

What is Rocket (rkt)?

rkt (pronounced "rock-it") is a app container runtime on Linux. rkt is designed to be secure, composable, and standards-based.

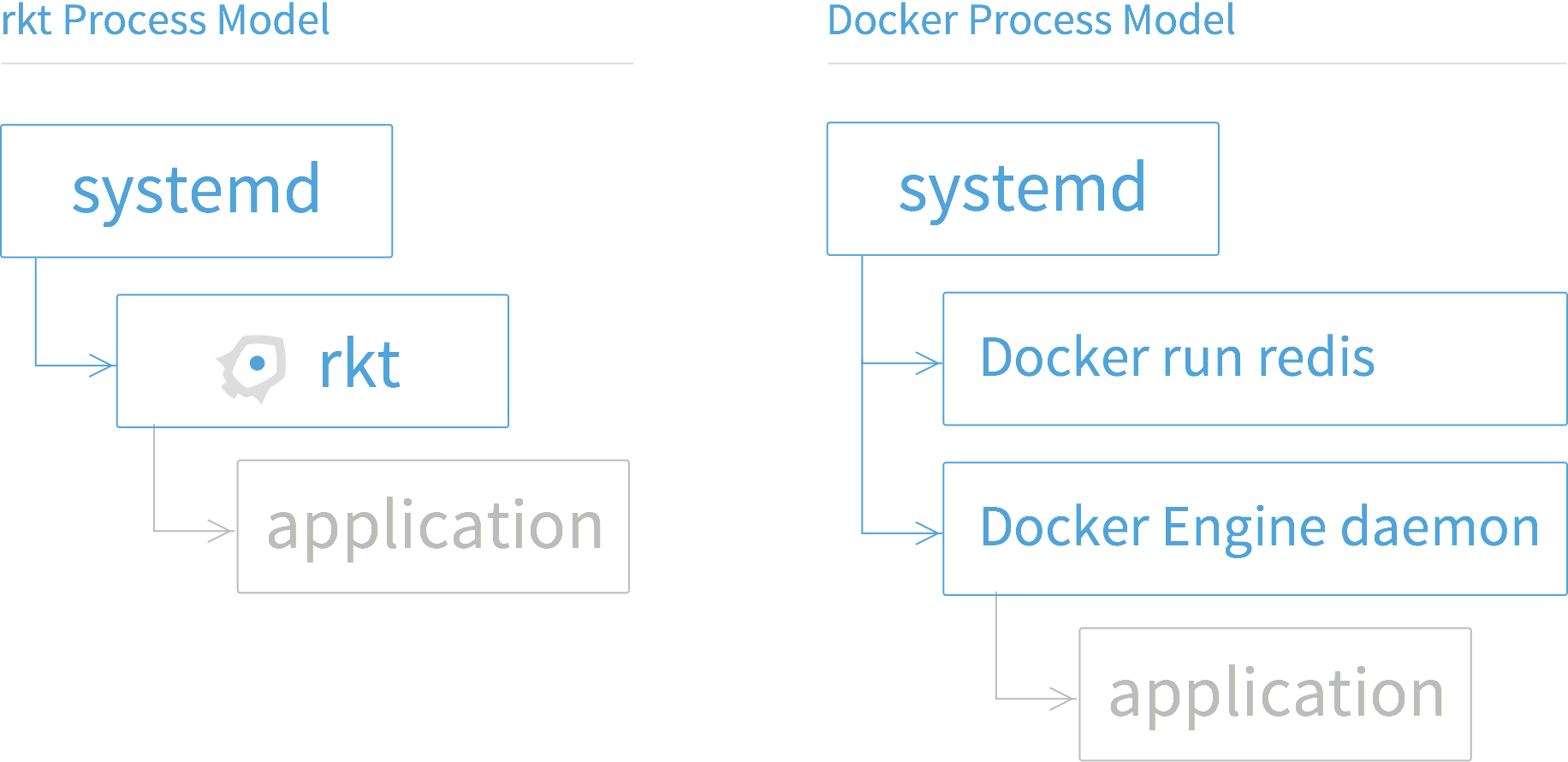
* *Security*: rkt is developed with a principle of "secure-by-default", and includes a number of important security features like support for [SELinux](https://github.com/coreos/rkt/blob/master/Documentation/selinux.md), [TPM measurement](https://github.com/coreos/rkt/blob/master/Documentation/devel/tpm.md), and running app containers in [hardware-isolated VMs](https://github.com/coreos/rkt/blob/master/Documentation/running-lkvm-stage1.md).
* *Composability*: rkt is designed for first-class integration with init systems ([systemd](https://github.com/coreos/rkt/blob/master/Documentation/using-rkt-with-systemd.md), upstart) and cluster orchestration tools (fleet, [Kubernetes](https://github.com/coreos/rkt/blob/master/Documentation/using-rkt-with-kubernetes.md), [Nomad](https://github.com/coreos/rkt/blob/master/Documentation/using-rkt-with-nomad.md)), and supports [swappable execution engines](https://github.com/coreos/rkt/blob/master/Documentation/devel/architecture.md).
* *Open standards and compatibility*: rkt implements the [appc specification](https://github.com/coreos/rkt/blob/master/Documentation/app-container.md), supports the [Container Networking Interface specification](https://github.com/appc/cni), and can also run [Docker images](https://github.com/coreos/rkt/blob/master/Documentation/running-docker-images.md).

**rkt is an implementation of the “App Container” spec.**

"App Container" (appc) is a specification describing how applications can be packaged, distributed, and executed in a portable and self-contained way. The specification defines an image format, an image discovery mechanism, a deployable grouping, and an execution environment.

Core goals of specification are

* Designing for fast downloads and starts of App Containers
* Ensuring images are cryptographically verifiable and highly cacheable
* Designing for composability and independent implementations
* Using common technologies for cryptography, archiving, compression and transport
* Using the DNS namespace to name and discover images



Installation

|  |
| --- |
| wget https://github.com/coreos/rkt/releases/download/v1.1.0/rkt-v1.1.0.tar.gz  tar xzvf rkt-v1.1.0.tar.gz  cd rkt-v1.1.0  ./rkt help |

For Non root usage

|  |
| --- |
| sudo groupadd rkt  export WHOAMI=$(whoami); sudo gpasswd -a $WHOAMI rkt  sudo ./scripts/setup-data-dir.sh |

rkt's native image format (ACI) and runtime/execution environment (pods) are defined in the App Container spec.

The image format defined by App Container (appc) and used in rkt is the **Application Container Image**, or **ACI**. An ACI is a simple tarball bundle of a rootfs (containing all the files needed to execute an application) and an Image Manifest, which defines things like default execution parameters and default resource constraints

To build ACI images we need to use acbuild cli tool

Download acbuild binary from <https://github.com/appc/acbuild/releases>

Or build it yourself

git clone <https://github.com/appc/acbuild>

cd acbuild

./build

./bin/acbuild help

Sample script

|  |
| --- |
| #!/usr/bin/env bash  acbuild begin  acbuild set-name example.com/nginx  acbuild dependency add registry-1.docker.io/library/nginx:latest  acbuild mount add html /usr/share/nginx/html  acbuild port add http tcp 80  acbuild set-exec -- /usr/sbin/nginx -g "daemon off;"  acbuild write nginx.aci  acbuild end |

Execute the ACI using rkt

|  |
| --- |
| rkt run --insecure-options=image ./nginx.aci --volume html,kind=host,source=/root/html/test --port=http:8888 |

Create sample html file

echo “Hello, world! from rkt” >> html/test/index.html

Verify using

curl localhost:8888