété d'être difficilement inversible, i.e. s'il est relativement aisé, connaissant l'algorithme de hashage, de calculer à partir d'un mot de passe en clair son hashage correspondant, la réciproque est un problème *très* difficile.

A la création d'un nouveau compte une étoile est placée dans le deuxième champ (`passwd`) du fichier `/etc/passwd`, elle permet d'empêcher l'accès au compte tant qu'un mot de passe n'est pas défini. **Ce champ ne doit jamais être vide !**

Le standard d'encryptage, par défaut, des mots de passe est DES, lequel n'encode que les 8 premiers caractères d'un mot de passe. Il est désormais considéré comme insuffisant au regard de la puissance de calcul d'un simple PC de bureau et c'est pourquoi les distributions *GNU/Linux* propose une alternative : le hashage *MD5*. De fait, *MD5* n'est pas meilleur que DES (du point de vue cryptographique), mais *MD5* n'impose pas de limite sur la longueur du mot de passe. Un mot de passe long est plus dur à casser donc plus sûr. *FreeBSD* de son côté utilise le hashage *MD5* par défaut, on peut éventuellement le remplacer par un autre algorithme tel que *blowfish* (c.f. le fichier `/etc/login.conf`, ligne `:passwd_format=...`).

NOTE



Avec *Debian GNU/Linux* le système propose d'activer à l'installation l'encodage des mots de passe par *MD5*, valider toujours ce choix.

1.4 Analyse du fichier `/etc/group`

Ce fichier référence l'ensemble des groupes du système. Une ligne concerne un groupe et elle est structurée en quatre champs : `groupname:password:gid:member, ...`

- `groupname` : le nom du groupe.
- `password` : le mot de passe hashé, initialisé au caractère `*`. En général, on n'utilise pas de mot de passe de groupe.
- `gid` : identifiant du groupe. Entier codé sur deux octets. Les valeurs inférieures à 10 sont réservées aux groupes systèmes.
- `member, ...` : liste des membres du groupe.

```
# $FreeBSD: src/etc/group,v 1.32.2.1 2006/03/06 22:23:10 rwatson Exp $
#
wheel:*:0:root
daemon:*:1:
kmem:*:2:
sys:*:3:
tty:*:4:
operator:*:5:root
mail:*:6:
```

```
bin:*:7:
news:*:8:
man:*:9:
games:*:13:
staff:*:20:
sshd:*:22:
smmsp:*:25:
mailnull:*:26:
guest:*:31:
bind:*:53:
proxy:*:62:
authpf:*:63:
_pflogd:*:64:
_dhcp:*:65:
uucp:*:66:
dialer:*:68:
network:*:69:
audit:*:77:
www:*:80:
nogroup:*:65533:
nobody:*:65534:
fremens:*:2323:ikare
squid:*:100:
```

1.5 Sécurisation des comptes

Historiquement, les mots de passe étaient stockés dans le fichier `/etc/passwd` (universellement accessible en lecture)¹, mais à mesure que les ordinateurs ont gagné en performance, il est devenu extrêmement dangereux de les laisser dans ce fichier.

1.5.1 GNU/Linux

GNU/Linux propose désormais de stocker les mots de passe dans un fichier à part : `/etc/shadow` (uniquement accessible au *super*-utilisateur. Ce mécanisme communément appelé “*shadow password*” est disponible sur toutes les distributions. En particulier, sur *Debian* cette option est explicitement proposée au moment de l’installation, il convient de l’adopter. La seule contre-indication à l’utilisation de cette stratégie concerne l’éventuel usage de votre machine comme serveur NIS/NFS.

Quand cette stratégie est activée, le champ mot de passe (deuxième champ) du fichier `/etc/passwd` est marqué par un `x`. Le fichier `/etc/shadow` contient donc l’encodage (ou hashage) des mots de passe et fournit des services supplémentaires non disponibles dans `/etc/passwd`. Attention, les deux fichiers sont nécessaires, l’un ne remplaçant pas l’autre !

Le fichier `/etc/shadow` est un fichier texte qui contient une ligne par utilisateur. Il est structuré en neuf champs, comme suit : `loginname:password:lcdate:mindays:maxdays:ndays1:ndays2-expiration:reserved` La sémantique associée à ces champs est la suivante :

- *loginname* : le nom de login de l’utilisateur, identique à celui du fichier `/etc/passwd` il permet ainsi d’établir le lien entre les deux fichiers.
- *password* : le hashage du mot de passe. S’il s’agit d’un hashage MD5, il commence par la séquence `1`.
- *lcdate* : date du dernier changement de mot de passe, exprimée relativement au temps Unix™, exprimé en jours, i.e. nombre de jours écoulés depuis le 1^{er} Janvier 1970. Généralement rempli par la commande `/usr/bin/passwd`
- *mindays* : Intervalle minimum de jours entre deux changements de mot de passe. Généralement positionné à 0, pour signifier qu’il n’y a pas d’intervalle.

¹ Le “jeu” consistait à récupérer une copie du fichier `/etc/passwd` puis à faire tourner un programme de “crack”. Ce dernier passe en revue un dictionnaire de mots de passe qu’il a hashé un à un et compare au hashage présent dans le fichier des mots de passe. En cas d’égalité on a retrouvé le mot de passe (la fonction de hashage est injective). Ce genre de programme parcourt donc l’espace des mots de passe possible ; il lui faut donc de la puissance de calcul et du temps. Or les PC de bureau d’aujourd’hui ont une puissance de calcul non négligeable !

- *maxdays* : Intervalle maximum de jours entre deux changements de mot de passe. Cela permet donc à l'administrateur de renforcer la stratégie de limitation dans le temps des mots de passe. Le délai maximum est obtenu comme somme de ce champ et du 7^e.
- *ndays1* : Nombre de jours, pour prévenir l'utilisateur, avant l'expiration du mot de passe.
- *ndays2* : Nombre de jours s'écoulant entre l'expiration du mot de passe et la désactivation du compte.
- *expiration* : Date d'expiration effective du compte, en nombre de jour depuis le 1^{er} Janvier 1970.
- *reserved* : Champ réservé pour un usage futur, vide pour le moment.

Un exemple typique de ligne extraite du fichier `/etc/shadow` :

```
ikare:$1$IzYw9H3u$Q7T2rxyPVKAlabue$.1n0:12144❶[1]:0:180❷[2]:7❸[3]::12294❹[4]:
```

- ❶ Le dernier changement de mot de passe a eu lieu le 2 avril 2003 (12144 jours depuis 01.01.1970) .
- ❷ Le mot de passe a une validité de 180 jours.
- ❸ Il y a un délai de 7 jours pour changer le mot de passe.
- ❹ Le compte expire le 30 août 2003 (12294 jours depuis le 01.01.1970).

Il existe un fichier `/etc/gshadow` dont le rôle est similaire à celui du fichier `/etc/shadow`, en pratique peu ou pas utilisé. On active rarement (voir jamais, par convention) les mots de passe de groupe.

1.5.2 xBSD

Les *flavors* xBSD stockent les comptes utilisateurs au sein du fichier `/etc/master.passwd`, uniquement accessible au *super*-utilisateur. Ce fichier est utilisé pour générer automatiquement le fichier `/etc/passwd`.

Ce fichier est au format texte et contient une ligne par utilisateur. Il est structuré en dix champs, comme suit : `loginname:password:uid:gid:userclass:change:expire:gecos:homedir:shell` La sémantique associée à ces champs est la suivante :

- *loginname* : le nom de login de l'utilisateur
- *password* : le hashage du mot de passe, préfixé par la séquence `1` s'il s'agit d'un hashage *md5* ou d'un `2` s'il s'agit d'un hashage *blowfish*.
- *uid* : identifiant utilisateur.
- *gid* : identifiant du groupe principal.
- *userclass* : classe d'appartenance de l'utilisateur telle que définie dans le fichier `/etc/login.conf`.
- *change* : exprimé en nombre de secondes à partir de l'origine des temps Unix (1^{er} Janvier 1970 [Epoch]), donne la date d'expiration du présent mot de passe. Si la valeur est à zéro, il n'y a pas d'expiration du mot de passe.
- *expire* : même unité que précédemment, mais exprimant la date d'expiration du compte. Si la valeur est à zéro, il n'y a pas d'expiration du compte.
- *gecos* : zone de commentaire, nom réel de l'utilisateur ...
- *homedir* : répertoire de domiciliation de l'utilisateur.
- *shell* : interpréteur de commande de l'utilisateur.

Un exemple typique de ligne extraite du fichier `/etc/master.passwd` :

```
ikare:$1$iIVcBt5z$y0qex.EtFzuwBYwN2dZjj1:2323:2323::1179450000❶[1]:0❷[2]:Ivan K ↔
;:/home/ikare:/bin/sh
```

- ❶ Expiration du présent mot de passe le 18 Mai 2007 à 05:00:00 (date `-r 11794500000`).
- ❷ Pas d'expiration du compte.

1.6 Commandes de gestion des utilisateurs

Dans ce qui suit les commandes sont préfixées par un # qui symbolise un accès *super*-utilisateur (*root*). Le mot qui précède ce caractère est le nom de machine (*tux* pour *GNU/Linux* et *chuck* pour *FreeBSD*).

AVERTISSEMENT



Vous pouvez tester les commandes en utilisant l'accès étendu **sudo**, néanmoins je vous prie instamment de faire une copie initiale des fichiers que vous allez modifier. Puis quand vous aurez terminé ce paragraphe, de restaurer les fichiers dans leur état initiaux.

1.6.1 Gérer les utilisateurs avec `user{add,mod,del}` sous *GNU/Linux*

Première forme de la commande `useradd` : Invoquée sous cette forme, la commande crée un nouveau compte utilisateur en utilisant les valeurs spécifiées (sur la ligne de commande) et les valeurs système par défaut si les options ne sont pas précisées. Selon les options choisies, le nouveau compte sera créé et des fichiers initiaux seront copiés dans son espace de travail.

```
SYNOPSIS :
useradd-c comment-d homedir-e expire_date-f inactive_date-g initial_group ←
    -G group,-m -k skel_dir-p passwd-s shell-u uid-ologin
EXEMPLE : ajouter un utilisateur.

tux:/# useradd -c "Bernard's Account" -d /home/.bernard -e "2003-08-31" - ←
    s \
/bin/bash -u 668 -g gnu -G operator -m bernard
tux:/# passwd bernard
Enter new UNIX password:
Retype new UNIX password:
passwd: password updated successfully
tux:/#
```

Deuxième forme de la commande `useradd` : Cette deuxième forme permet de lister les paramètres par défaut ou modifier les valeurs par défaut à partir des valeurs données en ligne de commande.

```
SYNOPSIS :
useradd-D -g default_group-b default_home-f default_inactive-e ←
    expire_date-s default_shell
EXEMPLE : Lister les paramètres par défaut, puis modifier trois des ←
    paramètres.

tux:/# useradd -D
GROUP=100
HOME=/home
INACTIVE=-1
EXPIRE=
SHELL=/bin/bash
SKEL=/etc/skel

tux:/# useradd -D -g 200 -s /bin/tcsh -e "2003-12-31"
tux:/# useradd -D
GROUP=200
HOME=/home
INACTIVE=-1
EXPIRE=2003-12-31
SHELL=/bin/tcsh
SKEL=/etc/skel
```

usermod La commande `usermod` permet de modifier les caractéristiques d'un utilisateur.

```
SYNOPSIS :
usermod-c comment-d homedir-m-e expire_date-f inactive_date-g ↔
    initial_group-G group,-l loginname-p passwd-s shell-u uid-o-L-Ulogin
EXEMPLE : Ajouter l'utilisateur pascal dans le groupe operator
(il est déjà membre du groupe principal gnu et des groupes root et staff)
et lui mettre une date d'expiration au 31.12.2003 :

tux:/# usermod -G root,staff,operator -e 2003-12-31 pascal
```

userdel La commande **userdel** permet d'enlever un compte utilisateur (s'il existe). L'option **-r** permet en plus la suppression du contenu et du répertoire de domiciliation de l'utilisateur, ainsi que son répertoire de mail.

```
SYNOPSIS :
userdel-rlogin
EXEMPLE : Enlever un utilisateur

tux:/# userdel -r bernard
```

1.6.2 Gérer les groupes avec group{add,mod,del} sous GNU/Linux

groupadd La commande **groupadd** permet de créer un nouveau groupe.

```
SYNOPSIS :
groupadd-g gid-ologin
EXEMPLE : Ajouter un groupe

tux:/# groupadd -g 666 gnu
```

groupmod La commande **groupmod** permet de modifier les paramètres d'un groupe.

```
SYNOPSIS :
groupmod-g gid-o-n group_namegroup
EXEMPLE : Modifier le gid du groupe gnu.

tux:/# groupmod -g 667 gnu
```

groupdel La commande **groupdel**, enlève un groupe (s'il existe).

```
SYNOPSIS :
groupdelgroup
EXEMPLE : Enlever le groupe gnu.

tux:/# groupdel gnu
```

1.6.3 Cas particulier de FreeBSD

La commande **pw** permet au *super*-utilisateur de gérer simplement les utilisateurs. Sa syntaxe est très proche de ces homologues "Linuxienne". Cette commande fait les mises-à-jour dans les fichiers `/etc/passwd`, `/etc/master.passwd` et `/etc/group`. sous *OpenBSD*, par contre on utilise les mêmes commandes que pour *GNU/Linux*

– Première forme de la commande **pw useradd** :

```
SYNOPSIS :
pw-V etcdiruseraddnameuid-C config-q-n name-u uid-c comment-d dir-e date-p ↔
    date-g group-G grouplist-m-k dir-w method-s shell-o-L class-h fd-N-P-Y
EXEMPLE : Créer un nouvel utilisateur et lui attribuer un mot de passe
passwd :
```

```
salusa# pw useradd ivan -u 2324 -c "Ivan" -d /home/.ivan \
-e "31-05-2007" -g fremens -G wheel,operator -m -s /bin/tcsh -N
ivan:*:2324:2323::0:1180555200:Ivan:/home/.ivan:/bin/tcsh
salusa# pw useradd ivan -u 2324 -c "Ivan" -d /home/.ivan \
-e "31-05-2007" -g fremens -G wheel,operator -m -s /bin/tcsh
salusa# passwd ivan
Changing local password for ivan
New Password:
Retype New Password:
salusa#
```

- Deuxième forme de la commande **pw useradd** (sémantique équivalente à son homologue *Linux*), qui permet de spécifier les valeurs par défauts (-D) :

SYNOPSIS :

```
pw-V etcdiruseraddnameuid-D-C config-q-b dir-e days-p days-g group-G ←
groupulist-k dir-u min,max-i min,max-w method-s shell-y path
```

- **pw usershow**

SYNOPSIS :

```
pw-V etcdirusershownameuid-n name-u uid-F-P-7-a
EXEMPLE : Vérifier la présence de l'utilisateur bernard :
```

```
salusa# pw usershow ivan -P
Login Name: ivan                #2324                Group: fremens                #2323
Full Name: Ivan
Home: /home/.ivan                Class:
Shell: /bin/tcsh                Office: [None]
Work Phone: [None]                Home Phone: [None]
Acc Expire: Thu May 31 00:00:00 2007 Pwd Expire: [None]
Groups: wheel,operator
salusa#
```

- La commande **pw usernext** permet d'obtenir le prochain couple uid/gid disponible (séparé par les " : ") :

SYNOPSIS :

```
pw-V etcdirusernext-C config-q
EXEMPLE :
```

```
salusa# pw usernext
2325:2324
salusa#
```

- La commande **pw usermod** :

SYNOPSIS :

```
pw-V etcdirusermodnameuid-C config-q-n name-u uid-c comment-d dir-e date- ←
p date-g group-G groupulist-l name-m-k dir-w method-s shell-L class-h ←
fd-N-P-Y
EXEMPLE :
```

```
salusa# pw usermod -G wheel,operator,daemon,sys -n ivan
salusa#
```

- La commande **pw userdel** :

```
SYNOPSIS :
pw-V etcdiruserdelnameuid-n name-u uid-r-Y
EXEMPLE :

chuck# pw userdel -n ivan
```

– La première forme de la commande **pw groupadd** :

```
SYNOPSIS :
pw-V etcdirgroupaddgroupgid-C config-q-n group-g gid
EXEMPLE :

chuck# pw groupadd -g 999 -n trainees
```

– La deuxième forme de la commande **pw groupadd** :

```
SYNOPSIS : ↔

pw-V etcdirgroupaddgroupgid-C config-q-n group-g gid-M members-o-h fd-N-P-Y
```

– La commande **pw groupdel** :

```
SYNOPSIS :
pw-V etcdirgroupdelgroupgid-C config-q-n group-g gid
EXEMPLE :

chuck# pw groupdel trainees

... ou bien ...

chuck# pw groupdel -g 999
```

– La commande **pw groupmod** :

```
SYNOPSIS :
pw-V etcdirgroupmodgroupgid-C config-q-n group-g gid-l name-M members-m ↔
newmembers-h fd-N-P-Y
```

– La commande **pw groupshow** :

```
SYNOPSIS :
pw-V etcdirgroupshowgroupgid-g name-g gid-F-P-a
EXEMPLE :

chuck# pw groupshow wheel
wheel:*:0:root,pascal
```

– La commande **pw groupnext** qui retourne le prochain gid >disponible :

```
SYNOPSIS :
pw-V etcdirgroupnextgroupgid-q
EXEMPLE :

chuck# pw groupnext
2324
```

- La commande **pw lock** qui permet de verrouiller l'accès à un compte :

```
SYNOPSIS :  
pw-V etcdirdirlocknameuid-C config-q
```

- La commande **pw unlock** qui a un effet symétrique à la commande précédente :

```
SYNOPSIS :  
pw-V etcdirdirunlocknameuid-C config-q
```

1.6.4 Vérification

Sous *GNU/Linux* deux commandes permettent de vérifier l'intégrité des fichiers de gestion des utilisateurs. Ce sont la commande **/usr/sbin/pwck**, qui vérifie les fichiers `/etc/passwd` et `/etc/shadow` et la commande **/usr/sbin/grpck** qui vérifie les fichiers `/etc/group` et `/etc/gshadow`.

1.6.5 Edition en ligne

Elle se font par la commande **vipw** pour le fichier `/etc/passwd` (et respectivement `/etc/master.passwd` pour *FreeBSD* et `/etc/shadow` pour *GNU/Linux*); elle synchronise ces deux fichiers. La commande **vigr** agit de même sur les fichiers `/etc/group` et `/etc/gshadow` pour *GNU/Linux* uniquement.

1.7 Programmes de gestion des utilisateurs sous FreeBSD

Si l'implémentation de la commande **pw** est très complète, s'interface directement avec les fichiers `/etc/passwd` et `/etc/group`, et permet de créer des scripts complexes, FreeBSD propose néanmoins une série de programmes simples permettant la gestion interactive des utilisateurs.

1.7.1 adduser

adduser est un programme simple pour ajouter de nouveaux utilisateurs. Il crée les entrées dans les fichiers système `/etc/passwd` et `/etc/group`. Il crée également le répertoire utilisateur pour le nouvel utilisateur, y copie les fichiers de configuration par défaut (dotfiles) à partir de `/usr/share/skel`, et peut éventuellement envoyer à l'utilisateur un courrier électronique de bienvenue.

Exemple 1.1 Ajout d'un utilisateur avec adduser

```

salusa# adduser
Username: danielle
Full name: Danielle
Uid (Leave empty for default): 2326
Login group [danielle]: fremens
Login group is fremens. Invite danielle into other groups? []:
Login class [default]:
Shell (sh csh tcsh nologin) [sh]: tcsh
Home directory [/home/danielle]:
Use password-based authentication? [yes]: yes
Use an empty password? (yes/no) [no]: no
Use a random password? (yes/no) [no]: no
Enter password:
Enter password again:
Lock out the account after creation? [no]: no
Username   : danielle
Password   : *****
Full Name  : Danielle
Uid        : 2326
Class      :
Groups     : fremens
Home       : /home/danielle
Shell      : /bin/tcsh
Locked     : no
OK? (yes/no): yes
adduser: INFO: Successfully added (danielle) to the user database.
Add another user? (yes/no): no
Goodbye!
salusa#

```

1.7.2 rmuser

Le programme **rmuser** permet de supprimer complètement et de manière interactive un utilisateur du système. Elle effectue les opérations suivantes:

1. Supprime les entrées appartenant à l'utilisateur de la **crontab** (s'il y en a).
2. Supprime les tâches **at** appartenant à l'utilisateur.
3. Tue tous les processus appartenant à l'utilisateur.
4. Supprime l'utilisateur du fichier de mots de passe local.
5. Supprime le répertoire l'utilisateur (s'il lui appartient).
6. Supprime les courriers électroniques en attente pour l'utilisateur dans `/var/mail`.
7. Supprime tous les fichiers temporaires appartenant à l'utilisateur des zones de stockages temporaires comme `/tmp`.
8. Et enfin, supprime l'utilisateur de tous les groupes auxquels il appartient dans `/etc/group`. Si le groupe est vide d'utilisateur à la suite de l'opération, le groupe est également supprimé!

Exemple 1.2 Suppression d'un utilisateur avec rmuser

```

salusa# rmuser danielle
Matching password entry:

danielle:*:2325:2325::0:0:User &:/home/danielle:/bin/sh

Is this the entry you wish to remove? yes
Remove user's home directory (/home/danielle)? yes
Removing user (danielle): mailspool home passwd.
salusa#

```

1.7.3 chpass

chpass modifie les informations de la base de données des utilisateurs comme les mots de passe, les interpréteurs de commandes, et les informations personnelles. Utilisé sans options, en dehors du nom facultatif de l'utilisateur, **chpass** ouvre un éditeur affichant les informations de l'utilisateur. Quand l'utilisateur quitte l'éditeur, la base de données utilisateur est mise à jour avec les nouvelles informations.

Exemple 1.3 Modification d'un utilisateur avec chpass

```
#Changing user information for danielle.
Login: danielle
Password: $1$9zDHGBah$2v99d67vDnmq0hPhtsF880
Uid [#]: 2325
Gid [# or name]: 2325
Change [month day year]:
Expire [month day year]:
Class:
Home directory: /home/danielle
Shell: /bin/tcsh
Full Name: Danielle
Office Location:
Office Phone:
Home Phone:
Other information:
```

1.8 Autres commandes

TAB. 1 Autres commandes

commande	sémantique
id	Imprime l'uid, le nom de login, le gid, le nom du groupe : <code>ikare:~ >id</code> <code>uid=2323(ikare) gid=2323(ikare)</code> <code>groups=2323(ikare), 0(wheel),</code> <code>1001(fremens)</code>
groups	Affiche les groupes de l'utilisateur : <code>ikare:~</code> <code>>groupsikare wheel fremens</code>
passwd	Permet à l'utilisateur de changer son mot de passe (moyennant la connaissance de l'ancien). Invoqué sous l'identité du <i>super</i> -utilisateur avec pour argument un nom d'utilisateur, elle permet de changer inconditionnellement son mot de passe.
su	Permet de changer d'identité. Typiquement permet à un utilisateur de devenir <i>super</i> -utilisateur à condition de connaître son mot de passe. Sur les <i>flavors</i> BSD, seuls les utilisateurs membre du groupe <i>wheel</i> sont autorisés à utiliser cette commande. Par défaut, les seules variables d'environnement modifiées sont : <code>USER</code> , <code>LOGNAME</code> , <code>HOME</code> et <code>SHELL</code> . La commande su - permet l'exécution d'un <i>login shell</i> et par conséquent le changement d'environnement.

2 Exercices

2.1 Exercices d'applications

Les exercices suivants sont à pratiquer sous *FreeBSD*. Indiquez sur papier votre réponse puis testez-
là.

2.1.1 Exercice 1

A l'aide des utilitaires **grep** et **awk** déterminer si l'utilisateur *bin* existe sur votre système. Si oui, donner son uid.

Bien s'assurer de ne renvoyer au plus qu'une seule réponse !

2.1.2 Exercice 2

Quels sont les `gid` de vos différents groupes ?

2.1.3 Exercice 3

Quels sont les membres du groupe *operator* ?

Utiliser **grep** et **awk**

2.1.4 Exercice 4

[nécessite l'accès **sudo**] Créer l'utilisateur *test* avec la commande **useradd** en utilisant toutes les valeurs par défaut. Vérifier que le compte est bien créé.

Ajouter cet utilisateur au groupe *operator* et vérifier ensuite ces groupes d'appartenance

Se connecter à ce compte via le login puis via la commande **su**. Expliquer la différence.

Que faut-il faire pour que la connexion au login fonctionne pour cet utilisateur ?

3 Exercices d'évaluation

3.1 Gestion des utilisateurs

Création et modification d'utilisateurs et de groupes

1. Créez un groupe "asr"
2. Créez un utilisateur "asrStag", avec création automatique de son répertoire de domiciliation `/home/.asrStag`. Vous prendrez soin de lui associer un shell `csh`.
3. Affectez cet utilisateur au groupe `asr`
4. Vérifiez dans les différents fichiers que tout s'est bien passé
5. Connectez vous à ce compte en utilisant **su**.
6. Vérifiez votre identité en utilisant la commande **id** (testez différents paramètres)
7. Dans un autre *shell*, ajoutez le compte `asrStag` au groupe `wheel`
8. Vérifiez le changement dans la session `asrStag`.

3.2 Droits

3.2.1 Commandes

Un certain nombre de commandes permettent de modifier les droits des fichiers, ainsi que leur propriétaire. Consultez le **man**:

- **chmod**
- **chown**
- **chgrp**

3.2.2 Exercice 1

Exercices à faire sous les compte `asrdev` et `asrStag`

1. Créez un petit texte contenant le **man** de **chmod**, qui soit lisible par tout le monde, mais pas modifiable (même pas par vous).
2. Donnez les droits de modifications au groupe `asr`. Vérifiez en utilisant le compte `asrStag`.
3. Créez un répertoire nommé `secret`, dont le contenu soit visible uniquement par vous même. Les fichiers placés dans ce répertoire sont-ils lisibles par d'autres membres de votre groupe ?
4. Créer un répertoire nommé `connaisseurs` tel que les autres utilisateurs ne puissent pas lister son contenu mais puissent lire les fichiers qui y sont placés. On obtiendra :

```
$ ls connaissances l-
s : connaissances: Permission denied $ cat connaissances/toto <...le cont-
enu du fichier toto (s'il existe)...>
```

3.2.3 Exercices 2

Pour faire cet exercice, vous devez créer deux comptes dans le groupe `asr`, `stag1` et `stag2`. Donnez ensuite les commandes nécessaires pour chaque étape suivante :

1. `stag1` et `stag2` créent dans un répertoire `~/adminUsers/exos/droits/`
2. `stag2` crée un fichier exécutable `~/adminUsers/exos/droits/sc.sh`, qui contient les lignes suivantes :

```
#!/bin/sh echo je suis execute par `whoami` stat $0 echo stag2 est pass-
e par la
```
3. Dans son repertoire `~/adminUsers/exos/droits/`, `stag1` fait un lien dur vers le fichier de `stag2`, ayant pour nom `sonScript`
4. `stag1` exécute le fichier `sonScript`
5. `stag1` modifie le fichier `sonScript` de façon à ce que l'exécution de ce fichier affiche en plus le message : `stag1 est passe par la`
6. `stag2` veut être le seul à pouvoir modifier le fichier `sc.sh`

3.3 Scripts

3.3.1 Listage de répertoire

Ecrire un script `csh listerc_dir.sh` permettant d'afficher le contenu d'un répertoire en séparant les fichiers et les (sous)répertoires : `./listrc_dir.sh` affichera :

```
-----
Fichiers dans /etc/rc.d rc
rc.local
rc.sysinit
-----
Repertoires dans /etc/rc.d init. d
rc0.d rc1.d rc2.d rc3.d rc4.d rc5.d rc6.d
```

3.3.2 Affichage des droits

Créez un script `testfichier.sh`, qui précisera le type du fichier passé en paramètre, ses permissions d'accès pour l'utilisateur. Par exemple :

Le fichier `/etc` est un répertoire : `"/etc"` est accessible par `root` en lecture écriture exécution Le fichier `/home/ikare/asr-est/doamines.txt` est un fichier ordinaire qui n'est pas vide : `"/home/ikare/asr-est/doamines.txt"` est accessible par `ikare` en lecture écriture.