BuildKit CLI for kubectl

Daniel Hiltgen & Patrick Devine
VMware Inc.

How did you build?


```
$ docker build -t myimage:v1 ./

$ docker run --rm -it myimage:v1
```

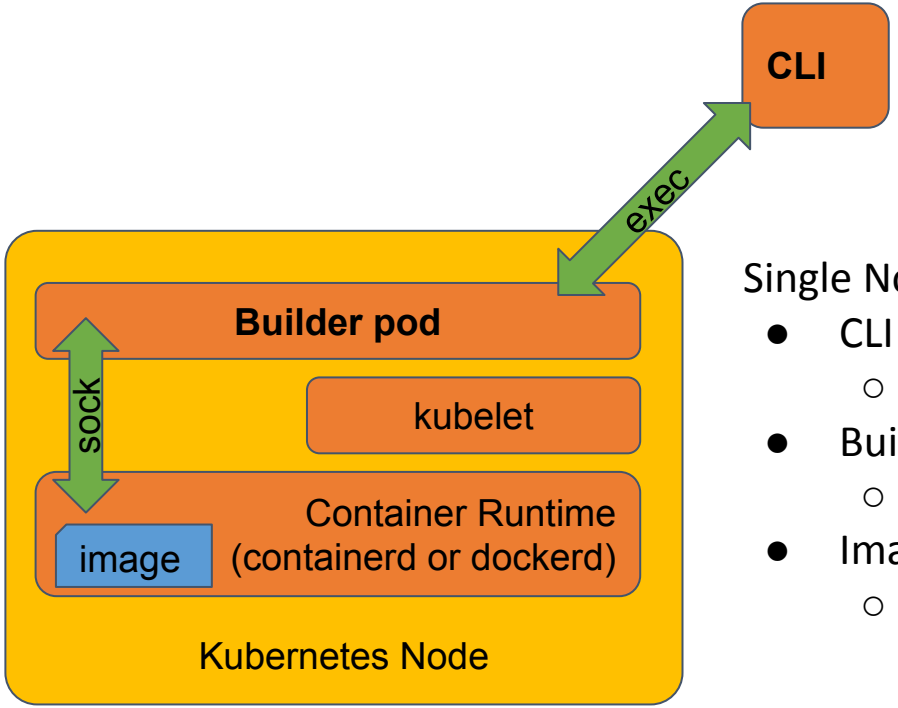
How do you build in K8s?


```
$ kubectl build -t myimage:v1 ./

$ kubectl run --rm -it --image myimage:v1 mypod
```

Live demo of kubectl build

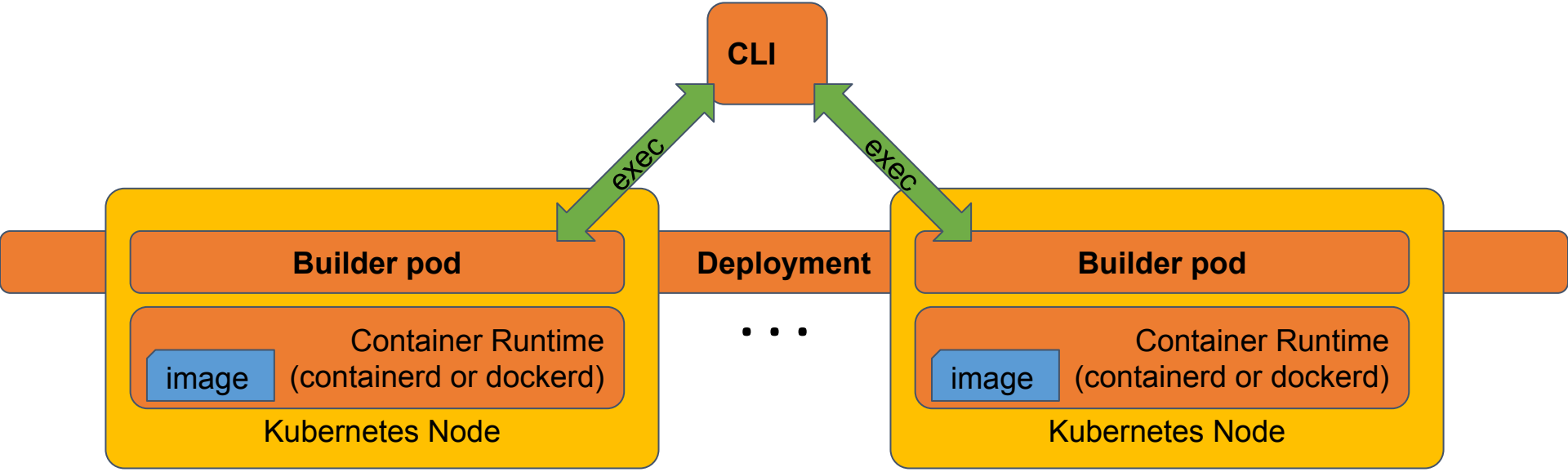
How does it work?



Single Node Highlights

- CLI “exec’s” into Builder Pod
  - Bidirectional BuildKit gRPC API
- Builder Pod mounts runtime sock
  - Default behavior, but optional
- Image loaded into runtime, visible to kubelet
  - W/o sock, disables runtime image loading

How does it work?



Multi Node Highlights

- Builder Deployment Pods use anti-affinity to distribute across nodes
- CLI “exec’s” into all pods
- Image built on one node
- After build, image replicated to other node runtimes through CLI

Why did we build this tool?

- Lots of tools in the “kubernetes image build” space
- None as popular as “docker build”
  - “docker build” - extremely easy and “just works”
  - Others often require a registry, adding complexity
  - No setup, single CLI command, images immediately available locally
- Container Runtime Evolving
    - Industry moving away from dockerd, to containerd and others
    - docker CLI (and thus build UX) tied to dockerd runtime
    - What happens to scripts/automation?
    - What happens to users “muscle memory”

Solution: Drop in replacement for “docker build” via kubectl

Power of BuildKit + K8s

BuildKit - powerful, modern toolkit to convert source to build artifact

- Compatible with latest Dockerfile features
- Fast, efficient local builds, smart build context handling
- Multi-architecture support (x86, arm, etc.)
- Advanced Caching support (local and via registry)
- Build Farms

https://github.com/moby/buildkit

Inner Loop Demo

Live demo of inner loop showing quick developer iteration

Multi-arch Support

- How do you make `kubectl run -it --rm --image=myimage:v1 mypod` work in a mixed multi-arch cluster? **Need a multi-arch image**

- Three different ways to multi-arch:
  1. Native builds (not yet)
  2. Emulated environments (not yet)
  3. **Cross-compilation (works now)**

- Have to use a registry

Multi-arch Support


```
FROM --platform=$BUILDPLATFORM golang:1.14-alpine AS builder
...
WORKDIR /project
ARG TARGETOS
ARG TARGETARCH
ENV GOOS=$TARGETOS GOARCH=$TARGETARCH
RUN CGO_ENABLED=0 go build -a -ldflags '-extldflags "-static"' -o thisisfine tif.go

FROM scratch AS release-linux
COPY --from=builder /project/thisisfine /thisisfine
ENTRYPOINT ["/thisisfine"]

FROM mcr.microsoft.com/windows/nanoserver:1809 AS release-windows
COPY --from=builder /project/thisisfine /thisisfine.exe
ENTRYPOINT ["\\thisisfine.exe"]

FROM release-$TARGETOS
```

$ kubectl build ./ -t ghcr.io/pdevine/thisisfine -f Dockerfile.cross --push --platform=linux/amd64,linux/arm/v7,linux/arm64,windows/amd64

What flavours of K8s?

- Versions: 1.14+
- Runtimes: containerd and dockerd
- Distros: most “just work”
    - K3d and similar with limitations (push only, no image loading)

Try it out!

- New project: v0.1.x
- Native Packaging available
  - brew, choco, deb, rpm
- https://github.com/vmware-tanzu/buildkit-cli-for-kubectl#getting-started

- If you have a kubernetes environment, give it a try!
- File issues for bugs or feature requests
- Pull Requests welcome!