Alpaquita Linux
Setting up OpenRC init system

Alpaquita Linux
Revision 1.0
June 2024

bellsoft

BellSoft software contains open source software. Additional information about third party code is available at https://bell-sw.com/third_party_licenses. You can also get more information on how to get a copy of source code by contacting info@bell-sw.com.

The use of any software product referenced in this document is governed by the applicable license agreement, which is not modified in any way by the terms of this notice.

Contents

1. Overview

The most popular init system on Linux distributions today is systemd. Besides being just an init system, `systemd` provides many additional components, which are intended to replace various common Linux services, such as syslog, cron, etc. This is why some people consider `systemd` design as too complex and incompatible with the Unix philosophy.

Alpaquita Linux, in contrast, uses OpenRC, a more lightweight init system that works with the init program `/usr/sbin/init` provided by BusyBox. Unlike `systemd`, OpenRC does not tend to replace common Linux services. In Alpaquita, some functionality that is normally covered by `systemd` in systemd-based distributions is covered by separate utilities.

The document provides a quick overview of some OpenRC init system features in Alpaquita Linux and includes examples of how to perform the tasks that you perform in a `systemd`-based OS distribution.

2. Basic commands

You can manage the OpenRC service using one of the following commands:

- `rc-service <service> <cmd>`
- `/etc/init.d/<service> <cmd>`

For example:

```
rc-service sshd reload
```

```
/etc/init.d/sshd reload
```

The following table describes basic OpenRC commands and their `systemd` alternatives:

Command	OpenRC	systemd
Start service	`rc-service <service> start`	`systemctl start <service>`
Stop service	`rc-service <service> stop`	`systemctl stop <service>`
Restart service	`rc-service <service> restart`	`systemctl restart <service>`
Reload service	`rc-service <service> reload`	`systemctl reload <service>`
Show service status	`rc-service <service> status`	`systemctl status <service>`
Enable service at startup	`rc-update add <service>`	`systemctl enable <service>`
Disable service at startup	`rc-update del <service>`	`systemctl disable <service>`

Command	OpenRC	systemd
List enabled services	`rc-update`	`systemctl list-unit-files --state enabled`
List all services	`rc-status -s`	`systemctl list-unit-files`

For more information, see the following documentation:

- Working with OpenRC
- OpenRC to systemd Cheatsheet

3. Configuration files

/etc/inittab

This is a configuration file for the `/usr/sbin/init` program from BusyBox. It is basically used to start OpenRC or `getty` processes on specific ttys depending on the requested action.

For more information, see /etc/inittab init(8) configuration for BusyBox.

/etc/rc.conf

This is the global OpenRC configuration file. In OpenRC, configuration files are simply shell scripts, therefore you do not need to learn a special syntax as in `systemd`.

/etc/conf.d

This directory contains configuration files for individual OpenRC init scripts. It somewhat corresponds to */etc/systemd/* in systemd-based distributions.

4. Logging

Messages that you normally observe by invoking `journalctl` under `systemd`, you can find in the following files when using OpenRC:

- /var/log/rc.log
- /var/log/dmesg
- /var/log/messages

/var/log/rc.log

This file is for OpenRC specific logging that is disabled by default. To enable it, add the following lines to */etc/rc.conf*:

```
rc_logger="YES"         # enable logging
rc_quiet_openrc="NO"    # disable quiet mode
```

/var/log/dmesg

This file contains the initial `dmesg` log. It is populated by the OpenRC service `bootmisc`. This log is useful in case of long uptime of your system when initial kernel messages have been flushed.

/var/log/messages

This file contains system log messages. Unlike `systemd`, OpenRC does not provide any utilities for system logging. In Alpaquita, BusyBox services `syslogd` and `klogd` are used for system logging.

Logging kernel messages

By default, `klogd` is disabled and kernel messages are not logged to */var/log/messages*. To enable and start it by running the following commands:

```
rc-update add klogd boot
rc-service klogd start
```

Log rotation

`systemd` provides the `systemd-journald` service for system logging and a wide range of options for log rotation. `syslogd` provides the following two options:

```
-s SIZE         Max size (KB) before rotation (default 200KB, 0=off)
-b N            N rotated logs to keep (default 1, max 99, 0=purge)
```

The options can be adjusted in */etc/conf.d/syslog* file. For example, you can set the following options for `syslogd`.

```
SYSLOGD_OPTS="-s 1024 -b 4"
```

For more sophisticated log rotation, consider installing the **logrotate** package.

Log rules

`journalctl` helps you filter logs in many different ways (by service, priority, facility, and so on). Such powerful filtering is possible, because the `systemd-journald` service stores logs in binary format. `syslogd` stores logs as plain text. For plain text files, you can use `grep`, but you can also set up log rules in */etc/syslog.conf* for different types of messages to be logged in different files as in the following example.

```
cron.*      /var/log/cron     #all messages of cron facility
*.err       /var/log/error    #all messages with priorities less than or equal
to err
```

For more information, see syslog.conf.txt.

> **Note:**
>
> If you need more advanced system logging, consider installing the **syslog-ng** package.

5. Writing init scripts

OpenRC uses init shell scripts to run services rather than special service configuration files. An init script must start with a shebang `#!/usr/sbin/openrc-run`. The init script code is expected to be compatible with POSIX shell.

Below you can see an example of an init script that moves all tasks to a `cgroup` with specific CPUs and memory nodes assigned to it:

```
#!/usr/sbin/openrc-run

: ${CPUS:="0"}
: ${MEMS:="0"}
# $RC_SVCNAME is a name of the service. By default, it equals a script name.
: ${CGROUP_NAME:=$RC_SVCNAME}
: ${CGROUP_CPUSET_PATH="/sys/fs/cgroup/cpuset"}

# This function specifies dependency information which is needed to start and
# stop the service in the right order in relation to other services.
depend()
{
    # The service must start after the 'cgroups' service.
    after cgroups
    # Don't stop the service when shutting the system down.
    keyword -shutdown
}

# This function contains commands necessary to start the service.  Functions
# 'ebegin', 'eend', 'einfo' are defined by openrc-run and used for logging.
start()
{
    cd "$CGROUP_CPUSET_PATH"

    ebegin "Creating cgroup $CGROUP_NAME"
    mkdir "$CGROUP_NAME"
    eend $? || return $?

    echo "$MEMS" > "$CGROUP_NAME/cpuset.mems"
    echo "$CPUS" > "$CGROUP_NAME/cpuset.cpus"

    einfo "Moving all tasks to cgroup $CGROUP_NAME"
    cat tasks | xargs -n1 echo >"$CGROUP_NAME/tasks" 2>/dev/null
```

```
    return 0
}

# This function contains commands necessary to stop the service.
stop()
{
    cd "$CGROUP_CPUSET_PATH"

    einfo "Removing all tasks from cgroup $CGROUP_NAME"
    cat "$CGROUP_NAME/tasks" | xargs -n1 echo >tasks 2>/dev/null

    ebegin "Removing cgroup $CGROUP_NAME"
    rmdir "$CGROUP_NAME"
    eend $? || return $?
}
```

If you save this script to */etc/init.d/isoltasks*, you can use it as a regular service.

```
# /etc/init.d/isoltasks start
 * Creating cgroup isoltasks ...                [ ok ]
 * Moving all tasks to cgroup isoltasks
# /etc/init.d/isoltasks stop
 * Removing all tasks from cgroup isoltasks
 * Removing cgroup isoltasks ...                [ ok ]
```

For more information about built-in functions and environment variables used in init scripts, see `man openrc-run` (provided by package **openrc-doc**).

See also:

- Writing Init Scripts
- Handbook:X86/Working/Initscripts

6. Mounting filesystems

nfs

To mount nfs targets, install the **nfs-utils** package first.

```
apk add nfs-utils
```

OpenRC does not automatically mount nfs targets specified in */etc/fstab*. Enable the `netmount` service for that.

```
rc-update add netmount default
```

tmpfs

Under `systemd`, */tmp* is automatically mounted as tmpfs. In Alpaquita, to mount */tmp* as tmpfs, configure the */etc/fstab* as in the following example:

```
tmpfs   /tmp    tmpfs   rw,nosuid,noatime,nodev,size=1G,mode=1777   0 0
```

7. Console font

A console font is set by the OpenRC service `consolefont`. Before using it, install some fonts first, for example:

```
apk add terminus-font
```

To change the default font, set the `consolefont` variable in */etc/conf.d/consolefont* to the name of a file found in */usr/share/consolefonts/*, for instance:

```
consolefont="ter-v20b.psf.gz"
```

The `consolefont` service is not enabled by default. To enable and start it, run the following commands:

```
rc-update add consolefont boot
rc-service consolefont start
```

To preview the font, use utility `setfont`.

```
setfont ter-v20b.psf.gz
```

> **Note:**
>
> `setfont` changes the font only for the console currently in use.

8. getty on the serial console

Under `systemd`, it is enough to specify `console=ttyS0` in the kernel command line and `systemd` automatically starts a getty for it. In Alpaquita, additional configuration is needed. Use */usr/sbin/init* or OpenRC for it.

/usr/sbin/init

To enable `getty` on the serial console using */usr/sbin/init*, uncomment or add this line to */etc/inittab*:

```
ttyS0::respawn:/sbin/getty -L ttyS0 115200 vt100
```

For the changes to take effect without a reboot, reload the init process */etc/inittab*. Use the following command:

```
kill -HUP 1
```

OpenRC

To enable and start `getty` on the serial console using OpenRC, set up the `agetty.ttyS0` service.

```
apk add agetty
ln -s /etc/init.d/agetty /etc/init.d/agetty.ttyS0
rc-update add agetty.ttyS0 boot
rc-service agetty.ttyS0 start
```

You can edit `agetty` options in the */etc/conf.d/agetty.ttyS0* file. See the */etc/conf.d/agetty* file for option description and examples.

9. Single-user mode

If you need to do something that is not possible at the default OpenRC runlevel, you can try to do it in the single-user mode (also known as the rescue mode). You can enter the single-user mode at different stages of a booting process, namely:

- initramfs (dracut)
- /usr/sbin/init
- OpenRC

initramfs (dracut)

Alpaquita uses **dracut** for creating initramfs images. dracut provides the `rd.break` kernel command line option that lets you drop to a shell right before switching to a root filesystem mounted on */sysroot*. If you exit from the shell, the booting process continues as usual.

> **Note:**
>
> Shutdown commands (`halt`, `poweroff`, `reboot`) do not work without the `-f` option in the dracut shell. The `-f` option forces a shutdown without relying on a */usr/sbin/init* process, which is not started yet at that moment.

It is also possible to drop to a shell at earlier stages. For more information, see `man dracut.cmdline` (provided by the **dracut-doc** package).

/usr/sbin/init

To enable single-user mode provided by */usr/sbin/init*, add `single` to the kernel command line. In this case, */etc/inittab* configuration file is ignored and you drop to a shell right after switching to a root filesystem, but before any OpenRC service starts.

If you want to get to the *default* runlevel from this stage, run the following commands:

```
openrc sysinit
```

```
openrc default
kill -HUP 1 && exit
```

> **Note:**
>
> This works only if services started by the *sysinit* and *default* runlevels do not affect a tty you are currently using (mainly, `agetty.*` services).

OpenRC

Single-user mode provided by OpenRC is a special runlevel called *single*. When you start it, all services except those in the sysinit runlevel are stopped.

To get to the *single* runlevel from the *default* runlevel, run the following command:

```
openrc single
```

To get back to the *default* runlevel, run the command as in the next example:

```
openrc default
```

> **Note:**
>
> A tty you are using must be set up via */etc/inittab*, not via `agetty.*` OpenRC services.

For more information, see `man openrc` (provided by the **openrc-doc** package).

Alpaquita Linux
Setting up OpenRC init system

bellsoft