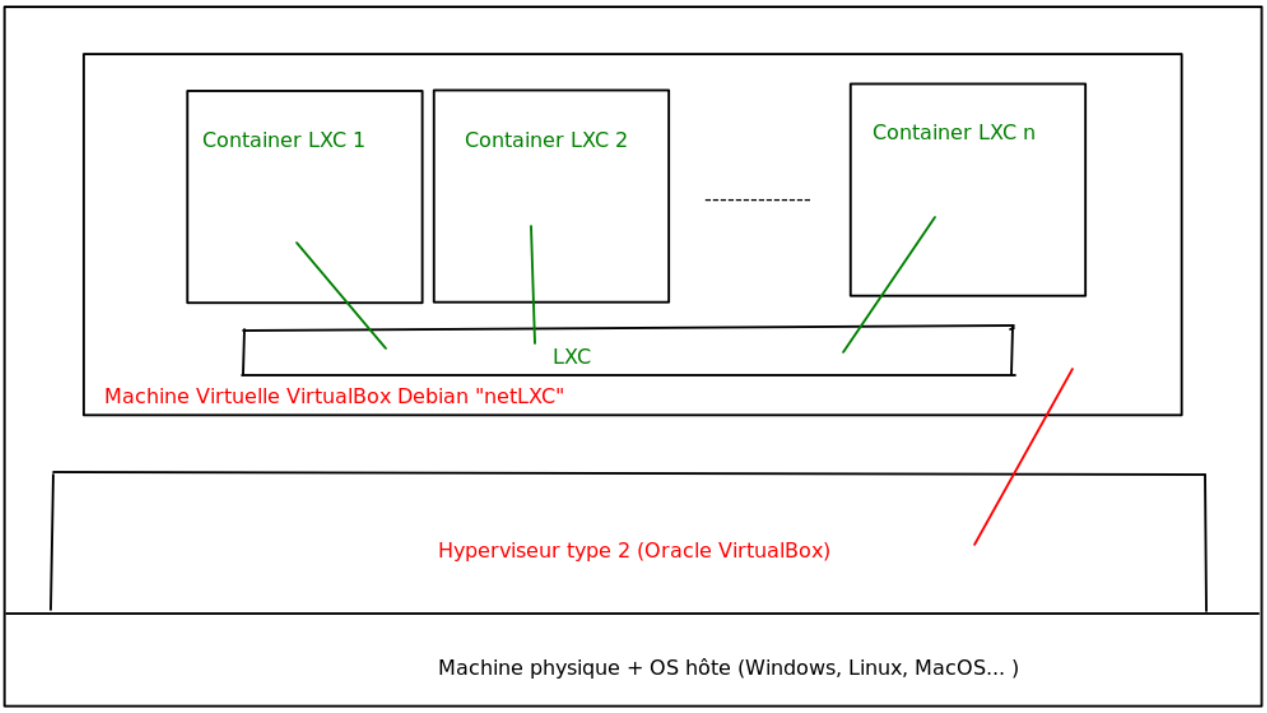
Construire une plate-forme netLXC de virtualisation de réseaux (version 09/2020)

Nous allons ici construire pas à pas une **appliance virtuelle d'apprentissage des réseaux** à base de LXC : netLXC.

Les objectifs

- maquetter rapidement un réseau à base de machines Linux,
- offrir un environnement de virtualisation convivial, facile à mettre en oeuvre, économe en ressources mais puissant aux étudiants,
- maquetter n'importe quel réseau depuis n'importe où ou presque... une terrasse de café conviendra très bien si l'on a un embryon de réseau... ! 🙂,
- travailler sur un produit de virtualisation léger, pérenne et utilisé en environnement de production : LXC,
- être toujours assuré de travailler avec des machines dont le noyau Linux est le plus récent : **les containers LXC utilisent le noyau Linux de la machine hôte.**

L'architecture et les briques logicielles mises en oeuvre



Tout est décrit ici ! :-)

La construction

Le socle

Le socle du produit est une machine virtuelle VirtualBox à base de Debian up-to-date.

netLXC sur Debian 10 (Buster)

Attention : cette installation a été réalisée sur Debian 10 implantée sur Oracle VirtualBox.

Notamment,la configuration réseau des containers sollicite le serveur DHCP de la machine hôte Virtualbox. **Dans un autre contexte, il vous faudra adapter cette documentation** notamment au niveau de la gestion des connexions réseaux (ce qui peut être plus complexe).

```
root@Debian10netLXC:/home/btssio# lsb_release -a
No LSB modules are available.
Distributor ID: Debian
Description:    Debian GNU/Linux 10 (buster)
Release:    10
Codename:   buster
root@Debian10netLXC:/home/btssio#
```

Installation des logiciels applicatifs

- l'analyseur de trame **wireshark**,
- l'interface d'administration **webmin**,
- une **calculette de programmation** (changements de base, opérations logiques),
- une **calculette IP** en mode commande (**ipcalc**), et en mode graphique (gip)
- **net-tools**,
- **netdiscover**,
- **traceroute**, **nmap** et **zenmap**,
- **ssh**,
- **netcat**,
- **gparted**...

Installation du produit LXC

- Ici, une présentation très simple de LXC

```
root@netLXC:/home/btssio# apt install lxc bridge-utils vlan libvirt-daemon-
system libvirt-dev libvirt-clients debootstrap vde2 openvswitch-switch
```

Vérification de la bonne installation de LXC !

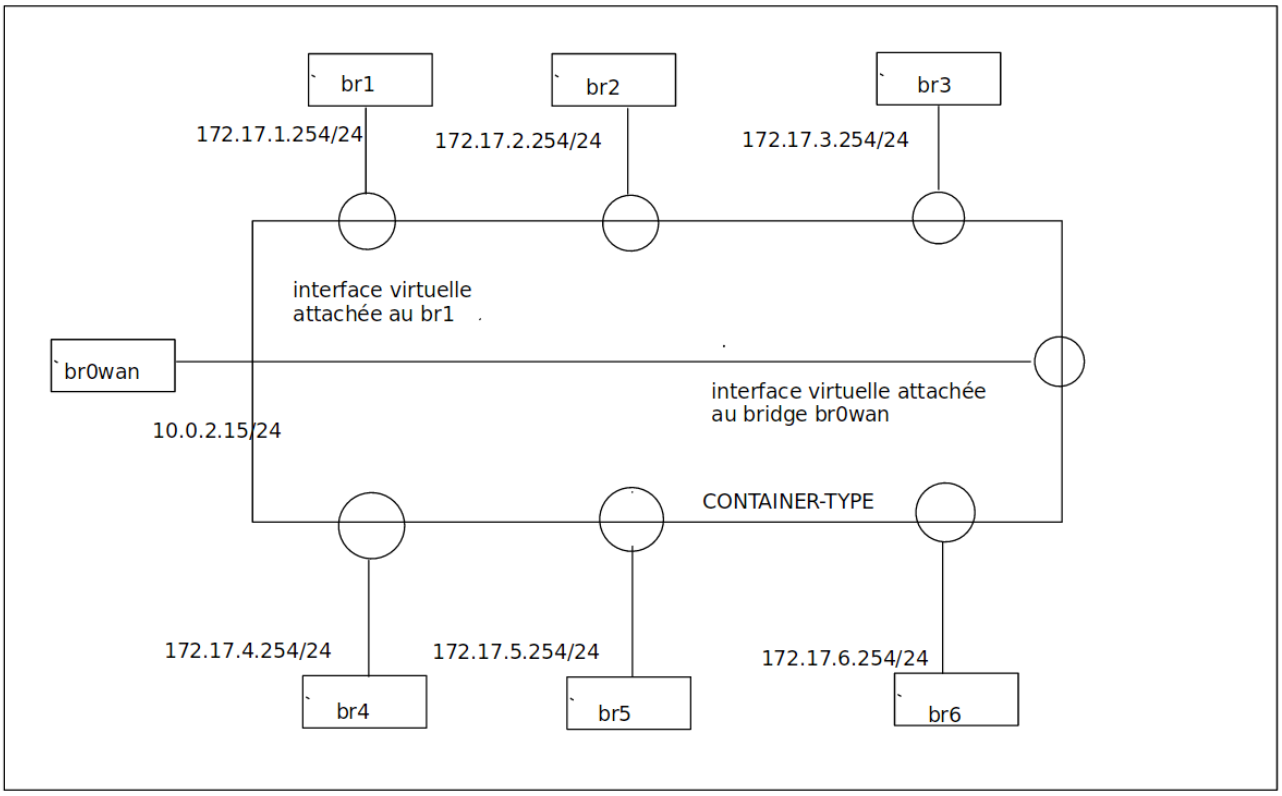
```
root@netLXC:/home/btssio# lxc-checkconfig
```

```
Kernel configuration not found at /proc/config.gz; searching...
Kernel configuration found at /boot/config-4.9.0-7-amd64
--- Namespaces ---
Namespaces: enabled
Utsname namespace: enabled
Ipc namespace: enabled
Pid namespace: enabled
User namespace: enabled
Network namespace: enabled

--- Control groups ---
Cgroup: enabled
Cgroup clone_children flag: enabled
Cgroup device: enabled
Cgroup sched: enabled
Cgroup cpu account: enabled
Cgroup memory controller: enabled
Cgroup cpuset: enabled

--- Misc ---
Veth pair device: enabled
Macvlan: enabled
Vlan: enabled
Bridges: enabled
Advanced netfilter: enabled
CONFIG_NF_NAT_IPV4: enabled
CONFIG_NF_NAT_IPV6: enabled
CONFIG_IP_NF_TARGET_MASQUERADE: enabled
CONFIG_IP6_NF_TARGET_MASQUERADE: enabled
CONFIG_NETFILTER_XT_TARGET_CHECKSUM: enabled
FUSE (for use with lxcfs): enabled

--- Checkpoint/Restore ---
checkpoint restore: enabled
CONFIG_FHANDLE: enabled
CONFIG_EVENTFD: enabled
CONFIG_EPOLL: enabled
CONFIG_UNIX_DIAG: enabled
CONFIG_INET_DIAG: enabled
CONFIG_PACKET_DIAG: enabled
CONFIG_NETLINK_DIAG: enabled
File capabilities: enabled

Note : Before booting a new kernel, you can check its configuration
usage : CONFIG=/path/to/config /usr/bin/lxc-checkconfig

root@netLXC:/home/btssio#
```

Configuration du réseau de la machine hôte netLXC

On va créer quelques bridges virtuels qui nous serviront par la suite pour nos expérimentations...

- bridge **br0wan** qui servira de passerelle vers l'extérieur (via la machine hôte netLXC),
- bridges “bruts de fonderie” : **br1**, **br2**, **br3**, **br4**, **br5**, **br6** pour les réseaux locaux.



MV netLXC VirtualBox

Pour créer un bridge, **le paquet “*bridge-utils*” doit être préalablement installé.**

Créer un bridge est très simple... A titre d'exemple, ici, on créé le bridge **br5** :

```
root@netLXC:/var/lib/lxc# brctl addbr br5
```

La commande “**brctl show**” nous montre la liste des bridges créés :

```
root@netLXC:/home/btssio# brctl show
bridge name bridge id        STP enabled    interfaces
br0wan      8000.0800274ad2c0    no        eth0
br1     8000.000000000000    no
br2     8000.000000000000    no
br3     8000.000000000000    no
br4     8000.000000000000    no
br5     8000.000000000000    no
br6     8000.000000000000    no
root@netLXC:/home/btssio#
```

On va maintenant configurer “en dur” les bridges en modifiant le fichier **/etc/network/interfaces**

```
root@netLXC:/home/btssio# cat /etc/network/interfaces
# This file describes the network interfaces available on your system
# and how to activate them. For more information, see interfaces(5).

source /etc/network/interfaces.d/*

# The loopback network interface
auto lo
iface lo inet loopback


# bridge br0wan, notre passerelle vers l'extérieur, vers l'Internet et peut-
être au delà...
auto br0wan
iface br0wan inet dhcp

        bridge_ports enp0s3
        bridge_fd 0
        bridge_maxwait 0


# bridge br1 pour expérimenter...
auto br1
iface br1 inet static
       bridge_ports vbr1
       bridge_fd 0
       bridge_maxwait 0




# bridge br2 pour expérimenter...
auto br2
iface br2 inet static
       bridge_ports vbr2
       bridge_fd 0
       bridge_maxwait 0



# bridge br3 pour expérimenter...
auto br3
iface br3 inet static
       bridge_ports vbr3
       bridge_fd 0
       bridge_maxwait 0


# bridge br4 pour expérimenter...
auto br4
iface br4 inet static
```

```
       bridge_ports vbr4
       bridge_fd 0
       bridge_maxwait 0

# bridge br5 pour expérimenter...
auto br5
iface br5 inet static
       bridge_ports vbr5
       bridge_fd 0
       bridge_maxwait 0

# bridge br6 pour expérimenter...
auto br6
iface br6 inet static
       bridge_ports vbr6
       bridge_fd 0
       bridge_maxwait 0


auto enp0s3
iface enp0s3 inet dhcp
root@netLXC:/home/btssio#
```

On relance le réseau :

```
root@netLXC:/home/btssio# systemctl restart networking
```

Les bridges sont bien actifs et configurés IP…

```
root@netLXC:/home/btssio# ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group
default qlen 1
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast
master br0wan state UP group default qlen 1000
    link/ether 08:00:27:8e:ef:f1 brd ff:ff:ff:ff:ff:ff
    inet 10.0.2.15/24 brd 10.0.2.255 scope global enp0s3
       valid_lft forever preferred_lft forever
10: lxcbr0: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc noqueue state
DOWN group default qlen 1000
    link/ether 00:16:3e:00:00:00 brd ff:ff:ff:ff:ff:ff
    inet 10.0.3.1/24 scope global lxcbr0
       valid_lft forever preferred_lft forever
11: br0wan: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state
UP group default qlen 1000
    link/ether 08:00:27:8e:ef:f1 brd ff:ff:ff:ff:ff:ff
```

```
    inet 10.0.2.15/24 brd 10.0.2.255 scope global br0wan
       valid_lft forever preferred_lft forever
    inet6 fe80::a00:27ff:fe8e:eff1/64 scope link
       valid_lft forever preferred_lft forever
12: br1: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state
UNKNOWN group default qlen 1000
    link/ether ba:9b:f8:42:eb:ae brd ff:ff:ff:ff:ff:ff
    inet 172.17.1.254/24 brd 172.17.1.255 scope global br1
       valid_lft forever preferred_lft forever
    inet6 fe80::b89b:f8ff:fe42:ebae/64 scope link
       valid_lft forever preferred_lft forever
13: br2: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state
UNKNOWN group default qlen 1000
    link/ether 06:f1:b2:a2:3a:71 brd ff:ff:ff:ff:ff:ff
    inet 172.17.2.254/24 brd 172.17.2.255 scope global br2
       valid_lft forever preferred_lft forever
    inet6 fe80::4f1:b2ff:fea2:3a71/64 scope link
       valid_lft forever preferred_lft forever
14: br3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state
UNKNOWN group default qlen 1000
    link/ether ee:f5:57:e9:2a:25 brd ff:ff:ff:ff:ff:ff
    inet 172.17.3.254/24 brd 172.17.3.255 scope global br3
       valid_lft forever preferred_lft forever
    inet6 fe80::ecf5:57ff:fee9:2a25/64 scope link
       valid_lft forever preferred_lft forever
15: br4: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state
UNKNOWN group default qlen 1000
    link/ether 42:12:8c:6e:e6:43 brd ff:ff:ff:ff:ff:ff
    inet 172.17.4.254/24 brd 172.17.4.255 scope global br4
       valid_lft forever preferred_lft forever
    inet6 fe80::4012:8cff:fe6e:e643/64 scope link
       valid_lft forever preferred_lft forever
16: br5: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state
UNKNOWN group default qlen 1000
    link/ether be:49:1f:da:64:8b brd ff:ff:ff:ff:ff:ff
    inet 172.17.5.254/24 brd 172.17.5.255 scope global br5
       valid_lft forever preferred_lft forever
    inet6 fe80::bc49:1fff:feda:648b/64 scope link
       valid_lft forever preferred_lft forever
17: br6: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state
UNKNOWN group default qlen 1000
    link/ether 0e:99:63:3a:4f:93 brd ff:ff:ff:ff:ff:ff
    inet 172.17.6.254/24 brd 172.17.6.255 scope global br6
       valid_lft forever preferred_lft forever
    inet6 fe80::c99:63ff:fe3a:4f93/64 scope link
       valid_lft forever preferred_lft forever
root@netLXC:/home/btssio#
```

On vérifie l'accès vers l'Internet ainsi que la résolution de noms...

```
root@netLXC:/home/btssio# host www.free.fr
```

```
www.free.fr has address 212.27.48.10
www.free.fr has IPv6 address 2a01:e0c:1::1
root@netLXC:/home/btssio#
```

La résolution de noms est ici assuré par le relais DNS de VirtualBox (10.0.2.3#53)

En cas de problème, tentez :

```
root@netLXC:/home/btssio# echo 'nameserver 208.67.222.222' >
/etc/resolv.conf
```

Les containers

Création d'un "super-template-deb" Debian

Ce container "**super-template-deb**" nous permettra de créer par clonage un "**template d'expérimentation**"

- Comment retrouver le nom de code de la distribution Debian ?

```
root@netLXC2017:/var/lib/lxc# lxc-create -n super-template-deb  -t debian --
--r buster
debootstrap est /usr/sbin/debootstrap
Checking cache download in /var/cache/lxc/debian/rootfs-jessie-amd64 ...
Copying rootfs to /var/lib/lxc/super-template-deb/rootfs...Generating
locales (this might take a while)...
  fr_FR.UTF-8... done
Generation complete.
insserv: warning: c

...

Current default time zone: 'Europe/Paris'
Local time is now:      Sun Dec 18 17:23:08 CET 2016.
Universal Time is now:  Sun Dec 18 16:23:08 UTC 2016.


root@netLXC2017:/var/lib/lxc#
```

Configuration "réseau" a minima du container "super-template-deb" et installation des paquets de logiciels qui vont bien...

Le principe d'une telle configuration est décrit ici et là ou encore là-bas. Nous passerons donc vite sur les explications de cette étape...

- Nous allons modifier le fichier de configuration du container "**super-template-deb**" : **/var/lib/lxc/super-template-deb/config** en rajoutant en fin de fichier les lignes suivantes :

```
##############################################################################
######
```

```
################################################################################
########
# CONNEXIONS RESEAU !!!!!
###################################
# br0wan : connexion vers l'extérieur !
# Filasse virtuelle qui attache le container au bridge br0wan (accès direct
à l'extérieur, à l'Internet... )
# Réseau de type Ethernet
#lxc.net.0.type = veth
# Cette interface est attachée au bridge virtuel br0wan de l'hôte
#lxc.net.0.link = br0wan
# L'interface virtuelle est nommée eth0wan (côté container)
#lxc.net.0.name = vers_br0wan
# Le port du bridge se nomme port1_br0wan (vu de l'hôte). Nom de la
connexion < 16 caractères
# Attention, ce port est UNIQUE sur un bridge donné...
#lxc.net.0.veth.pair = port1_br0wan
###################################
# br1
###################################
# Ici, on tire un câble entre le container et le bridge br1
#lxc.net.1.type = veth
#lxc.net.1.link = br1
#lxc.net.1.name = vers_br1
# Attention, ce port est UNIQUE sur un bridge donné...
#lxc.net.1.veth.pair = port1_br1
###################################
# br2
###################################
# Ici, on tire un câble entre le container et le bridge br2
#lxc.net.2.type = veth
#lxc.net.2.link = br2
#lxc.net.2.name = vers_br2
# Attention, ce port est UNIQUE sur un bridge donné...
#lxc.net.2.veth.pair = port1_br2
###################################
# br3
###################################
# Ici, on tire un câble entre le container et le bridge br3
#lxc.net.3.type = veth
#lxc.net.3.link = br3
#lxc.net.3.name = vers_br3
# Attention, ce port est UNIQUE sur un bridge donné...
#lxc.net.3.veth.pair = port1_br3
###################################
# br4
###################################
# Ici, on tire un câble entre le container et le bridge br4
#lxc.net.4.type = veth
#lxc.net.4.link = br4
#lxc.net.4.name = vers_br4
```

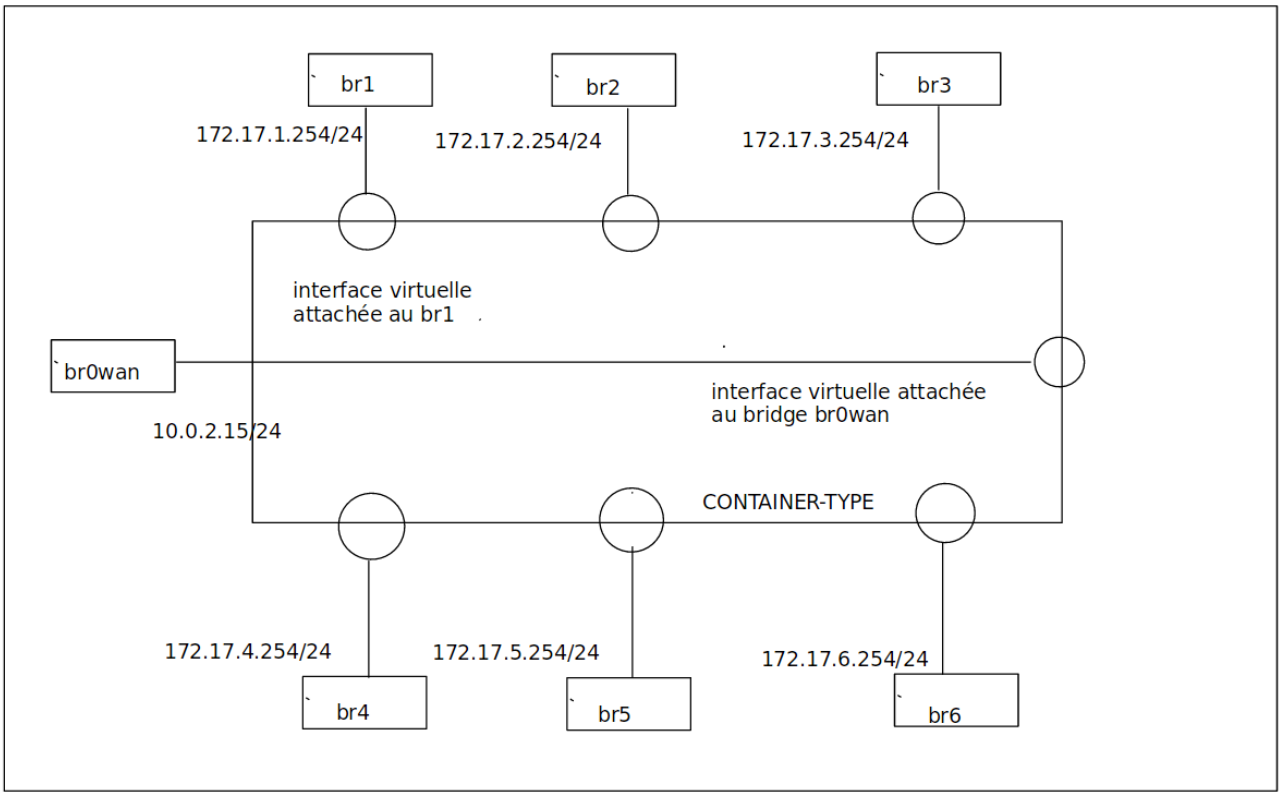
```
# Attention, ce port est UNIQUE sur un bridge donné...
#lxc.net.4.veth.pair = port1_br4
#################################
# br5
#################################
# Ici, on tire un câble entre le container et le bridge br5
#lxc.net.5.type = veth
#lxc.net.5.link = br5
#lxc.net.5.name = vers_br5
# Attention, ce port est UNIQUE sur un bridge donné...
#lxc.net.5.veth.pair = port1_br5
#################################
# br6
#################################
# Ici, on tire un câble entre le container et le bridge br6
#lxc.net.6.type = veth
#lxc.net.6.link = br6
#lxc.net.6.name = vers_br6
# Attention, ce port est UNIQUE sur un bridge donné...
#lxc.net.6.veth.pair = port1_br6
```

On en profite pour dé-commenter provisoirement les lignes suivantes :

```
...
...
# br0wan : connexion vers l'extérieur !
# Filasse virtuelle qui attache le container au bridge br0wan (accès direct
à l'extérieur, à l'Internet... )
# Réseau de type Ethernet
lxc.net.0.type = veth
# Cette interface est attachée au bridge virtuel br0wan de l'hôte
lxc.net.0.link = br0wan
# L'interface virtuelle est nommée eth0wan (côté container)
lxc.net.0.name = vers_br0wan
# Le port du bridge se nomme port1_br0wan (vu de l'hôte). Nom de la
connexion < 16 caractères
# Attention, ce port est UNIQUE sur un bridge donné...
lxc.net.0.veth.pair = port1_br0wan
...
```

On a attaché le container au bridge **br0wan** pour avoir “communication a minima” avec les serveurs de dépôts Debian...

Le principe de fonctionnement est le suivant : dans le fichier **/var/lib/lxc/super-template-deb/config sont pré-configurées 6 configurations** “prêtes à l'emploi“ (ou presque...) permettant de connecter le container sur l'un ou l'autre des bridges de la machine hôte.



MV netLXC VirtualBox

- On démarre le container...

```
root@netLXC:/home/btssio# lxc-start -n super-template-deb
```

- On s'attache au container “super-template-deb” :

```
root@netLXC:/home/btssio# lxc-attach -n super-template-deb
```

- On vérifie le fichier **/etc/network/interfaces**. On doit avoir quelque chose comme cela...

```
# root@super-template-deb:~# cat /etc/network/interfaces
# This file describes the network interfaces available on your system
# and how to activate them. For more information, see interfaces(5).

source /etc/network/interfaces.d/*

# The loopback network interface
auto lo
iface lo inet loopback

# bridge br0wan, notre passerelle vers l'extérieur, vers l'Internet et peut-
être au delà...
auto eth0wan
iface eth0wan inet dhcp
        bridge_fd 0
        bridge_maxwait 0
```

```
# Interface eth0 inchangée
#auto eth0
#iface eth0 inet dhcp

root@super-template-deb:~#
```

- On contrôle la résolution de noms (ici, normalement assurée par le relais DNS de Virtualbox...).
- On installe un système minimal...

```
root@super-template-deb:~# apt update && apt install sysvinit.core
```

- On ajoute le “simple” utilisateur “btssio” (mot de passe = btssio) :

```
root@super-template-deb:~# adduser btssio
```

- On arrête le container **depuis la machine hôte** par la commande infra :

```
root@netLXC2017:/home/btssio# lxc-stop -n super-template-deb  -k
```

- On redémarre le container :

```
root@netLXC2017:/home/btssio# lxc-start -n super-template-deb
```

- On installe les paquets “intéressants” 😃

```
root@super-template-deb:~# apt install locales dialog netbase net-tools
iproute2 openssh-server tcpdump ifupdown vim nano inetutils-ping traceroute
netcat nmap dnsutils ssh man bridge-utils vde2 iptables isc-dhcp-server
bind9
```

- Donnons un dernier coup de polish en évitant la tentative de démarrage des services “**isc-dhcp-server**” et “**bind9**”

```
root@super-template-deb:~# systemctl disable isc-dhcp-server
```

```
root@super-template-deb:~# systemctl disable bind9
```

A ce stade, notre super-template est OK ! 😃 On va pouvoir cloner comme un biologiste moyen ! 😎

Fabrication, par clonage, d'un "template d'expérimentation"

Nous allons ici fabriquer un “template d'expérimentation” à partir de notre “super-template” :

L'objectif de ce clonage est **de ne modifier EN AUCUN CAS le “super-template-deb”** qui nous permettra de refaire un container “template” d'expérimentation “propre” en cas de problème...

Clonage et configuration réseau du template "templex"

```
root@netLXC:/home/btssio# lxc-copy -n super-template-deb -N templex
Created container templex as copy of super-template-deb
```

```
root@netLXC:/home/btssio#
```

- Pour aller plus loin...
- Mise en pratique à l'aide du Quick Guide netLXC !

~~DISCUSSION~~