

II. Compte rendu TP2

Table des matières

II. Compte rendu TP2.....	1
1. Installation de la machine	2
a. Par VirtualBox	2
b. Par RemoteLabz	2
2. Installation de LXC.....	3
3. Premier conteneur	3
a. Créer un invité Ubuntu	3
b. Récupérer la liste des conteneurs utilisables sur la machine	3
c. Démarrer le conteneur	3
d. Récupérer sur l'hôte diverses informations.....	3
e. Après s'être connecté à la console du conteneur vérifier	4
f. Se déconnecter du conteneur et lancer dans l'invité les commandes précédentes depuis l'hôte. Puis arrêter le conteneur.	4
4. Limitation de ressources en ligne de commande	4
a. Lancer le conteneur, pendant son exécution modifier les ressources comme suit :	4
b. Vérifier les modifications à l'aide d'une commande système	5
5. Gestion du réseau en mode physique	5
a. Modifier la configuration du conteneur pour mettre le réseau en mode réseau physique. Puis démarrer le conteneur.	Erreur ! Signet non défini.
b. Donner l'ensemble des éléments de la configuration réseau sur l'hôte et sur l'invité	Erreur ! Signet non défini.
6. Installation d'un package dans le conteneur : installation d'un serveur Apache dans le conteneur	6
7. Limitation de ressources dans le fichier de configuration de l'invité LXC.....	6
a. Mémoire limitée à 256 Mo	7
b. Utilisation de 50% maximum du processeur	7
8. Scripting, écrire un script assurant	7
a. La création d'un nouveau conteneur avec les limitations de la question 3 <= ?	7
b. La modification du type de réseau	7
c. L'installation d'un serveur Apache	7
9. Modification du template de la machine Ubuntu pour que	7
a. Toutes les machines créées le soient avec les limites de 256Mo de mémoire et 50% maximum de processeur	7
b. Le package iputils soit présent dès la création d'un conteneur	7
10. Et seulement avec LXC ?	Erreur ! Signet non défini.

1. [Installation de la machine](#)

Nous pouvions faire le TP en passant par VirtualBox ou par le RemoteLabz. J'ai installé la machine avec les deux environnements pour m'entraîner à manipuler les deux.

a. Par VirtualBox

Il fallait que les 2 interfaces soient en DHCP. J'ai donc changé le fichier de configuration Netplan :

```
sudo nano /etc/netplan/00-installer-config.yaml
network
  ethernets:
    enp0s3:
      dhcp4: true
    enp0s8:
      dhcp4: true
  version 2
```

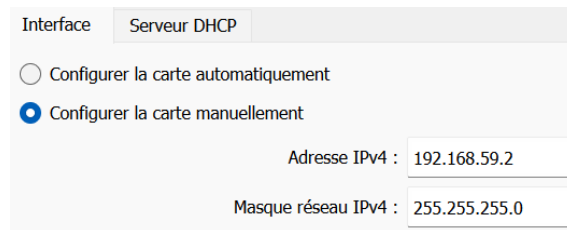
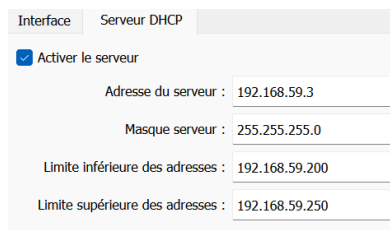
```
netplan apply
systemctl restart systemd-networkd
```

Concernant le serveur SSH, il était déjà installé puisque j'ai repris la machine du TP1.

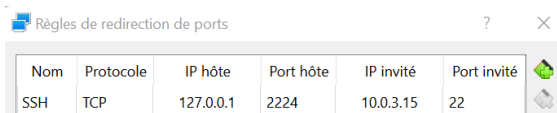
[Configuration de l'interface privée \(192.168.59.200/24\)](#)

Dans VirtualBox j'ai changé, dans Fichier > Gestionnaire de Réseau Hôte, la configuration de mon « VirtualBox Host-Only Adapter » afin de respecter les propriétés suivantes :

- Adresse du serveur DHCP : 192.168.59.3/24
- Plage d'adresses distribuées : de 200 à 250



[Configuration de l'interface NAT \(10.0.3.15/24\)](#)



Nom	Protocole	IP hôte	Port hôte	IP invité	Port invité
SSH	TCP	127.0.0.1	2224	10.0.3.15	22

J'ai activé le forward de port sur cette interface pour rendre possible la connexion SSH. Pour plus de détails cf. le TP1.

Pour tester, j'ai réussi à me connecter aux deux interfaces en SSH depuis mon hôte :

```
ssh jade@192.168.59.200
ssh jade@127.0.0.1 -p 2224
```

Pour ce TP je suis passée par VirtualBox.

b. Par RemoteLabz

J'ai téléchargé le fichier OpenVPN que j'ai inséré à OpenVPN GUI. Une fois la machine lancée j'ai configuré le réseau de l'invité Ubuntu avec les informations fournies dans le Lab RT0702 de RemoteLabz.

```
sudo nano /etc/netplan/00-installer-config.yaml
network
  ethernets:
    ens3:
      addresses: [10.11.3.178/28]
      gateway4: 10.11.3.190
      nameservers:
        addresses: [8.8.8.8]
      dhcp4: no
```

version 2

```
netplan apply
shutdown -r 0
```

Avec la commande ping j'ai pu vérifier que la connexion à Internet fonctionnait, ce qui était le cas. Il me manquait seulement la connexion SSH. J'ai vérifié l'état du service et j'ai rajouté l'utilisateur « toto » comme utilisateur autorisé :

```
sudo systemctl status ssh
sudo ufw allow ssh
sudo nano /etc/ssh/sshd_config
AllowUsers toto
```

Enfin, j'ai vérifié la connexion SSH depuis mon hôte : `ssh toto@10.11.3.178`. Ceci fonctionnait.

2. [Installation de LXC](#)

J'ai téléchargé LXC après avoir mis à jour ma machine :

```
sudo apt-get update
sudo apt install lxc lxc-templates -y
```

Pour vérifier j'ai l'installation j'ai tapé la commande : `lxc-checkconfig`

On remarque que « `ls -l /usr/share/lxc/templates` » permet de lister les templates pour créer des conteneurs.

3. [Premier conteneur](#)

a. [Créer un invité Ubuntu](#)

LXC permet aussi bien de créer des conteneurs tout petits qui comportent uniquement le minimum vital ou des conteneurs plus conséquents intégrant une distribution complète. Pour information si on ne dispose de pas beaucoup de place on peut utiliser le template « `lxc-busybox` », qui est 3 fois moins volumineux que le template « `lxc-ubuntu` ».

On retrouve les configurations respectives de chaque template dans « `/usr/share/lxc/config` ».

J'ai créé mon conteneur avec la commande « `lxc-create -n Ubuntu -t ubuntu` ».

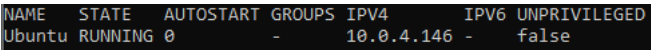
b. [Récupérer la liste des conteneurs utilisables sur la machine](#)

Par défaut « `lxc-create` » utilise une méthode de stockage « `dir` » qui met le rootfs (partition de système) du conteneur dans « `/var/lib/lxc` ».

c. [Démarrer le conteneur](#)

Pour démarrer le conteneur il suffit d'utiliser la commande « `lxc-start Ubuntu` ».

d. [Récupérer sur l'hôte diverses informations...](#)

Informations à récupérer :	Informations récupérées :
Informations d'exécution du conteneur	Pour avoir des informations d'exécution sur le conteneur j'ai tapé la commande « <code>lxc-ls -f</code> ». 
Etat du filesystem de l'hôte	Pour avoir des informations sur le filesystem de l'hôte j'ai tapé la commande « <code>lxc-info Ubun</code> ».

	<pre> root@rt0703ubuntu:/var/lib/lxc/Ubuntu# lxc-info Ubuntu Name: Ubuntu State: RUNNING PID: 23909 IP: 10.0.4.146 Link: veth6Axxv4 TX bytes: 1.65 KiB RX bytes: 1.88 KiB Total bytes: 3.54 KiB </pre>
Position du filesystem de l'invité sur l'hôte	« <code>cat /var/lib/lxc/Ubuntu/config</code> » permet d'afficher la configuration du conteneur.
Processus de l'hôte	« <code>ps auxef</code> » qui permet d'afficher les processus actifs du système.

e. Après s'être connecté à la console du conteneur vérifier ...

Lxc-attach : exécute une commande dans un conteneur. Si la commande n'est pas spécifiée, le shell par défaut de l'utilisateur en cours est exécuté.

`lxc-attach -n nom_machine -- la_commande`

Lxc-console : lance une console attachée à un TTY dans le conteneur spécifié. Si la connexion est coupée, la commande reprend la session là où elle en était. Attention, l'exécution d'une console implique un login. Le conteneur doit donc savoir gérer les logins. Par exemple, le template busybox ne sait pas faire, mais le template Debian sait. Il est possible de modifier le fichier « `/etc/inittab` » pour le personnaliser.

On peut également se connecter en SSH au conteneur avec la commande « `ssh ubuntu@10.0.4.146` ».

Ici je m'étais connecté avec la commande « `lxc-attach Ubuntu` ».

A vérifier ...	Remarques :
Vérifier l'accès et les droits dans l'invité	J'avais les droits root. J'ai remarqué que les droits du conteneur sont différents du host.
Etat du filesystem de l'invité, remarque (cf. partie d) ?	Pas de différences ?
Processus en exécution sur l'invité, remarque (cf. partie d) ?	Avec la commande « <code>ps auxef</code> » j'ai pu remarquer qu'il y avait moins de processus qui tournaient.

f. Se déconnecter du conteneur et lancer dans l'invité les commandes précédentes depuis l'hôte. Puis arrêter le conteneur.

Question à revoir... Je ne comprends pas.

4. Limitation de ressources en ligne de commande

a. Lancer le conteneur, pendant son exécution modifier les ressources comme suit :

Mémoire limitée à 256 Mo (= 256 000 000 octets)

Il faut noter qu'on ne peut pas descendre la limite de mémoire en dessous de ce dont le conteneur a besoin au moment où l'on souhaite la modifier. Il est donc préférable de vérifier la mémoire utiliser avant de changer cette limite avec la commande « `lxc-cgroup -n Ubuntu memory.current` ».

```

root@rt0703ubuntu:/var/lib/lxc/Ubuntu# lxc-cgroup -n Ubuntu memory.current
63561728

```

J'ai donc voulu changer la limite de la mémoire avec la commande « `lxc-cgroup -n Ubuntu memory.limit_in_bytes 256000000` ».

Malheureusement ça ne fonctionne pas je ne sais pas pourquoi...

```

root@rt0703ubuntu:/var/lib/lxc/Ubuntu# lxc-cgroup -n Ubuntu memory.limit_in_bytes 256000000
lxc-cgroup: Ubuntu: tools/lxc_cgroup.c: main: 111 Failed to assign '256000000' value to 'memory.limit_in_bytes' for '
Ubuntu'

```

Utilisation de 50% maximum du processeur

Je n'ai pas réussi à limiter la consommation en CPU d'un conteneur avec le paramètre « `cpu.weight` ». J'ai donc utilisé les commandes suivantes pour diminuer de 20% l'attribution du CPU au conteneur Ubuntu :

Ces commandes ne fonctionnent également pas... Je ne comprends pas pourquoi...

```
root@rt0703ubuntu:/var/lib/lxc/Ubuntu/rootfs/root# lxc-cgroup -n Ubuntu cpu.shares
lxc-cgroup: Ubuntu: tools/lxc_cgroup.c: main: 122 Failed to retrieve value of 'cpu.shares' for '/var/lib/lxc:Ubuntu'
```

b. Vérifier les modifications à l'aide d'une commande système

Je suppose que les modifications sont visibles dans « `cat /var/lib/lxc/Ubuntu/config` » normalement.

5. Gestion du réseau en mode physique

Depuis la version 2, le support réseau de LXC est basé sur un composant nommé « `lxc-net` » qui fait généralement parti du paquet LXC. Sa configuration s'articule autour du fichier « `etc/default/lxc-net` ».

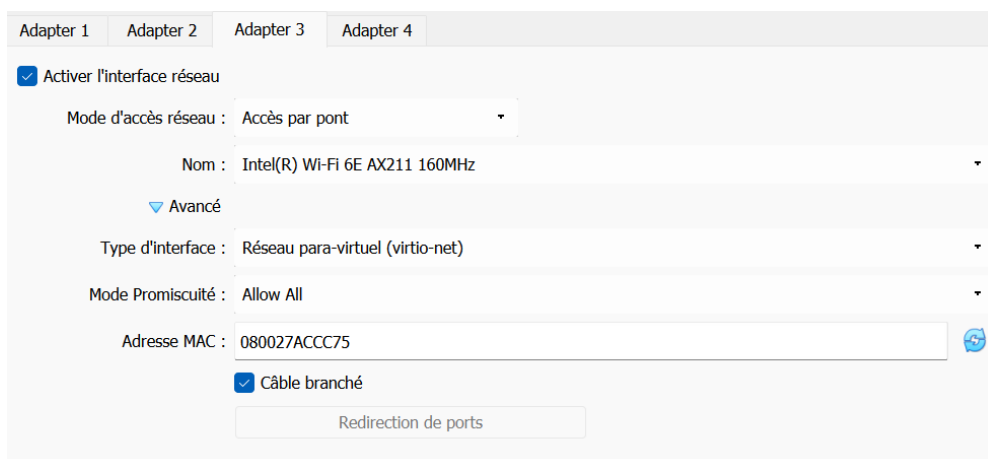
Dans un premier temps j'ai regardé les configurations réseaux qui étaient en place sur mon conteneur avec la commande « `cat /etc/default/lxc-net` ».

Cette commande m'a retourné le plan d'adressage utilisé qui est le 10.0.4.0/24 et que le DHCP est autorisé. Tout est géré par le service dnsmasq qui fait serveur DHCP et serveur DNS.

```
LXC_BRIDGE="lxcbr0"
LXC_ADDR="10.0.4.1"
LXC_NETMASK="255.255.255.0"
LXC_NETWORK="10.0.4.0/24"
LXC_DHCP_RANGE="10.0.4.2,10.0.4.254"
LXC_DHCP_MAX="253"
```

Pour cela j'ai dû mettre en place une 3^{ème} carte réseau, donc une 3^{ème} interface sur ma VM. En effet le mode physique utilise l'interface physique de l'hôte. On le précise dans « `lxc.network.link` ».

Dans un premier temps, j'ai activé sur Virtual Box une nouvelle interface en mode bridge que j'ai par la suite utilisé pour connecter directement mon conteneur.



Puis, il faut vérifier le nom de l'interface dans la VM avec la commande « `ip addr show` ». J'ai remarqué qu'une interface « `enp0s9` » s'était créée. Il a fallu que je modifie le fichier « `/etc/netplan/00-installer-config.yaml` » et que je relance les services pour que l'interface s'active.

Pour positionner le mode physique du conteneur j'ai modifié le fichier de configuration avec la commande suivante « `nano /var/lib/lxc/Ubuntu/config` »

Avant modifications	Après modifications
<pre># Network configuration lxc.net.0.type = veth lxc.net.0.link = lxcbr0 lxc.net.0.flags = up lxc.net.0.hwaddr = 00:16:3e:11:90:a3</pre> <p>Pour les différents types de réseaux cf. partie 8</p>	<pre>lxc.net.0.type = phys lxc.net.0.link = enp0s9 lxc.net.0.name = eth0 lxc.start.auto = 1 lxc.net.0.flags = up lxc.net.0.hwaddr = 08:00:27:ac:cc:75</pre> <p>A noter que l'adresse mac est l'adresse mac de mon interface enp0s9.</p>

J'ai ensuite redémarrer le conteneur en root avec « `lxc-start Ubuntu` ».

Quand j'exécutais une « `ip addr show` » je pouvais remarquer que l'interface n'était plus présente dans la VM. En effet, elle est maintenant directement utilisée par le conteneur Ubuntu. Le routeur wifi a attribué une IP par DHCP.

```
root@rt0703ubuntu:/home/jade# lxc-ls -f
NAME      STATE   AUTOSTART GROUPS IPV4      IPV6      UNPRIVILEGED
Ubuntu    RUNNING 1        -      192.168.1.10 2a01:e34:ec6e:24b0:a00:27ff:feac:cc75 false
```

On peut vérifier les configurations netplan avec « `cat /etc/netplan/10-lxc.yaml` » quand on est dans le conteneur (connexion avec « `lxc-attach Ubuntu` » par exemple).

6. [Installation d'un package dans le conteneur : installation d'un serveur Apache dans le conteneur](#)

J'étais dans le conteneur pour les commandes qui vont suivre.

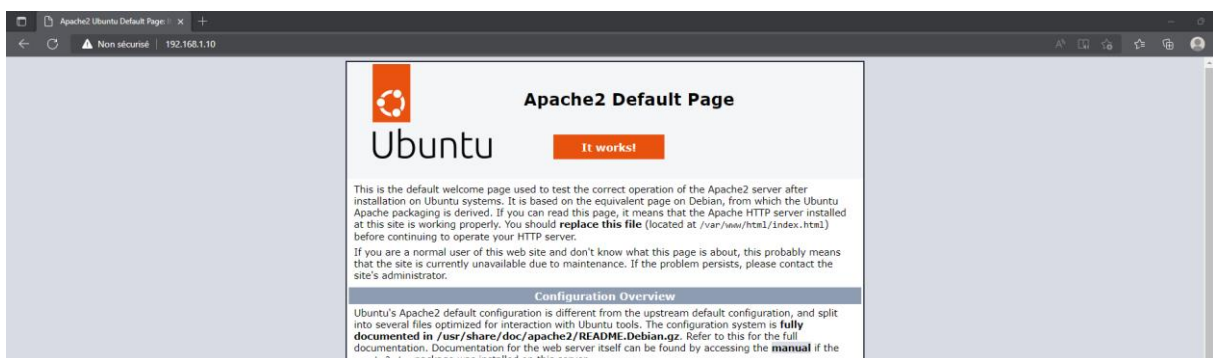
Commandes

Dans un premier temps j'ai mis à jour mon conteneur et télécharger Apache.

```
apt update
apt install apache2
service apache2 status
```

Adresse permettant d'accéder au serveur web et depuis quel hôte

Pour accéder au serveur web il suffit d'utiliser le protocole http et de mettre comme IP dans le navigateur celle du conteneur « `http://192.168.1.10` ».



7. [Limitation de ressources dans le fichier de configuration de l'invité LXC](#)

Je n'étais plus dans le conteneur pour les lignes suivantes. Pour rendre permanentes les limitations j'ai modifié le fichier de configuration avec la commande « `nano /var/lib/lxc/Ubuntu/config` ».

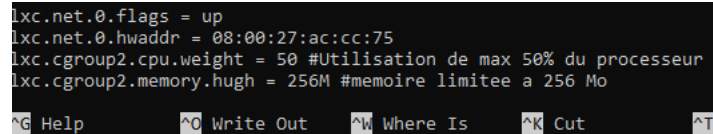
- a. Mémoire limitée à 256 Mo

J'ai rajouté la ligne : `lxc.cgroup2.cpu.weight = 50`

- b. Utilisation de 50% maximum du processeur

J'ai rajouté la ligne : `lxc.cgroup2.memory.hugh = 256M`

```
lxc.net.0.flags = up
lxc.net.0.hwaddr = 08:00:27:ac:cc:75
lxc.cgroup2.cpu.weight = 50 #Utilisation de max 50% du processeur
lxc.cgroup2.memory.hugh = 256M #memoire limitee a 256 Mo
```



8. Scripting, écrire un script assurant ...

PAS ENCORE TERMINE

- a. La création d'un nouveau conteneur avec les limitations vues précédemment
- b. La modification du type de réseau
- c. L'installation d'un serveur Apache

9. Modification du template de la machine Ubuntu pour que ...

- a. Toutes les machines créées le soient avec les limites de 256Mo de mémoire et 50% maximum de processeur
- b. Le package `iputils` soit présent dès la création d'un conteneur