CPE Lyon - 4ETI, 3IRC, 3ICS - Année 2022/2023

Administration Système

TP 3 - Utilisateurs, groupes et permissions

Dans ce quatrième TP, nous allons voir comment gérer les utilisateurs, les groupes et les permissions sur notre système, ainsi que les quotas disque.

Dans votre compte-rendu de TP, vous n'oublierez pas d'indiquer, en plus des réponses aux questions, les **commandes** qui vont ont permis d'obtenir ces réponses. Essayez de donner les commandes qui répondent **exactement** à la question posée (càd qui n'affichent pas d'autres informations que celle attendue).

Exercice 1. Gestion des utilisateurs et des groupes

1. Utilisez la commande `groupadd` pour créer deux groupes `dev` et `infra`
   groupadd dev
   groupadd infra
2. Créez ensuite 4 utilisateurs `alice`, `bob`, `charlie`, `dave` avec la commande `useradd`, en demandant la création de leur dossier personnel et avec `bash` pour shell
   useradd -m -s /bin/bash alice ; id. pour bob, charlie, dave
3. Ajoutez les utilisateurs dans les groupes créés :
   - `alice, bob, dave` dans `dev`
   - `bob, charlie, dave` dans `infra`

   usermod -G dev alice
   usermod -G dev,infra bob
   usermod -G infra charlie
   usermod -G dev,infra dave
4. Donnez deux moyens d'afficher les membres de `infra`
   On peut se servir du fichier /etc/group : `grep ^infra /etc/group | cut -d: -f4`, mais les éventuels utilisateurs ayant infra comme groupe primaire ne sont pas listés !
   La solution la plus simple consiste à faire appel à la commande members (à installer) : `members infra`, qui permet d'afficher *tous* les membres d'un groupe, et même de distinguer les membres primaires ou secondaires.
5. Faites de `dev` le groupe propriétaire des répertoires `/home/alice` et `/home/bob` et de `infra` le groupe propriétaire de `/home/charlie` et `/home/dave`
   chown alice :dev /home/alice
   chown bob :dev /home/bob
   chown charlie :infra /home/charlie
   chown dave :infra /home/dave
6. Remplacez le groupe primaire des utilisateurs :
   - `dev` pour `alice` et `bob`
   - `infra` pour `charlie` et `dave`

   usermod -g dev alice
   usermod -g dev bob
   usermod -g infra charlie
   usermod -g infra dave
7. Créez deux répertoires `/home/dev` et `/home/infra` pour le contenu commun aux membres de chaque groupe, et mettez en place les permissions leur permettant d'écrire dans ces dossiers.
   mkdir /home/dev

mkdir /home/infra
chgrp dev /home/dev
chgrp infra /home/infra
chmod g+w dev <== indispensable car c'est root qui a créé le dossier
chmod g+w infra

8. Comment faire pour que, dans ces dossiers, seul le propriétaire d'un fichier ait le droit de renommer ou supprimer ce fichier ?
   il faut placer le "sticky bit" : chmod +t dev ; chmod +t infra
9. Pouvez-vous ouvrir une session en tant que `alice` ? Pourquoi ?
   Non, car ce compte n'a pas encore de mot de passe
10. Activez le compte de l'utilisateur `alice` et vérifiez que vous pouvez désormais vous connecter avec son compte.
    passwd alice
11. Comment obtenir l'uid et le gid de `alice` ?
    id -u alice
    id -g alice
12. Quelle commande permet de retrouver l'utilisateur dont l'uid est `1003` ?
    cut -d : -f1,3 /etc/passwd | grep " :1003" | cut -d : -f1
    ou aussi : getent passwd 1003 | cut -d : -f1
13. Quel est l'id du groupe `dev` ?
    grep ^dev /etc/group | cut -d : -f3
14. Quel groupe a pour `gid` 1002 ? (⚠ Rien n'empêche d'avoir un groupe dont le **nom** serait 1002...)
    grep " :1003" /etc/group | cut -d : -f1
    ou getent group 1003 | cut -d : -f1
15. Retirez l'utilisateur `charlie` du groupe `infra`. Que se passe-t-il ? Expliquez.
    gpasswd -d charlie infra => impossible car infra est le groupe primaire de charlie
16. Modifiez le compte de `dave` de sorte que :
    - il expire au **1er juin 2021**
    - il faut changer de mot de passe avant 90 jours
    - il faut attendre 5 jours pour modifier un mot de passe
    - l'utilisateur est averti 14 jours avant l'expiration de son mot de passe
    - le compte sera bloqué 30 jours après expiration du mot de passe

    sudo chage -E 06/01/2021 -m 5 -M 90 -I 30 -W 14 dave
    =>
    Last password change : Jan 20, 2015
    Password expires : Apr 19, 2015
    Password inactive : May 19, 2015
    Account expires : Jun 01, 2019
    Minimum number of days between password change : 5
    Maximum number of days between password change : 90
    Number of days of warning before password expires : 14
17. Quel est l'interpréteur de commandes (Shell) de l'utilisateur `root` ?
    grep ^root /etc/passwd | cut -d : -f7 => /bin/bash
18. Si vous regardez la liste des comptes présents sur la machine, vous verrez qu'il en existe un nommé *nobody*. A quoi correspond-il ?
    Nom conventionnel d'un compte d'utilisateur à qui aucun fichier n'appartient, qui n'est dans aucun groupe qui a des privilèges et dont les seules possibilités sont celles que tous les "autres utilisateurs" ont. Il est courant de lancer des démons en tant que nobody, spécialement pour des serveurs, de façon à limiter les dommages qui pourrait être occasionnés par un utilisateur malicieux qui aurait réussi à prendre leur contrôle
19. Par défaut, combien de temps la commande `sudo` conserve-t-elle votre mot de passe en mémoire ? Quelle commande permet de forcer `sudo` à oublier votre mot de passe ?
    15 minutes (man sudo)
    sudo -k

Exercice 2. Gestion des permissions

1. Dans votre `$HOME`, créez un dossier `test`, et dans ce dossier un fichier `fichier` contenant quelques lignes de texte. Quels sont les droits sur `test` et `fichier` ?
   test : drwxrwxr-x
   fichier : -rw-rw-r–
2. Retirez tous les droits sur ce fichier (même pour vous), puis essayez de le modifier et de l'afficher en tant que `root`. Conclusion ?
   root n'est pas affecté par les droits (ceci lui permet par exemple de retrouver un fichier égaré...)
3. Redonnez vous les droits en écriture et exécution sur `fichier` puis exécutez la commande `echo "echo Hello" > fichier`. On a vu lors des TP précédents que cette commande remplace le contenu d'un fichier s'il existe déjà. Que peut-on dire au sujet des droits ?
   les droits ne sont pas affectés par echo ; echo remplace le *contenu* du fichier
4. Essayez d'exécuter le fichier. Est-ce que cela fonctionne ? Et avec sudo ? Expliquez.
   Non, car on n'a pas le droit de lire le fichier (ça n'a rien à voir avec une absence de shebang ou autre). Avec sudo on peut car root surpasse les permissions et peut donc lire le contenu du fichier
5. Placez-vous dans le répertoire `test`, et retirez-vous le droit en lecture pour ce répertoire. Listez le contenu du répertoire, puis exécutez ou affichez le contenu du fichier `fichier`. Qu'en déduisez-vous ? Rétablissez le droit en lecture sur `test`.
   On ne peut plus lister le contenu du dossier ; en revanche, si on connaît le chemin complet vers les fichiers, on peut afficher / exécuter ceux sur lesquels on a ces droits
6. Créez dans test un fichier `nouveau` ainsi qu'un répertoire `sstest`. Retirez au fichier `nouveau` et au répertoire `test` le droit en écriture. Tentez de modifier le fichier `nouveau`. Rétablissez ensuite le droit en écriture au répertoire `test`. Tentez de modifier le fichier `nouveau`, puis de le supprimer. Que pouvez-vous déduire de toutes ces manipulations ?
   Le droit d'écriture sur un dossier ne donne pas le droit d'écrire dans les fichiers qu'il contient ; il donne le droit de créer ou supprimer des fichiers dans ce dossier
7. Positionnez vous dans votre répertoire personnel, puis retirez le droit en exécution du répertoire `test`. Tentez de créer, supprimer, ou modifier un fichier dans le répertoire `test`, de vous y déplacer, d'en lister le contenu, etc…Qu'en déduisez vous quant au sens du droit en exécution pour les répertoires ?
   Tout est impossible ; la commande ls liste quand même le contenu (on a le droit en lecture sur le dossier) mais ne permet pas d'afficher les infos sur les fichiers du dossier. Le droit x sur les dossiers donne l'autorisation d'accéder / traverser le dossier (par exemple avec cd)
8. Rétablissez le droit en exécution du répertoire `test`. Positionnez vous dans ce répertoire et retirez lui à nouveau le droit d'exécution. Essayez de créer, supprimer et modifier un fichier dans le répertoire `test`, de vous déplacer dans `ssrep`, de lister son contenu. Qu'en concluez-vous quant à l'influence des droits que l'on possède sur le répertoire courant ? Peut-on retourner dans le répertoire parent avec "`cd ..`" ? Pouvez-vous donner une explication ?
   R et W ne servent pas à grand chose si on n'a pas le droit en exécution
   https ://unix.stackexchange.com/questions/21251/execute-vs-read-bit-how-do-directory-permissions-in-linux-work
   cd .. fonctionne car avec bash, le dossier parent est calculé en enlevant la dernière partie du chemin courant : https ://stackoverflow.com/questions/24111236/when-we-do-not-have-execute-permission-on-current-directory-why-does-cd-wo
9. Rétablissez le droit en exécution du répertoire `test`. Attribuez au fichier `fichier` les droits suffisants pour qu'une autre personne de votre groupe puisse y accéder en lecture, mais pas en écriture.
   chmod g=r fichier
10. Définissez un umask très restrictif qui interdit à quiconque à part vous l'accès en lecture ou en écriture, ainsi que la traversée de vos répertoires. Testez sur un nouveau fichier et un nouveau répertoire.
    umask 077

11. Définissez un umask très permissif qui autorise tout le monde à lire vos fichiers et traverser vos répertoires, mais n'autorise que vous à écrire. Testez sur un nouveau fichier et un nouveau répertoire.
umask 022
12. Définissez un umask équilibré qui vous autorise un accès complet et autorise un accès en lecture aux membres de votre groupe. Testez sur un nouveau fichier et un nouveau répertoire.
umask 027
13. Transcrivez les commandes suivantes de la notation classique à la notation octale ou vice-versa (vous pourrez vous aider de la commande `stat` pour valider vos réponses) :
    - `chmod u=rx,g=wx,o=r fic`
    chmod 534 fic
    - `chmod uo+w,g-rx fic` en sachant que les droits initiaux de `fic` sont `r--r-x---`
    chmod 602 fic
    - `chmod 653 fic` en sachant que les droits initiaux de `fic` sont `711`
    chmod u-x,g+r,o+w fic
    - `chmod u+x,g=w,o-r fic` en sachant que les droits initiaux de `fic` sont `r--r-x---`
    chmod 520 fic
14. Affichez les droits sur le ***programme*** `passwd`. Que remarquez-vous ? En affichant les droits du fichier `/etc/passwd`, pouvez-vous justifier les permissions sur le programme `passwd` ?
présence d'un indicateur particulier : 's' => setuid
un utilisateur lance ce programme en prenant l'identité de root => indispensable pour qu'un utilisateur puisse modifier son mot de passe

Pour les plus rapides :

15. **Access Control Lists (ACL)** : suivez le tutoriel de cette page : https://doc.ubuntu-fr.org/acl.
16. **Quotas disques** : suivez le tutoriel de cette page : https://doc.ubuntu-fr.org/quota.