

支持wasm的shim汇总

- <https://github.com/containerd/runwasi>
 - containerd-shim-wasmtime-v1: 使用wasmtime作为engine
- <https://github.com/deislabs/containerd-wasm-shims>
 - This project aims to provide containerd shim implementations that can run Wasm / WASI workloads using **runwasi** as a library. This means that by **installing these shims onto Kubernetes nodes**, we can add a **runtime class** to Kubernetes and **schedule Wasm workloads on those nodes**.
 - spin shim: powered by **Fermyon Spin** engine. Spin is an open source framework for building and running fast, secure, and composable **cloud microservices with WebAssembly**.
 - 支持两种触发方式: http, redis
 - slight(SpiderLightning) shim: In addition, the slight shim comes with an increasing array of WebAssembly component capabilities, including underlying implementations, to consume **common application level services**.

支持wasm的shim汇总

- <https://github.com/dmccgowan/containerd-wasm>
 - 容器中的进程执行`wasmtime xxx.wasm`来实现

性能对比测试

分别用wasmtime与wasmedge制作的镜像（使用runc运行）

- 使用wasmedge是相对好一点
 - 体积小、性能好
 - wasmtime没有提供类似wasmedge-slim的docker镜像
- 分别用runc运行两个镜像（左 wasmtime，右 wasmedge）

```
root@VM-0-17-ubuntu:~/test-perf# \time -v ./test-wasmtime.sh
Command exited with non-zero status 1
  Command being timed: "./test-wasmtime.sh"
    User time (seconds): 4.67
    System time (seconds): 2.85
Percent of CPU this job got: 10%
Elapsed (wall clock) time (h:mm:ss or m:ss): 1:12.07
Average shared text size (kbytes): 0
Average unshared data size (kbytes): 0
Average stack size (kbytes): 0
Average total size (kbytes): 0
Maximum resident set size (kbytes): 27748
Average resident set size (kbytes): 0
Major (requiring I/O) page faults: 0
Minor (reclaiming a frame) page faults: 539523
Voluntary context switches: 171679
Involuntary context switches: 33215
Swaps: 0
File system inputs: 0
File system outputs: 0
Socket messages sent: 0
Socket messages received: 0
Signals delivered: 0
Page size (bytes): 4096
Exit status: 1
```

```
ot@VM-0-17-ubuntu:~/test-perf# \time -v ./test-runc.sh
Command being timed: "./test-runc.sh"
  User time (seconds): 4.35
  System time (seconds): 2.91
Percent of CPU this job got: 17%
Elapsed (wall clock) time (h:mm:ss or m:ss): 0:41.33
Average shared text size (kbytes): 0
Average unshared data size (kbytes): 0
Average stack size (kbytes): 0
Average total size (kbytes): 0
Maximum resident set size (kbytes): 27776
Average resident set size (kbytes): 0
Major (requiring I/O) page faults: 0
Minor (reclaiming a frame) page faults: 537035
Voluntary context switches: 161602
Involuntary context switches: 34059
Swaps: 0
File system inputs: 0
File system outputs: 0
Socket messages sent: 0
Socket messages received: 0
Signals delivered: 0
Page size (bytes): 4096
Exit status: 0
```

TAG

[wasmtime4](#)

Last pushed 14 minutes ago by [leviyanx](#)

DIGEST

[3e70dd71805b](#)


OS/ARCH

linux/amd64


COMPRESSED SIZE ⓘ

33.56 MB

docker pull leviyanx/runc-wa...



docker pull leviyanx/runc-wa...



COMPRESSED SIZE ⓘ
3.62 MB

性能对比测试

- crun(wasmedge) vs containerd-wasm-shim (wasmtime)
 - 左：crun运行仅含wasm文件的镜像（wasmedge），右：shim运行运行仅含wasm文件的镜像（wasmtime）
 - crun with wasmedge是oci运行时，containerd-wasm-shim不是oci运行时

```
root@VM-0-17-ubuntu:~/test-perf# \time -v ./test-crun.sh
./test-crun.sh
Command exited with non-zero status 1
  Command being timed: "./test-crun.sh"
  User time (seconds): 2.65
  System time (seconds): 1.75
  Percent of CPU this job got: 82%
  Elapsed (wall clock) time (h:mm:ss or m:ss): 0:05.32
  Average shared text size (kbytes): 0
  Average unshared data size (kbytes): 0
  Average stack size (kbytes): 0
  Average total size (kbytes): 0
  Maximum resident set size (kbytes): 26360
  Average resident set size (kbytes): 0
  Major (requiring I/O) page faults: 0
  Minor (reclaiming a frame) page faults: 493704
  Voluntary context switches: 48480
  Involuntary context switches: 19677
  Swaps: 0
  File system inputs: 0
  File system outputs: 0
  Socket messages sent: 0
  Socket messages received: 0
  Signals delivered: 0
  Page size (bytes): 4096
  Exit status: 1
```

```
root@VM-0-17-ubuntu:~/test-perf# \time -v ./test-containerd-wasm-shim.sh
./test-containerd-wasm-shim.sh
Command exited with non-zero status 1
  Command being timed: "./test-containerd-wasm-shim.sh"
  User time (seconds): 2.58
  System time (seconds): 1.80
  Percent of CPU this job got: 82%
  Elapsed (wall clock) time (h:mm:ss or m:ss): 0:05.31
  Average shared text size (kbytes): 0
  Average unshared data size (kbytes): 0
  Average stack size (kbytes): 0
  Average total size (kbytes): 0
  Maximum resident set size (kbytes): 26476
  Average resident set size (kbytes): 0
  Major (requiring I/O) page faults: 0
  Minor (reclaiming a frame) page faults: 492764
  Voluntary context switches: 48066
  Involuntary context switches: 19545
  Swaps: 0
  File system inputs: 0
  File system outputs: 0
  Socket messages sent: 0
  Socket messages received: 0
  Signals delivered: 0
  Page size (bytes): 4096
  Exit status: 1
```

Containerd shim api

- <https://github.com/containerd/containerd/blob/main/runtime/v2/README.md>
 - **I/O** for a container is provided by the client to the shim via fifo on Linux, named pipes on Windows, or log files on disk.
 - **root filesystem** for the containers (Shims are responsible for **mounting the filesystem** into the rootfs directory of the bundle)
 - **Events**

Tasks

Topic	Compliance	Description
runtime.TaskCreateEventTopic	MUST	When a task is successfully created
runtime.TaskStartEventTopic	MUST (follow TaskCreateEventTopic)	When a task is successfully started
runtime.TaskExitEventTopic	MUST (follow TaskStartEventTopic)	When a task exits expected or unexpected
runtime.TaskDeleteEventTopic	MUST (follow TaskExitEventTopic or TaskCreateEventTopic if never started)	When a task is removed from a shim

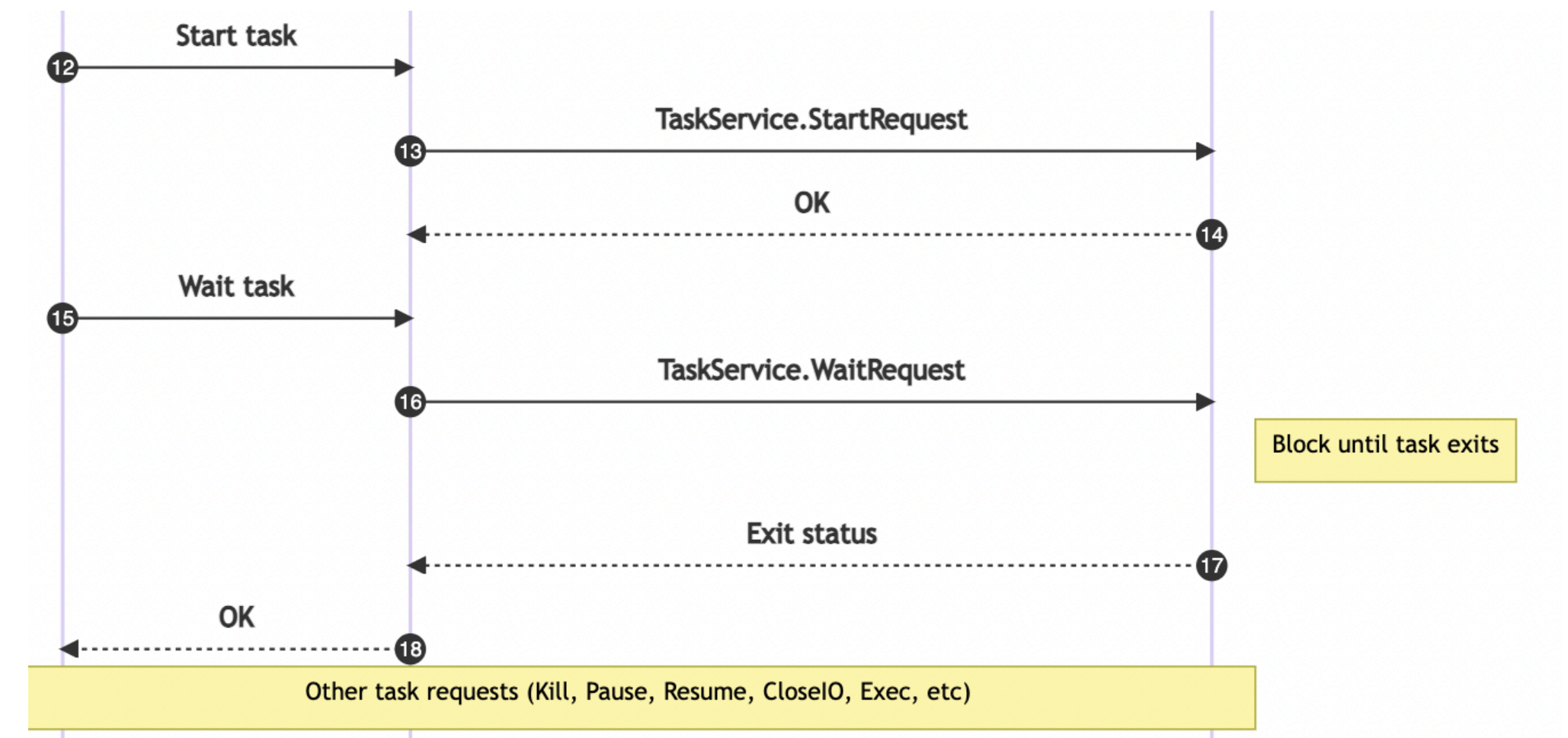
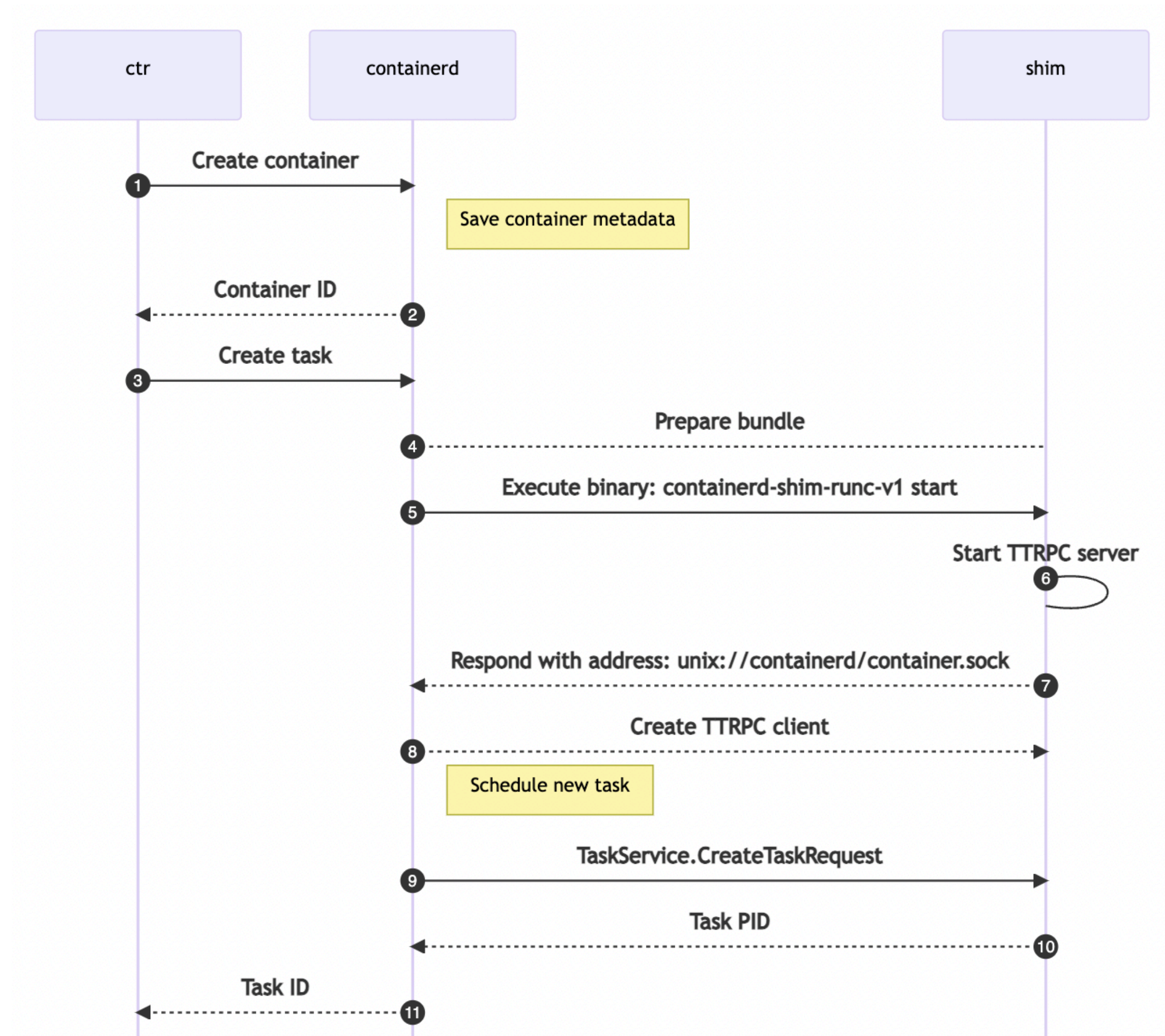
- task

execs

Topic	Compliance	Description
runtime.TaskExecAddedEventTopic	MUST (follow TaskCreateEventTopic)	When an exec is successfully added
runtime.TaskExecStartedEventTopic	MUST (follow TaskExecAddedEventTopic)	When an exec is successfully started

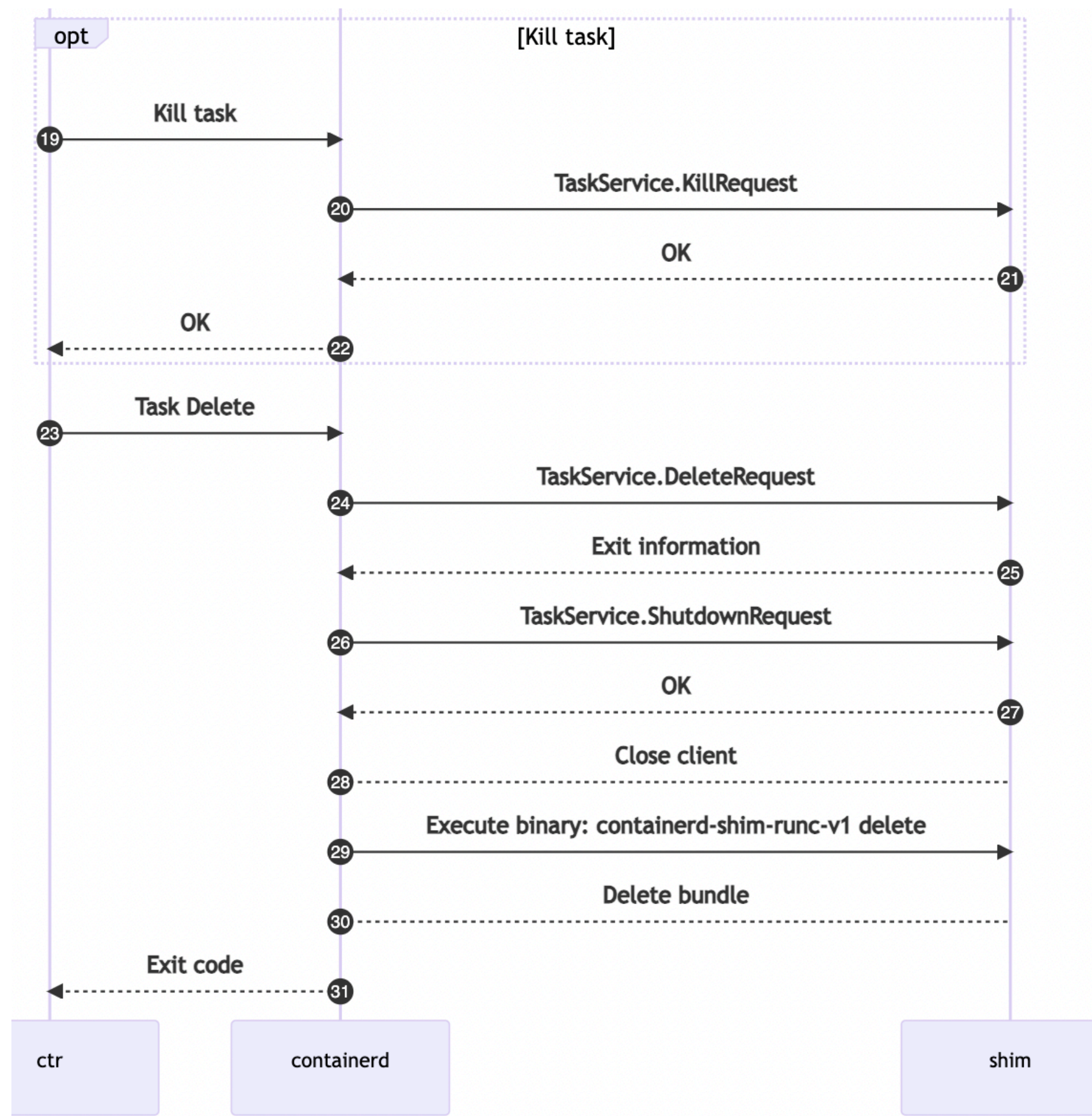
Containerd shim api

flow of `ctr run`



runwasi

flow of `ctr run`



runwasi

runwasi如何实现shim api

- 思想
 - 假装container，但是一些工作并不实际去做
 - // When a cri sandbox is specified we just assume it's the sandbox container and treat it as such by **not actually running the code** (which is going to be wasm).
 - // this is **janky** since it is internal data, but check that this is seen as a "real" container
 - 如果遇到传进来的是一个一般的cri container，就不处理
 - // If it is cri, then this is the "pause" container, which **we don't need to deal with**
 - rootfs实际并没有挂载根目录（未实现）
 - cgroup;