LXC

Overview

What is LXC

About LXC

> Linux Containers (LXC) is an operating-system-level virtualization method for running multiple isolated Linux systems (containers) on a control host using a single Linux kernel.
>
> -- Wikipedia

References

Homepage: https://linuxcontainers.org/lxc

Wikipedia: https://en.wikipedia.org/wiki/LXC

Alpine Linux Dev Container

OpenRC init

Overview

Alpine linux uses Busybox' init as process 1, which in turn runs OpenRC as a service manager.

OpenRC features its own process 1 implementation - `openrc-init` - which suits container startup perfectly well.

We can reduce (minimally) startup and administration dependencies by replacing `/sbin/init` with `/sbin/openrc-init`.

Follow this steps to switch to OpenRC init:

- Assure terminal configuration on openrc startup
- Configure OpenRC
- Configure the container to start `openrc-init`.

Terminal Configuration

See Busybox getty

After setting up the busybox start script, enable the service for at least the console and tty1. We add tty2 also.

```
sudo true
CONSOLES="console tty1 tty2"
cd /etc/conf.d
for t in $CONSOLES; do sudo ln -s bgetty bgetty.$t; done
cd /etc/init.d
for t in $CONSOLES; do sudo ln -s bgetty bgetty.$t; done
for t in $CONSOLES; do sudo rc-update add bgetty.$t; done
```

This does not yet run the gettys. If you want to test one use e.g.: `rc-service bgetty.tty2 start`.

Note: The console is switched around during startup. Including it as a bgetty terminal service seems to print out console startup messages on tty1 and then start the login shell there.

Autologin a User

getty runs a login command, which can be replaced with the `-l` commandline option, however it is not possible to specify options. So we need to write a login script which is then called by getty:

```
sudo true
sudo tee /usr/local/bin/login.$USER <<EOF
#!/bin/sh
exec login -p -f $USER
EOF
chmod +x /usr/local/bin/login.$USER
```

The `/usr/local/bin/login.`*`USER`* script logs in *`USER`* without prompt and without authentication.

Add a corresponding options line to the bgetty configuration file:

```
sudo true
sudo tee -a /etc/conf.d/bgetty <<EOF
bgetty_options="-n -l /usr/local/bin/login.$USER"
EOF
```

After starting the respective terminal the users login shell is run immediately. If the user exits the shell she is logged in immediately again.

Configure OpenRC

The following settings in `/etc/rc.conf` are optional

```
rc_parallel="YES"
rc_env_allow="*"
rc_sys="lxc"
rc_tty_number=4
```

- rc_parallel: start up independent services in parallel. gives confusing output on start, but allows almost immediate interaction
- rc_env: pass all environment variables through to the shell.
- rc_sys: only important setting

- rc_tty_number: normally 12 tty's are allocated, we reduce them because we probably don't ever need 'em.

Container Shutdown with OpenRC init

Use `openrc-shutdown` to shut down the container. Standard init commands like `reboot` and `shutdown` won't work.

Configure the LXC Container

Add this to the container `config`:

```
lxc.tty.max = 4
lxc.init.cmd = /sbin/openrc-init
```

Busybox getty

Overview

Alpine Linux uses by default agetty for managing terminal lines which is an extra package. We can use busybox' getty instead and save space.

Service configuration

The service for running busybox getty will be called `bgetty`.

Create a default configuration file

```
sudo true
sudo tee /etc/conf.d/bgetty <<EOF
# Set the baud rate of the terminal line
# 0 .. leave alone
baud="0"
# set the terminal type
#term_type="linux"

# extra options to pass to getty for this port
bgetty_options=""
EOF
```

Create a service script

```
sudo true
sudo tee /etc/init.d/bgetty <<EOF
#!/sbin/openrc-run
# Copyright (c) 2017 The OpenRC Authors.
# See the Authors file at the top-level directory of this distribution and
# https://github.com/OpenRC/openrc/blob/master/AUTHORS
```

```
#
# This file is part of OpenRC. It is subject to the license terms in
# the LICENSE file found in the top-level directory of this
# distribution and at https://github.com/OpenRC/openrc/blob/master/LICENSE
# This file may not be copied, modified, propagated, or distributed
# except according to the terms contained in the LICENSE file.
#
# Adapted for busybox getty gl@x-net.at

description="start busybox getty on a terminal line"
supervisor=supervise-daemon
port="${RC_SVCNAME#*.}"
respawn_period="${respawn_period:-60}"
term_type="${term_type:-linux}"
baud="${baud:-0}"
command=/sbin/getty
command_args_foreground="${bgetty_options} ${baud} ${port} ${term_type}"
pidfile="/run/${RC_SVCNAME}.pid"

depend() {
	after local
	keyword -prefix
	provide getty
}

start_pre() {
	if [ -z "$port" ]; then
		eerror "${RC_SVCNAME} cannot be started directly. You must create"
		eerror "symbolic links to it for the ports you want to start"
		eerror "getty on and add those to the appropriate runlevels."
		return 1
	else
		export EINFO_QUIET="${quiet:-yes}"
	fi
}

stop_pre()
{
	export EINFO_QUIET="${quiet:-yes}"
}
```

```
EOF
sudo chmod +x /etc/init.d/bgetty
```

Busybox

About Busybox

> BusyBox is a software suite that provides several Unix utilities in a single executable file. It runs in a variety of POSIX environments such as Linux, Android, and FreeBSD, although many of the tools it provides are designed to work with interfaces provided by the Linux kernel. It was specifically created for embedded operating systems with very limited resources. The authors dubbed it "The Swiss Army knife of Embedded Linux",[10] as the single executable replaces basic functions of more than 300 common commands. It is released as free software under the terms of the GNU General Public License, version 2.
>
> -- Wikipedia

Busybox References

Homepage: https://www.busybox.net/

Wikipedia: https://en.wikipedia.org/wiki/BusyBox