# LXC使用文档

## LXC介绍：

1. **LXC介绍：**

LXC为Linux Container的简写。Linux Container容器是一种内核虚拟化技术，可以提供轻量级的虚拟化，以便隔离进程和资源，而且不需要提供指令解释机制以及全虚拟化的其他复杂性。相当于C++中的NameSpace。容器有效地将由单个操作系统管理的资源划分到孤立的组中，以更好地在孤立的组之间平衡有冲突的资源使用需求。

1. **组成：**

LXC可以在操作系统层次上为进程提供虚拟的执行环境，一个虚拟的执行环境就是一个容器，同时可以为容器绑定特定的cpu和memory节点，分配特定比例的cpu时间、IO时间，限制可以使用的内存大小（包括内存和是swap空间），提供device访问控制，提供独立的namespace（网络、pid、ipc、mnt、uts）。

LXC主要包含三个方面：**cgroup**、**network**和**rootfs**。

1. **Cgroups**是control groups的缩写，是Linux内核提供的一种可以限制、记录、隔离进程组（process groups）所使用的物理资源（如：cpu,memory,IO等等）的机制。LXC主要通过cgroup实现对于资源的管理和控制。
2. **Network**是为容器提供的网络环境设置。
3. **rootfs**用于指定容器的虚拟根目录，设定此项以后，容器内所有进程将会把此目录根目录，不能访问此目录之外的路径，相当于chroot的效果。
4. **优势：**

与传统虚拟化技术相比，它的优势在于：

1. 与宿主机使用同一个内核，性能损耗小；
2. 不需要指令级模拟；
3. 不需要即时(Just-in-time)编译；
4. 容器可以在CPU核心的本地运行指令，不需要任何专门的解释机制；
5. 避免了准虚拟化和系统调用替换中的复杂性；
6. 轻量级隔离，在隔离的同时还提供共享机制，以实现容器与宿主机的资源共享。

## 创建 LXC 容器

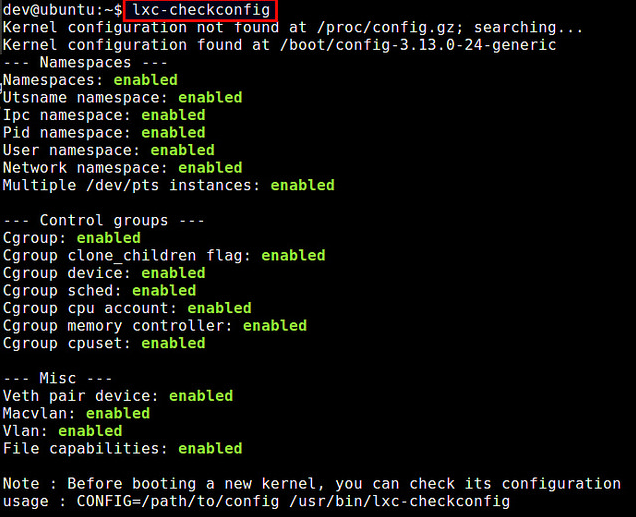
**安装**

使用下面的命令安装 LXC 在用户态的工具：

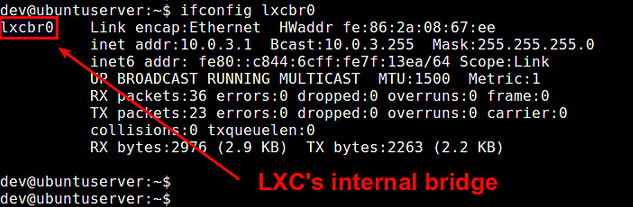
$ sudo apt-get install lxc

然后检查当前内核是否支持 LXC。如果所有结果都是“enable”，说明内核支持：

$ lxc-checkconfig



安装完 LXC 工具后，就能看到 LXC 自动创建了一块桥接网卡（lxcbr0，可以在 /etc/lxc/default.conf 中设置）。

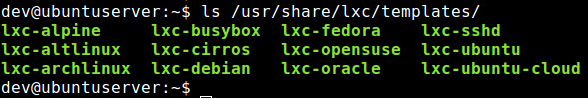


当创建了 LXC 容器后，它的网口会自动链接到这个桥接网卡上，然后这个容器就能和外部世界通信了。

**创建**

为了在指定环境下创建 LXC 容器，需要一个相应的 LXC 模板。 LXC 提供的工具集成了一整套现成的 LXC 模板，你可以在 /usr/share/lxc/templates 目录下找到它们。

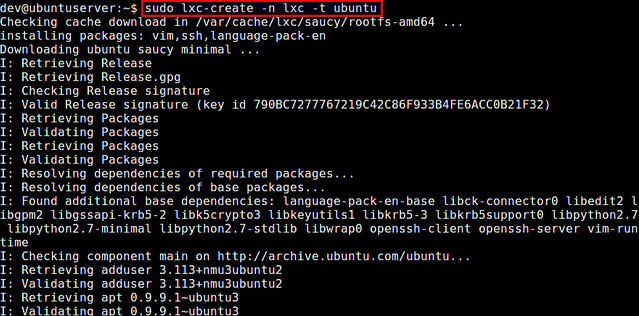
$ ls /usr/share/lxc/templates



一个 LXC 模板实质上就是一个脚本，用于创建指定环境下的容器。当你创建 LXC 容器时，你需要用到它们。

比如你要新建 Ubuntu 容器，使用下面的命令即可：

$ sudo lxc-create -n <container-name> -t Ubuntu

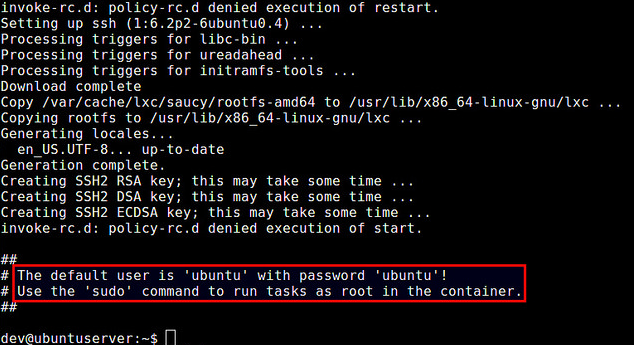


默认情况下，这个命令会创建一个最小的 Ubuntu 环境，版本号与你的宿主机一致。

当然你也可以创建任何你喜欢的版本，只要在命令里面加一个版本参数即可。举个例子，创建 Ubuntu 14.04的容器：

$ sudo lxc-create -n <container-name> -t ubuntu -- --release utopic

这个命令就会下载安装指定环境下的软件包，创建新容器。整个过程需要几分钟时间，与容器的类型有关，所以，需要耐心等待。



下载安装完所有软件包后，LXC 容器镜像就创建完成了，你可以看到默认的登录界面。容器被放到 /var/lib/lxc/<容器名> 这个目录下，容器的根文件系统放在 /var/lib/lxc/<容器名>/rootfs 目录下。

创建过程中下载的软件包保存在 /var/cache/lxc 目录下面，当你想另外建一个一样的容器时，可以省去很多下载时间。

用下面的命令看看主机上所有的 LXC 容器：

$ sudo lxc-ls –fancy

NAME STATE IPV4 IPV6 AUTOSTART

------------------------------------

test-lxc STOPPED - - NO

使用下面的命令启动容器。参数“-d”将容器作为后台进程打开。如果没有指定这个参数，可以在控制台界面上直接把容器的运行程序关闭（LCTT译注：Ctrl+C组合键）。

$ sudo lxc-start -n <container-name> -d

打开容器后，看看状态：

$ sudo lxc-ls --fancy

NAME STATE IPV4 IPV6 AUTOSTART

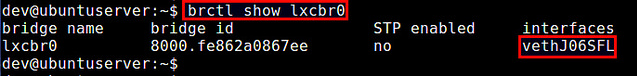
-----------------------------------------

lxc RUNNING 10.0.3.55 - NO

容器状态是“运行中”，容器 IP 是10.0.3.55。

可以看到容器的网络接口自动与 LXC 内部网桥（lxcbr0）连上了：

$ brctl show lxcbr0

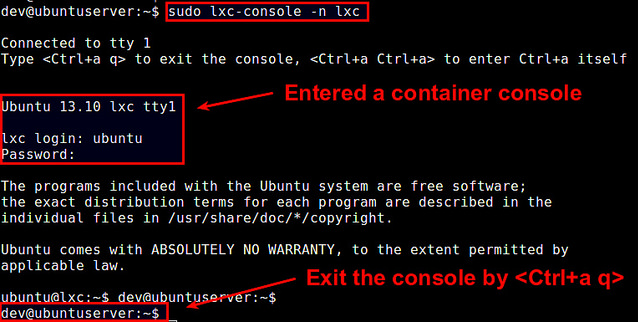


## 管理 LXC 容器

上文已经演示创建和启动 LXC 容器，现在来看看怎么使用一个正在运行着的容器。

第一步：打开容器控制台：

$ sudo lxc-console -n <container-name>



使用“Crtl+a q”组合键退出控制台。

停止、删除容器：

$ sudo lxc-stop -n <container-name>

$ sudo lxc-destroy -n <container-name>

复制容器，用下面的命令：

$ sudo lxc-stop -n <container-name>

$ sudo lxc-clone -o <container-name> -n <new-container-name>

## 常见命令总结

1. **lxc-version 用于显示系统LXC的版本号**（可以通过此命令判断系统是否安装了lxc）

用法：lxc-version

例如:lxc-version

1. **lxc-checkconfig 用于判断linux内核是否支持LXC**

用法：lxc-checkconfig

例如：lxc-checkconfig

1. **lxc-create用于创建一个容器**

用法：lxc-create -n name [-f config\_file]

-n 后面跟要创建的容器名字 例如：-n foo

-f 后面跟容器配置文件的路径

注：

1.采用lxc-create创建的容器，在停止运行后，不会被销毁，要采用lxc-destroy命令才能销毁

2.容器命令空间是全局的，系统中不允许存在重名的容器，如果-n 后面跟一个已经存在的容器名，创建会失败

例如：lxc-create --n foo --f foo.conf

1. **lxc-execute 用于在一个容器执行应用程序**

用法： lxc-execute -n name [-f config\_file] [ -s KEY=VAL ]command

-n 后面跟容器名字（容器名字用于管理容器）例如：-n foo

-f 后面跟容器配置文件的路径（如果没有配置文件，可以直接用-s指定配置选项，如果什么都没有，系统采用默认策略）例如：-f foo.conf

-s 后面跟配置键值对 例如：lxc.cgroup.cpu.shares=512

command 为要执行的命令 例如：/bin/bash

这个命令会mount /proc 并且会自动创建/销毁容器。

注：

1.如果容器还不存在，lxc-execute会自动创建一个,容器停止运行后会被自动销毁

2.用lxc-execute启动应用程序，配置优先级如下：

如果指定-f选项，那么之前创建容器（如果容器是已存在的）的配置文件不会被使用

如果指定-s选项，则在命令行中的配置键值对会覆盖配置文件（无论之前的还是-f指定的）相同配置

例如：lxc-execute --n foo --s lxc.cgroup.cpu.shares=512 /bin/bash

使用实际例子:

lxc-execute -n test /bin/bash

这个会启动一个lxc并给出类似的一个cmd窗口，网络是与操作系统共用的，这里好像仅仅是创建了一个命名空间

如果没有指定-f，默认的隔离将被使用，这个命令当你需要一个快速在一个隔离的环境中运行程序。在物理机上和container中都会运行lxc-init，在宿主机上面，这个程序用于转发lxc-kill 信号到已经启动的程序中 ，在container中，这个程序的pid为1，

它会fork出要执行的命令（pid为2）并执行。

1. **lxc-start 用于在容器中执行给定命令**

用法：lxc-start -n name [-f config\_file] [-c console\_file] [-d] [-s KEY=VAL]

[command]

-d 将容器当做守护进程执行

-f 后面跟配置文件

-c 指定一个文件作为容器console的输出，如果不指定，将输出到终端

-s 指定配置

如果没有指定命令，lxc-start 将要运行 /sbin/init

例如：lxc-start -n foo -f foo.conf -d /bin/bash

注：1.如果容器还不存在，lxc-start会自动创建一个,容器停止运行后会被自动销毁

2.lxc-start配置优先级与lxc-execute相同

3.lxc-start 与lxc.execute的异同：

lxc-start 和 lxc-execute都可以在容器中启动进程，区别在于lxc-start直接创建进程，lxc-execute先创建lxc-init进程，然后在lxc-init中fork一个进程来执行。(关于第4点，lxc-init所占的是一个什么样的地位？)

1. lxc-start用于在容器启动system，lxc-execute用于在容器执行应用程序
2. **lxc-kill 发送信号给容器中的第一个用户进程（容器内部进程号为2的进程）**

用法：lxc-kil -n name SIGNUM

-n 后面跟容器名

SIGNUM 信号 （此参数可选，默认SIGKILL）

例如：lxc-kill -n foo

1. **lxc-stop 用于停止容器中所有的进程**

用法：lxc-stop -n name

-n后面跟要停止的容器名

例如:lxc-stop --n foo

1. **lxc-destroy 用于销毁容器**

用法：lxc-destroy -n name

-n后面跟要停止的容器名

例如: lxc-destroy --n foo

1. **lxc-cgroup 用于获取或调整与cgroup相关的参数**

用法：lxc-cgroup -n name subsystem value

-n 后面跟要调整的容器名

例如： lxc-cgroup -n foo devices.list

lxc-cgroup -n foo cpuset.cpus "0,3"

1. **lxc-info 用户获取一个容器的状态**

用法:lxc-info -n name

-n后面跟操作的容器名

例如: lxc-info --n foo

注：容器的状态有：STARTING RUNNING STOPPING STOPPED ABORTING

1. **lxc-monitor 监控一个容器状态的变换，当一个容器的状态变化时，此命令会在屏幕上打印出容器的状态**

用法:lxc-monitor -n name

例如：lxc-monitor -n foo

1. **lxc-ls 列出当前系统所有的容器**

用法：lxc-ls

例如：lxc-ls

1. **lxc-ps 列出特定容器中运行的进程**

用法:lxc-ps

例如:lxc-ps -n foo

更多的命令介绍可以再man page获得

设置ubuntu自启动应用sudo vi /etc/rc.local

默认登录名为ubuntu,密码为ubuntu

root用户默认已失效，需要重新设置，sudo passwd root 命令

配置文件详解：<http://www.cnblogs.com/lisperl/archive/2012/04/16/2451215.html>

<http://m.blog.csdn.net/blog/linuxchyu/21240691>

网络信息配置地址：/var/lib/lxc/<container>/config

网桥配置（brctl命令详解）：<http://fp-moon.iteye.com/blog/1468650>

容器文件系统目录：/var/lib/lxc/（容器名）/rootfs/home/（容器名）

物理机网络设置：

sudo vim /etc/network/interfaces

重启服务：

sudo /etc/init.d/networking restart

配置DNS：

sudo nano /etc/resolv.conf

nameserver x.x.x.x

**创建桥接虚拟机步骤：**

1.添加新网桥（lxc默认为lxcbr0），并将eth0帮上去：使用brctl命令

2.更改虚拟机ip

sudo vim /etc/network/interfaces

文件内容：

auto lo

iface lo inet loopback

auto eth0

iface eth0 inet manual

auto lxcbr0

iface lxcbr0 inet static

address 172.29.153.174

network 172.29.153.0

netmark 255.255.255.0

broadcast 172.29.153.255

gateway 172.29.153.1

bridge\_ports eth0

bridge\_fd 9

bridge\_hello 2

bridge\_maxage 12

bridge\_stp off

3.更改默认虚拟机配置

/var/lib/lxc/<container>/config

lxc.network.ipv4 = 172.29.153.175

加入ip