TP - LIFPCA Programmation Concurrente et Administration Système

—

Administration Système – GNU/Linux

Sylvain Brandel, Yves Caniou, Guillaume Damiand, Meriem Ghali, Laurent Lefèvre, Thibaut Modrzyk, Grégoire Pichon, Alec Sadler, Florence Zara, Jerry Lacmou Zeutouo

Printemps 2024

Vous avez normalement fait au moins le DM0 qui introduit toutes les manipulations que nous ferons dans ce TP. Par ailleurs, en cours, on vous a demandé de lire les URLs suivantes en préparation de ce TP :

- `https://www.debian.org/doc/manuals/debian-faq/pkg-basics.fr.html`
- `https://www.debian.org/doc/manuals/debian-faq/pkgtools.fr.html`
- `https://www.debian.org/doc/manuals/debian-faq/uptodate.fr.html`

Elles contiennent les principales informations *théoriques* qu'il vaut mieux retenir... ;)

I Administration Système

Où il est « intéressant » de connaître les commandes `top source alias less tail head ps grep whoami chown chgrp passwd adduser addgroup sudo su lsmod modprobe apt-get aptitude dpkg man` et certaines de leurs options.

I.1 Création/re-création de VM

Pour ce TP, vous devez utiliser la machine virtuelle que nous vous avons créé et sur laquelle vous êtes administrateur. Vous pouvez revenir sur le TP1 pour les instructions pour vous y connecter (on supposera par la suite que `IP_VM` est son IP). Grâce au DM0, vous savez comment configurer un environnement agréable pour travailler, que ce soit sur votre compte Lyon 1 ou sur votre VM (notamment avec des alias comme `l='ls --color -l'`, un prompt `$1` défini correctement et en couleur, etc.).

I.2 Configurations d'environnement : confort d'utilisation avec la complétion "intelligente"

Vous pouvez remarquer que la complétion existe bien sur votre compte Lyon1, mais pas la complétion "intelligente". Nous avons vu dans le DM0 qu'elle était mise en place

automatiquement grâce à la commande `source /etc/bash_completion` sourcée au démarrage d'un nouveau shell. Mais ce fichier n'existe pas sur les installations Fedora, les admins ont visiblement oublié de l'installer...

Q.I.1) - À partir de la VM, comment connaître son adresse IP ?

Q.I.2) - Le fichier `/etc/bash_completion` est-il le seul intéressant ? Pour le savoir, regardons quel package l'a installé (à moins que vous vous souveniez de son nom ?) : tapez `dpkg -S /etc/bash_completion`. Cela permet justement de donner le package qui a installé le fichier donné.

Q.I.3) - Comment lister tous les fichiers installés par le package contenant `/etc/bash_completion` ?

Q.I.4) - On voit que le package installe notamment `/etc/profile.d/bash_completion.sh`. Qu'est-ce que ce fichier, et que contient-il (et pourquoi) ? Quelle différence avec `/etc/bash_completion` ?

Q.I.5) - On voit également le fichier `/usr/bin/dh_bash-completion`. Si on regarde son contenu, peut-on savoir à quoi il sert à priori ?

Q.I.6) - À l'aide de la commande `rsync` (ou `scp`), copiez le fichier `/etc/bash_completion` de votre VM sur votre compte Lyon1, par exemple dans `~/.bash_completion_VM`.

Q.I.7) - Sourcez-le. Est-ce que cela change quelque chose à l'environnement ? *A*ny comments ?

Les étudiants ont vu en LIFRES (UE Réseaux) la commande :
`ip a` (pour `ip address show`). On voit qu'il y a 127.0.0.1 (`localhost`, *c.-à-d.* la boucle locale, pour que les programmes réseaux puissent utiliser une interface réseau virtuelle et fonctionner correctement même en n'étant pas sur un réseau : c'est le cas de Xorg), et `ens3`. L'adresse internet est lisible après le mot `inet` (pour internet).

Pour lister tous les fichiers installés par un package :
`dpkg -L bash-completion`

Pour savoir ce qu'est le fichier, attention !
`file /etc/profile.d/bash_completion.sh`
Pour savoir ce qu'il contient, un cat ou nano (qui sera en lecture seule) fonctionne très bien : le point à voir est qu'il y a un test pour savoir si le shell est en mode interactif ou non, et qu'on est capable facilement de regarder et comprendre des scripts qui nous sont donnés par l'installation.

Pour savoir la différence, il faut taper :
`diff /usr/share/bash-completion/bash_completion /etc/bash_completion` même si ici la sortie est longuette (ça nous aménera à parler des patches plus loin). On voit simplement que quand on fait `source /etc/bash_completion`, ce qui est fait est juste un `source /usr/share/bash-completion/bash_completion`, et que ce dernier fichier contient le script qui permet de rendra la complétion intelligente (on ne rentrera pas dans les détails de ce script).

Le fichier `/usr/bin/dh_bash-completion` est un outil destiné aux développeurs de paquets Debian, pour leur permettre d'ajouter la complétion intelligente pour la commande contenue dans le package.

Il suffit de faire, depuis son compte Lyon 1 :
`rsync -av etudiant@IP_VM:/etc/bash_completion ~/.bash_completion_VM`

Ensuite, on ne devrait pas sourcer n'importe comment des trucs ! (ou en tout cas, ne jamais mettre dans la config un truc par défaut sans l'avoir testé avant). Ici, on s'assure bien qu'on était le seul à avoir pu se logguer sur la machine : ceci pourrait être un vecteur d'attaque...
`source ~/.bash_completion_VM`

Note : je n'ai pas pu tester. Le point important ici est la première approche sur comment récupérer des trucs sympas d'une installation pour faire la même chose autre part, que l'on soit admin ou non...
D'après la doc de `https://github.com/scop/bash-completion`, peut-être qu'en faisant quelque chose comme :
`mkdir -p ~/.local/share/bash-completion`
Puis copier les fichiers comme des fichiers de complétion locaux :
`scp -r etudiant@192.168.75.222:/usr/share/bash-completion/completions ~/.local/share/bash-`
éventuellement avec un `export BASH_COMPLETION_USER_DIR=~/.local/share/bash-completion`
si cela ne fonctionne toujours pas (pas réussi sur ma gentoo, mais le système est différent, donc...).

I.3 Retour sur les alias et fonctions

Q.I.8) - Toujours taper `ls -l --color` est long. Proposez un alias *l* pour faire la même chose, et rendez-le persistant !

Q.I.9) - La fonction suivante utilise le package `adb` pour communiquer avec un téléphone android. Que fait-elle ?

```
# $1 must be provide as date like 201509, 2015, or 20150926
function down_pics_from_phone {
    local photo_dir="/storage/sdcard1/DCIM/Camera"
    if [ "$1" == "" ] ;then
        adb shell "ls $photo_dir" > /dev/shm/list_pics2
        tr -d '\r' < /dev/shm/list_pics2 > /dev/shm/list_pics
        rm /dev/shm/list_pics2
        cat /dev/shm/list_pics
        echo List available in /dev/shm/list_pics
        echo Use $0 YYYYMMDD or subset to pull photos
        return
    fi
```

```
    adb shell "ls $photo_dir" | grep $1 > list_pics
    tr -d '\r' < list_pics > list_pics2
    for i in `cat list_pics2` ; do
        echo "Getting $i"
        adb pull $photo_dir/$i
    done
    rm list_pics list_pics2
}
```

C'est l'occasion d'un peu de révision shell avec les redirections, la commande `tr` qui est bien utile. Il est bon d'avoir un peu pratiqué du `sed` (voir les DM !) également même si pas utilisé ici.

On aura besoin du package `adb`, qui permet des communications USB avec son téléphone (copies notamment). Naturellement, ça signifie installer le package (voir plus loin dans le TP) sans possibilité réelle de l'utiliser puisque on ne peut connecter son téléphone directement à la VM. MAIS ça peut être utile pour soit à la maison...

Remarques : perso, j'ai également une fonction appelée `YCrm_pics_from_phone`...

I.4 Les utilisateurs et root – rappels de LIFSE

Q.I.10) - Avant tout, connectez-vous (ssh) sur la VM OpenStack en tant qu'utilisateur `etudiant` (reprenez l'énoncé du TP1 si besoin).

Q.I.11) - Quels sont tous les utilisateurs présents sur la machine ?

La question est ambiguë :
- `w` permet de savoir qui est connecté (on peut utiliser `who` qui fait presque la même chose).
- `cat /etc/passwd` permet de savoir quels sont les utilisateurs créés sur la machine...
Ici, l'idée est de faire la 2e commande.

Q.I.12) - Sur votre VM, créez un autre utilisateur de login `lurong` (nom complet : Gai Luron), et donnez-lui un mot de passe différent de celui pour `etudiant` (rappel : la commande `adduser` est plus pratique que `useradd`, c'est en fait un wrapper.).

```
$ sudo adduser lurong
[sudo] password for etudiant:
Adding user `lurong' ...
Adding new group `lurong' (1000) ...
Adding new user `lurong' (1000) with group `lurong' ...
Creating home directory `/home/lurong' ...
Copying files from `/etc/skel' ...
Enter new UNIX password:
```

```
Retype new UNIX password:
passwd: password updated successfully
Changing the user information for lurong
Enter the new value, or press ENTER for the default
        Full Name []: Gai Luron
        Room Number []:
        Work Phone []:
        Home Phone []:
        Other []:
Is the information correct? [Y/n] y
```

Q.I.13) - Depuis un shell appartenant à l'utilisateur `etudiant`, ouvrez un shell en tant qu'utilisateur `lurong`. Fermez ce shell pour revenir au shell de `etudiant`.

Depuis un shell existant, on peut entrer :

```
$ su - lurong
Password:
```

Avez-vous bien compris l'utilité du '-' dans la ligne de commande précédente ? Notez bien de quel utilisateur le mot de passe est demandé.
Pour sortir du shell, Control-D, ou `exit`.

Q.I.14) - Depuis votre PC physique, ouvrez un nouveau terminal (pour conserver le shell `etudiant` ouvert), et dans ce nouveau terminal ouvrez une connexion sur votre VM en vous connectant *directement* en tant qu'utilisateur `lurong`.

Au choix, `ssh lurong@IP_VM` ou bien `ssh IP_VM -l lurong`.

Q.I.15) - Regardez le contenu du fichier `/etc/passwd` (il est accessible en lecture pour tout le monde, vous pouvez voir son contenu avec une commande comme `less` ou `cat`, ou l'ouvrir dans votre éditeur de texte). Vous devriez trouver une ligne pour l'utilisateur `lurong` contenant entre autres le nom complet que vous avez entré à la création, et d'autres informations qui ont été ajoutées automatiquement comme le répertoire personnel `/home/lurong` et le shell par défaut `/bin/bash`. Notez que le mot de passe ne se trouve pas dans ce fichier (il y a un `x` dans la colonne où il aurait pu se trouver, qui signifie qu'il est stocké ailleurs).

Q.I.16) - Regardez maintenant le contenu du fichier `/etc/shadow`. Celui-ci n'est accessible que par `root`. Dans ce fichier, il y a bien une information sur le mot de passe, mais pas le mot de passe lui-même. Pourquoi ?

Vous aurez essayé de taper `sudo cat` pour afficher le fichier, après avoir vu que les utilisateurs n'ont pas le droit d'afficher ce fichier – ça doit être sû sans même essayer ;) et vous avez bien noté de quel utilisateur le mot de passe est demandé dans ce cas ? – Mais c'est resté une tentative car **`lurong` n'est pas dans les utilisateurs sudo : ça doit**

être configuré dans le fichier `/etc/sudoers`... Il faut donc faire la manipulation de l'exercice en tant que etudiant, qui lui est sudo. Et c'est l'occasion de comprendre comment comment configurer la commande sudo si vous avez le temps en fin de TP.

Le fichier `/etc/shadow` contient un hash du mot de passe, *etc.*. (fait en cours).

Q.I.17) - Quel est l'`UID` (User IDentifier, le nombre qui identifie l'utilisateur de manière unique) de l'utilisateur `lurong` ?

Soit avec `id lurong`, soit la 3ème colonne de `/etc/passwd`.

Q.I.18) - Comment connaître les groupes auxquels il appartient ?

Soit avec `groups lurong`, ou la commande `groups` exécutée en tant que lurong, soit dans le fichier `/etc/group` qui est également accessible en lecture pour tous.

I.5 Utilisateurs et gestion des droits

Q.I.19) - Arrangez-vous pour avoir deux terminaux ouverts, l'un avec un shell appartenant à `etudiant`, l'autre avec un shell `lurong`. Si vous avez bien suivi les consignes ci-dessus, vous l'avez déjà fait. Vous devriez aussi avoir des invites de commandes différentes dans ces deux terminaux.

Q.I.20) - Dans les deux terminaux, vérifiez que vous êtes bien celui que vous pensez à l'aide de la commande `whoami`.

Q.I.21) - Vérifiez que chaque utilisateur se trouve, pour l'instant, dans son propre répertoire personnel (commande `pwd`).

Q.I.22) - Dans le shell sur la VM, sous l'identité `etudiant`, créez un fichier avec la commande `touch poi` et exécutez `ls -l`. Modifiez ce fichier avec la commande de votre choix.

Q.I.23) - Dans le shell sous l'identité `lurong`, retrouvez le fichier `poi`. Faites un `ls -l` dessus et regardez son contenu. Essayez de le modifier. Que se passe-t-il ?

Le fichier n'est pas accessible en écriture, donc on peut faire `ls -l`, regarder le contenu mais pas l'éditer.

C'est aussi l'occasion de noter que le répertoire de `etudiant` est en lecture et exécution pour tous ! Qu'est-ce que cela signifie ?

Q.I.24) - Revenez sous le shell de `etudiant` et exécutez la commande `chmod go+w poi`. Réessayez de modifier le fichier en tant que `lurong` (cela devrait marcher maintenant). Faites un `chmod go-w poi` en tant que `etudiant`, et vérifiez que la commande a bien coupé les droits en écriture.

Q.I.25) - Exécutez `chmod go-r poi` en tant que `etudiant` et vérifiez que les droits en lecture ont été coupés. Exécutez `chmod 644 poi` en tant que `etudiant` pour revenir à la situation précédente.

Explication : `chmod go+w` = pour Group et Other, ajouter (+) le droit Write. `chmod go-w` = idem, mais supprime (-). `chmod 644` : notation octale (6 = 4 + 2 = read + write, 4 = read).

Q.I.26) - en tant que `lurong`, exécutez `chmod go+rw poi`. Vous devriez obtenir une erreur : vous n'êtes pas propriétaire du fichier, vous n'avez pas le droit de modifier ses permissions.

Q.I.27) - Ajoutez `lurong` au group `etudiant` à l'aide de la commande `usermod`, et vérifiez.

C'est naturellement une commande système, il faut donc l'exécuter avec des droits `root`, par exemple `sudo usermod -a -G etudiant lurong`. Pour vérifier, en tant que `lurong`, il suffit de (re-)taper `groups` (peut-être se déconnecter/reconnecter avant ?).

Q.I.28) - Est-ce que maintenant `lurong` peut modifier le fichier `poi` ?

Q.I.29) - Dans une console sur la VM, sous l'identité `root`, faites un `chmod go-rwx /etc/passwd`. En tant que `etudiant`, exécutez maintenant `ls -l` et expliquez ce que vous voyez (et pas ce que vous croyez voir). Remettez les bons droits au fichier `/etc/passwd`.

C'est plus évident dans le compte de `etudiant` que si on liste `/etc/passwd` : dans ce dernier cas, on lit `root` et on pense que tout est normal, alors que le 0 à sa gauche est l'UID qui n'a pas été résolu en login ! En listant le répertoire de connexion de `etudiant` et en voyant 1004, ça saute un peu plus aux yeux.

Q.I.30) - En local (la question n'a pas de sens sinon), comment passer en mode console ? Revenir au mode graphique ?

Normalement, Ctrl-Alt-F1-F6 en fontion du nombre de tty ouverts, et Alt-F7 pour la première session graphique.

Sur les installations Fedora de l'université, F1 correspond à la session graphique, les autres sont des consoles.
L'intérêt de passer en mode console peut se manifester pour exécuter une commande `top` car on trouve que le système est un peu lent. Le système n'a plus à mettre à jours les nombreuses fenêtres, *etc.*. On peut voir aussi la quantité de RAM utilisée, l'espace SWAP utilisé : un système peut être ralenti par les nombreux accès disques si la machine manque de mémoire et fait des accès au SWAP.

I.6 Les packages et leur gestion

Q.I.31) - Quelle est la liste complète des packages installés ?

`dpkg --list` (ou `dpkg --get-selection`)

Q.I.32) - Que dois-je faire si je souhaite trouver un package contenant le mot clé `keyring` par exemple ?

Je dois déjà faire un `apt update` pour mettre la base de données des packages et leurs versions à jour : si je ne le fais pas, je peux trouver des résultats obsolètes, ou tenter dans la foulée d'installer un package qui n'existe même plus. Ensuite `apt search keyring`. On remarque qu'on peut donner plusieurs mots clés à la suite.

Certains packages en lien avec `keyring` concernent des clés d'authentifications : ce sont grâce à elles que des communications sécurisée (chiffrées et authentifiées) sont mises en place lors de l'installation de packages.

Q.I.33) - Faire une mise à jour de la base des packages.

```
sudo apt update
```

Q.I.34) - Démarrer une mise à jour du système (pour ne pas perdre de temps vous pouvez annuler la mise à jour en répondant « n » à la dernière question).

```
sudo apt upgrade
```

Nous allons maintenant installer un serveur web sur votre VM. Un serveur web très connu est apache2, mais nous allons jouer dans un premier temps avec Nginx (à prononcer « engine-X »).

Q.I.35) - Comment savoir si Nginx est installé ?

`apt policy nginx` ou bien `dpkg -l nginx` qui doit contenir « ii ». Mais nécessite de répondre à la question suivante d'abord, pour connaître le nom du package lié à Nginx. **Ne pas oublier d' utiliser la tabulation intelligente quand c'est possible !**

Q.I.36) - Quel est le package concernant nginx ?

```
apt search nginx
```

Q.I.37) - Voir si nginx est installé.
Quelle version serait installée si on décidait d'installer nginx ?

```
apt show nginx | grep Version
```

Q.I.38) - Installer le package `nginx`

```
sudo apt install nginx
```

Q.I.39) - Quels sont les fichiers installés par nginx ?

`dpkg -L nginx` ou l'option longue : `dpkg --listfiles nginx`, mais en réalité la majorité des fichiers sont installés par les dépendances du package, surtout `nginx-common`

Q.I.40) - Voir les fichiers de configuration de nginx.

`dpkg --listfiles nginx-common | grep etc`, principalement dans `/etc/nginx/`

Q.I.41) - Où sont les fichiers de log de nginx

`/var/log/nginx/`

Q.I.42) - À votre avis, votre VM a t-elle un serveur ssh installé ? Comment le savoir ? Où sont ses fichiers ?

Q.I.43) - Vous pouvez regarder les packages `adb` (voir la question sur les fonctions), `french-conjugator`, `anki`, `nmap`, *etc.*

Attention, tout scan non authorisé d'une machine est considéré comme une attaque devant la loi...

Q.I.44) - Installez le package `zmap` avec la commande `apt-get install zmap` (c'est un petit package, et ce n'est pas son installation ici qui est importante). Contrairement à `apt`, `apt-get` ne fait pas le ménage des packages téléchargés : on peut donc trouver les archives `.deb` dans `/var/cache/apt/archives/`.
Copiez le package debian qui s'y trouve dans le répertoire de connexion, et explosez l'archive.

```
ar -x zmap_2.1.1-2_amd64.deb
```

Q.I.45) - Dans un sous-répertoire `tmp`, explosez l'archive `control.tar.gz`. Que contient-elle ? À quoi servent ces fichiers ?

`mkdir tmp ; cd tmp/ ; tar xzvf ../control.tar.gz`
Contient les fichiers `./md5sums`, `./conffiles` et `./control`.
On peut s'intéresser à `./md5sums`, qui contient les hashes des différents fichiers du package, pour vérifier leur intégrité.

Q.I.46) - Dans un sous-répertoire `pmp`, explosez `data.tar.xz` et commentez.

`mkdir ../pmp ; cd $_ ; unxz ../data.tar.xz ; tar xvf ../data.tar`
Ce sont tous les fichiers, dans une arborescence locale, qui sont normalement installé à la racine du système du fichier.

I.7 Gestion des services avec systemd

Nginx est un logiciel prévu pour tourner en tâche de fond sur la machine. Il est lancé au démarrage et tourne en permanence même si personne n'est connecté à la machine. On appelle cela un *démon* (daemon en anglais). Son rôle est de répondre aux requêtes HTTP reçues, donc c'est aussi un *serveur*. Ce serveur fournit un *service* aux clients.

Q.I.47) - Vérifiez que votre machine répond bien quand un navigateur web l'interroge sur le port 80. Plusieurs solutions :

- Lancer votre navigateur web habituel sur `http://IP_VM/` (cela ne marchera que si vous avez un accès direct à la VM, donc pas si vous tentez une connexion depuis l'extérieur de l'université ni depuis le wifi).
- Depuis un shell qui tourne sur votre VM, lancez la commande `links http://localhost/` (si besoin, installez `links` au préalable). Links est un navigateur en mode texte, peu convivial mais cela présente entre autres l'avantage de pouvoir être lancé facilement via une connexion SSH.
- Si vous voulez vous amuser : faire un tunnel SSH pour accéder à `IP_VM:80` depuis votre PC physique.

Vous devriez voir apparaître la page d'accueil par défaut de nginx (« Welcome to nginx ! »). Si vous le voulez, vous pouvez aussi voir et modifier le contenu de cette page dans le fichier `/var/www/html/index.nginx-debian.html`.

Q.I.48) - Interrogez systemd pour avoir le statut du service nginx avec la commande :

```
systemctl status nginx.service
```

Q.I.49) - Vérifiez « à la main » que le processus nginx tourne : `ps aux | grep nginx`

Q.I.50) - Coupez le service nginx avec la commande :

```
sudo systemctl stop nginx.service
```

Q.I.51) - Ré-essayez de charger la page web : vous aurez une erreur du type « connection refused ».

Q.I.52) - Interrogez systemd pour avoir le status du service nginx avec la même commande que ci-dessus. La ligne « Active » doit être passée de « active (running) » à « inactive (dead) ».

Q.I.53) - Vérifiez « à la main » que le processus nginx ne tourne plus : `ps aux | grep nginx`

Q.I.54) - Redémarrez le service :

```
sudo systemctl start nginx.service
```

Q.I.55) - Regardez à quoi ressemble le fichier de description du service pour systemd : `/etc/systemd/system/multi-user.target.wants/nginx.service`
Vous n'avez pas besoin de comprendre les détails, mais ce fichier contient au moins :

- La commande pour démarrer le démon (`ExecStart`)
- La commande pour arrêter le démon (`ExecStop`)
- Un descriptif (`Description`)
- Les dépendances (`After`, `WantedBy`)

Q.I.56) - Pour lister les services disponibles, lancez la commande `systemctl`. Retrouvez la ligne concernant nginx avec `systemctl | grep nginx`.

Q.I.57) - Comment savoir si le serveur ssh est lancé (c'est-à-dire, comment connaître le status du serveur ssh) ?

I.8 Les logs !

Q.I.58) - Naviguez dans `/var/log`. Quels sont tous ces fichiers, leurs droits et que contiennent-ils ?

> Insister sur messages, daemon, X et... auth.log pour les connexions ssh !

Q.I.59) - Lancez la commande `tail -f /var/log/nginx/access.log`, puis, pendant que cette commande tourne, rechargez la page d'accueil de nginx. Vous devriez voir apparaître une ligne par requête dans le fichier de log (`tail -f` les affiche au fur-et-à-mesure).

Q.I.60) - Chargez la page `http://IP_VM/404/`. Comme cette page n'existe pas, vous verrez apparaître la page d'erreur 404 (Not Found) de nginx.

Q.I.61) - Installez maintenant le paquet `apache2`. Ce paquet correspond à la version 2 du serveur web Apache httpd. C'est un concurrent direct de nginx, qui va essayez d'écouter sur le port 80 (Ce qui est impossible pour l'instant car nginx est déjà en écoute dessus, mais faisons semblant de ne pas savoir pourquoi un instant ...).

Q.I.62) - Forcez un lancement d'apache :

```
sudo systemctl start apache2
```

Q.I.63) - Essayez de recharger la page d'erreur 404 : c'est toujours nginx qui répond !

> Attention, le paquet apache2 a installé un fichier HTML correspondant à sa page d'accueil par défaut. Si vous chargez la page d'accueil, vous aurez peut-être celle d'apache, servie par nginx.

Q.I.64) - Regardez le status du service apache2 avec `systemctl status`. Vous devriez obtenir « Active : inactive (dead) », et la commande vous donne quelques lignes de logs qui devraient vous mettre sur la voie de la raison de l'échec du lancement. Vous pouvez retrouver les détails avec la commande `journalctl`.

Q.I.65) - Devant l'échec de cette tentative de lancer deux serveurs web sur la même machine, nous décidons d'abandonner et de désinstaller apache.

II Reviser en s'amusant : le jeu de piste

Une petite exercices pour réviser la gestion des paquets et des logs : un jeu de piste. Chaque étape vous demande une manipulation, à réaliser sur votre VM, et faire cette manipulation vous donne accès à l'étape suivante, et ainsi de suite. Le point de départ est ici : `https://asr-lyon1.gitlabpages.inria.fr/prog-concurrente/jeu-de-piste.html`. Toutes les commandes utilisées pendant le jeu de piste sont au programme, vous devez les connaître pour l'examen final.

III Si le temps le permet : copie de fichiers à distance (vu en LIFSE en L2)

Vous devez apprendre à utiliser ssh pour copier des fichiers à distance. Une excellent alternative est `rsync`.

Si vous souhaitez copier votre `.bashrc` sur plusieurs machines, vous pouvez faire des cas particulier pour certaines machines en mettant des conditions sur le `HOSTNAME`.

Q.III.1) - Faites le en utilisant la ligne de commande, via `scp`, `pscp` (`PuTTY`) ou `rsync`.

Pour copier `machin` dans le repertoire `/tmp/` de la machine virtuelle : `scp -Cr machineacopier etudiant@192.168.77.3:/tmp/` l'option `-C` permet de compresser et l'option `-r` de copier de manière récursive un répertoire.
Attention : utilisation d'une autre clé ssh avec rsync `r`sync -e "ssh -i /.ssh/blabla" –progress root@IP /tmp/

Certains logiciels reposant sur la bibliothèque `FUSE` (File System in User Space) permettent de configurer un système de fichiers en réseau en tant que simple utilisateur. Vous pourrez en installer et utiliser un pour associer à un répertoire du compte etudiant de la VM un répertoire que vous utilisez à l'université pour les TPs de LIFPCA.

Q.III.2) - Qu'est-ce qu'un disque réseau ? Donner un exemple de logiciel utilisant ce type de bibliothèque.

Cela permet d'associer un disque (windows) ou un répertoire (unix) à un répertoire de fichiers situés sur un serveur réseaux. Une fois cette association faite, la synchronisation est automatique, les fichiers peuvent être utilisés comme s'ils étaient locaux à la machine. C'est ce qui est utilisé par vos comptes à l'université, ou des systèmes comme `DropBox`, ou mieux `Owncloud` ou `Pydio` que vous pouvez installer sur votre machine à la maison.

Q.III.3) - Installez le logiciel `sshfs`.

`sudo apt-get install sshfs` puis répondez oui aux questions.

Q.III.4) - Regardez le manuel de la commande `sshfs`, utilisez-la pour que le répertoire `votrevm:/home/etudiant/univ/` corresponde au répertoire `"$HOME"/LIFPCA/VM` sur votre poste de travail de l'université.

Utiliser la commande
`sshfs <votre login>@pedagolinux710.univ-lyon1.fr:<votre répertoire>/LIFPCA/ /home/etudiant/un`
attention, pour le nom du repertoire. `~/` ne fonctionne pas, il faut mettre le nom complet : `/home/etu/?/p??????????/`.

Q.III.5) - Quel est l'intérêt de faire des systèmes de fichiers dans « l'espace utilisateur » ?

L'accès aux fichiers est une opération du système (espace noyau). Si un utilisateur veut avoir accès à un fichier qui se trouve ailleurs sur le réseau, il doit le télécharger ou utiliser un logiciel qui le télécharge à sa place. Un logiciel qui n'est pas « prévu pour » ne peut donc pas le faire. En configurant depuis l'espace utilisateur un système de fichiers, on

permet à un utilisateur (non administrateur) de créer un système de fichiers réseau. Tous les logiciels seront donc capables de manipuler les fichiers de ce système, même s'il ne sont pas réellement présents sur la machine. Par exemple, sans avoir à faire de copie, vous pourrez compiler vos codes sur la VM et les modifier sur votre poste de travail. Pour cela, si le logiciel est installé, vous n'avez pas besoin de devenir administrateur : il vous suffit d'avoir le droit de lire et modifier les différents répertoires concernés.

IV Pour aller plus loin

IV.1 Configuration de `sudo`

`sudo` permet d'autoriser à un utilisateur l'exécution d'une ou plusieurs commandes avec des droits privilégiés. C'est une commande **additionnelle** au système.

Q.IV.1) - Comment la machine ASR7, dont vous avez obtenu une copie pour créer votre VM, a été configurée pour donner la possibilité à `etudiant` de lancer n'importe quelle commande avec `sudo` ?

Q.IV.2) - Comment configurer `sudo` pour donner le droit à `lurong` d'exécuter la commande `cat` avec les droits `root` (afin par exemple de pouvoir afficher le contenu de `/etc/shadow`) ?

Q.IV.3) - Comment se fait-il que la commande `sudo` puisse donner le droit à un utilisateur de lancer un processus (donc à priori avec les droits propres à cet utilisateur) avec des droits `root` ?

C'est du cours...
Par ailleurs, le bit SetUID ne permet-il à des processus que de gagner les droits `root` ? Non... il s'agit de gagner les droits du propriétaire pendant l'exécution de la commande. Dans nos exemples, le propriétaire est à chaque fois `root` (pour `sudo, chsh` *etc.*) mais ça pourrait être différent.

IV.2 Les "fichiers" du noyau Linux

Q.IV.4) - Naviguez dans `/boot`, `/usr/src/`, `/lib/modules/`
Peut-on lire la même chose en local ?

S'assurer que les étudiants ont toujours bien 2 voire 3 consoles, et qu'ils les utilisent plutôt que se déconnectent et reconnectent.
Et on peut lire le même genre de choses en local, mais pas la même chose car machine différente...

Quelles sont les capacités du noyau chargé ?

`lsmod` pour lire les modules chargés, *i.e.*, ces capacités immédiates. On voit plus loin `modprobe`...

Q.IV.5) - Faire `dmesg`, distant et local.
Repérer le type de processeur, et de disque dur.
Faites `cat /proc/cpuinfo` et `cat /proc/meminfo`
Quelles informations obtenez-vous ?
Peut-on les obtenir en local et/ou distant ?
Du coup, quel utilisateur peut exécuter la commande ?
Comment était-il possible de le savoir avant ?

Q.IV.6) - Listez `/proc/` et commentez ce que vous y trouver.

Parler des répertoire pid, et voir ce qu'on y trouve : c'est directement en lien avec le CM. Insister sur les droits d'accès, du type de système de fichiers utilisé pour accéder à ces informations.

Des remarques sur `/proc/sys/net/ipv4/ip_forward` ?

Q.IV.7) - Quels systèmes de fichiers le noyau installé est-il capable de lire ?

```
cat /proc/filesystems
```

Q.IV.8) - Qu'est-ce que `/dev/` ?
Qu'est-ce que `/dev/shm/` ? Qu'est-ce que `/dev/disk/by-uuid/` ?

`/dev/` contient les fichiers liés aux périphériques. `/dev/shm/` contient un système de fichier en mémoire RAM utilisé pour faire communiquer les applications en mémoire partagée (SHared Memory). `/dev/disk/by-uuid` contient la liste des disques, classés par UUID, c'est à dire par identifiant unique (en pratique ce répertoire contient des liens symboliques vers les fichiers `/dev/sd...`)

IV.3 Jouons à casser notre environnement ;-)

Comme le titre l'indique, ne pas faire ces manipulations sur votre compte habituel pour ne pas prendre de risque.

— Exécutez la commande `PATH=` (équivalente à `PATH=''`, c'est-à-dire avec une chaîne vide à droite du `=`), puis essayez `ls` et `cd`. Expliquez ce qu'il se passe.

Avec un `PATH` vide, le shell ne trouve plus les commandes externes comme `ls`. On peut encore les exécuter en entrant leur chemin complet comme `/bin/ls`. Les commandes internes du shell comme `cd` sont encore disponibles.

IV.4 Préserver une session de la déconnexion avec `screen`

Un scénario assez classique :

— Un utilisateur lance une commande sur son serveur

— L'heure tourne, l'utilisateur doit rentrer chez lui

— De chez lui, l'utilisateur se reconnecte à son serveur via SSH, et aimerait reprendre la main sur ce qu'il a démarré avant de partir.

La commande `screen` (ou un de ses petits frères comme `tmux` ou le couple `dvtm/dtach`) permet de répondre à ce problème (et bien d'autres comme la possibilité de découper votre terminal en sous-fenêtres).

— Dans un terminal lancé sur votre VM, entrez la commande `screen`, et validez le message d'accueil avec Entrée si besoin.

— Entrez quelques commandes : tout se passe normalement.

— Entrez Control-a puis d. Vous devriez revenir au shell initial, mais ce que vous avez démarré dans screen tourne toujours. Si un calcul est en cours, le calcul continue.

— Entrez la commande `screen -r` : votre environnement screen est de retour.

— Refaites Control-a puis d. Déconnectez-vous complètement de la VM.

— (Pour respecter strictement le sénario ci-dessus, il faudrait rentrer chez vous ici ...)

— Reconnectez-vous via SSH et faites `screen -r` : votre environnement est encore là.

Une alternative est la commande `nohup` qui permet de lancer une commande qui survit à la fin de la session de l'utilisateur courant (mais ne permet pas de reprendre la main facilement dessus).