# OpenRC vs rc.d

Compare and contrast

# Quick OpenRC FAQ

**Q. Is OpenRC an init system?**

Yes - **IF** you want to use it that way. OpenRC was originally designed *only* as an rc system. This is how TrueOS uses it, but other projects do use OpenRC as an init system.

**Q. Is OpenRC infected by the GPL license?**

Yes and No. OpenRC itself is available under the 2-clause BSD license, but some individual applications may use the GPL for their service files. Most of the GPL services are Linux specific and TrueOS has provided BSD-licensed replacements for many of them.

**Q. What are OpenRC runlevels? Isn’t that a Linux thing?**

While the term “runlevel” originated with the sysV init system and refers to numeric states of the system, OpenRC “runlevels” refer to something different. Each OpenRC “runlevel” is a logical group of services that are started in a designated order. Service dependency handling is always evaluated within the “runlevel”.

There are two predefined runlevels in OpenRC: **boot** is the first runlevel. Only base system services (*/etc/init.d*) can be placed in **boot**. **default** has no restrictions on what can be placed in it.

# Quick differences between OpenRC and rc.d

| | **OpenRC** | **rc.d** |
|---|---|---|
| Enable service at boot | \`rc-update add [service] [runlevel]\` | \`sysrc [service]_enable=YES\` |
| Disable service at boot | \`rc-update delete [service] [runlevel]\` | \`sysrc [service]_enable=NO\` |
| Check service status | \`rc-status\` or \`service [name] status\` | \`service [name] status\` (not well supported by most services) |
| Interact with a service | \`service [name] [action]\` | \`service [name] (one)[action]\` |
| Location of service files | [/usr/local]/etc/**init.d** | [/usr/local]/etc/**rc.d** |
| Written in | C | Shell |

# rc.d example: /usr/local/etc/**rc.d**/cupsd (print/cups)

```
#!/bin/sh
#
# $FreeBSD$
#
# PROVIDE: cupsd
# REQUIRE: DAEMON
# KEYWORD: shutdown
#
. /etc/rc.subr

name="cupsd"
rcvar="cupsd_enable"
start_precmd="${name}_prestart"
command="/usr/local/sbin/cupsd"
extra_commands="reload"

...
```

```
…

cupsd_prestart()
{
        if [ -n "$TZ" ]; then
                export TZ
        fi
}

load_rc_config ${name}
: ${cupsd_enable=NO}
run_rc_command "$1"
```

# OpenRC example: /usr/local/etc/**init.d**/cupsd

```
#!/sbin/openrc-run
command=/usr/local/sbin/cupsd
command_args="-f"
name="cupsd"
supervisor=supervise-daemon
pidfile="/var/run/cupsd.pid"

depend() {
	provide cups
	need localmount
	after dbus avahi-daemon
}
```

**Value added by this shorter file:**

- Easier to read/modify
- Service is supervised (auto-restarted if binary crashes)
- Ensures that dbus/avahi services are started before this (if they are enabled)
- Service status is **always available**

## Writing an OpenRC service “from scratch”

**SERVICE FILE (init.d/dummy)**

```
#!/sbin/openrc-run
name=”dummy”
description=”dummy service”
command=/usr/local/bin/dummy
command_args=”-foreground  ${dummy_args}”
supervisor=supervise-daemon
pidfile=”/var/run/dummy.pid”
output_log=”/var/log/dummy.log”
error_log=${output_log}

depend(){
        need localmount
        after network
}
```

**NOTES:**

<- This is *not* #!/bin/sh

<- Arguments for the command: “dummy_args” comes from the rc.conf files **OR** conf.d/[service_name]

<- Logfiles for standard output/error are easily specified

<- “need”: Ensure this service is running first

<- “after”: If enabled, start this service first

# OpenRC dependency handling

- Keywords for the “depend()” section:
  - **need**: These services are automatically started prior to the current service. If any are stopped or restarted, this service is also stopped or restarted in the proper order. This service *will not be started* if another required service is unavailable or cannot start.
  - **use**: These services are automatically started prior to this service *only if* enabled in the runlevel.
  - **want**: specify a service to start after another named service.
  - **after/before**: Set start/stop order based on these other services.
  - **provide**: alternate name this service provides. Example: *ntpd* and *openntpd* can both provide the “ntp” name, allowing other services to just need/use “ntp” instead of a specific ntp service.
  - **config**: List of files that the service will monitor and recalculate dependencies if changed
  - **keyword**: Additional flags for configuring service behavior (-shutdown, -timeout, -jail)