GNU/Linux Administration - Support #132

Lightweight VPS Server with LXC on Debian 7 Wheezy

07/01/2013 04:15 PM - Daniel Curtis

Status:	Closed	**Start date:**	07/01/2013
Priority:	High	**Due date:**	
Assignee:	Daniel Curtis	**% Done:**	100%
Category:	Jails / Container	**Estimated time:**	2.00 hours
Target version:		**Spent time:**	8.00 hours

Description

Why LXC?

So I'm doing web development, and I'm using Debian Wheezy as my development environment, which doesn't have the same version of software than stable, which is what we usually use as target servers. I used to use chroots for this, but I found them painful to manage, especially when running daemons on the same ports than on the host machine.

People like to use virtualization for this, such as VirtualBox (esp. with Vagrant) but I didn't want that since it forces you to start a whole virtual machine every time you want to develop. Running a virtual machine is quite a heavy process and they constantly use resources even if they don't do anything. The big plus of using containers is that you nearly don't have any performance hit, and your container doesn't take any resource to run. Only processes that run will use some resources. Not to mention my initial available computing architecture was PowerPC and many virtualization solutions just will not work due to the lack of virtualization support on the PowerPC processors.

The main drawback of using LXC is that you can only run systems that support the same kernel as your host. Basically it means that you won't be able to create a BSD, Solaris or Windows container on your Debian host. But you'll still be able to create any Linux container, whatever distribution you want to install.

I won't go in detail about what LXC is. If you want more information, check out this Wikipedia page: http://en.wikipedia.org/wiki/LXC.

New VPS Node Checklist

When adding a new VPS server, make sure to have the following puppet configuration parameters set:

1. Puppet Node SSL signed by the Puppet Master
2. Puppet Node defined
3. Puppet Classes Debian base, APT client, Kerberos client, LXC vServer, and Puppet client included
4. Unique Node /etc/fstab defined
5. Unique Node /etc/rc.local defined
6. Unique Node /etc/network/interfaces defined
7. Virtual Server template files defined

Cgroups configuration

Cgroups are a kernel feature that's needed for lxc to run properly. Start by adding the following to your /etc/fstab file:

```
cgroup  /sys/fs/cgroup  cgroup  defaults  0   0
```

Mount it:

```
sudo mount /sys/fs/cgroup
```

Installing the packages

We'll be using lxc to create and manage the containers, and libvirt to manage the network:

```
sudo apt-get install lxc libvirt-bin dnsmasq-base
```

First container creation

Do this for each container you want to create. Let's create our first container:

```
sudo lxc-create -n myfirstcontainer -t debian
```

Answer to the different questions (just go with the defaults, except when it asks you for a root password if you want to put a custom password).

Once the container is created (this takes some time because it needs to download all the base packages and install a base system), chroot in it and create some TTY devices. I found that the lxc-console command didn't work if I didn't create these devices first:

```
sudo chroot /var/lib/lxc/myfirstcontainer/rootfs
mknod -m 666 /dev/tty1 c 4 1
mknod -m 666 /dev/tty2 c 4 2
mknod -m 666 /dev/tty3 c 4 3
mknod -m 666 /dev/tty4 c 4 4
mknod -m 666 /dev/tty5 c 4 5
mknod -m 666 /dev/tty6 c 4 6
```

Exit the chroot (by pressing Ctrl-D or by typing exit) and try to start it:

```
sudo lxc-start -n myfirstcontainer
```

You'll get some warnings and permission errors. You can safely ignore them. Now open a new shell and try to open a console to the container:

```
sudo lxc-console -n myfirstcontainer
```

You should be brought to a login prompt. Log in as root using the password you provided during the container creation phase. At this point you already have a fully functional container. The next step is to setup an ip address for your container so you can ssh into it, make http requests, etc. To shut down the container cleanly, use the halt command like on your host machine (if it's stuck and the halt command doesn't work, use the stop command):

```
sudo lxc-stop -n myfirstcontainer
```

Network configuration

We'll be using libvirt to manage the network bridge that we'll create for our containers. Start by enabling the default network configuration that comes with libvirt (you can skip the -c option if you don't have any other virtualization solution installed, such as VirtualBox):

```
sudo virsh -c lxc:/// net-define /etc/libvirt/qemu/networks/default.xml
sudo virsh -c lxc:/// net-start default
sudo virsh -c lxc:/// net-autostart default
```

The default network libvirt uses is 192.168.122.0/24. If you want to change it, run:

```
virsh -c lxc:/// net-edit default
```

and adapt the settings accordingly.

The last line will allow the interface to be automatically started on boot, so you don't have to do it manually. Now if you run ifconfig you'll see you have a new interface named virbr0.

Now that our bridge is defined, we need to set the network interface of our container. To do this, add the following lines to /var/lib/lxc/myfirstcontainer/config (feel free to replace the ip address to match your bridge configuration):

    lxc.network.type = veth

```
lxc.network.flags = up
lxc.network.link = virbr0
lxc.network.ipv4 = 192.168.122.2/24
```

Start your machine with the lxc-start command, lxc-console into it, run ifconfig and check that you have an eth0 interface defined. I still don't know why but it looks like the routing won't work by default. To solve this, you can edit your /etc/rc.local file and add this before the exit line (adapt it if your network configuration is different):

```
ip route add default via 192.168.122.1
```

or adding a gateway to the container configuration:

```
lxc.network.ipv4.gateway = 192.168.122.1
```

Now you can either restart your container or execute this command in a shell, and you should be able to connect to your container from your host machine.

SSH

I don't know why but the base ssh install seems to have a problem with the keys generation, making it unusable. To fix it, run lxc-console to go in your container and reinstall ssh:

```
apt-get install --reinstall openssh-server
```

Now you should be able to ssh to your container from your host:

```
ssh root@192.168.122.2
```

Mount Points

You'll probably want to mount some of your host directories in your container. Here's an example to mount the directory /home/you/directory_to_mount to /srv/mountpoint. Add the following lines to /var/lib/lxc/myfirstcontainer/config:

```
sudo mkdir /var/lib/lxc/myfirstcontainer/rootfs/srv/mountpoint
lxc.mount.entry = /home/you/directory_to_mount /var/lib/lxc/myfirstcontainer/rootfs/srv/mountpoint
 none defaults,bind 0 0
```

Automatically start your containers

To automatically start your containers at boot, all you have to do is to put symlinks to your containers config files in the /etc/lxc/auto/ directory. For example for your previously created container (it's important that your symlink has the same name as your container):

```
sudo ln -s /var/lib/lxc/myfirstcontainer/config /etc/lxc/auto/myfirstcontainer
```

History

#1 - 07/01/2013 04:16 PM - Daniel Curtis

- Description updated

#2 - 07/03/2013 12:34 PM - Daniel Curtis

There were problems setting up virtual containers using LXC and libvirt. The simplest way I could resolve the problem was to create a self-contained virtual network on the VPS host running the linux container. The resolutions was as follows:

1. Create a routed virtual network using libvirt
2. Configure linux containers network configuration accordingly
3. Add firewall/iptables rule to VPS host forwarding traffic to the virtual network
4. Enable IP Forwarding on VPS host
5. Setup a gateway and network route on the DMZ router (for internal networking)
6. Add firewall rule to DMZ router ALLOWING traffic from virtual network to Internet (for external networking)

Create a routed virtual network using libvirt

- lxc-route.xml

```
<class code="xml">
<network>
  <name>lxc-route</name>
  <uuid>43f3f9b8-d709-9292-722d-90f69e52ebb5</uuid>
  <forward dev='br0' mode='route'>
    <interface dev='br0'/>
  </forward>
  <bridge name='virbr0' stp='on' delay='0' />
  <ip address='192.168.100.1' netmask='255.255.255.0'>
    <dhcp>
      <range start='192.168.100.2' end='192.168.100.254' />
    </dhcp>
  </ip>
</network>
</class>
```

Configure linux containers network configuration accordingly

- /var/liv/lxc/container/config

1. networking
   lxc.utsname = container
   lxc.network.type = veth
   lxc.network.flags = up
   lxc.network.link = virbr0
   lxc.network.ipv4 = 192.168.100.10/24
   lxc.network.ipv4.gateway = 192.168.100.1

Add firewall/iptables rule to VPS host forwarding traffic to the virtual network

```
* @/etc/rc.local@, before *exit* line:
iptables -I FORWARD 1 -i virbr0 -o virbr0 --dest 192.168.100.0/24 -j ACCEPT
```

Enable IP Forwarding on VPS host

- /etc/sysctl.conf

```
net.ipv4.ip_forward = 1
```

To enable the changes made in sysctl.conf you will need to run the command:

```
sysctl -p /etc/sysctl.conf
```

#3 - 07/09/2013 12:36 PM - Daniel Curtis

There was a problem moving the first virtual network to the new virtual network. I had first thought the problem to be in the iptables command specified in /etc/rc.local so I changed the command to allow forwarding to the correct virtual network.

This however did not fix the problem, so I had checked the /etc/resolve.conf to find that the DNS server set was the virtual router; instead of the real, DMZ router. The DMZ router manages the entire DNS infrastructure for the DMZ network. Once the /etc/resolv.conf was set to the correct values I was able to connect out to the Internet.

#4 - 08/20/2013 08:16 AM - Daniel Curtis

- File lxc-debian-wheezy added

There are a couple of problems with the LXC Debian container creation script, as well as a lack of support for PowerPC repositories. I managed to find a script capable of creating working Debian containers on PowerPC hosts.

#5 - 08/23/2013 05:46 PM - Daniel Curtis

- Project changed from Website Hosting to 21

#6 - 09/16/2013 02:07 PM - Daniel Curtis

When adding a new VPS server, make sure to have the following puppet configuration parameters set:

1. Puppet Node SSL signed by the Puppet Master
2. Puppet Node defined
3. Puppet Classes Debian base, APT client, Kerberos client, LXC vServer, and Puppet client included
4. Unique Node /etc/fstab defined
5. Unique Node /etc/rc.local defined
6. Unique Node /etc/network/interfaces defined
7. Virtual Server template files defined

#7 - 09/17/2013 03:17 PM - Daniel Curtis

- Description updated

#8 - 02/15/2015 09:44 PM - Daniel Curtis

- Project changed from 21 to GNU/Linux Administration

- Category set to Jails / Container

Files

lxc-debian-wheezy	7.68 KB	08/20/2013	Daniel Curtis