Chainguard

Modern Build Systems for Containers

FOSDEM
3 Feb 2024

Adrian Mouat
Technical Community Advocate @ Chainguard


ISUZU
BT

Docker is doing to apt what apt did to tar

Bryan Cantrill
@bcantrill@mastodon.social

Container Images are Packaging

- Just a filesystem and some metadata
- Not that difficult to create
  - Running and maintaining them might be another story
  - And a different talk

Old Skool Docker Build

```
FROM golang

WORKDIR /work

COPY . /work

RUN go build -o hello ./cmd/server

ENTRYPOINT ["/work/hello"]
```

Problems

- Big image
  - 100s of MBs
- CVEs
  - At least according to scanners
- Poor reproducibility

Distroless Multistage Docker Build

```
FROM cgr.dev/chainguard/go:latest-dev as builder
WORKDIR /work
COPY . /work/
RUN CGO_ENABLED=0 go build -o hello ./cmd/server

#FROM gcr.io/distroless/static-debian11
FROM cgr.dev/chainguard/static
COPY --from=builder /work/hello /hello
ENTRYPOINT ["/hello"]
```

Distroless Build

- Much smaller!
- No CVEs
- Still not reproducible
- Need to have runtime image that works for you
  - Static, Node, Ruby, Java, Python…

KO Punch

- https://github.com/ko-build/ko
- Just do
  - `ko build ./cmd/app`
- No Docker

hachyderm.io/@adrianmouat

Hang On, How Is Distroless Made?

- GCT Distroless
  - Built with Bazel
  - Basically took Debian and hacked things out
- Chainguard Distroless
  - Apko & Wolfi (and Alpine)

OK, Show me Bazel then

- You asked for it
- https://github.com/aspect-build/bazel-examples/tree/main/oci_go_image

Bazel

Bazel

- Excellent reproducibility
- Fast
- Can build minimal images
- But it's a bit a of a beast
  - Not for faint-hearted
- Bring your own base image

OK, Show me Apko

```
contents:
  keyring:
    - https://packages.wolfi.dev/os/wolfi-signing.rsa.pub
  repositories:
    - https://packages.wolfi.dev/os
  packages:
    - ca-certificates-bundle
    - wolfi-base
entrypoint:
  command: /bin/sh -l
```

Apko

- Simple
- Declarative
- Reproducible
- Low CVE
- Composes well with Dockerfiles

Apko

- Dependent on Alpine/Wolfi
- Create own packages with melange

- Also see `rules_apko` for Bazel!


ISUZU
ISUZU

Canonical Chiselled Containers

- Canonical version of distroless
- Minimal
- Limited number of images
  - .net
  - JRE
- https://hub.docker.com/u/ubuntu

Canonical Chiseled Containers

- Idea of "slices"
  - "chisel" bits out of packages
    - Instead of having modular packages
  - Seems very manual
- It's own ecosystem

Buildpacks

- Aim to be easy to use
- Pick up requirements.txt etc
- Doesn't seem to produce small images
  - Or low CVE ones
  - It could?
- Feels a bit one size fits all

Buildpacks.io

Buildkit and Dagger

- Buildkit is powerful
  - Dockerfile just scratches the surface
- Dagger takes advantage of this
- Designed for CI/CD
  - Solve "Works on my machine"

Dagger

Dagger Example

```
Contents, err := client.Container().
        From("alpine:latest").
        WithDirectory("/src", project,
         dagger.ContainerWithDirectoryOpts{
           Include: []string{"*.md"},}).
        WithWorkdir("/src").
        WithExec([]string{"ls", "/src"}).
        Stdout(ctx)
```

Dagger

- Similar level to Bazel?
  - But simpler
- Building out plugins
  - Apko!
  - But also apk

Nix

- Don't need to understand all of Nix
- Or even install it
- Two approaches
  - pkgs.DockerTools
  - flakes

NixOS

Nix pkgs.DockerTools

```
pkgs.dockerTools.buildImage {
    name = "redis";
    tag = "latest";
    fromImage = null; #debian/bash
    copyToRoot = pkgs.buildEnv {
      name = "image-root";
      paths = [ pkgs.redis ];
      pathsToLink = [ "/bin" ];
    };
```

```
runAsRoot = ''
mkdir -p /data
config = {
    Cmd = ["/bin/redis-server"];
    WorkingDir = "/data";
        Volumes = {
            "/data" = {};
        };
     };
}
```

Nix pkgs.DockerTools

- I couldn't get to work on MacOS
  - Required KVM?
- Fully reproducible?
- Minimal?
- Full programming language
- Nix

Nix Flakes

- Nix creates stand-alone systems for everything
  - So why not just ship that in an image?
- Use with Dockerfile
- Simpler
- Works on my Mac
- https://mitchellh.com/writing/nix-with-dockerfiles

Nix Flakes

- Halfway Reproducible
  - Nix tries to be totally reproducible
  - Docker build will cause issues
- Should be able to make minimal
  - Not sure why there's a shell etc
  - Need to play more

Nix Flakes

- Final image is a bit weird
  - Not your typical filesystem
  - Makes debugging more difficult
    - E.g. how to set entrypoint to shell?

OK, So What Do You Recommend?

Org-wide Solutions

- Bazel
  - If you want the guarantees it gives you
- Dagger
  - If CI/CD is a pain
    - (it probably is)

Smaller Single Project Solutions

- Multistage Docker Build with distroless
- Apko
  - Simple, generic, reproducible
  - I'm biased
- Otherwise ecosystem specific e.g. ko, jib
- Nix? Maybe!


Peter Vardy
Peter Vardy
GUINNESS
SIX NATIONS