KubeCon | CloudNativeCon

North America 2024

KubeCon | CloudNativeCon

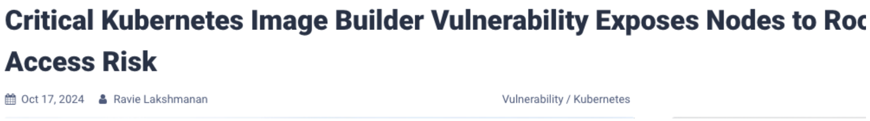
North America 2024

It’s Dangerous To Build It Alone!

Take This!

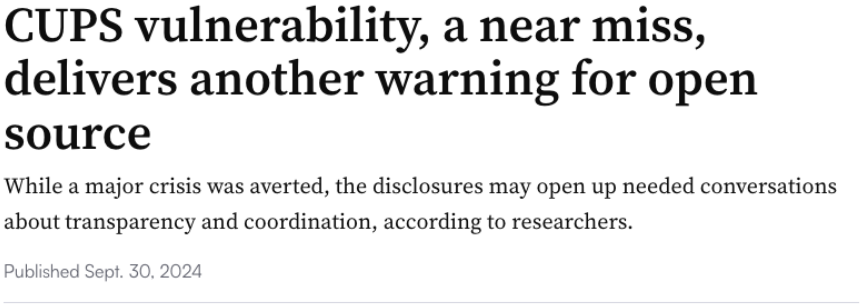
Ashna Mehrotra
Jeremy Rickard

Why Are You Here?



Critical Kubernetes Image Builder Vulnerability Exposes Nodes to Roc Access Risk

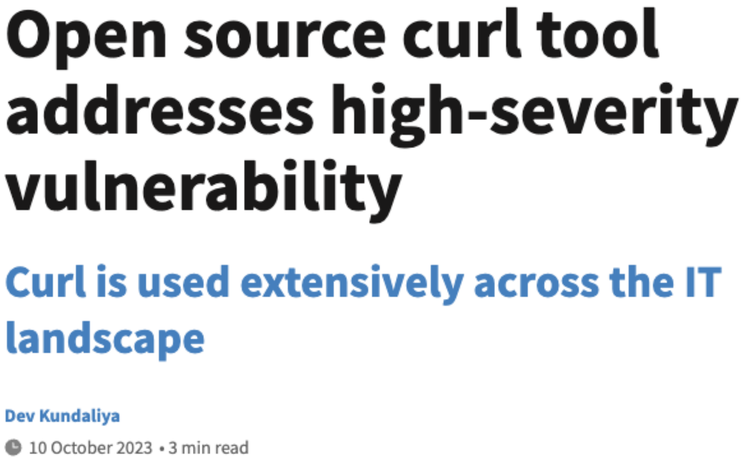
Oct 17, 2024 Ravie Lakshmanan Vulnerability / Kubernetes



CUPS vulnerability, a near miss, delivers another warning for open source

While a major crisis was averted, the disclosures may open up needed conversations about transparency and coordination, according to researchers.

Published Sept. 30, 2024

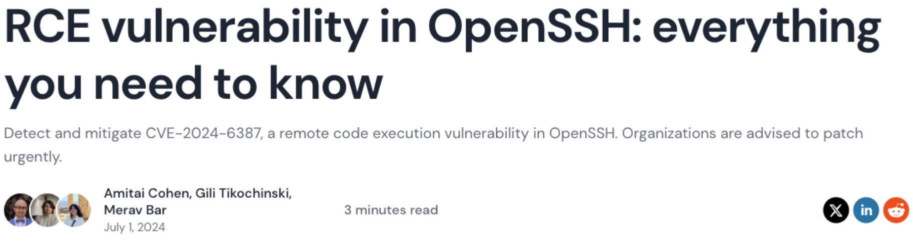


Open source curl tool addresses high-severity vulnerability

Curl is used extensively across the IT landscape

Dev Kundaliya

10 October 2023 • 3 min read

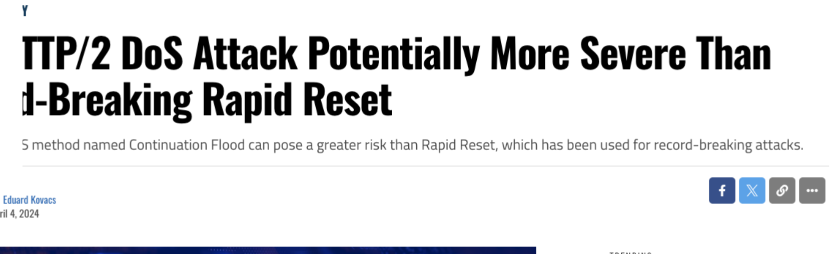


RCE vulnerability in OpenSSH: everything you need to know

Detect and mitigate CVE-2024-6387, a remote code execution vulnerability in OpenSSH. Organizations are advised to patch urgently.

Amitai Cohen, Gili Tikochinski, Merav Bar
July 1, 2024

3 minutes read

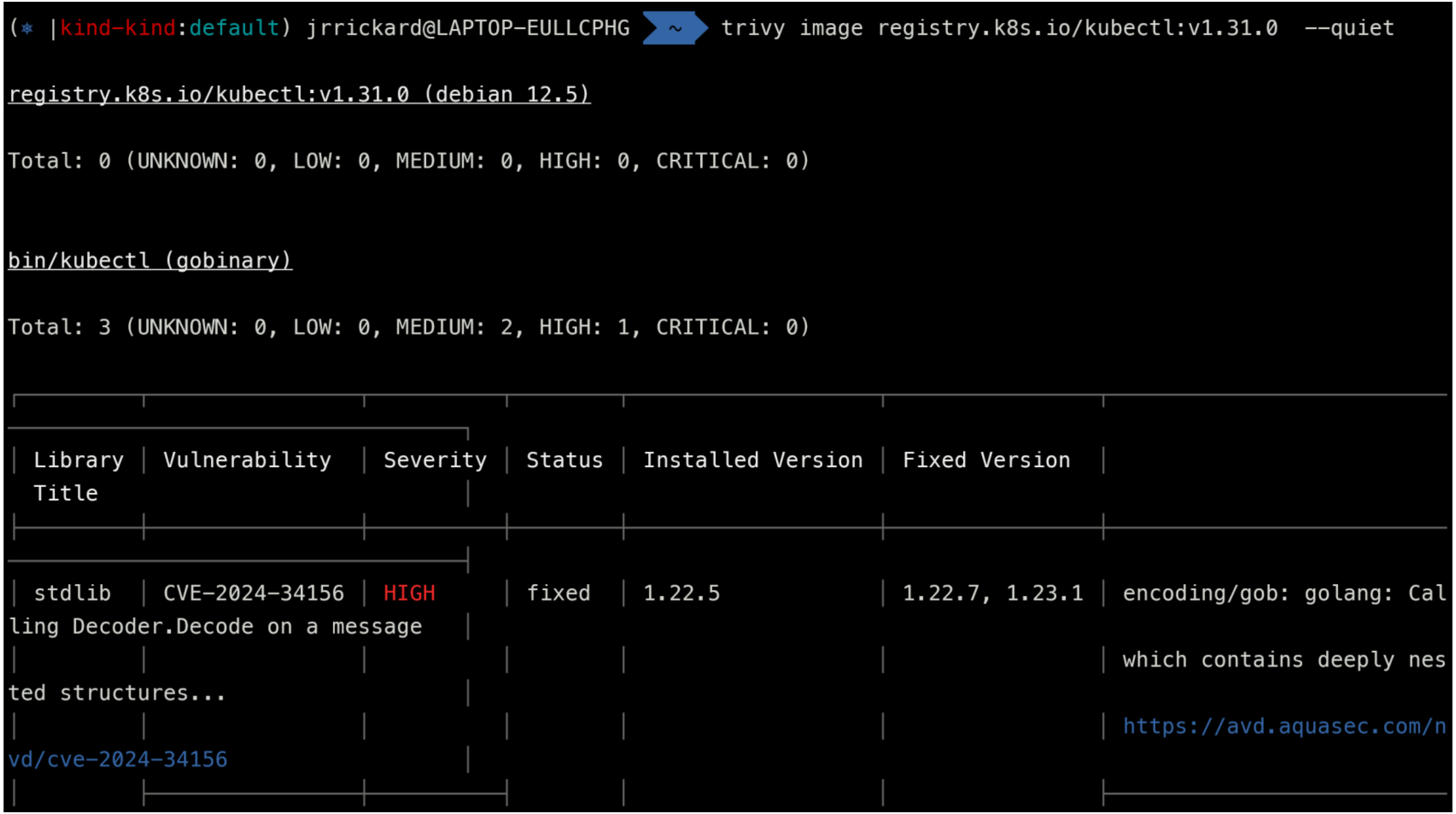


Y

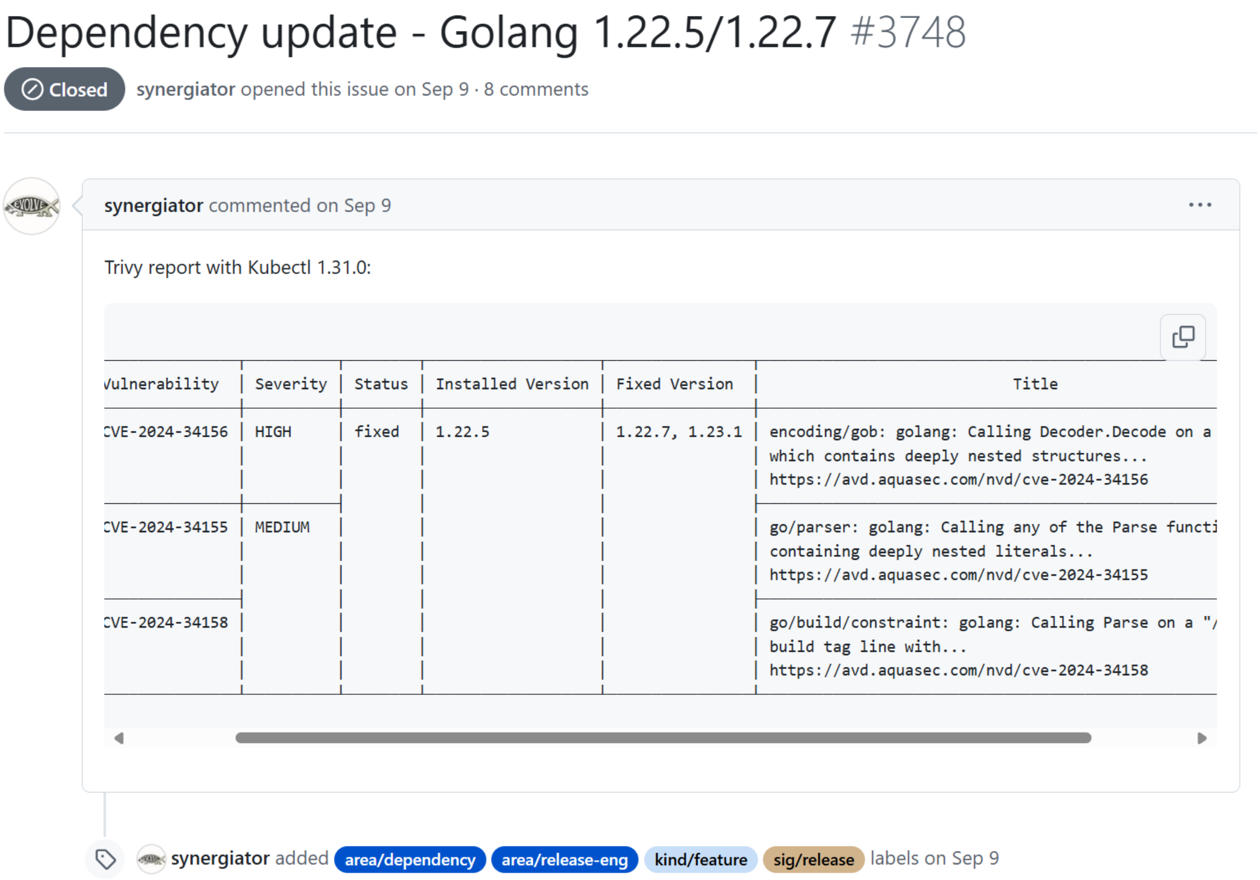
TTP/2 DoS Attack Potentially More Severe Than d-Breaking Rapid Reset

5 method named Continuation Flood can pose a greater risk than Rapid Reset, which has been used for record-breaking attacks.

By Eduard Kovacs
April 4, 2024



```
(⎈ |kind-kind:default) jrrickard@LAPTOP-EULLCPHG  ~  trivy image registry.k8s.io/kubectl:v1.31.0  --quiet

registry.k8s.io/kubectl:v1.31.0 (debian 12.5)

Total: 0 (UNKNOWN: 0, LOW: 0, MEDIUM: 0, HIGH: 0, CRITICAL: 0)


bin/kubectl (gobinary)

Total: 3 (UNKNOWN: 0, LOW: 0, MEDIUM: 2, HIGH: 1, CRITICAL: 0)

┌─────────┬────────────────┬──────────┬────────┬───────────────────┬────────────────┬────────────────────────
───────────────────────────────────┐
│ Library │ Vulnerability  │ Severity │ Status │ Installed Version │ Fixed Version  │
  Title                            │
├─────────┼────────────────┼──────────┼────────┼───────────────────┼────────────────┼────────────────────────
───────────────────────────────────┤
│ stdlib  │ CVE-2024-34156 │ HIGH     │ fixed  │ 1.22.5            │ 1.22.7, 1.23.1 │ encoding/gob: golang: Cal
ling Decoder.Decode on a message   │
│         │                │          │        │                   │                │ which contains deeply nes
ted structures...                  │
│         │                │          │        │                   │                │ https://avd.aquasec.com/n
vd/cve-2024-34156                  │
│         ├────────────────┼──────────┤        │                   │                ├────────────────────────
```



Dependency update - Golang 1.22.5/1.22.7 #3748

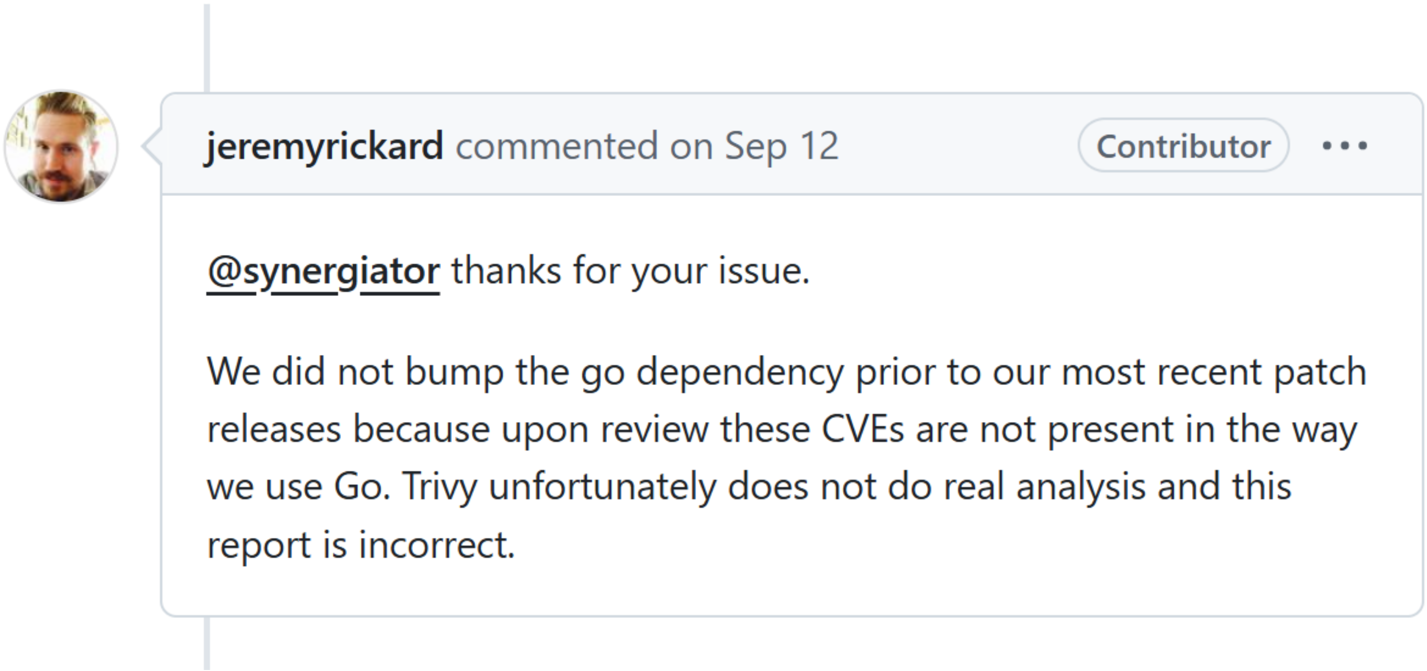
Closed synergiator opened this issue on Sep 9 · 8 comments

synergiator commented on Sep 9

Trivy report with Kubectl 1.31.0:

```
──────────────────┬──────────┬────────┬───────────────────┬────────────────┬──────────────────────────────────────────────────────
Vulnerability     │ Severity │ Status │ Installed Version │ Fixed Version  │                          Title
──────────────────┼──────────┼────────┼───────────────────┼────────────────┼──────────────────────────────────────────────────────
CVE-2024-34156    │ HIGH     │ fixed  │ 1.22.5            │ 1.22.7, 1.23.1 │ encoding/gob: golang: Calling Decoder.Decode on a
                  │          │        │                   │                │ which contains deeply nested structures...
                  │          │        │                   │                │ https://avd.aquasec.com/nvd/cve-2024-34156
──────────────────┼──────────┤        │                   │                ├──────────────────────────────────────────────────────
CVE-2024-34155    │ MEDIUM   │        │                   │                │ go/parser: golang: Calling any of the Parse functi
                  │          │        │                   │                │ containing deeply nested literals...
                  │          │        │                   │                │ https://avd.aquasec.com/nvd/cve-2024-34155
──────────────────┤          │        │                   │                ├──────────────────────────────────────────────────────
CVE-2024-34158    │          │        │                   │                │ go/build/constraint: golang: Calling Parse on a "/
                  │          │        │                   │                │ build tag line with...
                  │          │        │                   │                │ https://avd.aquasec.com/nvd/cve-2024-34158
──────────────────┴──────────┴────────┴───────────────────┴────────────────┴──────────────────────────────────────────────────────
```

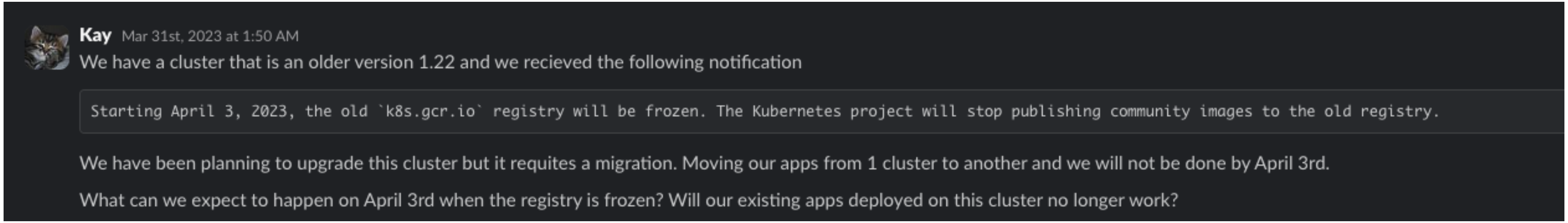
synergiator added area/dependency area/release-eng kind/feature sig/release labels on Sep 9



jeremyrickard commented on Sep 12

@synergiator thanks for your issue.

We did not bump the go dependency prior to our most recent patch releases because upon review these CVEs are not present in the way we use Go. Trivy unfortunately does not do real analysis and this report is incorrect.



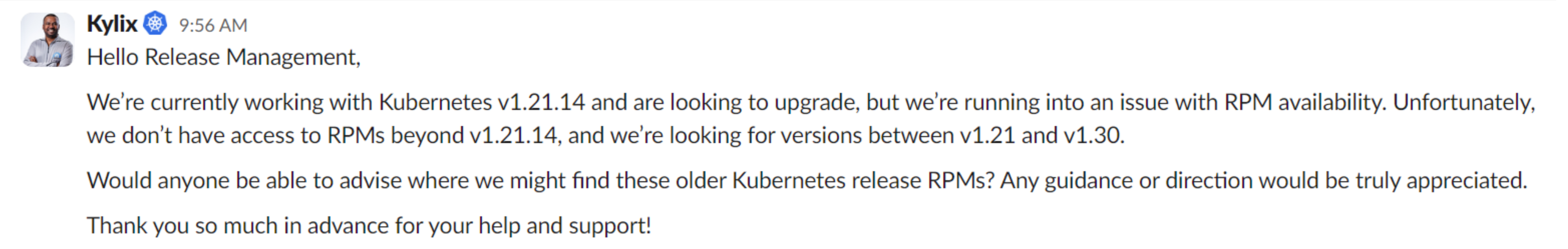
Kay Mar 31st, 2023 at 1:50 AM

We have a cluster that is an older version 1.22 and we recieved the following notification

```
Starting April 3, 2023, the old `k8s.gcr.io` registry will be frozen. The Kubernetes project will stop publishing community images to the old registry.
```

We have been planning to upgrade this cluster but it requites a migration. Moving our apps from 1 cluster to another and we will not be done by April 3rd.

What can we expect to happen on April 3rd when the registry is frozen? Will our existing apps deployed on this cluster no longer work?



Kylix 9:56 AM

Hello Release Management,

We're currently working with Kubernetes v1.21.14 and are looking to upgrade, but we're running into an issue with RPM availability. Unfortunately, we don't have access to RPMs beyond v1.21.14, and we're looking for versions between v1.21 and v1.30.

Would anyone be able to advise where we might find these older Kubernetes release RPMs? Any guidance or direction would be truly appreciated.

Thank you so much in advance for your help and support!

What do we do now?

Open Source Security Foundation

Secure Supply Chain Consumption Framework

https://github.com/ossf/s2c2f/blob/main/specification/README.md

Framework Practices and Requirements

- **Practice 1: Ingest It**

- **Use public package managers trusted by your organization (i.e. NuGet.org, npmjs.com, PyPi.org, etc.)**
- **Use an OSS binary repository manager solution (i.e. JFrog Artifactory, Azure Artifacts, etc.) – this is images too**
- Have a Deny List capability to block known malicious OSS from being consumed
- **Mirror a copy of all OSS source code to an internal location**

Framework Practices and Requirements

- Practice 1: Ingest It
- **Practice 2: Scan It**

- **Scan OSS for known vulnerabilities (i.e. CVEs, GitHub Advisories, etc.)**
- Scan OSS for licenses
- Scan OSS to determine if its end-of-life
- Scan OSS for malware
- Perform proactive security analysis of OSS

Framework Practices and Requirements

- Practice 1: Ingest It
- Practice 2: Scan It
- **Practice 3: Inventory It**

- Maintain an automated inventory of all OSS used in development
- Have an OSS Incident Response Plan

Framework Practices and Requirements

- Practice 1: Ingest It
- Practice 2: Scan It
- Practice 3: Inventory It
- **Practice 4: Update It**

- **Update vulnerable OSS manually**
- **Enable automated OSS updates**
- Display OSS vulnerabilities in developer contribution flow (i.e. Pull Requests).

Framework Practices and Requirements

- Practice 1: Ingest It
- Practice 2: Scan It
- Practice 3: Inventory It
- Practice 4: Update It
- **Practice 5: Audit It**

- Verify the provenance of your OSS
- Audit that developers are consuming OSS through the approved ingestion method
- **Validate integrity of the OSS that you consume into your build**
- **Validate SBOMs of OSS that you consume into your build**

Framework Practices and Requirements

- Practice 1: Ingest It
- Practice 2: Scan It
- Practice 3: Inventory It
- Practice 4: Update It
- Practice 5: Audit It
- **Practice 6: Enforce It**

- Securely configure your package source files (i.e. nuget.config, .npmrc, pip.conf, pom.xml, etc.)
- **Enforce usage of a curated OSS feed that enhances the trust of your OSS (this can also be enforcing registries in your cluster!)**

Framework Practices and Requirements

- Practice 1: Ingest It
- Practice 2: Scan It
- Practice 3: Inventory It
- Practice 4: Update It
- Practice 5: Audit It
- Practice 6: Enforce It
- **Practice 7: Rebuild It**

- **Rebuild the OSS in a trusted build environment, or validate that it is reproducibly built.**
- **Digitally sign the OSS you rebuild**
- **Generate SBOMs for OSS that you rebuild**
- **Digitally sign the SBOMs you produce**

Framework Practices and Requirements

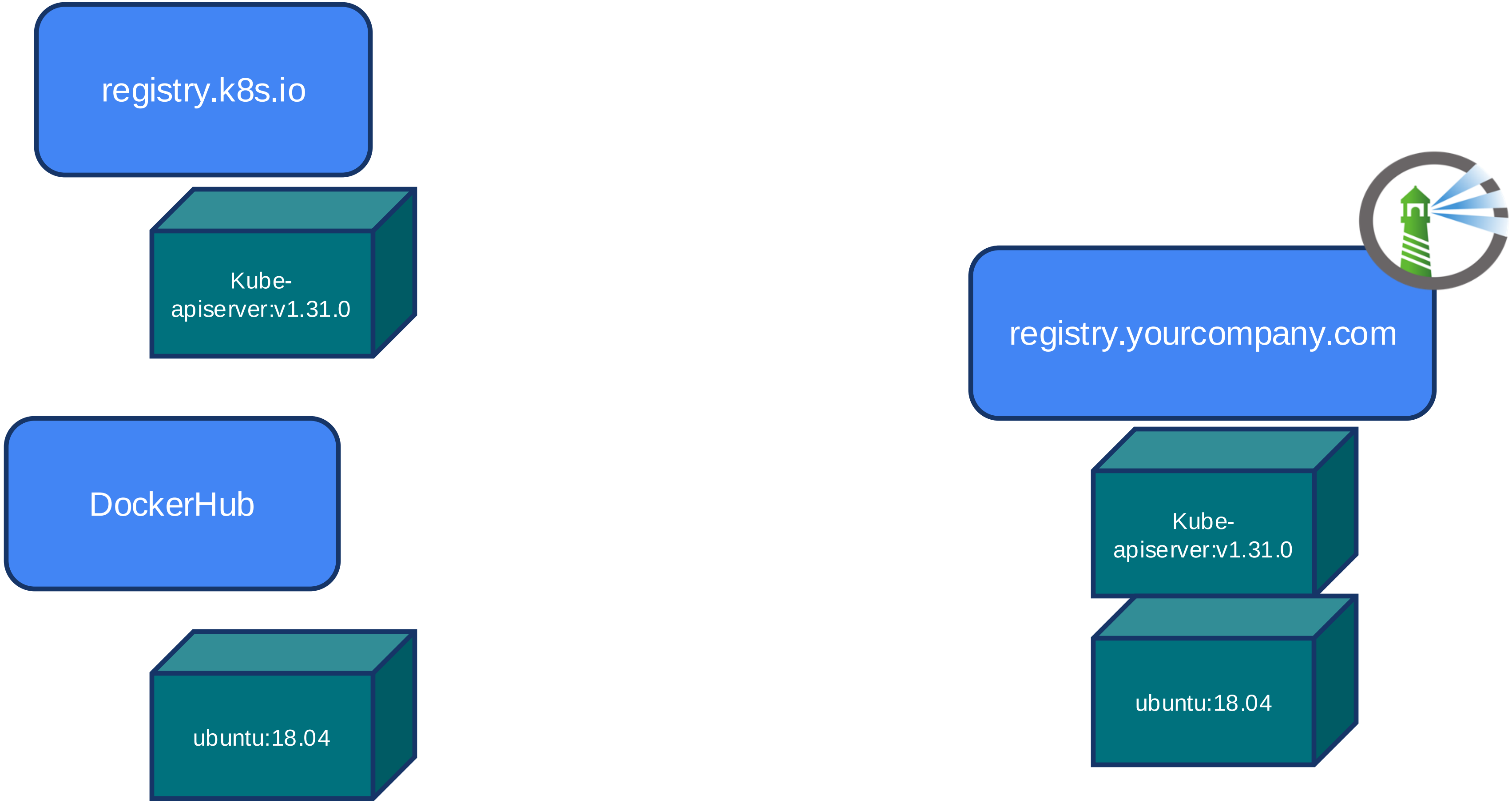
- Practice 1: Ingest It
- Practice 2: Scan It
- Practice 3: Inventory It
- Practice 4: Update It
- Practice 5: Audit It
- Practice 6: Enforce It
- Practice 7: Rebuild It
- **Practice 8: Fix It + Upstream**

- Implement a change in the code to address a zero-day vulnerability, rebuild, deploy to your organization, and *confidentially contribute the fix to the upstream maintainer*

Level 1	Level 2	Level 3	Level 4
Minimum OSS Governance Program	**Secure Consumption and Improved MTTR**	**Malware Defense and Zero-Day Detection**	**Advanced Threat Defense**
• Use package managers [ING-1] • Local copy of artifact [ING-2] • Scan with known vulns [SCA-1] • Scan for software licenses [SCA-2] • Inventory OSS [INV-1] • Manual OSS updates [UPD-1]	• Scan for end of life [SCA-3] • Have an incident response plan [INV-2] • Auto OSS updates [UPD-2] • Alerts on vulns at PR time [UPD-3] • Audit that consumption is through approved ingestion method [AUD-2] • Validate integrity of OSS [AUD-3] • Secure package source file configuration [ENF-1]	• Deny list capability [ING-3] • Clone OSS source [ING-4] • Scan for malware [SCA-4] • Proactive security reviews [SCA-5] • Enforce OSS provenance [AUD-1] • Enforce consumption from curated feed [ENF-2]	• Validate the SBOMs of OSS consumed [AUD-4] • Rebuild OSS on trusted infrastructure [REB-1] • Digitally sign rebuilt OSS [REB-2] • Generate SBOM for rebuilt OSS [REB-3] • Digitally sign protected SBOMs [REB-4] • Implement fixes [FIX-1]

Ingest It

Consuming Upstream Images



registry.k8s.io

Kube-apiserver:v1.31.0

DockerHub

ubuntu:18.04

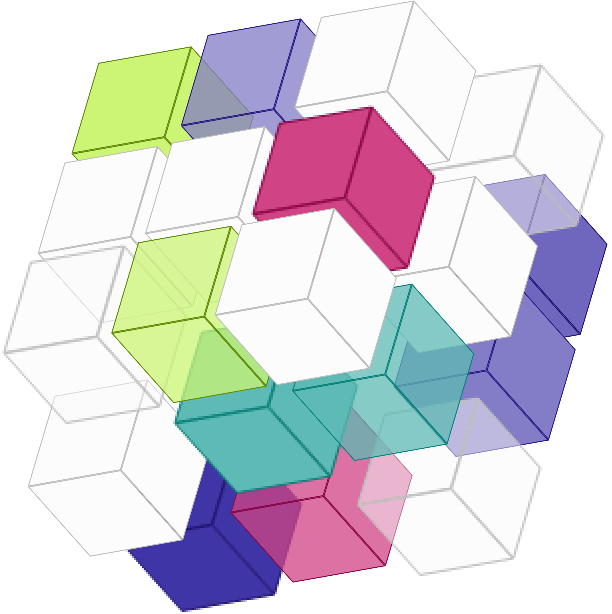
registry.yourcompany.com

Kube-apiserver:v1.31.0

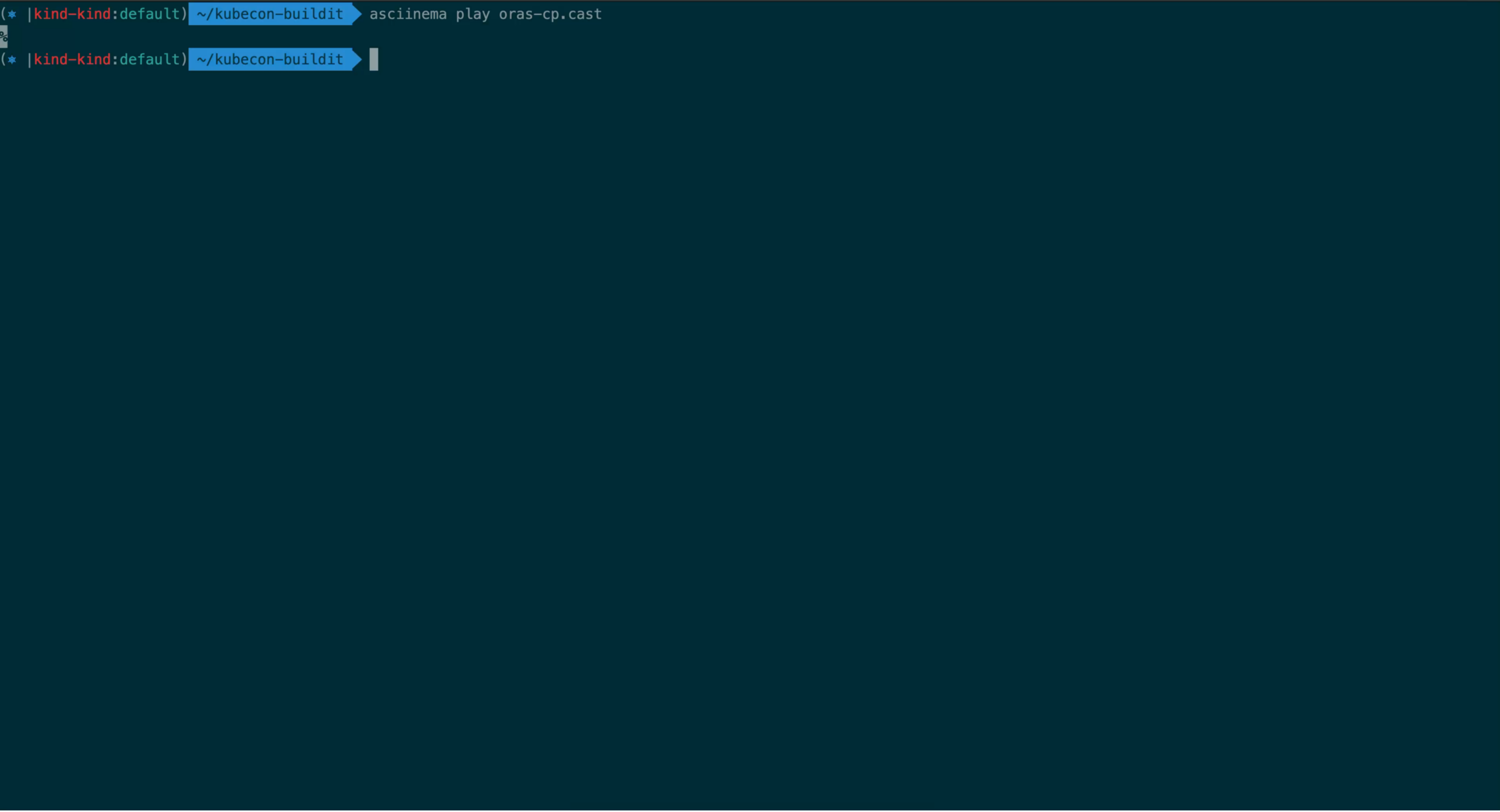
ubuntu:18.04

Let's do this with ORAS

- CNCF Sandbox Project
- ORAS provides CLI and client libraries to distribute artifacts across OCI-compliant registries.
- Copy Artifacts From One Registry to Another
- Attach Artifacts to Images (and discover them)
- List tags and examine manifests



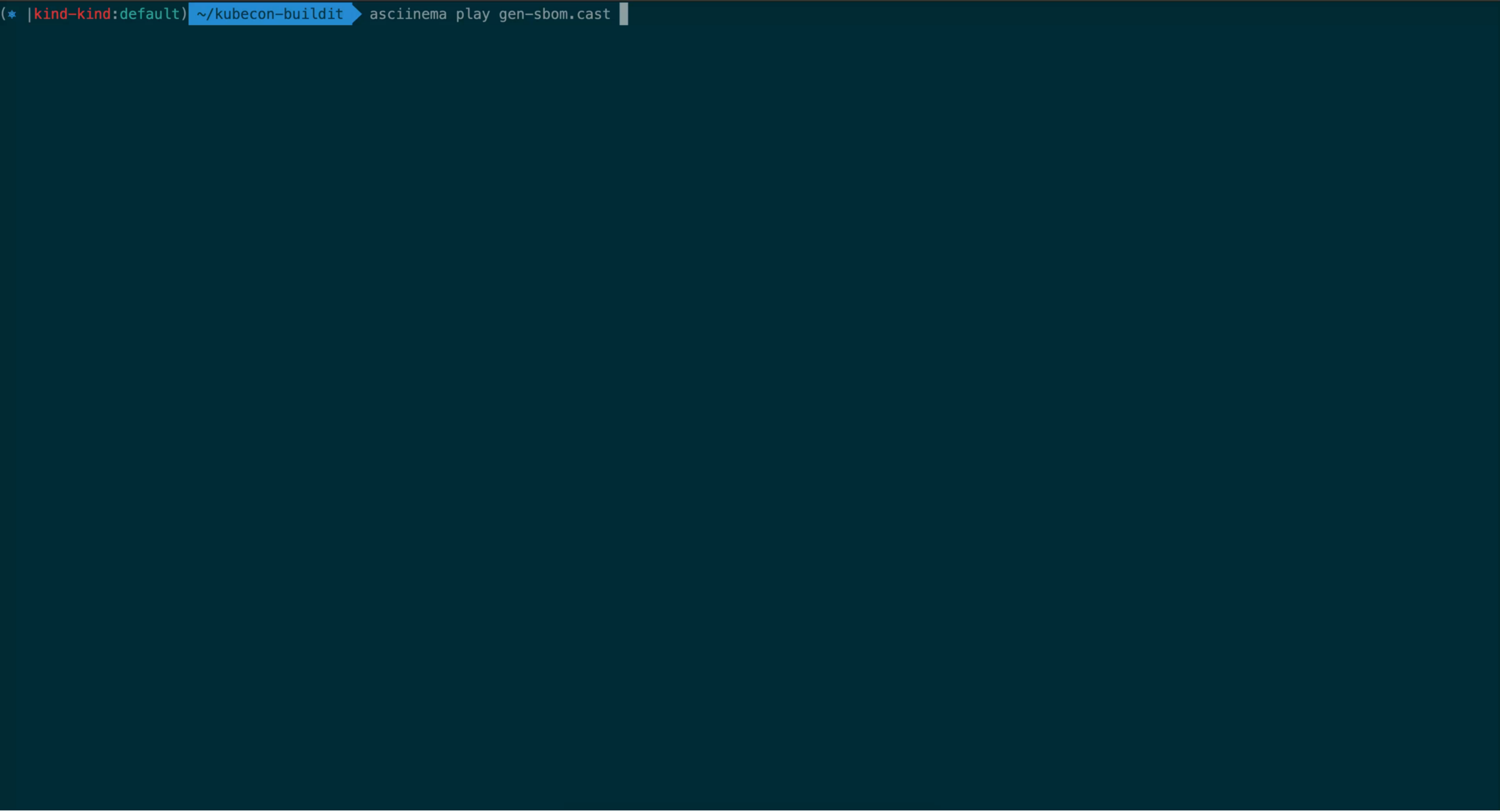
https://oras.land/



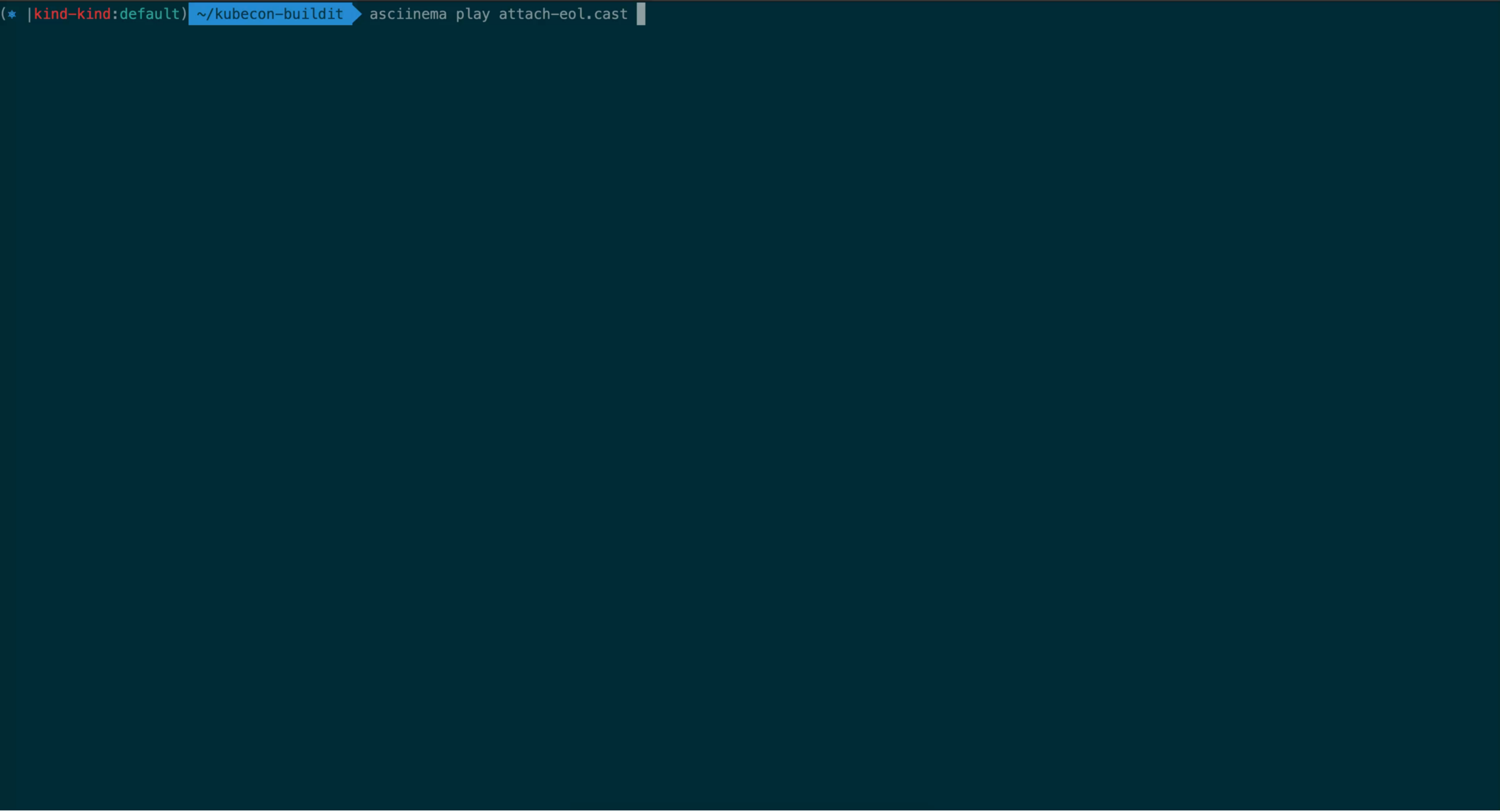
```
(⎈ |kind-kind:default) ~/kubecon-buildit  asciinema play oras-cp.cast
%
(⎈ |kind-kind:default) ~/kubecon-buildit 
```

Audit It and Enforce It

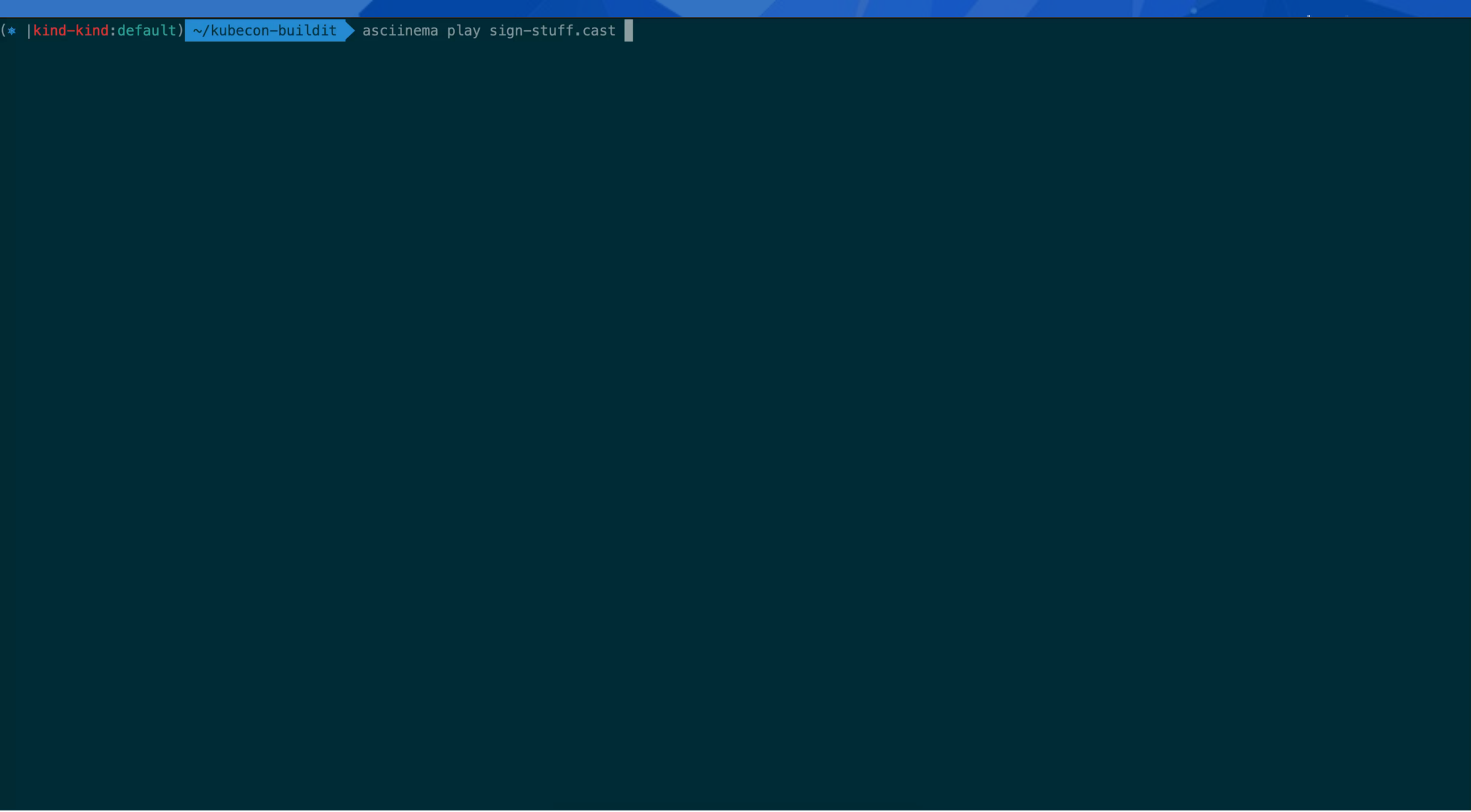
Adding additional context…



```
(⎈ |kind-kind:default) ~/kubecon-buildit  asciinema play gen-sbom.cast
```



```
(⎈ |kind-kind:default) ~/kubecon-buildit  asciinema play attach-eol.cast
```



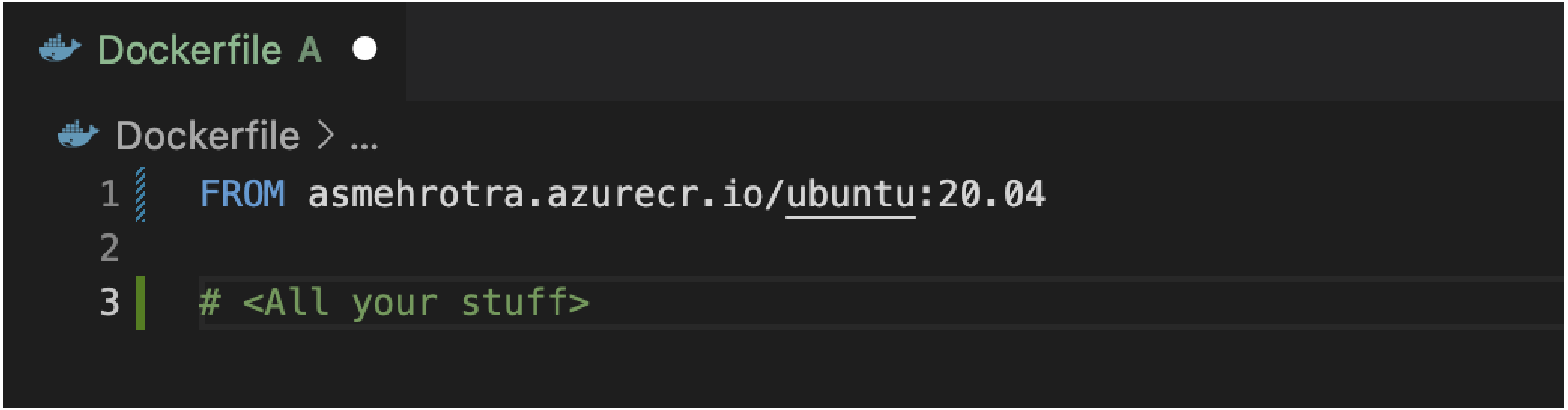
```
(✱ |kind-kind:default) ~/kubecon-buildit  asciinema play sign-stuff.cast
```

KubeCon | CloudNativeCon
North America 2024

Scan It and Update It

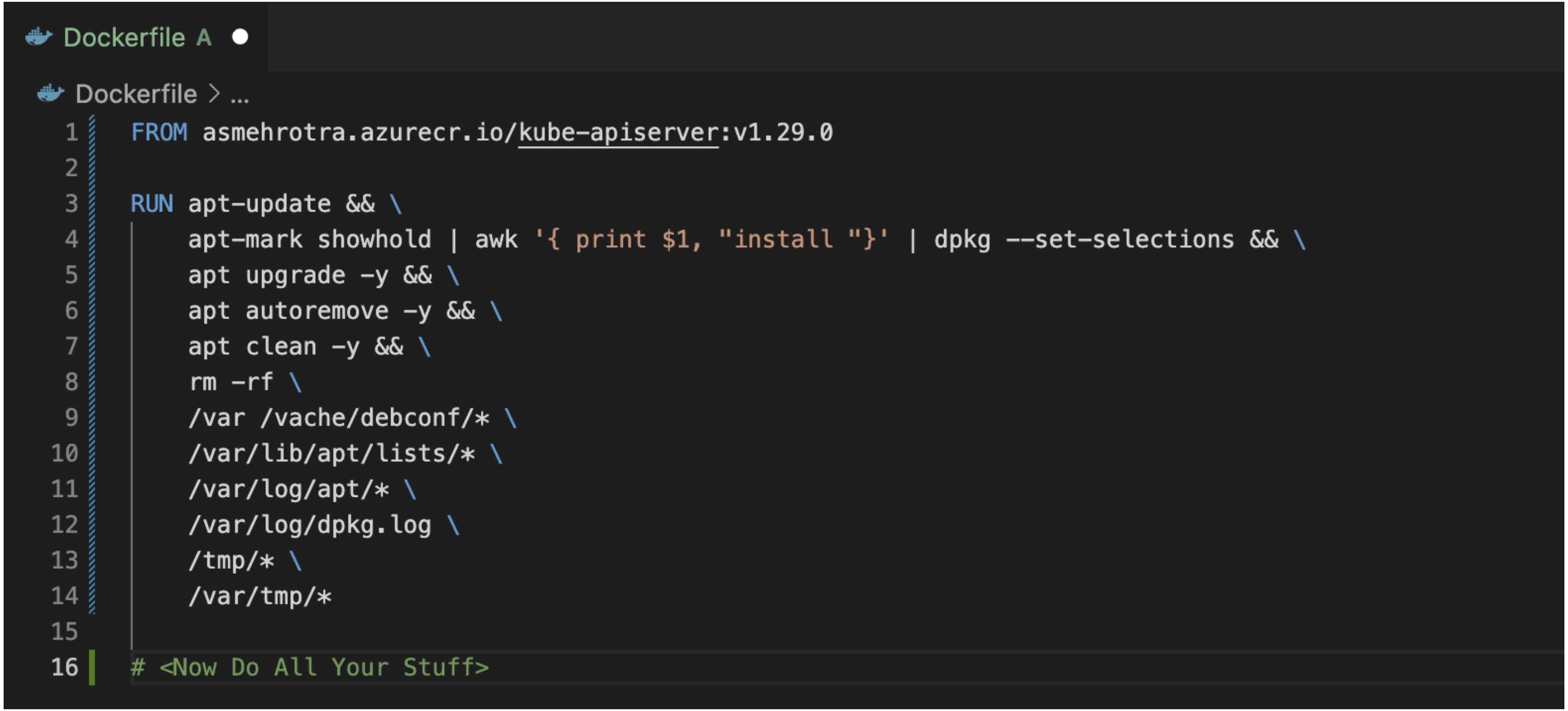
Patching Upstream Images with Copa

Wait for Base Image Updates



```
FROM asmehrotra.azurecr.io/ubuntu:20.04

# <All your stuff>
```

Rely on Package Managers

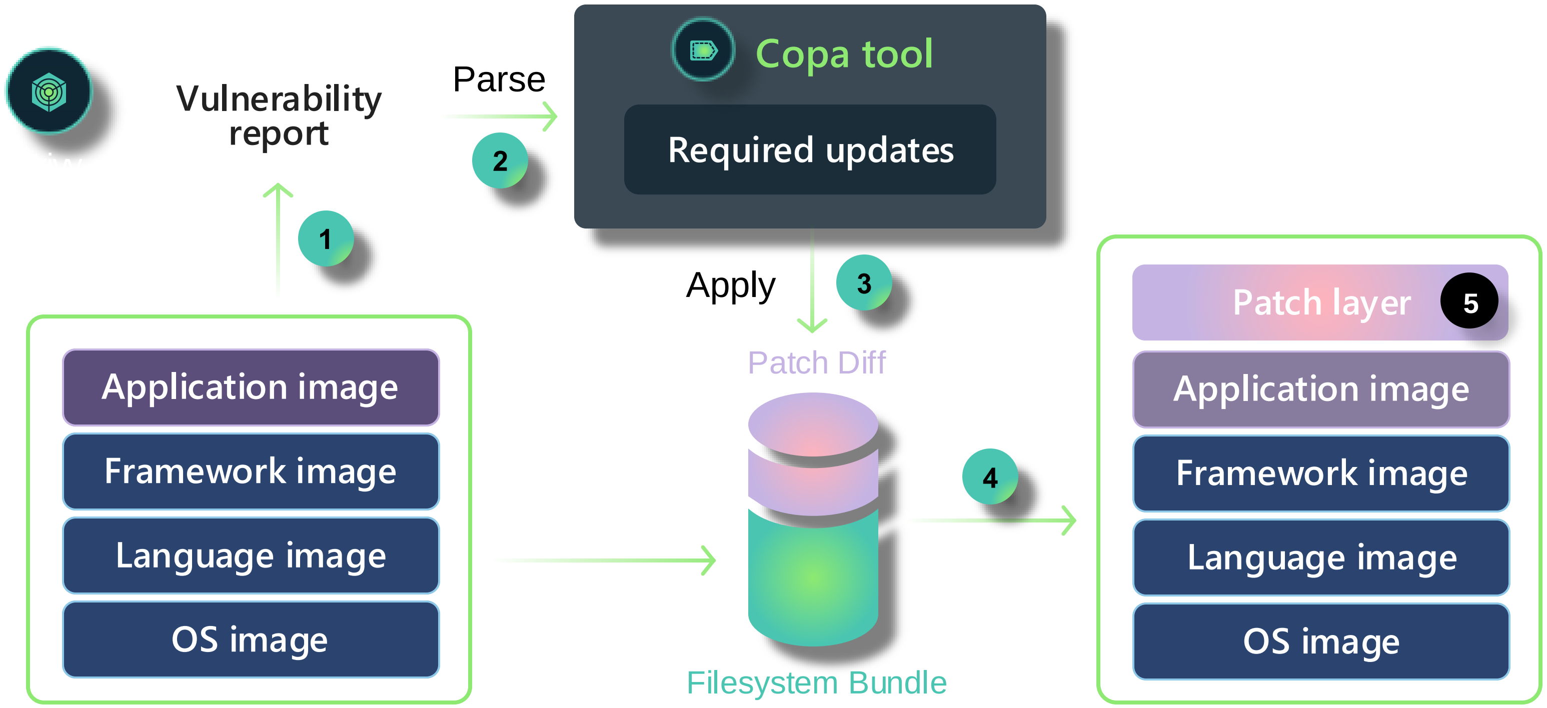


```dockerfile
FROM asmehrotra.azurecr.io/kube-apiserver:v1.29.0

RUN apt-update && \
    apt-mark showhold | awk '{ print $1, "install "}' | dpkg --set-selections && \
    apt upgrade -y && \
    apt autoremove -y && \
    apt clean -y && \
    rm -rf \
    /var /vache/debconf/* \
    /var/lib/apt/lists/* \
    /var/log/apt/* \
    /var/log/dpkg.log \
    /tmp/* \
    /var/tmp/*

# <Now Do All Your Stuff>
```

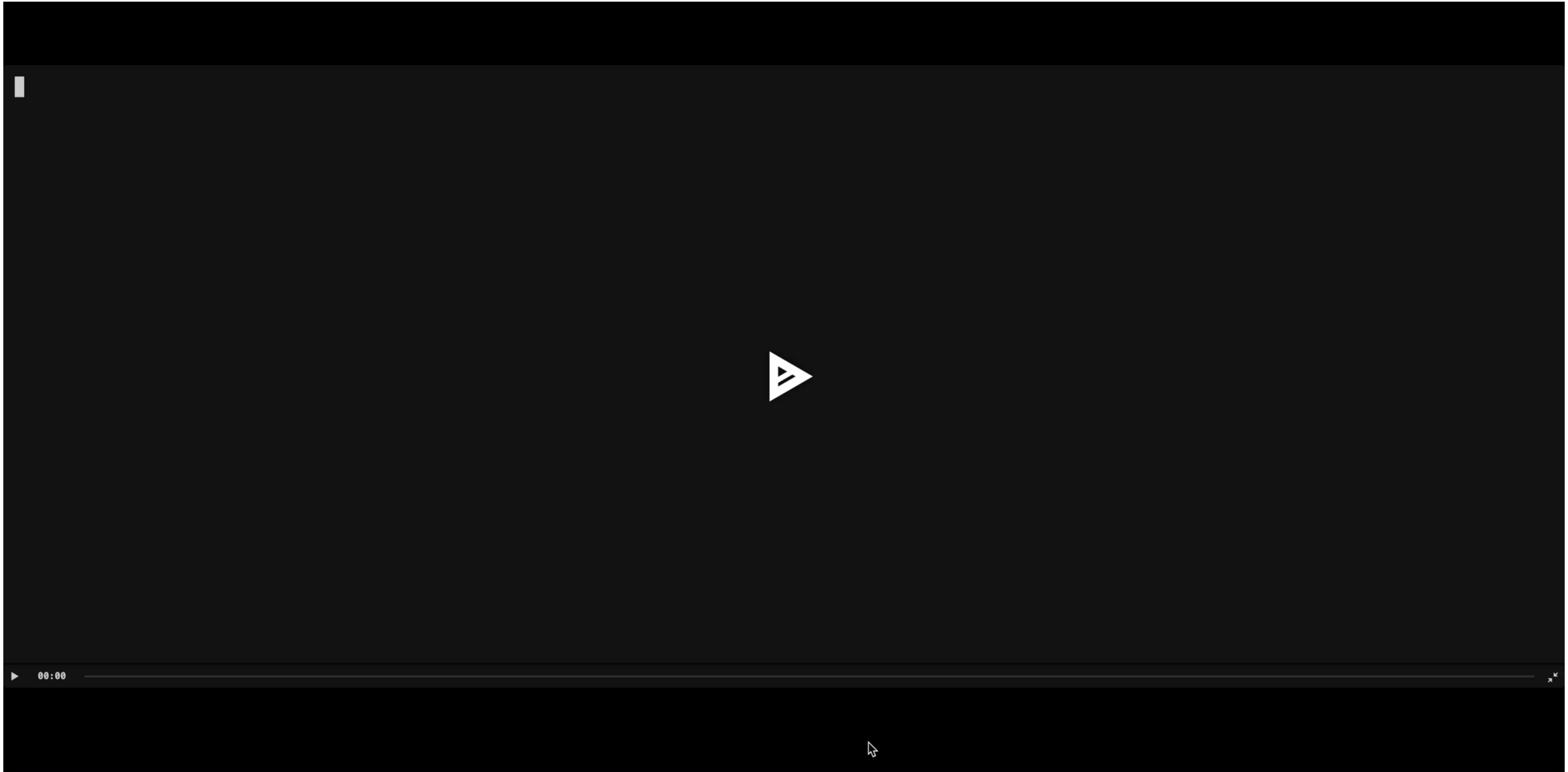
What is Copa?

- CNCF Sandbox project to patch container images
- CLI tool written in Go and based on BuildKit (Docker's default builder)
- Updates vulnerable or outdated packages in an image
  - Targeted patching by default uses Trivy reports
    - Trivy can scan images for OS vulnerabilities
    - But scanner report component is pluggable
  - Otherwise updates all outdated packages

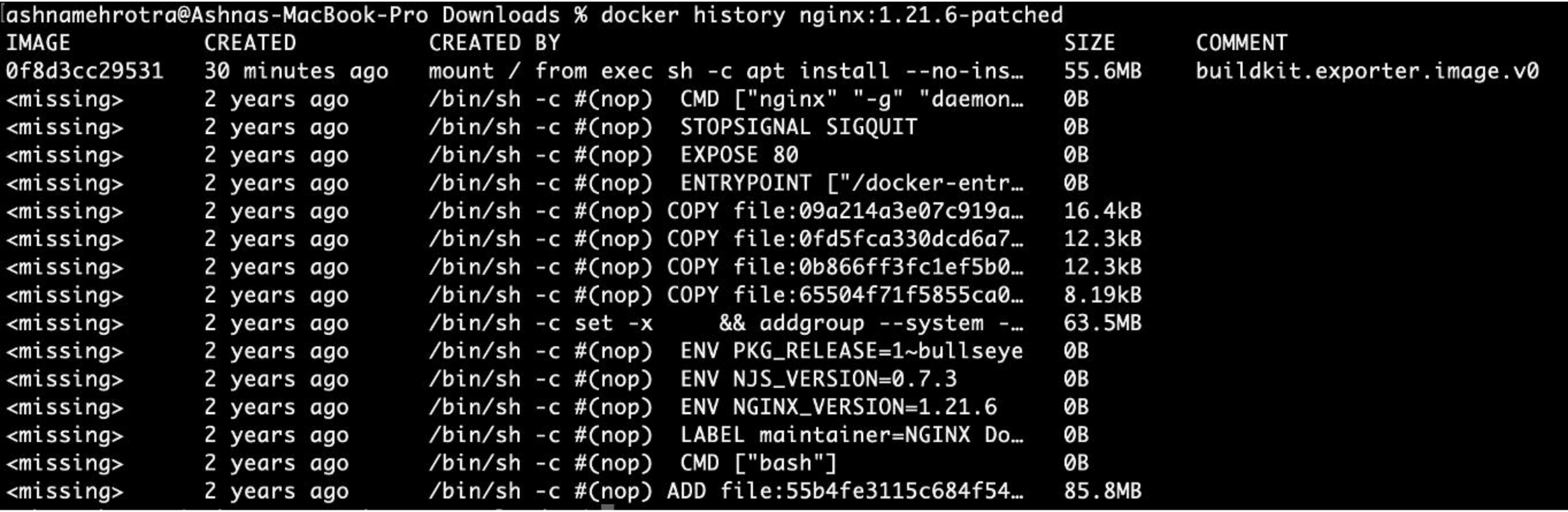
How Copa Works



Demo



Resulting Image



```
ashnamehrotra@Ashnas-MacBook-Pro Downloads % docker history nginx:1.21.6-patched
IMAGE          CREATED          CREATED BY                                      SIZE      COMMENT
0f8d3cc29531   30 minutes ago   mount / from exec sh -c apt install --no-ins…   55.6MB    buildkit.exporter.image.v0
<missing>      2 years ago      /bin/sh -c #(nop)  CMD ["nginx" "-g" "daemon…   0B
<missing>      2 years ago      /bin/sh -c #(nop)  STOPSIGNAL SIGQUIT           0B
<missing>      2 years ago      /bin/sh -c #(nop)  EXPOSE 80                    0B
<missing>      2 years ago      /bin/sh -c #(nop)  ENTRYPOINT ["/docker-entr…   0B
<missing>      2 years ago      /bin/sh -c #(nop) COPY file:09a214a3e07c919a…   16.4kB
<missing>      2 years ago      /bin/sh -c #(nop) COPY file:0fd5fca330dcd6a7…   12.3kB
<missing>      2 years ago      /bin/sh -c #(nop) COPY file:0b866ff3fc1ef5b0…   12.3kB
<missing>      2 years ago      /bin/sh -c #(nop) COPY file:65504f71f5855ca0…   8.19kB
<missing>      2 years ago      /bin/sh -c set -x     && addgroup --system -…   63.5MB
<missing>      2 years ago      /bin/sh -c #(nop)  ENV PKG_RELEASE=1~bullseye   0B
<missing>      2 years ago      /bin/sh -c #(nop)  ENV NJS_VERSION=0.7.3        0B
<missing>      2 years ago      /bin/sh -c #(nop)  ENV NGINX_VERSION=1.21.6     0B
<missing>      2 years ago      /bin/sh -c #(nop)  LABEL maintainer=NGINX Do…   0B
<missing>      2 years ago      /bin/sh -c #(nop)  CMD ["bash"]                 0B
<missing>      2 years ago      /bin/sh -c #(nop) ADD file:55b4fe3115c684f54…   85.8MB
```

Benefits

- Gives control to who patches container images
- Custom patching solution
    - Pluggable scanner report, or update all outdated packages
- Reduces time, cost, and complexity for container patching
- Easy to integrate into pipelines
    - Build time patching
    - Recurring patching
- Can integrate with Dependabot
    - Create image update PRs
- Can patch distroless images

Limitations

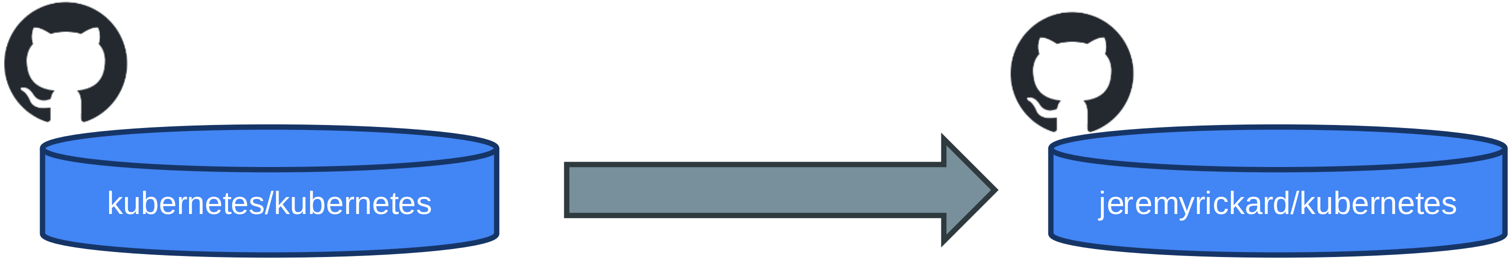
- App-specific vulnerabilities
- Windows images
- Dependency on individual package managers

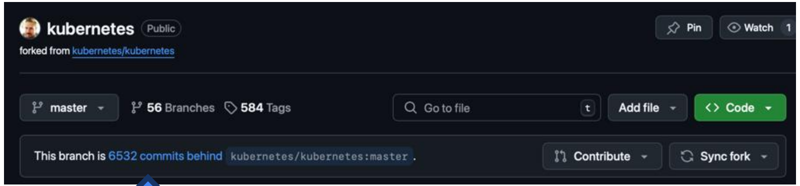
KubeCon | CloudNativeCon
North America 2024

Rebuild It

Rebuild Your Open Source Dependencies

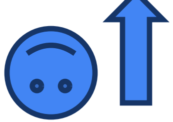
First things first….




kubernetes Public

forked from kubernetes/kubernetes

This branch is 6532 commits behind kubernetes/kubernetes:master .

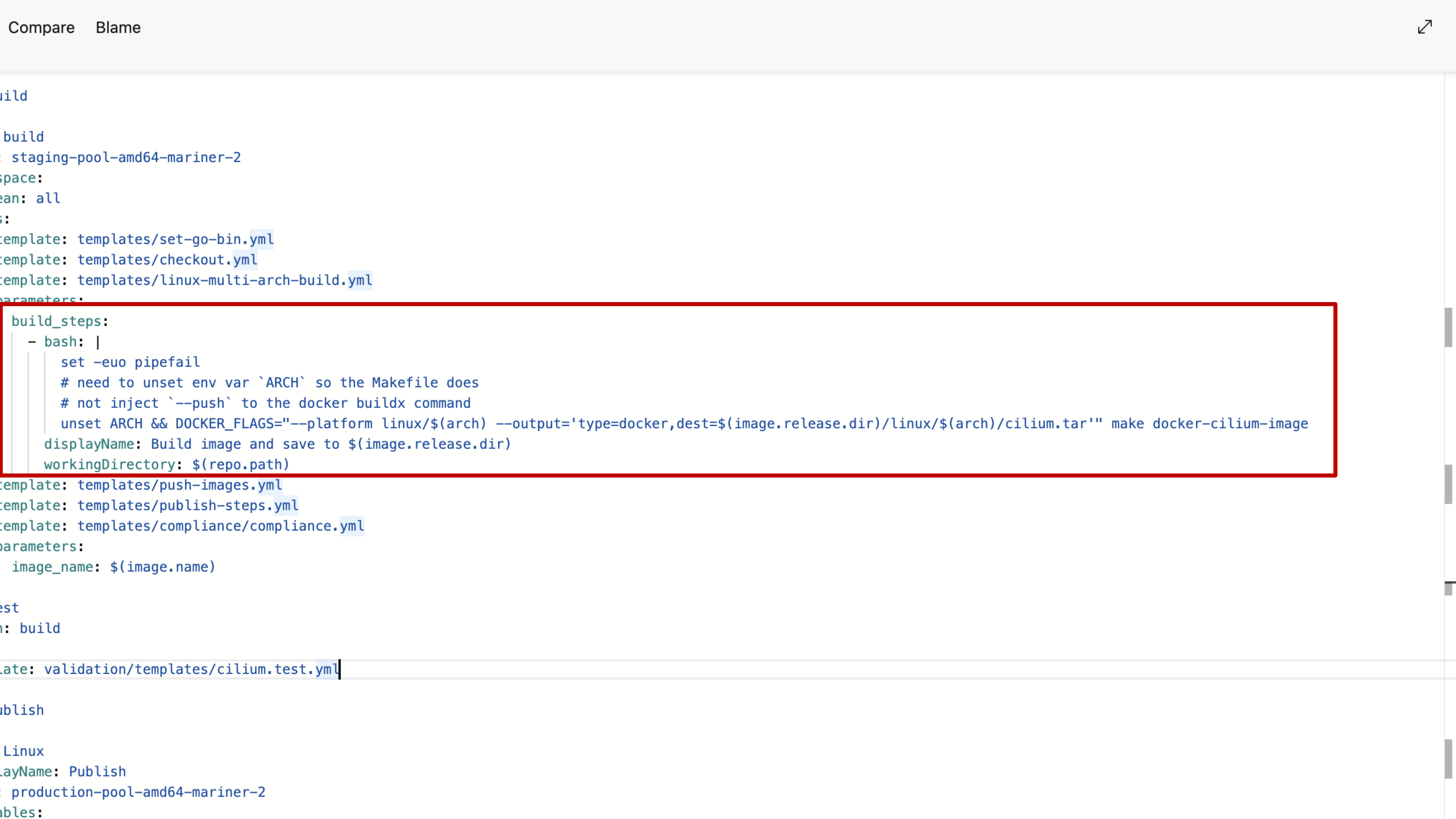


And then build it….

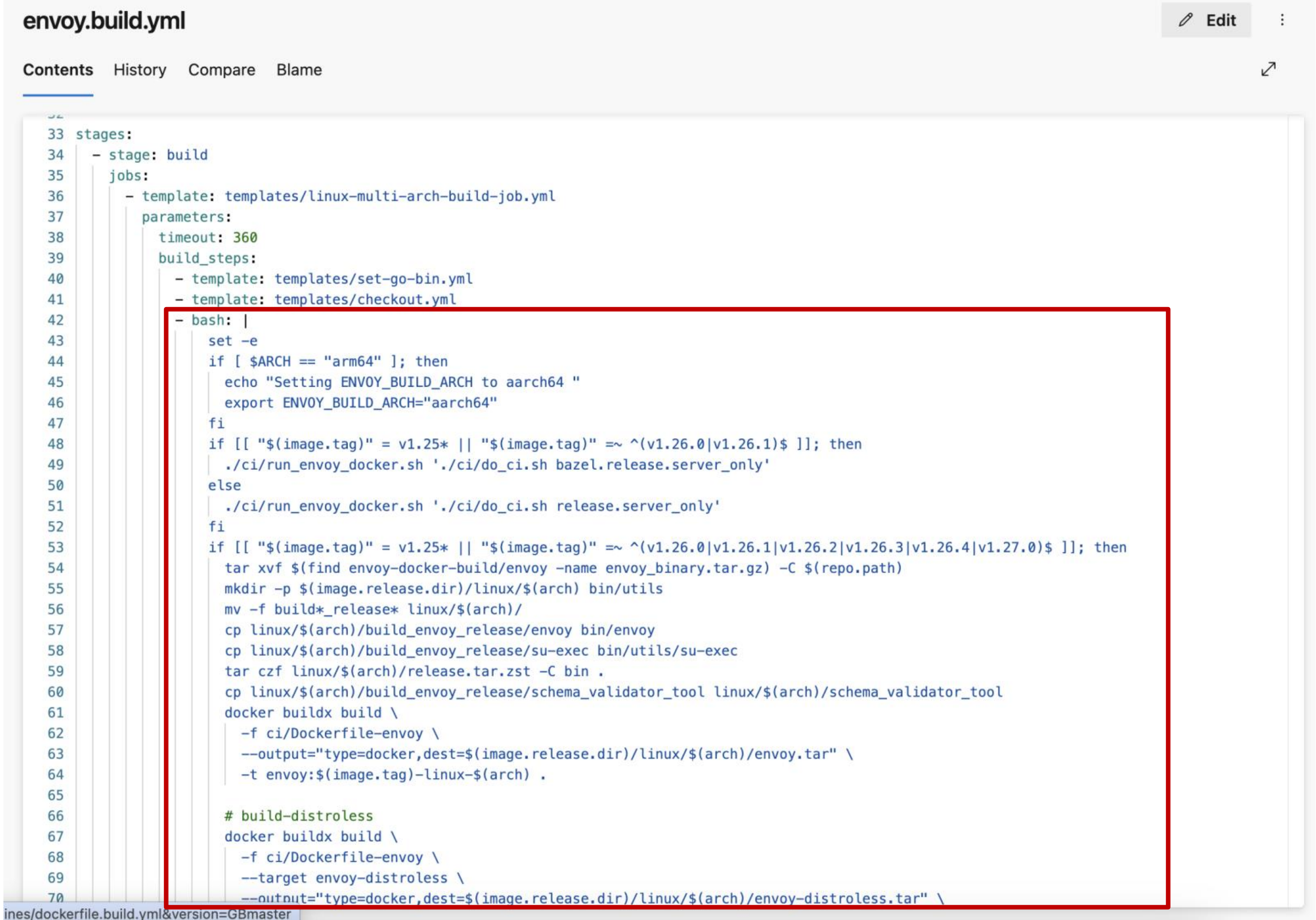
Step 1: Draw some circles

Step 2: Draw the rest of the owl!

- What tools will I need to build this?
- What commands do I need to run this?
- Do I need to build all of this?
- Does the build change between versions?
- How does each project get tested?

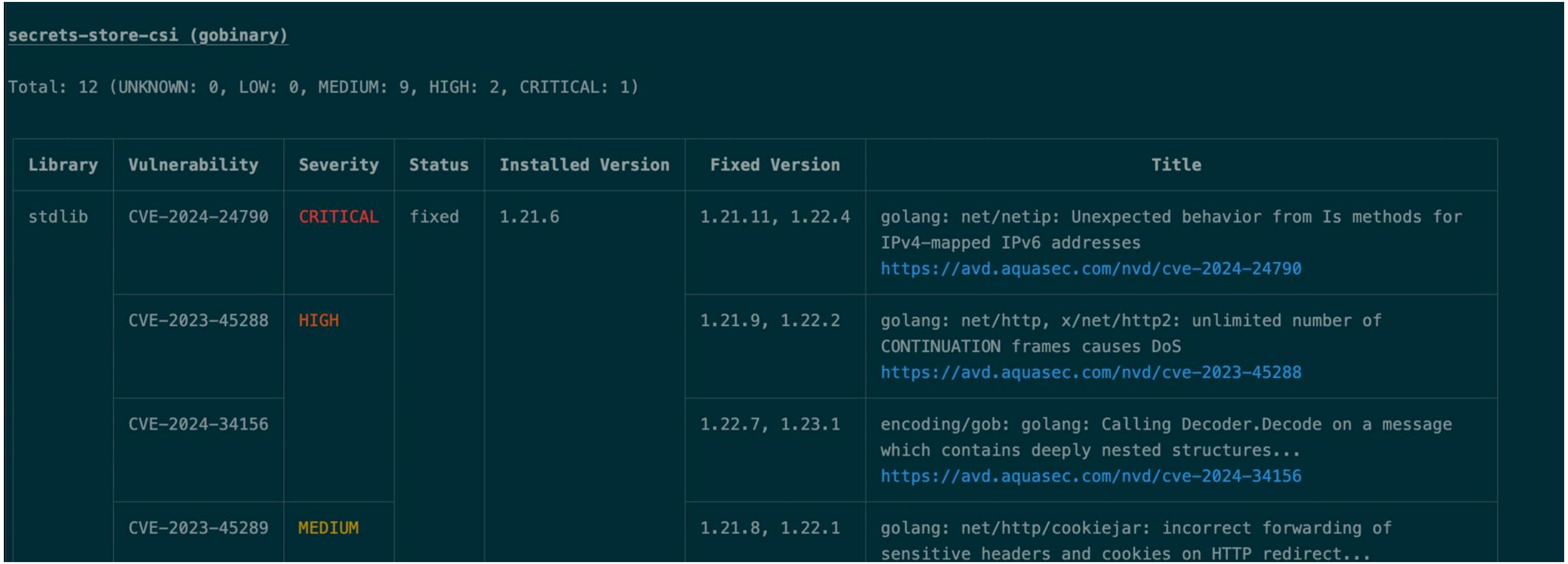


```
uild

 build
: staging-pool-amd64-mariner-2
space:
ean: all
s:
emplate: templates/set-go-bin.yml
emplate: templates/checkout.yml
emplate: templates/linux-multi-arch-build.yml
arameters:
 build_steps:
   - bash: |
       set -euo pipefail
       # need to unset env var `ARCH` so the Makefile does
       # not inject `--push` to the docker buildx command
       unset ARCH && DOCKER_FLAGS="--platform linux/$(arch) --output='type=docker,dest=$(image.release.dir)/linux/$(arch)/cilium.tar'" make docker-cilium-image
     displayName: Build image and save to $(image.release.dir)
     workingDirectory: $(repo.path)
emplate: templates/push-images.yml
emplate: templates/publish-steps.yml
emplate: templates/compliance/compliance.yml
arameters:
 image_name: $(image.name)

est
n: build

ate: validation/templates/cilium.test.yml

ublish

 Linux
ayName: Publish
: production-pool-amd64-mariner-2
ables:
```



```
stages:
  - stage: build
    jobs:
      - template: templates/linux-multi-arch-build-job.yml
        parameters:
          timeout: 360
          build_steps:
            - template: templates/set-go-bin.yml
            - template: templates/checkout.yml
            - bash: |
                set -e
                if [ $ARCH == "arm64" ]; then
                  echo "Setting ENVOY_BUILD_ARCH to aarch64 "
                  export ENVOY_BUILD_ARCH="aarch64"
                fi
                if [[ "$(image.tag)" = v1.25* || "$(image.tag)" =~ ^(v1.26.0|v1.26.1)$ ]]; then
                  ./ci/run_envoy_docker.sh './ci/do_ci.sh bazel.release.server_only'
                else
                  ./ci/run_envoy_docker.sh './ci/do_ci.sh release.server_only'
                fi
                if [[ "$(image.tag)" = v1.25* || "$(image.tag)" =~ ^(v1.26.0|v1.26.1|v1.26.2|v1.26.3|v1.26.4|v1.27.0)$ ]]; then
                  tar xvf $(find envoy-docker-build/envoy -name envoy_binary.tar.gz) -C $(repo.path)
                  mkdir -p $(image.release.dir)/linux/$(arch) bin/utils
                  mv -f build*_release* linux/$(arch)/
                  cp linux/$(arch)/build_envoy_release/envoy bin/envoy
                  cp linux/$(arch)/build_envoy_release/su-exec bin/utils/su-exec
                  tar czf linux/$(arch)/release.tar.zst -C bin .
                  cp linux/$(arch)/build_envoy_release/schema_validator_tool linux/$(arch)/schema_validator_tool
                  docker buildx build \
                    -f ci/Dockerfile-envoy \
                    --output="type=docker,dest=$(image.release.dir)/linux/$(arch)/envoy.tar" \
                    -t envoy:$(image.tag)-linux-$(arch) .

                  # build-distroless
                  docker buildx build \
                    -f ci/Dockerfile-envoy \
                    --target envoy-distroless \
                    --output="type=docker,dest=$(image.release.dir)/linux/$(arch)/envoy-distroless.tar" \
```

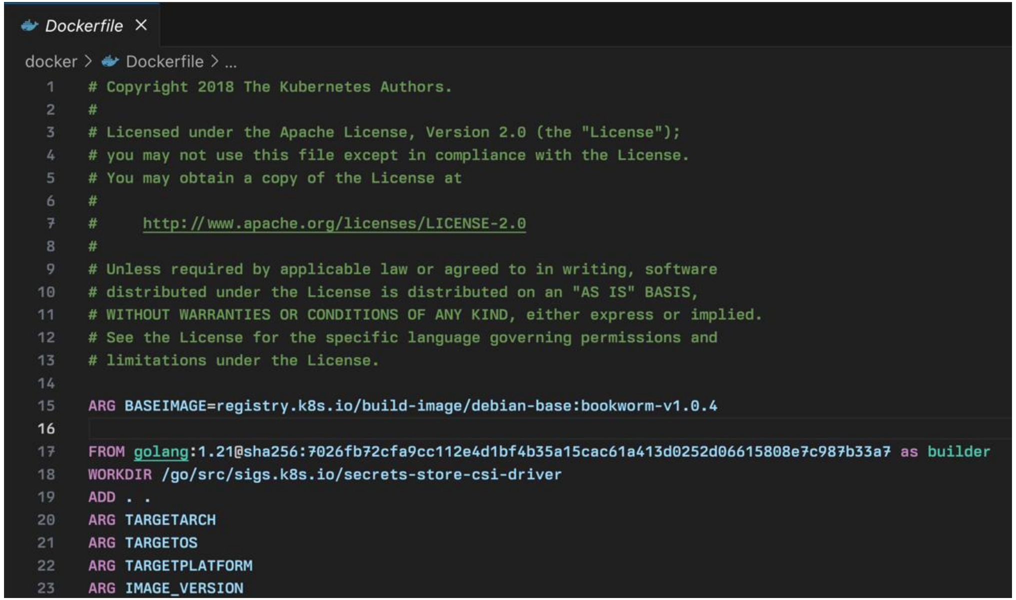
Now.....time to fix things



```
secrets-store-csi (gobinary)

Total: 12 (UNKNOWN: 0, LOW: 0, MEDIUM: 9, HIGH: 2, CRITICAL: 1)

 Library  Vulnerability   Severity  Status  Installed Version  Fixed Version                            Title

 stdlib   CVE-2024-24790  CRITICAL  fixed   1.21.6             1.21.11, 1.22.4  golang: net/netip: Unexpected behavior from Is methods for
                                                                                IPv4-mapped IPv6 addresses
                                                                                https://avd.aquasec.com/nvd/cve-2024-24790

          CVE-2023-45288  HIGH                                 1.21.9, 1.22.2   golang: net/http, x/net/http2: unlimited number of
                                                                                CONTINUATION frames causes DoS
                                                                                https://avd.aquasec.com/nvd/cve-2023-45288

          CVE-2024-34156                                       1.22.7, 1.23.1   encoding/gob: golang: Calling Decoder.Decode on a message
                                                                                which contains deeply nested structures...
                                                                                https://avd.aquasec.com/nvd/cve-2024-34156

          CVE-2023-45289  MEDIUM                               1.21.8, 1.22.1   golang: net/http/cookiejar: incorrect forwarding of
                                                                                sensitive headers and cookies on HTTP redirect...
```

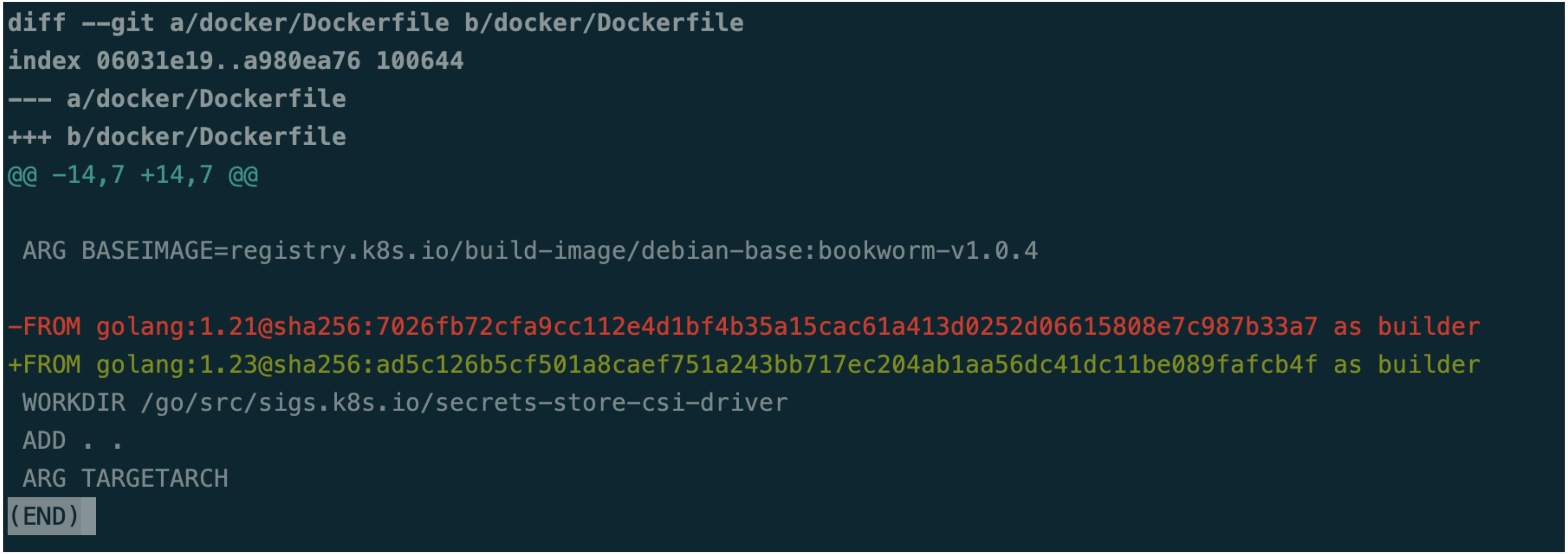
How is it built?

docker buildx build --no-cache \
    --build-arg IMAGE_VERSION=v1.4.6 \
    --output=type=docker,dest=/images/linux/amd64/driver.tar \
    -t kubeconbuildit.azurecr.io/secrets-store/driver:v1.4.6 \
    -f **docker/Dockerfile** .



```
# Copyright 2018 The Kubernetes Authors.
#
# Licensed under the Apache License, Version 2.0 (the "License");
# you may not use this file except in compliance with the License.
# You may obtain a copy of the License at
#
#     http://www.apache.org/licenses/LICENSE-2.0
#
# Unless required by applicable law or agreed to in writing, software
# distributed under the License is distributed on an "AS IS" BASIS,
# WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
# See the License for the specific language governing permissions and
# limitations under the License.

ARG BASEIMAGE=registry.k8s.io/build-image/debian-base:bookworm-v1.0.4

FROM golang:1.21@sha256:7026fb72cfa9cc112e4d1bf4b35a15cac61a413d0252d06615808e7c987b33a7 as builder
WORKDIR /go/src/sigs.k8s.io/secrets-store-csi-driver
ADD . .
ARG TARGETARCH
ARG TARGETOS
ARG TARGETPLATFORM
ARG IMAGE_VERSION
```

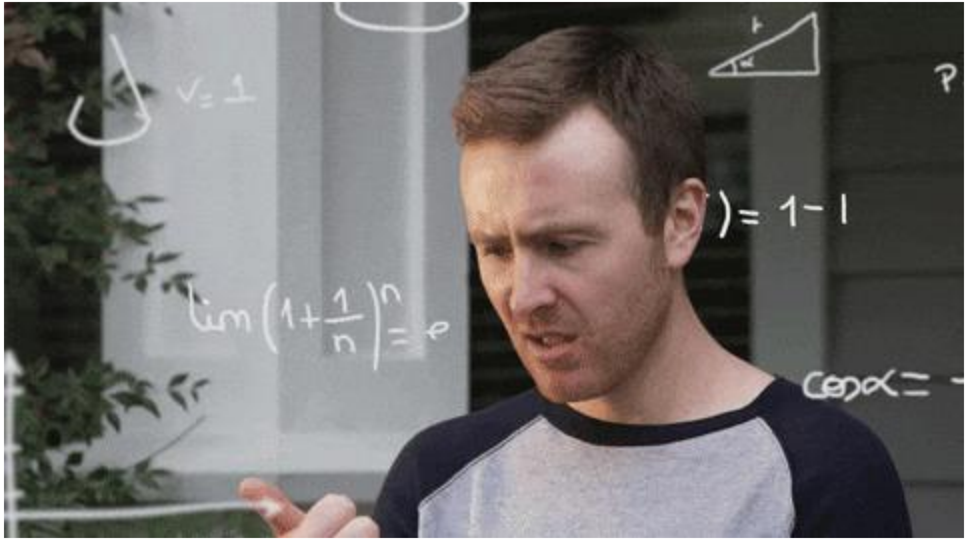
Let's fix it.....



```
diff --git a/docker/Dockerfile b/docker/Dockerfile
index 06031e19..a980ea76 100644
--- a/docker/Dockerfile
+++ b/docker/Dockerfile
@@ -14,7 +14,7 @@

 ARG BASEIMAGE=registry.k8s.io/build-image/debian-base:bookworm-v1.0.4

-FROM golang:1.21@sha256:7026fb72cfa9cc112e4d1bf4b35a15cac61a413d0252d06615808e7c987b33a7 as builder
+FROM golang:1.23@sha256:ad5c126b5cf501a8caef751a243bb717ec204ab1aa56dc41dc11be089fafcb4f as builder
 WORKDIR /go/src/sigs.k8s.io/secrets-store-csi-driver
 ADD . .
 ARG TARGETARCH
(END)
```

Doing this for MANY projects….

- Handling lots of patches?
- Handling build changes over time?
- Handling many different base images and dependencies?
- Have any fixes or patches broken things…



A better approach?

- Focus on a reduced set of base images
- Leverage packages
- Minimize Dependencies
- Standardize Container Builds

Overview of Chainguard Images — Chainguard Academy

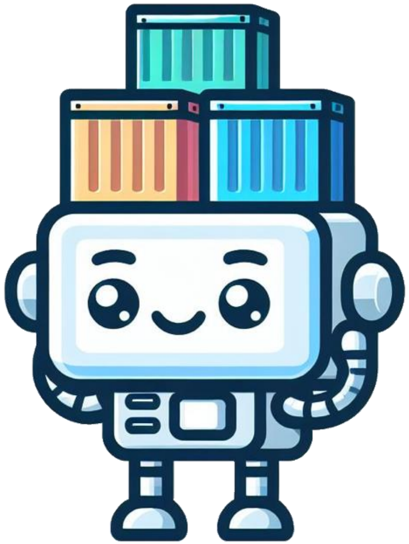
Chainguard Images - Home

Adding other constraints…

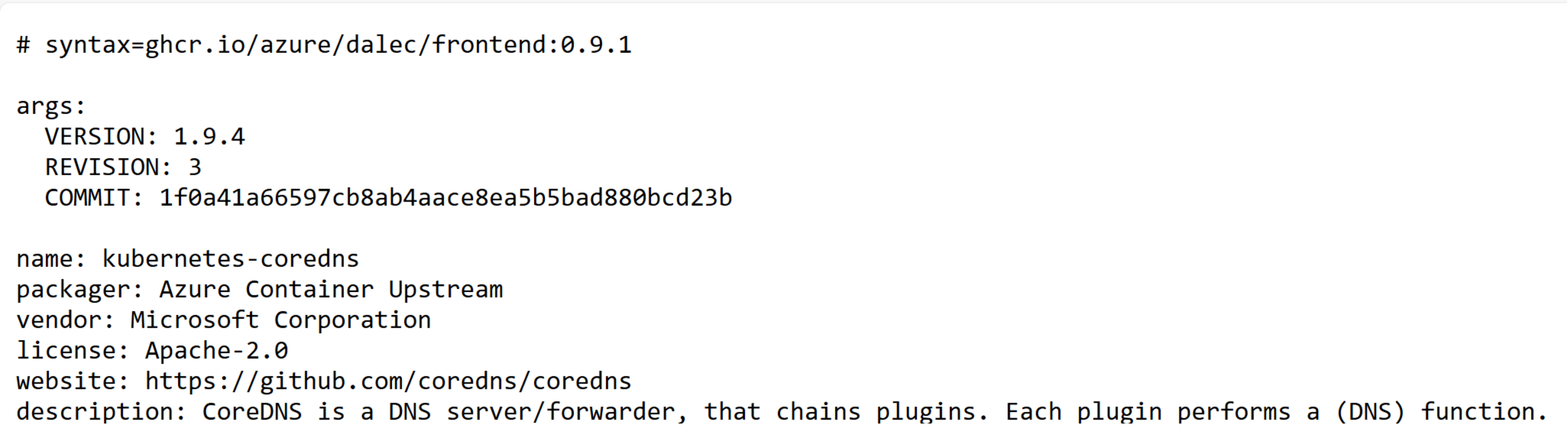
- We have a requirement to use Azure Linux (and make windows and Ubuntu binaries)
- We have a requirement to only use trusted packages
- We have a requirement to build with only Microsoft Go (or other trusted tool chains)
- We have a requirement to rebuild from mirrored source

DALEC

- Build Packages (and windows binaries) from OSS
- Build Images from those packages (and binaries)
- Leverage Azure Linux and Ubuntu for Packages
- Leverage BuildKit to orchestrate this with one set of tooling
- Get “SBOM” and “Provenance” for free
- Leverage a unified yaml config for all builds

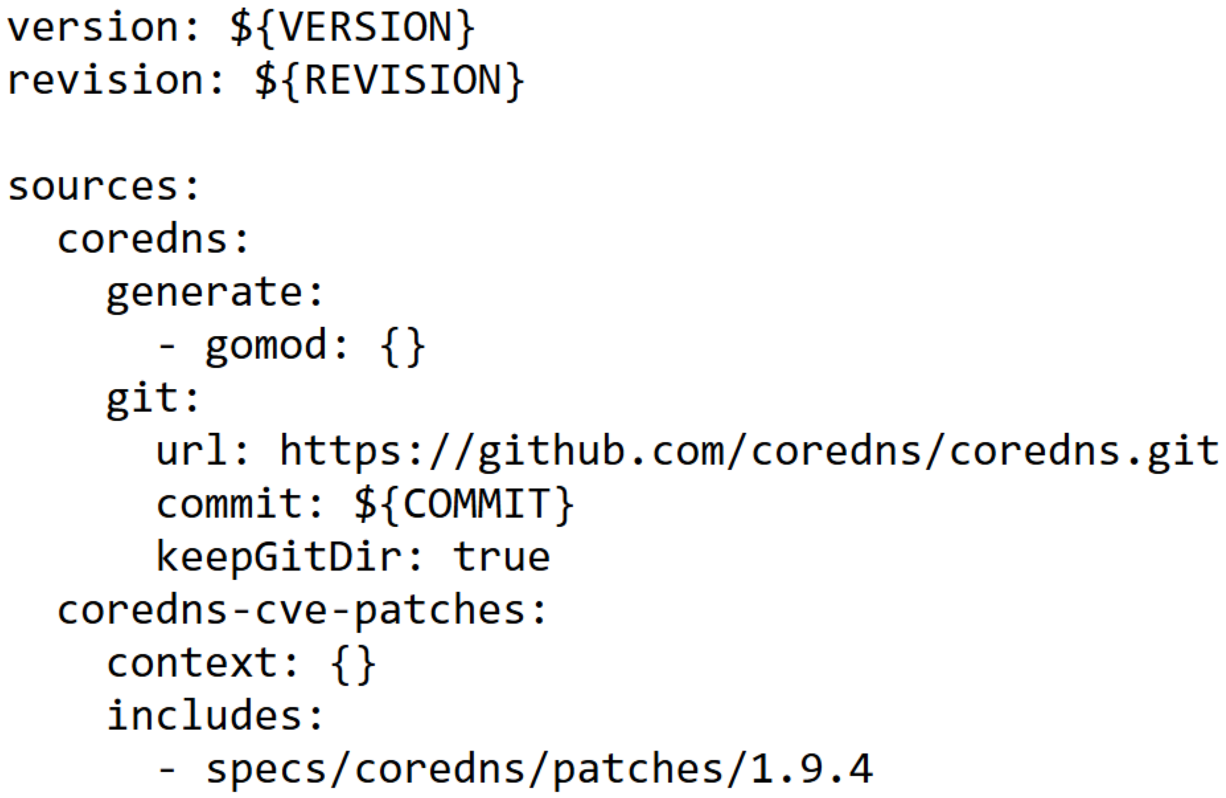


DALEC Spec



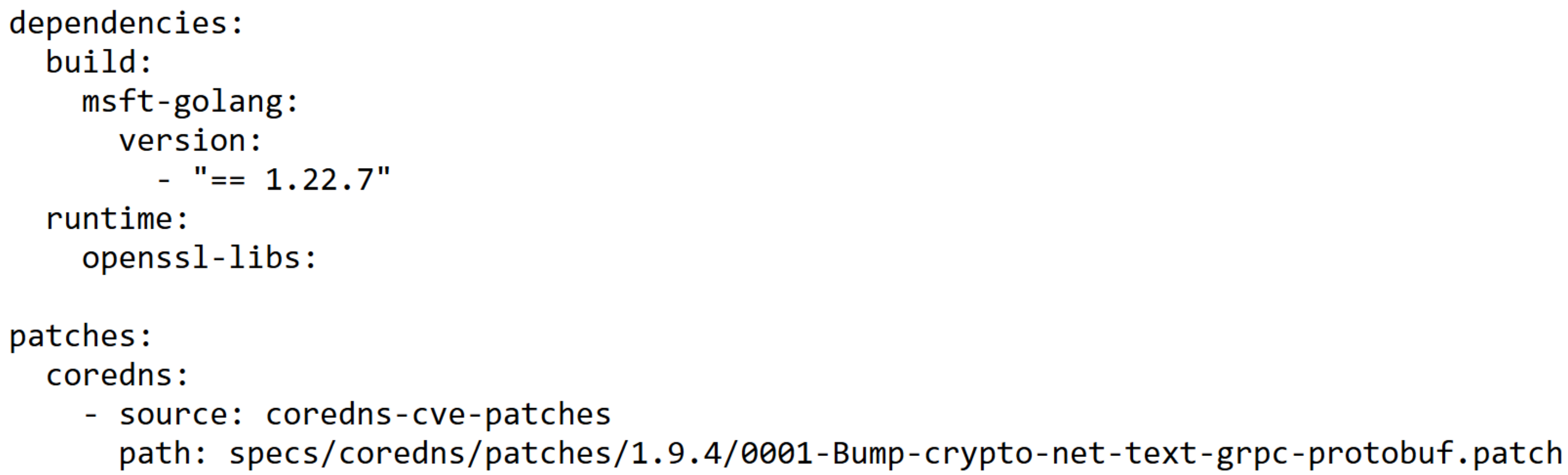
```
# syntax=ghcr.io/azure/dalec/frontend:0.9.1

args:
  VERSION: 1.9.4
  REVISION: 3
  COMMIT: 1f0a41a66597cb8ab4aace8ea5b5bad880bcd23b

name: kubernetes-coredns
packager: Azure Container Upstream
vendor: Microsoft Corporation
license: Apache-2.0
website: https://github.com/coredns/coredns
description: CoreDNS is a DNS server/forwarder, that chains plugins. Each plugin performs a (DNS) function.
```

DALEC Spec



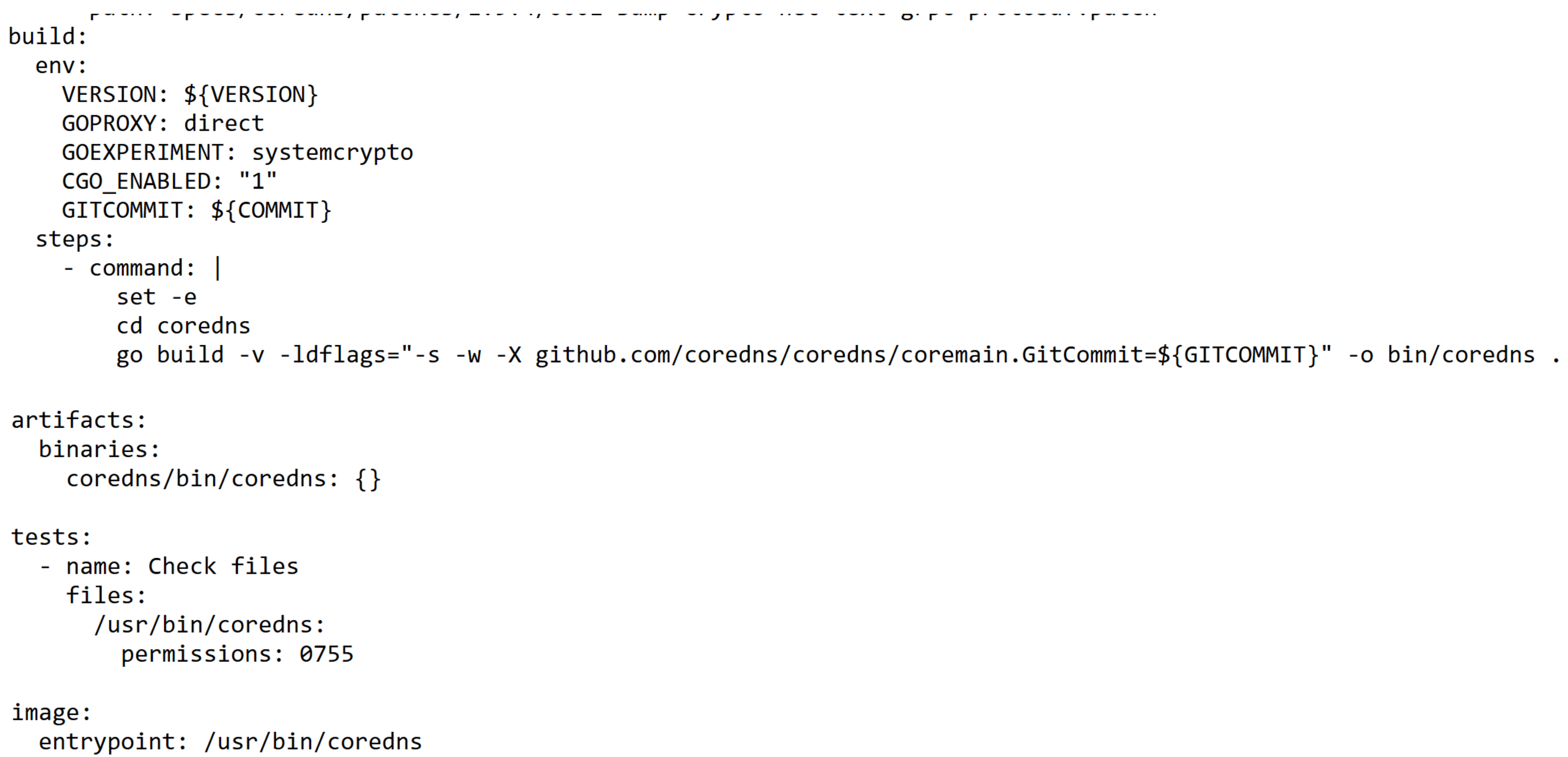
```yaml
version: ${VERSION}
revision: ${REVISION}

sources:
  coredns:
    generate:
      - gomod: {}
    git:
      url: https://github.com/coredns/coredns.git
      commit: ${COMMIT}
      keepGitDir: true
  coredns-cve-patches:
    context: {}
    includes:
      - specs/coredns/patches/1.9.4
```

DALEC Spec

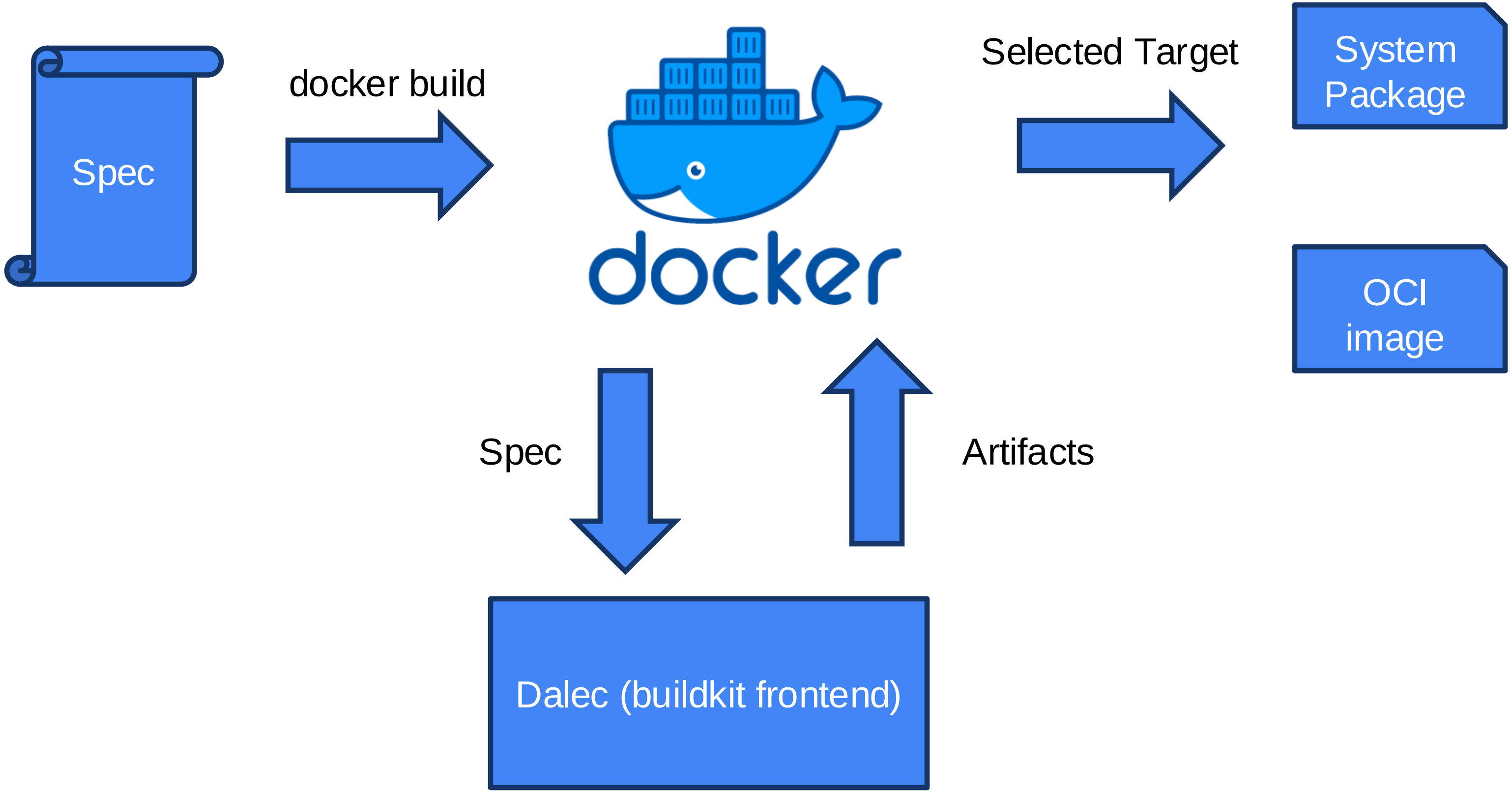


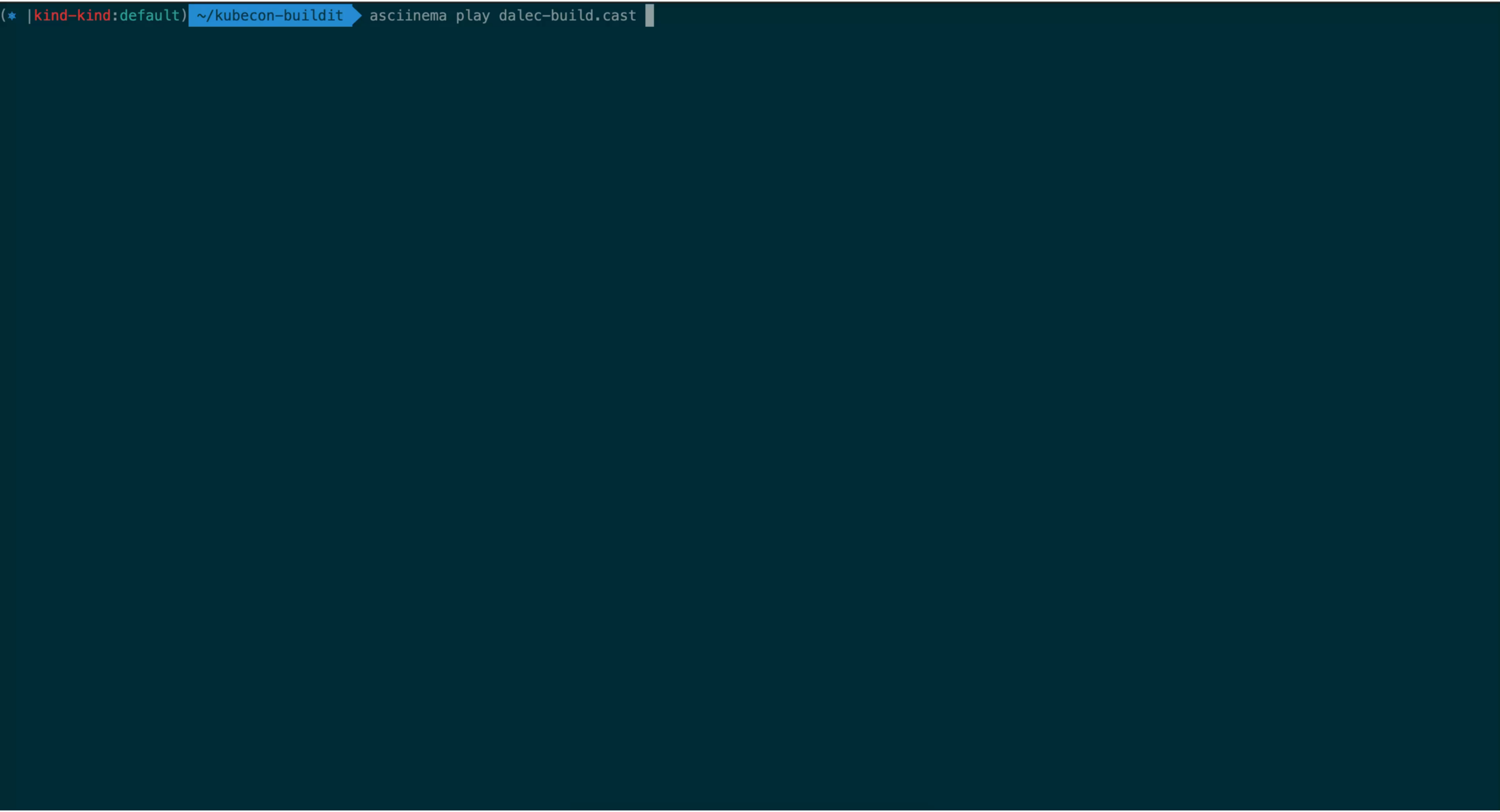
```yaml
dependencies:
  build:
    msft-golang:
      version:
        - "== 1.22.7"
  runtime:
    openssl-libs:

patches:
  coredns:
    - source: coredns-cve-patches
      path: specs/coredns/patches/1.9.4/0001-Bump-crypto-net-text-grpc-protobuf.patch
```

DALEC Spec



```
build:
  env:
    VERSION: ${VERSION}
    GOPROXY: direct
    GOEXPERIMENT: systemcrypto
    CGO_ENABLED: "1"
    GITCOMMIT: ${COMMIT}
  steps:
    - command: |
        set -e
        cd coredns
        go build -v -ldflags="-s -w -X github.com/coredns/coredns/coremain.GitCommit=${GITCOMMIT}" -o bin/coredns .

artifacts:
  binaries:
    coredns/bin/coredns: {}

tests:
  - name: Check files
    files:
      /usr/bin/coredns:
        permissions: 0755

image:
  entrypoint: /usr/bin/coredns
```


Spec
docker build
docker
Selected Target
System Package
OCI image
Spec
Artifacts
Dalec (buildkit frontend)



```
(⎈ |kind-kind:default) ~/kubecon-buildit  asciinema play dalec-build.cast
```

Summary

- Mirror OSS dependencies to start with
- Scan and Sign Before Use, Enforce / Validate on Use
- Copa can help you patch and update those
- Building from source is a better idea
- It’s not free though….make sure you allocate resources

Links

- https://github.com/ossf/s2c2f/blob/main/specification/README.md
- https://oras.land
- https://github.com/notaryproject/notation
- https://github.com/project-copacetic/copacetic
- https://edu.chainguard.dev/chainguard/chainguard-images/
- https://github.com/Azure/dalec

Thank You