Docker容器使用dumb-init作为进程supervisor和init系统

Published on May 3, 2022 UTC by Wang Ye <https://wangye.org>

经过一段时间的实践，目前我的IT基础设施基本上完成了向Docker容器化的迁移，通过容器的隔离大大减轻了运维的负担，尤其对于我个人来说，可以专注于业务代码的实现，在容器化的具体实践中也遇到一些问题，本文介绍的就是其中容器内进程管理和僵尸进程（Zombie Process）的消除。

1 Linux init进程

使用过Linux系统的同学可能对init这个进程并不陌生，此进程也是内核启动的第一个进程，PID为1，此进程包含多个任务，其中重要的任务之一：接管孤儿进程和消除僵尸进程。

孤儿进程指的是在其父进程执行完成或被终止后仍继续运行的一类进程，而僵尸进程是指子进程先于父进程结束，而父进程没有调用wait或者waitpid等待，这样结束的子进程将会变成僵尸进程，系统会自动将孤儿进程和僵尸进程过继给PID为1的init进程，这样init进程重新成为孤儿进程的父进程并给僵尸进程收尸（消除），避免大量的僵尸进程占用PID和消耗系统资源。

2 Docker容器的init实现

2.1 容器的PID 1号进程

我们所构建的Dcoker镜像往往最后的拉起的是最终执行的命令程序，例如nginx的Docker entrypoint命令可能是这样的：

```
ENTRYPOINT ["nginx", "-g", "'daemon off;'"]
```

Docker会将entrypoint命令进程作为PID 1号进程，例如上面nginx的例子，启动这个Docker通过观察启动的容器进程列表就会发现nginx的master进程作为PID 1号进程。

可以理解这个PID 1的进程就是容器内的init进程，那么对于同容器多进程的Docker应用该进程应当具备接管孤儿进程和消除僵尸进程的能力，除此以外还应当能够接收docker操作相应的信号并优雅（gracefully）的对子进程相应操作（转发信号），对于上文的nginx来说，nginx的架构master-slave就决定了master进程天然具备以上特性，所以将nginx的master进程设为容器的PID 1号进程并无不妥，除非你的nginx容器还需要运行其他进程，比如计划任务cron，这种情况下就不适合将nginx master进程设为1号进程了。

这时候我们需要有一种统一管理容器进程的能力，类似于系统默认的init进程。

2.2 使用**dumb-init**

dumb-init是Yelp开发的最简init系统用于实现容器内PID 1号进程，其重点解决了孤儿进程和消除僵尸进程接管和容器信号的转发。

dumb-init已经被打包进各大Linux发行版的软件库，可以直接通过`apt-get install dumb-init`（Debian、Ubuntu）或者`apk add dumb-init`（Alpine）这种形式安装。当然也可以直接通过源代码构建，这里就不详述了。

这里推荐两种调用方式：一是使用docker构建脚本的ENTRYPOINT指令；二是使用entrypoint脚本首行的shell bang约定调用。

1. 使用Docker构建脚本的ENTRYPOINT指令。该方式的Dockerfile的例子如下：

```
# Runs "/usr/bin/dumb-init -- /my/script --with --args"
ENTRYPOINT ["/usr/bin/dumb-init", "--"]

# or if you use --rewrite or other cli flags
# ENTRYPOINT ["dumb-init", "--rewrite", "2:3", "--"]

CMD ["/my/script", "--with", "--args"]
```

2. 使用entrypoint脚本的首行shell bang约定。该方式一般写在Docker构建使用的`entrypoint.sh`脚本的首行，例如：

```
#!/usr/bin/dumb-init /bin/sh
my-web-server &  # launch a process in the background
my-other-server  # launch another process in the foreground
```

通过dumb-init的帮助，我们可以优雅的实现多进程容器的良性生态，有效避免僵尸进程、停止容器进程收不到信号等问题，更多的使用方式可以参考dumb-init项目主页。当然类似的项目除了dumb-init还有很多，比如另外一个流行和常用的就是使用tini，关于tini和dumb-init使用方式类似，各位读者可以根据喜好选择。

3 总结

本人对于Linux容器化的部署实践也有一段时间，容器化确实大为减轻运维负担和消除平台兼容性，但是容器化也容易带来一些问题，本文所述就是我在部署容器过程中为解决某容器产生大量僵尸进程而进一步查阅资料找到的解决方案，希望对大家有所帮助。

External References (Links)

init这个进程
https://en.wikipedia.org/wiki/Init
dumb-init
https://github.com/Yelp/dumb-init
tini
https://github.com/krallin/tini