Building a Container Supervisor

Michael Crosby

dockercon 16

> whoami

Docker since 0.3 - maintainer

dockerui - author

libcontainer - author

nsinit - author

runc - author

OCI - maintainer

containerd - author

> man containerd

containerd

- Fast, lightweight container supervisor
- runc (OCI) multiplexer
- Container lifecycle operations
- rm -rf docker/daemon/execdrivers

> why

- runc integration
- Multiple runtime support
- Execution v2
- Decouple Execution from filesystem
- daemonless containers
- cleaner development

> events

- lock free event loop
- concurrency control
  - 10 > 100 at a time
- easier to new developers

Benchmarks

```
> ./benchmark -count 100
INFO[0001] 1.149902846 seconds
```

> container state

Managing state is easy when you don’t have any.

Don’t keep anything in memory.

Restore

```
> containerd --debug
DEBU[0000] containerd: container restored                id=0
DEBU[0000] containerd: container restored                id=1
DEBU[0000] containerd: container restored                id=2
DEBU[0000] containerd: container restored                id=3
DEBU[0000] containerd: container restored                id=4
DEBU[0000] containerd: container restored                id=5
DEBU[0000] containerd: container restored                id=6
...
```

> shim

- Exit code
- TTY / STDIO
- Reparenting to sysinit

> exit status

- Pipe + File
- O_CLOEXEC
- Multiple subscribers

O_CLOEXEC

```
if (mkfifo("exit-fifo", 0666) != 0) {
    printf("%s\n", strerror(errno));
    exit(EXIT_FAILURE);
}
int fd = open("exit-fifo", O_WRONLY | O_CLOEXEC, 0);
```

> stdio reattach

- FIFOs - the good, bad, and the stupid
- open() never blocks :trollface:
- fifos have a buffer
  - /proc/sys/fs/pipe-max-size

> re-parenting

- prctl - PR_SET_CHILD_SUBREAPER
- system init

> re-parenting rules

1. Your parent is the process that forked you, **your mommy**
2. If your parent dies, your new parent is PID 1*, **the creator**
3. If the parent(s) of your parent has the subreaper set, they will become your parent not PID 1, **your nana**
4. If you die then your parent dies before doing a wait4(), **you're a zombie**

PR_SET_CHILD_SUBREAPER

```
> ./parent
main() parent 27538
child process 27540 with parent 27539
parent 27539 exiting
child process 27540 with new parent 2391

> ps x | grep 2391
 2391 ?        Ss     0:00 /sbin/upstart --user
```

PR_SET_CHILD_SUBREAPER

```
> ./parent --subreaper
main() parent 27543
child process 27545 with parent 27544
parent 27544 exiting
child process 27545 with new parent 27543
```

> The OOM Problem

How do you connect to OOM notifications before the user process starts?

> runtime workflow

- create
  - initialize namespaces and config
- start
  - exec the user’s process
- delete
  - destroy the container

> runtime workflow

1. Create container
2. Register OOM handler
3. Exec the user's process

> code

https://github.com/crosbymichael/dockercon-2016

Thank you!