Rapport TP1 : Installation Gentoo

DRISSI Mohamed Reda
reda-mohamed@isty.uvsq.fr

8 octobre 2018

Table des matières

I Réponses

I.1 Exercice 1

Création de machine virtuelle.

I.2 Exercice 2

Téléchargement de l'image x64 depuis le site officiel.

I.3 Exercice 3

```
$ fdisk /dev/sda
```

Partitionnement du disque .
Nous choisissons :
— boot : 100Mo en ext2.
— swap : 256Mo
— / : 6Go en ext4
— home : 5.7Go en ext4

I.4 Exercice 4

```
$ mkfs.ext2 -L boot           /dev/sda1
$ mkfs.ext4 -L /    -T small  /dev/sda3
$ mkfs.ext4 -L home -T small  /dev/sda4
$ mkswap /dev/sda2
```

Formatte les partitions en leur attribuant un label, puisque nos partitions sont relativement petites, nous quadruplons le nombre d'inodes pour ne pas rencontrer de problèmes par la suite. (Au lieu d'un inode tous les 16Ko, maintenant c'est un inode tous les 4Ko)

I.5 Exercice 5

```
$ mount /dev/sda4 /mnt/gentoo
$ cd /mnt/gentoo
$ mkdir boot home
$ mount /dev/sda1 /mnt/gentoo/boot
$ mount /dev/sda4 /mnt/gentoo/home
$ swapon /dev/sda2
```

Montage des partitions et activation du swap.

I.6 Exercice 6

```
$ links https://www.gentoo.org/downloads/mirrors/
```

Télécharger le tarball stage3.

I.7 Exercice 7

```
$ tar xvf <tarball> --xattrs-include='*.*' --numeric-owner
```

Décompresser le tarball stage3.

I.8 Exercice 8

```
$ mirrorselect -i -o >> /mnt/gentoo/etc/portage/make.conf
```

Choix d'un mirroir rapide.

```
$ mkdir --parents /mnt/gentoo/etc/portage/repos.conf
$ cp /mnt/gentoo/usr/share/portage/config/repos.conf
  /mnt/gentoo/etc/portage/repos.conf/gentoo.conf
```

Configuration du repository ebuild de gentoo

```
$ cp --dereference /etc/resolv.conf /mnt/gentoo/etc/
$ mount --types proc /proc /mnt/gentoo/proc
$ mount --rbind /sys /mnt/gentoo/sys
$ mount --make-rslave /mnt/gentoo/sys
$ mount --rbind /dev /mnt/gentoo/dev
$ mount --make-rslave /mnt/gentoo/dev
```

Copie des données du DNS et montage des filesystèmes nécessaires.

```
$ chroot /mnt/gentoo /bin/bash
$ source /etc/profile
$ export PS1=(chroot) $PS1
```

Chroot vers le nouveau système.

```
$ emerge-webrsync
```

Configure portage, et installe le dernier snapshot depuis internet.

I.9 Exercice 9

Création du fichier /etc/locale.gen avec les locales dont on a besoin (ici fr_FR.UTF-8).

```
$ locale-gen
```

pour générer les locales spécifées dans /etc/locale.gen

```
$ locale -a
```

pour vérifier que les locales ont bien été générées.

```
$ eselect locale list
```

affiche les locales disponibles qu'on peut appliquer à tout le système.

```
$ eselect locale <valeur>
```

permet de choisir la locale <valeur> parmis les locales disponibles qui ont été affichés par la commande précédente.

```
$ env-update && source /etc/profile
```

permet de recharger l'environement.

```
$ date
```

permet de vérifier que l'heure locale est bien configurée. Dans mon cas, c'était bien le cas, sinon il faut utiliser la commande ntpd pour bien configurer, ou bien timedatectl (pour ceux qui ont installé systemd).

Une autre méthode pour configurer le fuseau horaire serait avec "timezone-data" :

```
$ echo "Europe/Paris" > /etc/timezone
$ emerge --config sys-libs/timezone-data
```

Pour avoir la liste des timezones disponibles :

```
$ ls /usr/share/zoneinfo/
```

Éditer la varibale "hostname" du fichier /etc/conf.d/hostname pour changer le nom d'hôte.

```
$ ifconfig
```

Permet de trouver le nom de l'interface réseau à configurer (dans le cas d'une VM cela sera ethX / enp0sX)

```
$ emerge --ask net-misc/dhcpcd
$ dhcpd <interface>
```

Lance la configuration automatique du protocole DHCP de l'interface choisie.

Éditer le fichier /etc/fstab pour configurer le montage automatique au démarrage des partitions.

I.10 Exercice 10

```
$ emerge --ask sys-process/htop
```

Installe htop

I.11 Exercice 11

```
$ emerge --ask sys-kernel/gentoo-sources
```

Télécharge les sources du noyau

I.12 Exercice 12

```
$ emerge --ask sys-apps/pciutils
```

Installe le package pciutils qui regroupe plusieurs outils qui permettent d'intéragir avec le bus PCI.

```
$ lspci
```

Liste le matériel de la machine, ceci est important puisqu'il est impossible de bien configurer le kernel dans certains cas sans connaître sans matériel, (dans la plupart des cas, avec du matériel récent, le kernel pourra être configuré automatiquement).

I.13 Exercice 13

```
$ cd /usr/src/linux && make menuconfig
```

Lance le menu de configuration, ici il faut activer/désactiver les options de compilation.

I.14 Exercice 14

```
$ make -j4 && make modules_install
$ make install
```

Compilation et installation du kernel, avec un maximum de 4 tâches parallèles autorisées.
Il est possible de laisser l'outil "genkernel" configurer automatiquement les options de compilation. Cependant cette option prendra du temps, puisque l'outil prends compte de la plupart des machines.

```
$ emerge sys-boot/grub:2
$ grub-install /dev/sda
$ grub-mkconfig -o /boot/grub/grub.cfg
```

Installe et configure le `grub`

I.15 Exercice 15

```
$ passwd
```

Écrase le mot de passe du root

```
$ emerge --ask app-admin/sysklogd
$ emerge --ask app-admin/logrotate
$ rc-update add sysklogd default
$ rc-update add logrotate default
```

Installe les outils de gestion de logs, et les configure pour être lancés au démarrage.

```
$ emerge --ask sys-process/cronie
$ emerge --ask sys-apps/mlocate
$ rc-update add cronie default
$ rc-update add sshd default
```

Installation et configuration du cron daemon, indexeur de fichiers, et l'accès à distance via ssh.

I.16 Exercice 16

```
$ logout
```

Sortir du chroot.

```
$ umount -l /mnt/gentoo/dev{/shm,/pts,}
$ umount -R /mnt/gentoo
$ reboot
```

Démonte les partitions et redémarre.

I.17 Exercice 17

Après redémarrage nous nous connectons en tant que root

```
$ useradd -a -G sudo,wheel -s /bin/bash red
$ passwd red
$ emerge app-admin/sudo
```

Ajoute un nouvel utilisateur red, lui donne un mot de passe, puis la possibilité d'exécuter des commandes en mode sudo. Puis on rajoute la ligne "%sudo ALL=(ALL :ALL) ALL" au fichier /etc/sudoers avec la commande $ visudo .

I.18 Exercice 18

```
$ emerge sys-fs/quota
$ /etc/init.d/quota start
$ rc-update add quota default
$ groupadd quotagroup
$ gpasswd --add red quotagroup
```

Installe et configure les quotas de disques.
Il faut ajouter l'option "usrquota" à la ligne concernant la partition à laquelle nous voulons limiter l'accès (ici /dev/sda4 aka /home) sur /etc/fstab.

```
$ mount -o remount /home
```

Démonte, applique les changements puis remonte la partition `home`

```
$ quotacheck --user --group --create-files /home
```

Création des fichiers de quotas nécessaires.

```
$ quotaon --print-state --all
```

Affiche si les quotas sont activés ou pas.
Les quotas fonctionnent par blocs et par inode, la taille de chaque bloc est de 1Kb (1024 octets) et par des limites soft et hard. Nous avons besoin dans notre cas de limiter les blocs à 200Mb donc 204800 blocs, et aucune limite sur les inodes.

```
$ setquota -u red 204000 204800 0 0 /home
```

Maintenant nous allons essayer de tester cela :

```
$ su red
$ cd && dd if=/dev/zero of=f.out bs=210M count=1
```

Nous obtenons une erreur après que 200Mb soient écrites.

I.19 Exercice 19

```
$ su red
$ cd && mkdir usr && cd usr
$ wget <hwloc-download-link>
$ tar -xvf hwloc-2.0.2.tar.bz2
$ cd hwloc-2.0.2
$ ./configure --prefix=$HOME/usr
$ make -j4 && make install
```

Télécharge et installe hwloc dans le chemin /home/$USER/usr

I.20 Exercice 20

```
$ echo "export PATH=$PATH:/home/$USER/usr/bin"
  >> ~/.bashrc && source ~/.bashrc
```

Permet de rajouter /home/$USER/usr/bin aux chemins de la variable "$PATH" et donc permet d'exécuter les binaires de /home/$USER/usr/bin sans devoir rentrer tout le chemin.