

容器运行时：runc

本篇是第八部分“生态篇”的第三篇。在这个部分，我会为你介绍 Docker 生态中的相关项目，以及如何参与到 Docker 项目中，最后会聊聊 Docker 未来的走向，上一篇，我为你介绍了容器运行时 containerd。本篇，我们来聊聊更底层的容器运行时 runc。

背景

在上篇中，我曾为你介绍过 Docker 为了变得更快、更好、更强大，而选择将其组件进行解耦。其中一个很重要的部分便是容器管理相关的组件，进而有了现在的 containerd 的存在。

而处于更底层，也是更核心的组件是则是容器运行时组件，这也是 Docker 首先分拆出来的重要组件之一，现在称之为 **runC**。

下文为了避免歧义，统一使用 runc 代指。

runc 是为了能使容器操作标准化，屏蔽掉不同系统/发行版之间的差异。

在 2015 年 Docker 联合 CoreOS 及其他容器行业的领导者一同成立了 OCI (Open Container Initiative) 组织，并且 Docker 将其镜像格式和容器运行时 runc 捐给了 OCI，以作为 OCI 工作的基石。

基本使用

在你安装完 Docker 后，runc 也会同时被安装到你的系统中。或者你可以在 runc 项目的 [release 页面](#) 直接下载其二进制文件即可。

准备 OCI Bundle

如果你想要使用 runc 启动容器，那类似的 Docker 和 containerd，需要先有一个容器镜像。对于 runc 而言，我们称之为 OCI Bundle。

OCI Bundle 是一组符合 OCI 规范的文件集合，你可以按照 OCI 规范自行创建。如果你已经安装了 Docker 的话，那使用 `docker export` 命令可以更方便的创建 OCI Bundle。

首先，需要准备 rootfs：

复制

```
(MoeLove) → ~ mkdir redis
(MoeLove) → ~ cd redis/
(MoeLove) → redis mkdir rootfs
(MoeLove) → redis docker export $(docker create redis:alpine) | tar -C rootfs -x
...# 省略部分输出
```

准备好 rootfs 后, 还需要有一个特定的配置文件, runc 提供了一个 `runc spec` 命令用于生成默认的配置文件的:

复制

```
(MoeLove) → redis runc spec
```

查看现在目录中的内容:

复制

```
(MoeLove) → tree -L 2
```

```
.
├── config.json
└── rootfs
    ├── bin
    ├── data
    ├── dev
    ├── etc
    ├── home
    ├── lib
    ├── media
    ├── mnt
    ├── opt
    ├── proc
    ├── root
    ├── run
    ├── sbin
    ├── srv
    ├── sys
    ├── tmp
    ├── usr
    └── var
```

19 directories, 1 file

上面的内容有没有感觉很熟悉? 在之前的内容中, 其实我们已经大致介绍过了。

我们需要对默认的 config.json 文件稍作调整, 以便于可以直接启动此镜像中的 redis-server。

只需要将 config.json 中的 args 从 sh 修改为 redis-server 即可:

复制

```
"args": [  
    "sh"  
],
```

修改为:

复制

```
"args": [  
    "redis-server"  
],
```

启动容器

复制

```
(MoeLove) → redis runc run redis  
... # 省略部分输出
```

简单的 `runc run` 命令便可启动一个容器, 这其实也是 OCI 组织中对用户体验的一种规范, 参考 `docker run`。

查看容器列表

打开另一个终端, 进入此目录, 执行 `runc list` 查看已启动的容器列表:

复制

```
(MoeLove) → redis runc list  
ID          PID          STATUS      BUNDLE      CREATED      0  
redis       474          running     /redis      2020-04-01T15:02:07.43938539Z  r
```

查看容器状态

复制

```
(MoeLove) → redis runc ps redis
PID    USER    TIME    COMMAND
  474  root      0:00  redis-server
(MoeLove) → redis runc state redis
{
  "ociVersion": "1.0.1-dev",
  "id": "redis",
  "pid": 474,
  "status": "running",
  "bundle": "/redis",
  "rootfs": "/redis/rootfs",
  "created": "2020-04-01T15:02:07.43938539Z",
  "owner": ""
}
```

进入容器

复制

```
(MoeLove) → redis runc exec redis redis-cli
127.0.0.1:6379> ping
PONG
```

总结

本篇，我为你介绍了一个很底层的容器运行时 runc，从上述的使用示例中你可以看到，它的基本操作与 docker 是很类似的。

但是它没有镜像下载之类的功能，这是由于 runc 的项目定位本就只涉及到了容器生命周期相关的内容，镜像管理并不在 runc 的项目目标中。

你可以通过修改 config.json 中的配置，来达到对容器的控制。(比较类似于 Dockerfile 中的一些参数)

下一篇，我将为你介绍如何参与 Docker 上游项目的开发，带你更深入地认识 Docker 项目。