chainguard

Workshop:
The Safe source for Open Source

Peter Andersson

peter.andersson@chainguard.dev

- Enterprise Sales Engineer Chainguard

- Partner SE Broadcom
- Sales Engineer Sysdig
- Technical Strategist SUSE
- Enterprise SE Black Duck
- Product Manager op5
-
- https://www.linkedin.com/in/itslav/

Spare time

- Just joined Home Guard
- Sailing
- Fiddling with SW and HW
- Favorite project now:
  Signal-K, connect different protocols in a boat

chainguard

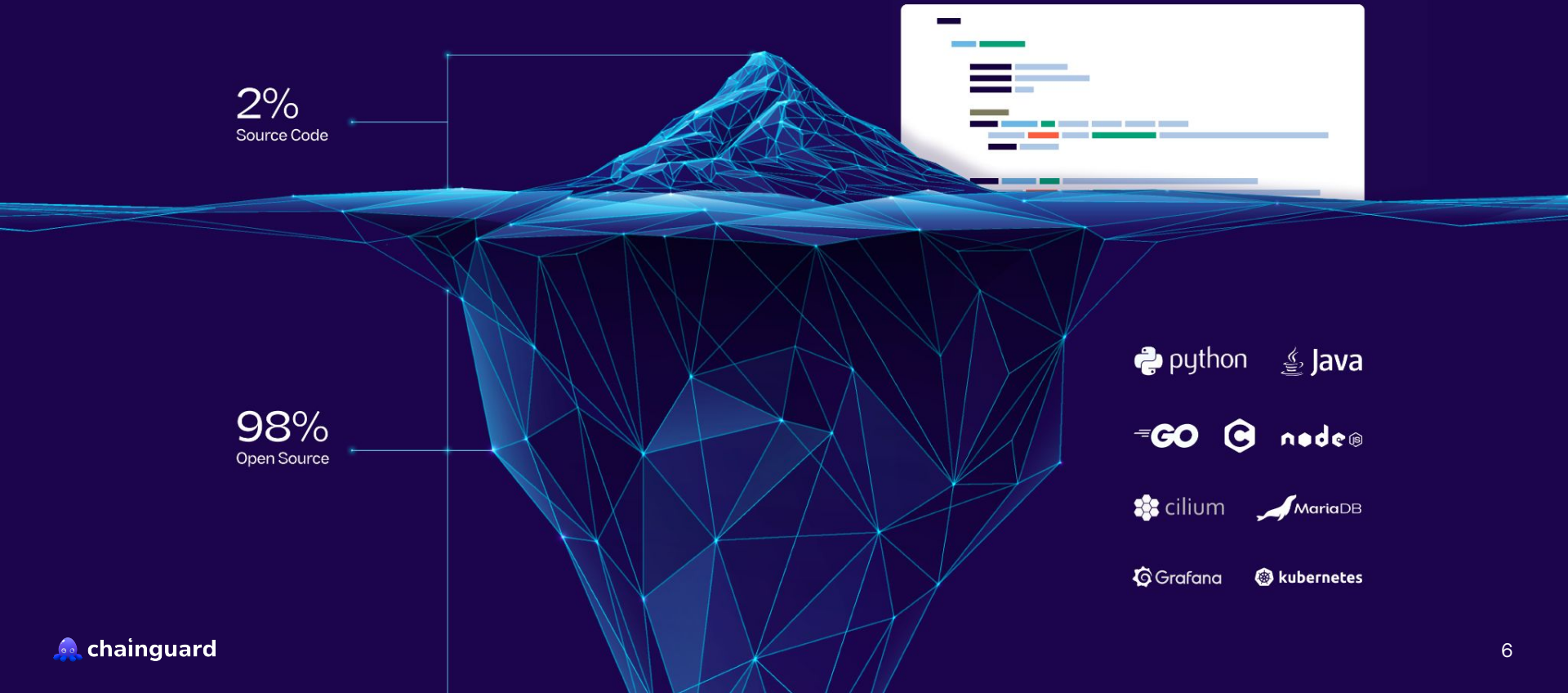
Agenda

- Intro to CVEs
- Intro to free Chainguard Images
- Show our example image
- Migrate to free Chainguard equivalent
- Using Multistage Builds
- Troubleshoot distroless containers
- Signing an image
- Scanners and more
- Q&A

Prerequisites

- Docker installed
  - Podman should also work
- Git installed
- Bash >4
- Trivy
- Grype
- Syft
- Cosign
- Jq
- Docker hub account

Open Source Software Has Transformed Software Development



artbleed Bug

d Bug is a serious vulnerability in the popular OpenSSL
software library. This weakness allows stealing the
otected, under normal conditions, by the SSL/TLS
d to secure the Internet. SSL/TLS provides communication
ivacy over the Internet for applications such as web, email,
ing (IM) and some virtual private networks (VPNs).

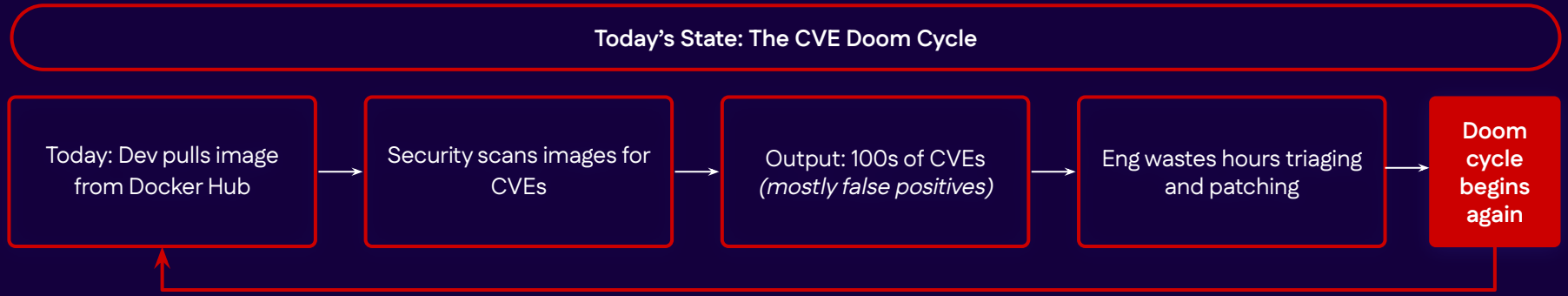
d bug allows anyone on the Internet to read the memory of
otected by the vulnerable versions of the OpenSSL
compromises the secret keys used to identify the service
o encrypt the traffic, the names and passwords of the
ctual content. This allows attackers to eavesdrop
s, steal data directly from the services and users and to
rvices and users.

HEARTBLEED PROJECT

SCAN

Vulnerability	Severity	Package
CVE-2018-5709	Negligible	krb5
CVE-2018-7738	Negligible	util-linux
CVE-2016-10228	Negligible	glibc
CVE-2019-7309	Negligible	glibc
E-2017-7245	Negligible	pcre3
7-7246	Negligible	pcre3
0654	Negligible	libtasn1-6
	Medium	krb5
	Medium	glibc
CV	Medium	libonig
CVE-2019-1	Medium	gnupg2

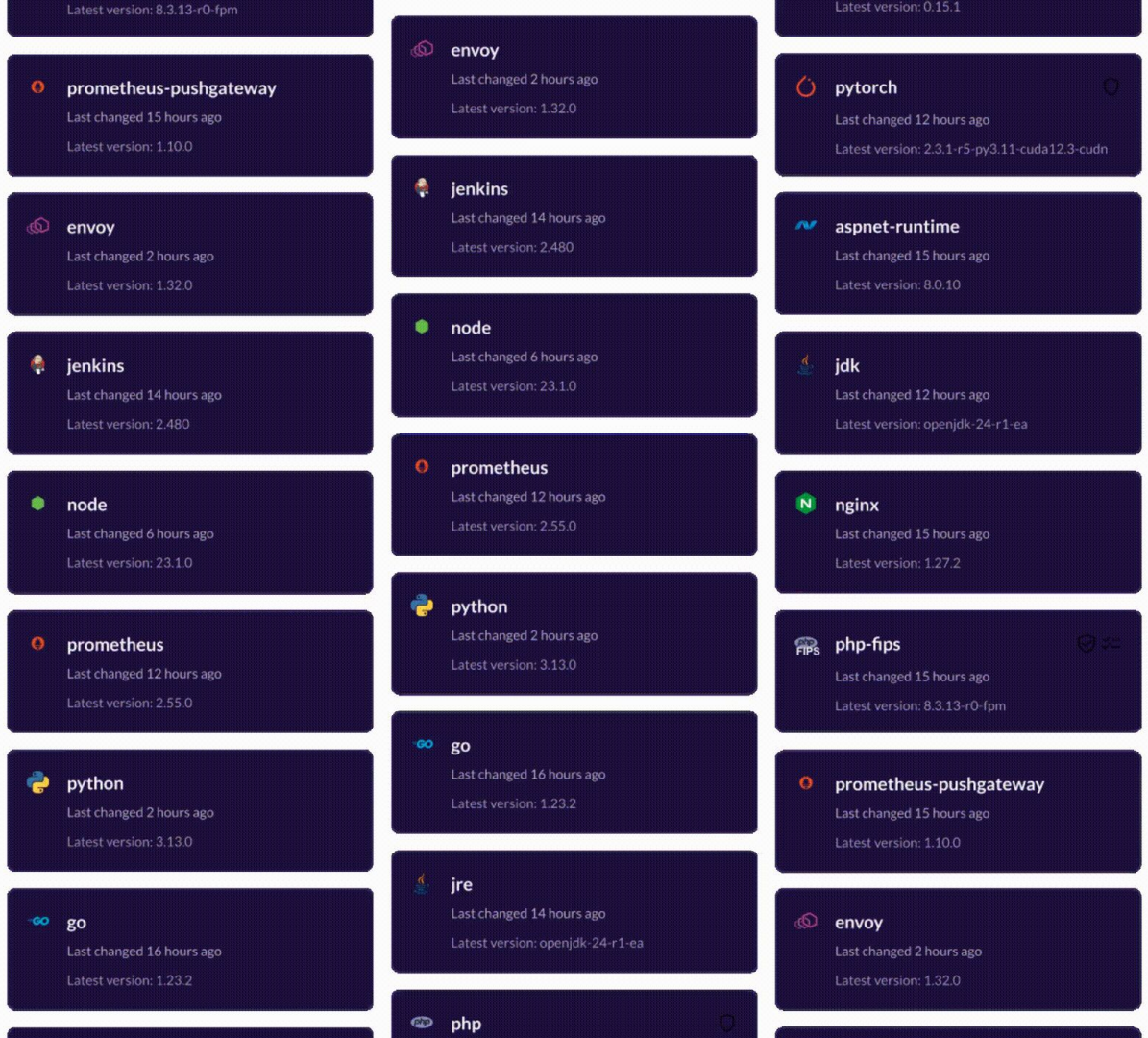
The Status Quo for CVE Management is Deeply Broken...



Chainguard Images

- Dedicated OS-Level STIG
- Kernel Independent FIPS
- HTML OSCAP Scan Reports
- SLAs for CVE Remediation
- Zero CVEs
- Minimal Attack Surface
- All Maintained Versions
- SBOMs and Attestation

chainguard



Chainguard

`cgr.dev/chainguard-private/python:latest`

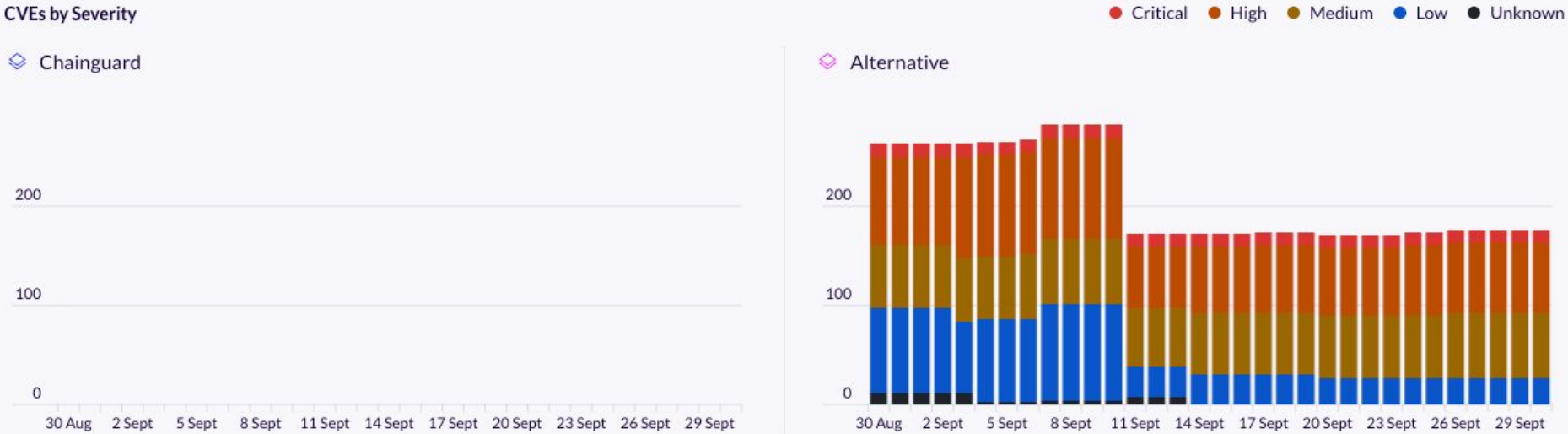
Latest CVE count	Daily average	Compressed size
0	0	22.39 MB

Alternative

`python:latest`

Latest CVE count	Daily average	Compressed size
176	210	392.85 MB

CVEs by Severity



~~Shift Left.~~ Start Left.

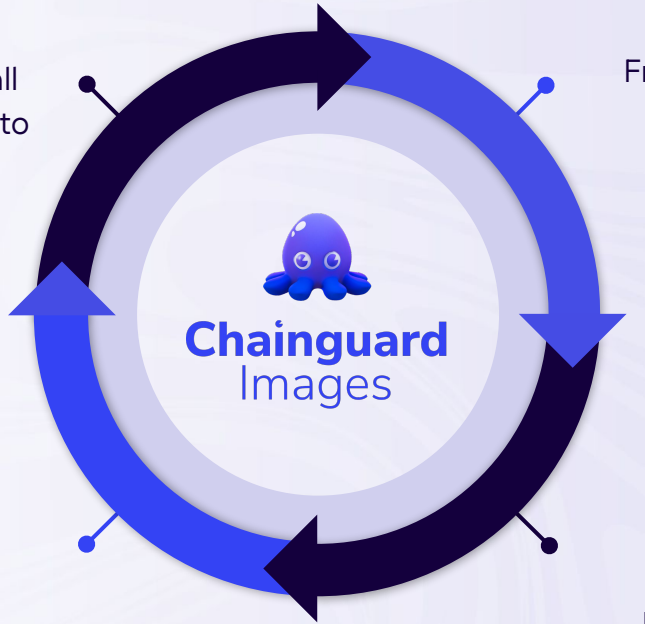
Delivered & Verified

Images with SBOMs attestations all signed with Sigstore and delivered to your registry of choice.

Rebuilt Daily

From upstream open source projects and minimized.

Wolfi



Scan & Patch CVEs

To fix any known or new vulnerabilities.

Check Behavior

Changes in behavior between package versions are checked.

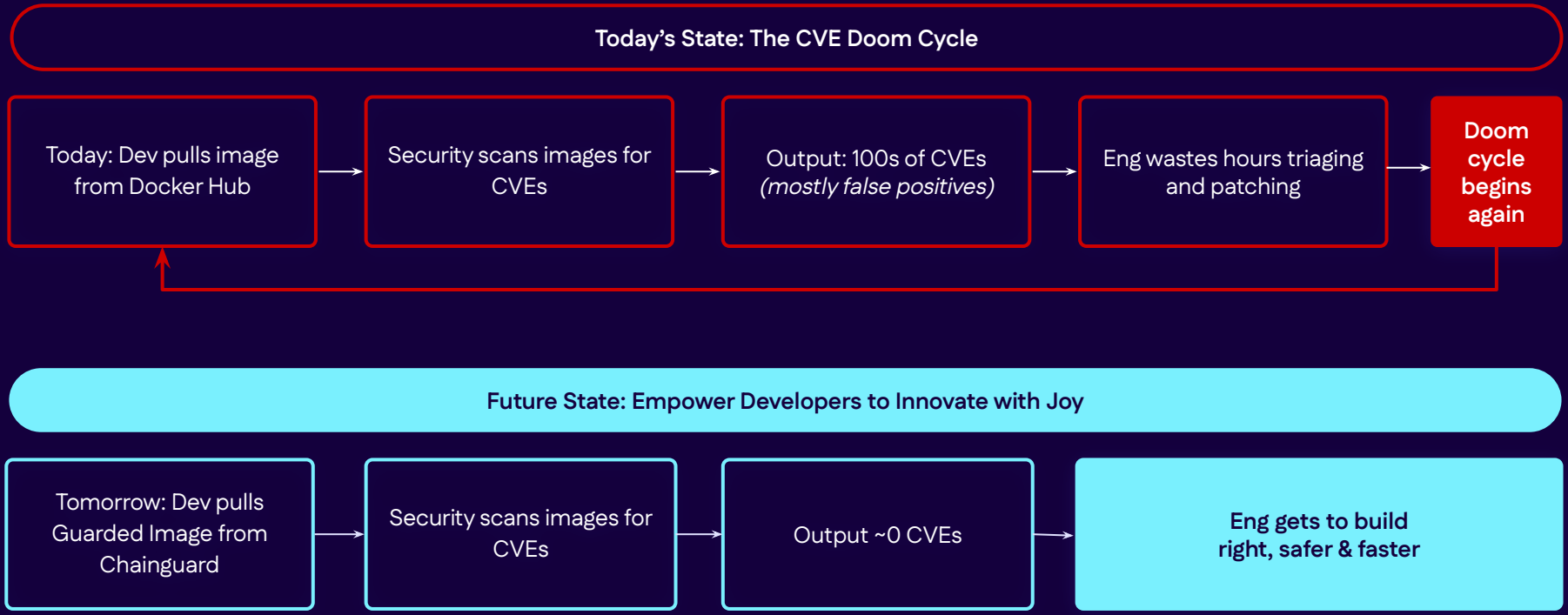
Practical

- Switching to a free Chainguard Image
- Grab the code from:
  - https://github.com/chainguard-dev/learning-labs-static/

Results

Build Based On	Size (MB)	CVEs (Grype)	CVEs (Scout)
golang	1330	329	72
cgr.dev/chainguard/go	1220	0	0
cgr.dev/chainguard/static	18	0	0

… With Chainguard, ~~Shift Left~~ Start Left to Build Right



So what is this "static" thing?

- Dynamic binaries
  - Link against other libraries
  - Often system libraries
- Static binaries are fully self contained
- Rust and Go code itself is statically linked
  - **Except** against system libraries

glibc and musl

- `glibc` is the "standard" Linux C library
  - But isn't good for static linking
  - Variant images available
- `musl` is an alternative C library
  - Can be statically linked
  - Sometimes compatibility concerns

Static Variants

- Sometimes need a few common libraries
- Almost static?!
- cgr.dev/chainguard/cc-dynamic
  - `glibc, libgcc`
- cgr.dev/chainguard/glibc-dynamic
  - `glibc, libgcc, libstdc++`

A word on FIPS

- FIPS is not covered by this lab
- You are responsible for creating binaries and images which solely use FIPS cryptography
- go-fips image
  - Overview and advice
- glibc-openssl-fips image
  - Possibly useful as a base in multistage

Static Binaries and Rust

- `cgr.dev/chainguard/glibc-dynamic` image should work
- Otherwise use musl target
  - E.g. `cargo build \`

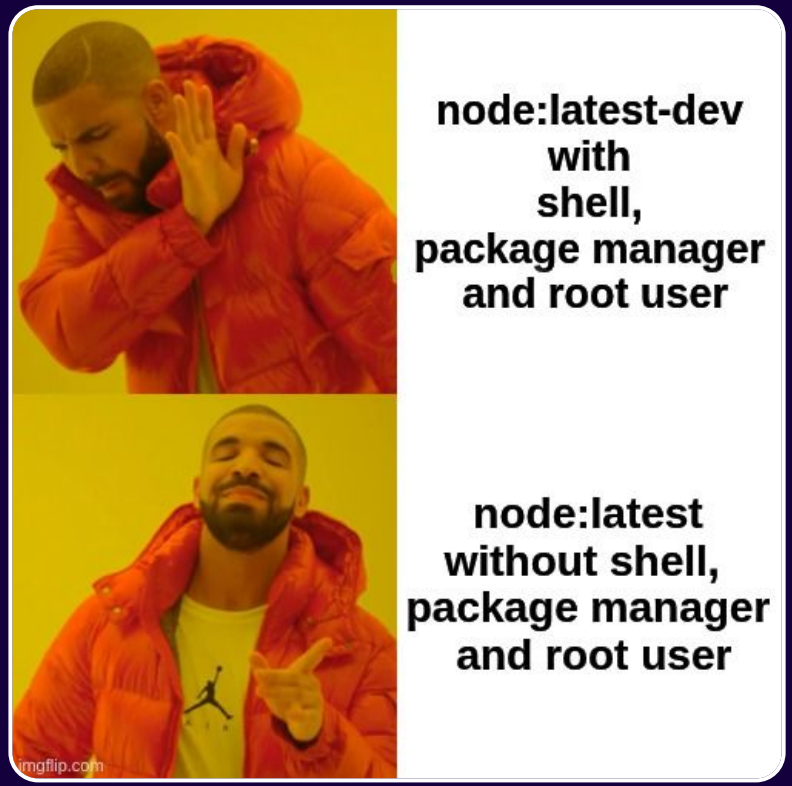
    `--target=x86_64-unknown-linux-musl`

What's "distroless"?

- Chainguard Images are often described as distroless
  - Contain minimum number of dependencies
  - No shell or package manager by default
  - But `latest-dev` variants available

Different Image Flavours

- PROD and DEV Images
  - DEV Image comes with a:
    - Root-Access
    - Package-Manager
    - Shell
  - PROD Images comes without:
    - Root-User (Non-root User only)
    - Package Manager
    - Shell
- Different Tags (Versions)
  - Standard
  - EOL Image Support
  - Immutable Digests vs. Mutable Tags
  - Epoch Tags



Practical 2

- Debugging Distroless Containers

Debugging Distroless

- Note `latest-dev` variants
- Docker Debug
- Ephemeral containers
- cdebug

How we keep out CVEs

- Cut down dependencies
- Keep things up-to-date
- Apply patches when necessary
- Issue Security Advisories

Wrap Up

- Simple to change to ~~Chainguard~~ Distroless Images
- Major advantages in size and security
- Large number of images available
  - Include –dev variants

You want to build a ~~Chocolate~~ Container Factory?

Introducing – Sigstore!

Digital Signatures for Containers

- Open Source Project (Chainguard founders involved)

Components:

- Cosign: sign/verify
- Fulcio: ephemeral certs
- Rekor: transparency log

Goal: make trust cryptographically verifiable

sigstore

cosign

Howto Sign an SBOM with Cosign

- https://edu.chainguard.dev/open-source/sigstore/cosign/how-to-sign-an-sbom-with-cosign/
- (And for mature organisations, use admission controller to only allow untampered signed images

  https://edu.chainguard.dev/open-source/sigstore/policy-controller/how-to-install-policy-controller/)

Further Resources

- Chainguard Images Directory
- Chainguard Academy
- Docker Debug
- cdebug
- Statically Linking Go