Run containerd in the Docker daemon

- Enable embedded containerd
- Verify the configuration
- Connect directly to embedded containerd

Starting with Docker Engine 29.7.0, the Docker daemon can run containerd in the same process as `dockerd`. By default, the daemon starts and manages containerd as a separate process.

> Embedded containerd is an experimental feature. Its behavior may change, or the feature may be removed from a future release.

Embedded mode changes how the daemon starts and communicates with containerd. The embedded server still listens on a containerd socket, but the daemon and BuildKit reach it over an in-memory connection instead. Task shims continue to run as separate processes and connect over a socket.

The performance benefit is greatest when Docker uses the containerd image store. Any image interactions use containerd's content API for both metadata requests and the bytes that make up image manifests, configurations, and layers. A separate containerd process splits these reads and writes into gRPC messages that cross a Unix socket or named pipe. Embedded mode sends the same messages over an in-memory connection, avoiding operating system socket calls and kernel transport for this data-heavy path.

The embedded server doesn't include the container runtime interface (CRI). You can't enable `embedded-containerd` and `cri-containerd` at the same time.

Enable embedded containerd

Add the `embedded-containerd` feature to the daemon configuration file:

```
{
  "features": {
    "embedded-containerd": true
  }
}
```

Restart the Docker daemon:

```
$ sudo systemctl restart docker
```

The `embedded-containerd` feature takes precedence over a containerd address set with the `--containerd` daemon flag. This behavior means that packaged service configurations that set a containerd address don't prevent embedded containerd from starting.

> Embedded containerd keeps its state under the Docker data root, in `/var/lib/docker/containerd/daemon` by default. A containerd installed on the host keeps its own state elsewhere, such as `/var/lib/containerd`. If the daemon used such a containerd before, the containers and images stored there aren't available in embedded mode.

You can also enable the feature when starting `dockerd` manually:

```
$ sudo dockerd --feature embedded-containerd
```

Verify the configuration

Run `docker info` and check for the experimental mode warning:

```
$ docker info
...
NOTE: Running with experimental embedded-containerd mode. In a future release,
this mode may be used by default when no system containerd is available, rather
than starting and supervising a separate containerd process. The option used to
enable this mode may also change.
```

Connect directly to embedded containerd

The embedded server provides an endpoint for containerd clients such as `ctr` and `nerdctl`. On Linux, the endpoint is `<exec-root>/containerd/containerd.sock`. With the default daemon configuration, this path is:

```
/var/run/docker/containerd/containerd.sock
```

On Windows, the server uses a named pipe by default. Check the Docker daemon startup logs for the endpoint address.

Docker Engine stores containers in the `moby` containerd namespace by default. For example, use `ctr` on Linux to list them:

```
$ sudo ctr --address /var/run/docker/containerd/containerd.sock \
  --namespace moby containers list
```

> The endpoint is useful for debugging, but the daemon owns the state behind it. Don't treat it as a general-purpose containerd endpoint: its address and the layout of the namespaces can change, and changes that other clients make can conflict with the daemon.

□□□ #2
□ □□□ □□□ 2026-10-31 19:22:09 UTC
□ □□□ □□□ 2026-10-31 19:22:10 UTC