Rocky LXC Guide

For **RHCSA** & **RHCE** Practice

Version 0.2 alpha by: Trevor Smale

Nov 4th 2024

Overview

Intro

LXC Containers are minimal, low resource environments much like Docker or Podman containers that can be run in ProxMox. This is a guide for proxmox users to create a container for practicing for the RHCSA & RHCE with low stakes machines that can be created and destroyed from templates.

🪶 LXC

LXC containers are lighter and faster than virtual machines (VMs) because they share the host's kernel and resources, avoiding the overhead of running a separate operating system for each instance. This allows containers to use less memory and CPU compared to VMs, which need to emulate full hardware environments through hypervisors.

🧐 In this Guide

We will go through the following procedures:

1. Adding & Configuring a new user with sudo privledges
2. Installing, Configuring and Activating SSH services for remote access
3. Installing useful packages like VIM, Python, GO, SSH, Ansible, Podman ++
4. Converting completed LXC container into a template for replication

🔍 Pre-Requisites

- Must have a ProxMox VE setup.
- Must have prior understanding of how to find and download LXC templates within ProxMox.
- Must have prior linux command line experience.

⚙️ Recommended Container Specs

- Rocky 9 LXC Template
- Processors = 1
- Memory = 512 MiB
- Swap = 512 MiB
- Bootdisk = 4GiB (1GiB Req'd for packages)

⚙️ Configuration Procedure

👨‍🔧 Add user and make them Super🦄

1. Create a new user: **`useradd 'username'`**
2. Create a password for that new user **`passwd 'username'`**
3. Give sudo privleges to that new user **`usermod -aG wheel 'username'`**
4. Update the sudoers file **`visudo`**

Changes to sudoers file by adding a line above or below the root user privileges. This can be found near bottom add user below root and give same privs

`'username' ALL=(ALL) ALL\`

5. **Write** changes and **Quit vi (esc :wq)**
6. Become this new user. **`su 'username'`**
7. test **`sudo`** powers **`sudo dnf update`** 👍

☎️ Install and Configure SSH 🔧

Adding SSH involves several steps that must be done correctly in order for things to work properly

1. **`dnf install openssh`**
2. **`dnf install openssh-server`**
3. **`systemctl status sshd`**
4. **`systemctl enable sshd`**
5. **`systemctl start sshd`**
6. **`vi /etc/ssh/sshd_config`**

These are the settings we must modify using vi. We do this by first un-commenting the line by removing the **#** mark, then altering the text. You will find these lines sequentially from top to bottom:

Fields in the **sshd_config** file that need to be altered:

- **AuthorizedKeysFile**
- **PasswordAuthentication yes**
- **Hostkey /etc/ssh/ssh_host_ed25519_key**
- **PermitRootLogin no**
- **PubkeyAuthentication yes**
- **AuthorizedKeysFile .ssh/authorized_keys**

7. **Write** changes and **Quit vi (esc :wq)**
8. Restart sshd service:

`systemctl restart sshd`

9. Test your SSH connection 👍

Download and install packages

Follow the **numbered steps** of this installation procedure and everything should install without a hitch. I assume that you are install everything as **root**.

1. Update repository reference info:

```
dnf update -y
```

2. Install most packages:

```
dnf install -y bash sudo shadow bash-completion ca-certificates glibc gcc ncurses iptables net-tools iproute procps-ng psmisc policycoreutil
policycoreutils-python-utils selinux-policy firewalld logrotate curl wget rsync tar unzip git vim tmux lsof strace tcpdump iotop sysstat bind-utils nmap iperf3 tree podman
```

3. Install enterprise package repo:

```
dnf install -y epel-release
```

4. Install podman-compose

```
dnf install podman-compose
```

5. Install Ansible:

```
dnf install -y ansible
```

Additional steps for latest GO binary release

1. **Using wget to retrieve latest binary**

```
wget https://go.dev/dl/go1.22.1.linux-amd64.tar.gz
```

2. **Unpacking binary into /user/local**

```
tar -C /usr/local -xzf go1.22.1.linux-amd64.tar.gz
```

3. **Adding user binary PATH**

Adds the Go binary path to your system's PATH so you can run Go commands from any location in the terminal.

```
export PATH=$PATH:/usr/local/go/bin
```

4. **Adding Home PATH**

sets GOPATH, which is the workspace for Go projects, to the go directory in your home folder.

```
export GOPATH=$HOME/go
```

5. **Adding general binary PATH**

Adds the Go workspace binary path (where Go binaries like `go install` places compiled binaries) to your system's PATH.

```
export PATH=$PATH:$GOPATH/bin
```

6. **Add PATHS to Bash RC**

```
vim ~/.bashrc
```

Paste these into the bash rc file:

- `export PATH=$PATH:/usr/local/go/bin`
- `export GOPATH=$HOME/go`
- `export PATH=$PATH:$GOPATH/bin`

7. **Reload Bash RC**

```
source ~/.bashrc
```

8. **Confirm everything works**

```
go version
```

© Template Conversion

- Ensure that you are satisfied with the containers operation.
- **Test SSH** Access to both the Root and new sudo privileged user.
- Once satisfied, convert the container into a template.
  - do this by right clicking on the container in the left hand column of the pve list and choosing **convert to template**.
- Be patient and allow for the process to proceed uninterrupted.
- Once the template is created, it is trivial to make copies either by **GUI**, **TUI** or **CLI**

📦 Full Package List

Package Name	Description
bash	Essential shell for Linux systems
sudo	Run commands as root user
shadow	Manages user account passwords
bash-completion	Autocompletion for bash commands
ca-certificates	Certificate authority files for SSL
glibc	Standard C library for Linux
gcc	GNU C Compiler for building software
ncurses	Terminal handling library for text
iptables	Command-line firewall utility
net-tools	Legacy networking tools (e.g., ifconfig)
iproute	Modern networking tools (e.g., ip)
procps-ng	Utilities for process management (e.g., ps)
psmisc	Utilities like killall and fuser
policycoreutils	Security policies for SELinux
policycoreutils-python-utils	Python tools for SELinux policies
selinux-policy	Default SELinux policy for system
firewalld	Dynamic firewall management daemon
logrotate	Automatically rotates system logs
curl	Tool to transfer data from URLs
wget	Command-line downloader for HTTP/FTP
rsync	File copying and synchronization tool
tar	Archiving utility for file packaging
unzip	Utility for extracting zip archives
git	Version control system for code
vim	Highly configurable text editor
tmux	Terminal multiplexer for session management
lsof	List open files in the system
strace	Trace system calls and signals
tcpdump	Packet analyzer for network traffic
iotop	Monitor disk I/O usage in real-time
sysstat	Performance monitoring utilities
bind-utils	DNS query tools (e.g., dig)
nmap	Network scanner and port mapper
iperf3	Network performance measurement tool
tree	Display directories in a tree format
epel-release	Extra packages for Enterprise Linux
ansible	Automation tool for configuration management
podman	Container management similar to Docker
podman-compose	Compose tool for Podman containers