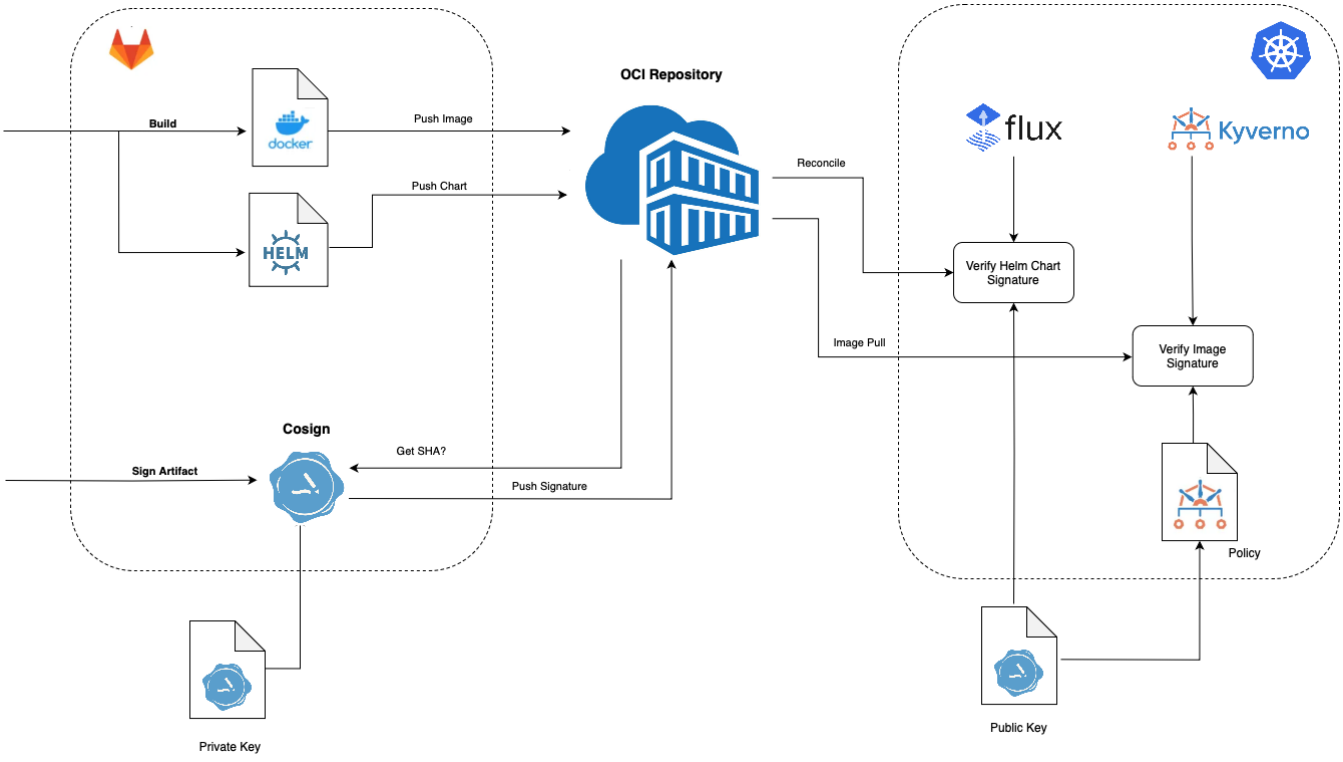
Cosign

Overview

Cosign is a command line utility that can sign and verify software artifact, such as container images and blobs.

In Kubernetes, we can use FluxCD and Kyverno to verify Helmcharts and Docker images respectively.



Installation

https://docs.sigstore.dev/system_config/installation/

```
wget https://github.com/sigstore/cosign/releases/download/v2.2.3/cosign-darwin-arm64
chmod +x cosign-darwin-arm64
sudo cp cosign-darwin-arm64 /usr/local/bin/cosign
```

Signing a Blob

```
> cosign sign-blob <file> --bundle cosign.bundle
```

Output:

```
Using payload from: gotk-components.yaml
Generating ephemeral keys...
Retrieving signed certificate...

        The sigstore service, hosted by sigstore a Series of LF Projects, LLC, is provided pursuant to the
Hosted Project Tools Terms of Use, available at https://lfprojects.org/policies/hosted-project-tools-terms-of-
use/.
        Note that if your submission includes personal data associated with this signed artifact, it will be
part of an immutable record.
        This may include the email address associated with the account with which you authenticate your
contractual Agreement.
        This information will be used for signing this artifact and will be stored in public transparency logs
and cannot be removed later, and is subject to the Immutable Record notice at https://lfprojects.org/policies
/hosted-project-tools-immutable-records/.

By typing 'y', you attest that (1) you are not submitting the personal data of any other person; and (2) you
understand and agree to the statement and the Agreement terms at the URLs listed above.
Are you sure you would like to continue? [y/N] y
Your browser will now be opened to:
https://oauth2.sigstore.dev/auth/auth?
access_type=online&client_id=sigstore&code_challenge=1cboQoe9SlmimuCDgYZheH1MrByX9epG7ceMPs9uKWg&code_challenge_
method=S256&nonce=2ddexWMzBTyUEvDVBOwx9Mb15Zc&redirect_uri=http%3A%2F%2Flocalhost%3A64841%2Fauth%
2Fcallback&response_type=code&scope=openid+email&state=2ddexTMg9MYyDZlV9aKb6QkSLWn
Successfully verified SCT...
using ephemeral certificate:
-----BEGIN CERTIFICATE-----
...
-----END CERTIFICATE-----

tlog entry created with index: 77825201
using ephemeral certificate:
-----BEGIN CERTIFICATE-----
...
-----END CERTIFICATE-----

Wrote bundle to file cosign.bundle
MEQCIEI2pfffhoN3xAFLF0ffuACLk00Z4NVr6XlLFUFOlTT+AiAXivuc3Zja7e4hjR3r63NOlZ9pvysWoaB3CADNaCMmAQ==
```

Verifying

```
cosign verify-blob gotk-components.yaml --bundle cosign.bundle --certificate-identity=jmehan@yahoo.com --
certificate-oidc-issuer=https://github.com/login/oauth
```

```
Verified OK
```

Generating a Key Pair

```
cosign generate-key-pair
```

```
Enter password for private key:
Enter password for private key again:
Private key written to cosign.key
Public key written to cosign.pub
```

Signing and Verifying a Container

Get the digest of the image

```
docker inspect --format='{{index .RepoDigests 0}}' ncydacrinprogress.azurecr.io/cybersecuritydome/kafka-azure-sink:22.0.1-3868980
```

```
ncydacrinprogress.azurecr.io/cybersecuritydome/kafka-azure-sink@sha256:
2fbb556a6a2b68466def031067c18411693c6a9f3b5e4b16c1677e28c0029172
```

Sign

```
cosign sign --key cosign.key ncydacrinprogress.azurecr.io/cybersecuritydome/kafka-azure-sink@sha256:
2fbb556a6a2b68466def031067c18411693c6a9f3b5e4b16c1677e28c0029172
```

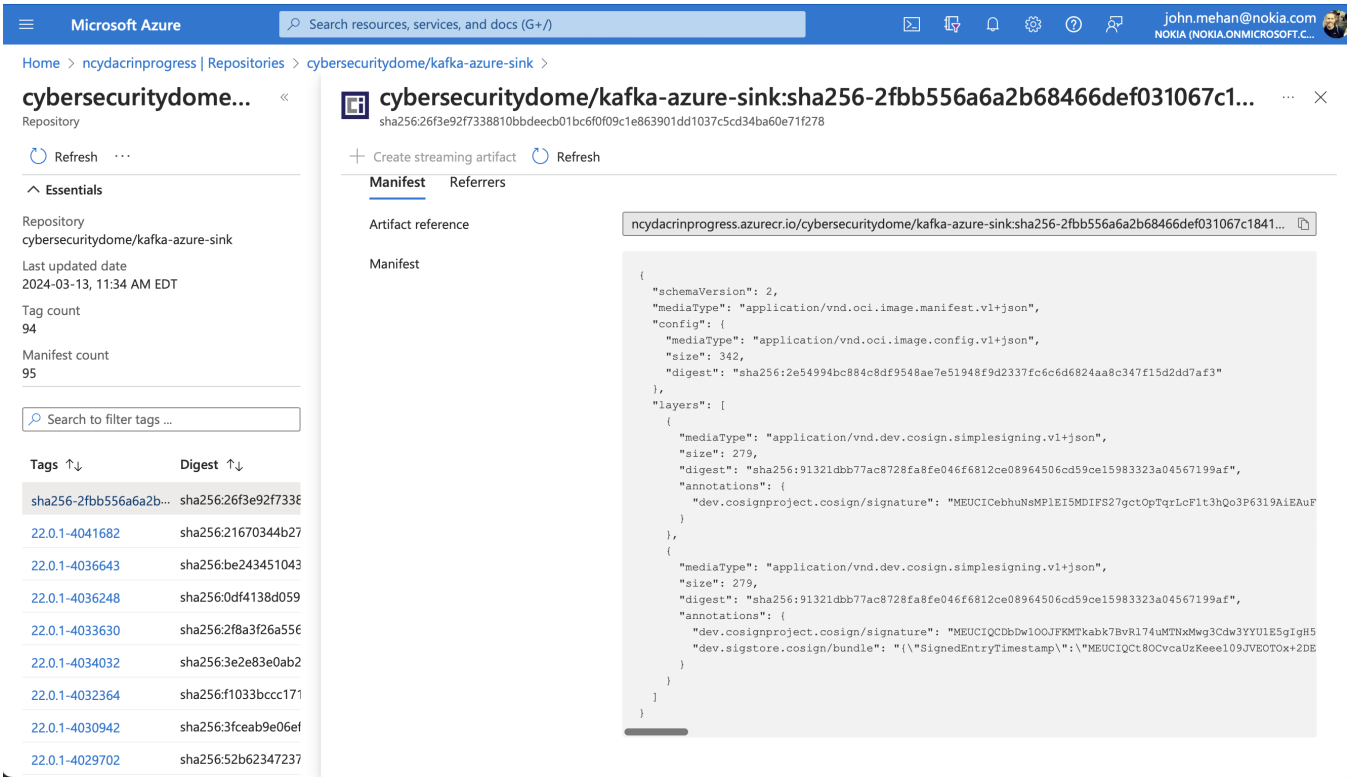
```
Enter password for private key:
WARNING: "ncydacrinprogress.azurecr.io/cybersecuritydome/kafka-azure-sink" appears to be a private repository, please confirm uploading to the transparency log at "https://rekor.sigstore.dev"
Are you sure you would like to continue? [y/N] y

        The sigstore service, hosted by sigstore a Series of LF Projects, LLC, is provided pursuant to the Hosted Project Tools Terms of Use, available at https://lfprojects.org/policies/hosted-project-tools-terms-of-use/.
        Note that if your submission includes personal data associated with this signed artifact, it will be part of an immutable record.
        This may include the email address associated with the account with which you authenticate your contractual Agreement.
        This information will be used for signing this artifact and will be stored in public transparency logs and cannot be removed later, and is subject to the Immutable Record notice at https://lfprojects.org/policies/hosted-project-tools-immutable-records/.

By typing 'y', you attest that (1) you are not submitting the personal data of any other person; and (2) you understand and agree to the statement and the Agreement terms at the URLs listed above.
Are you sure you would like to continue? [y/N] y


tlog entry created with index: 77828277
Pushing signature to: ncydacrinprogress.azurecr.io/cybersecuritydome/kafka-azure-sink
```

View Container in Azure



Verify Signature

```
cosign verify --key cosign.pub ncydacrinprogress.azurecr.io/cybersecuritydome/kafka-azure-sink@sha256:
2fbb556a6a2b68466def031067c18411693c6a9f3b5e4b16c1677e28c0029172
```

```
Verification for ncydacrinprogress.azurecr.io/cybersecuritydome/kafka-azure-sink@sha256:
2fbb556a6a2b68466def031067c18411693c6a9f3b5e4b16c1677e28c0029172 --
The following checks were performed on each of these signatures:
  - The cosign claims were validated
  - Existence of the claims in the transparency log was verified offline
  - The signatures were verified against the specified public key
```

```
[
 {
   "critical": {
     "identity": {
       "docker-reference": "ncydacrinprogress.azurecr.io/cybersecuritydome/kafka-azure-sink"
     },
     "image": {
       "docker-manifest-digest": "sha256:2fbb556a6a2b68466def031067c18411693c6a9f3b5e4b16c1677e28c0029172"
     },
     "type": "cosign container image signature"
   },
   "optional": {
     "Bundle": {
       "SignedEntryTimestamp": "MEUCIQCt8OCvcaUzKeee109JVEOTOx+2DEKA5SCEd5R
/BaXA1QIgI91Ebfv6MEx5F2OW05yU8kSxS3kwrXAP5/beU7CQBc0=",
       "Payload": {
         "body":
"eyJhcGlWZXJzaW9uIjoiMC4wLjEiLCJraW5kIjoiaGFzaGVkcmVrb3JkIiwic3BlYyI6eyJkYXRhIjp7Imhhc2giOnsiYWxnb3JpdGhtIjoic2h
hMjU2IiwidmFsdWUiOiI5MTMyMWRiYjc3YWM4NzI4ZmE4ZmUwNDZmNjgxMmNlMDg5NjQ1MDZjZDU5Y2UxNTk4MzMyM2EwNDU2NzE5OWFmIn19LCJ
zaWduYXR1cmUiOnsiY29udGVudCI6Ik1FVUNJUUNEYkR3MU9PSkZLTVRrYWJrN0J2Umw3NHVNVE54TXdnM0NkdzNZWVUxRTVnSWdINWp1UmJRd2t
nSnozZEZJNkEreFIxc2tnV2N3NmFsTUdublFKaU9PNC80PSIsInB1YmxpY0tleSI6eyJjb250ZW50IjoiTFMwdExTMUNSVWRKVGlCUVZVSk1TVU1
nUzBWWkxTMHRMUzBLVFVacmQwVjNXVWhMYjFwSmVtb3dRMEZSV1VsTGIxcEplbW93UkVGUlkwUlJaMEZGTmpnNE56a3pPVlZtVkRsUFVFMUlkbE5
VTjA5Q1psUXhlRUYyWVFwcFVsQmlRakZJZVdGeUsyNUdRMVZYVm5aWU4wVjJhVVZRVEhoVVdsSk9VVEpCTkU5UVMwRnJSRzh4WlROSVNUaFBSbFJ
5T1ZwQlNYbFJQVDBLTFMwdExTMUZUa1FnVUZWQ1RFbERJRXRGV1MwdExTMHRDZz09In19fX0=",
         "integratedTime": 1710344087,
         "logIndex": 77828277,
         "logID": "c0d23d6ad406973f9559f3ba2d1ca01f84147d8ffc5b8445c224f98b9591801d"
       }
     }
   }
 }
]
```


Signing and Verifying a Helm Chart

Assuming you have pushed a helm chart to your oci repository

```
helm push <app-name>-<app-version>.tgz oci://<registry-host>/<org>/charts
```

Sign

```
cosign sign --key cosign.key <registry-host>/<org>/charts/<app-name>:<app-version>

ex:
cosign sign --key cosign.key ncydacrinprogress.azurecr.io/charts/kowl:22.0.1-4040670
```

Add the public key to the cluster

```
kubectl -n flux-system create secret generic cosign-pub --from-file=cosign.pub=cosign.pub

ex:
cd ~/cosign
kubectl -n ncyd-flux create secret generic cosign-pub --from-file=cosign.pub=cosign.pub
```

Modify helmrelease to verify the helmchart

```
apiVersion: helm.toolkit.fluxcd.io/v2beta2
kind: HelmRelease
metadata:
  name: <app-name>
spec:
  interval: 1h
  chart:
    spec:
      chart: <app-name>
      version: <app-version>
      sourceRef:
        kind: HelmRepository
        name: helm-charts
      verify:
        provider: cosign
        secretRef:
          name: cosign-pub
```

When using a customization override

kowl.yaml

```
---
# set $patch: delete to exclude from installation
#$patch: delete

apiVersion: helm.toolkit.fluxcd.io/v2beta1
kind: HelmRelease
metadata:
  name: kowl
  namespace: ncyd-flux
spec:
  chart:
    spec:
      version: '22.0.1-4040670'
      verify:
        provider: cosign
        secretRef:
          name: cosign-pub
  values:
    imagePullSecrets:
      - name: regcred
    image:
      registry: ncydacrinprogress.azurecr.io
```

Reconcile and watch for events in the namespace

```
flux reconcile kustomization ncyd-flux -n ncyd-flux

kubectl events -n ncyd-flux --watch
```

Error from source-controller logs:

```
chart verification error: not implemented


{"level":"error","ts":"2024-03-14T15:19:53.977Z","msg":"Reconciler error","controller":"helmchart","
controllerGroup":"source.toolkit.fluxcd.io","controllerKind":"HelmChart","HelmChart":{"name":"ncyd-flux-kowl","
namespace":"ncyd-flux"},"namespace":"ncyd-flux","name":"ncyd-flux-kowl","reconcileID":"d103f844-f750-49cf-ba56-
a631037fd585","error":"chart verification error: not implemented"}
```

I think we need to define an oci repository. Currently we have a helm repository.

Definen OCI Repository

Use it..

kowl.yaml

```
---
# set $patch: delete to exclude from installation
#$patch: delete

apiVersion: helm.toolkit.fluxcd.io/v2beta1
kind: HelmRelease
metadata:
  name: kowl
  namespace: ncyd-flux
spec:
  chart:
    spec:
      version: '22.0.1-4040670'

      sourceRef:
        kind: HelmRepository
        name: ncyd-oci-virtual
      verify:
        provider: cosign
        secretRef:
          name: cosign-pub

  values:
    imagePullSecrets:
      - name: regcred
    image:
      registry: ncydacrinprogress.azurecr.io
```

If the signature is bad

```
> kubectl events -n ncyd-flux --watch

0s (x4 over 20s)       Warning   ChartVerificationError       HelmChart/ncyd-flux-kowl
chart verification error: failed to verify oci://ncydacrinprogress.azurecr.io/charts/kowl:22.0.1-4042072: no
matching signatures: invalid signature when validating ASN.1 encoded signature


> f logs -f source-controller-f7dbc4597-vwwj2

{"level":"error","ts":"2024-03-15T16:08:05.400Z","msg":"Reconciler error","controller":"helmchart","
controllerGroup":"source.toolkit.fluxcd.io","controllerKind":"HelmChart","HelmChart":{"name":"ncyd-flux-kowl","
namespace":"ncyd-flux"},"namespace":"ncyd-flux","name":"ncyd-flux-kowl","reconcileID":"c29c0b58-cf10-4d1d-a290-
ecb997acf1ac","error":"chart verification error: failed to verify oci://ncydacrinprogress.azurecr.io/charts
/kowl:22.0.1-4042072: no matching signatures: invalid signature when validating ASN.1 encoded signature"}
```

Process for Using Cosign with HelmCharts in Flux

- Need to generate keys

```
cosign generate-key-pair
```

- Need to store public key in k8s

```
kubectl -n flux-system create secret generic cosign-pub --from-file=cosign.pub=cosign.pub

ex:
cd ~/cosign
kubectl -n ncyd-flux create secret generic cosign-pub --from-file=cosign.pub=cosign.pub
```

- Need to use OCI helmrepository

```
---
apiVersion: source.toolkit.fluxcd.io/v1beta2
kind: HelmRepository
metadata:
  name: ncyd-oci-virtual
  namespace: ncyd-flux
spec:
  type: "oci"
  interval: 1m0s
  secretRef:
    name: regcred
  url: oci://ncydacrinprogress.azurecr.io/charts
```

- Need to sign helmchart

```
cosign sign --key cosign.key <registry-host>/<org>/charts/<app-name>:<app-version>

ex:
cosign sign --key cosign.key ncydacrinprogress.azurecr.io/charts/kowl:22.0.1-4040670
```

- Need to update helm releases to use OCI and reference cosign public key

kowl.yaml

```
apiVersion: helm.toolkit.fluxcd.io/v2beta1
kind: HelmRelease
metadata:
  name: kowl
  namespace: ncyd-flux
spec:
  chart:
    spec:
      sourceRef:
        kind: HelmRepository
        name: ncyd-oci-virtual
      verify:
        provider: cosign
        secretRef:
          name: cosign-pub
  values:
...
```

Signing and Verifying Images

https://fluxcd.io/blog/2022/02/security-image-provenance/

https://kyverno.io/docs/writing-policies/verify-images/

Install Kyverno

```
helm repo add kyverno https://kyverno.github.io/kyverno/
helm repo update
helm install -n ncyd-flux kyverno kyverno/kyverno
```

Update the kyverno deployment

```
f edit deployment/kyverno-admission-controller
```

```
     containers:
      - args:
...
        - --imagePullSecrets=regcred
```

Create Policy:

```
apiVersion: kyverno.io/v1
kind: Policy
metadata:
  name: check-image
spec:
  validationFailureAction: Enforce
  background: false
  webhookTimeoutSeconds: 30
  failurePolicy: Fail
  rules:
    - name: check-image
      match:
        any:
        - resources:
            kinds:
              - Pod
      verifyImages:
      - imageReferences:
        - "ncydacrinprogress.azurecr.io/cloudhut/kowl:*"
        attestors:
        - count: 1
          entries:
          - keys:
              publicKeys: |-
                -----BEGIN PUBLIC KEY-----
                MFkwEwYHKoZIzj0CAQYIKoZIzj0DAQcDQgAE6887939UfT9OPMHvST7OBfT1xAva
                iRPbB1Hyar+nFCUWVvX7EviEPLxTZRNQ2A4OPKAkDo1e3HI8OFTr9ZAIyQ==
                -----END PUBLIC KEY-----
```

> *https://kyverno.io/docs/writing-policies/verify-images/sigstore/*
>
> ***Note***
>
> *The public key may either be defined in the policy directly or reference a standard Kubernetes Secret elsewhere in the cluster by specifying it in the format `k8s://<namespace>/<secret_name>`. The named Secret must specify a key `cosign.pub`containing the public key used for verification. Secrets may also be referenced using the `secret{}` object. See `kubectl explain clusterpolicy.spec.rules.verifyImages.attestors.entries.keys` for more details on the supported key options.*

Testing

```
> kubectl events -n ncyd-flux --watch


0s                      Warning   error                       HelmRelease
/kowl                                                 Helm install failed: 1 error occurred:
                        * admission webhook "mutate.kyverno.svc-fail" denied the request:

resource Deployment/ncyd/kowl was blocked due to the following policies

check-image:
  autogen-check-image: 'failed to verify image ncydacrinprogress.azurecr.io/cloudhut/kowl:v1.5.0:
    .attestors[0].entries[0].keys: GET https://ncydacrinprogress.azurecr.io/oauth2/token?scope=repository%!A
(MISSING)cloudhut%!F(MISSING)kowl%!A(MISSING)pull&service=ncydacrinprogress.azurecr.io:
    UNAUTHORIZED: authentication required, visit https://aka.ms/acr/authorization
    for more information.'




Last Helm logs:
0s                      Warning   error                       HelmRelease
/kowl                                                reconciliation failed: Helm install failed: 1 error
occurred:
                        * admission webhook "mutate.kyverno.svc-fail" denied the request:

resource Deployment/ncyd/kowl was blocked due to the following policies

check-image:
  autogen-check-image: 'failed to verify image ncydacrinprogress.azurecr.io/cloudhut/kowl:v1.5.0:
    .attestors[0].entries[0].keys: GET https://ncydacrinprogress.azurecr.io/oauth2/token?scope=repository%!A
(MISSING)cloudhut%!F(MISSING)kowl%!A(MISSING)pull&service=ncydacrinprogress.azurecr.io:
    UNAUTHORIZED: authentication required, visit https://aka.ms/acr/authorization
    for more information.'
```

Script to Sign Helm Charts and Docker Images

signArtifacts.sh

```
#!/bin/bash

#source optional env file
set -a
. ~/cosign/env
set -e +a

#************ SAMPLE ENV FILE ************
#BUILD="22.0.1-4040670"
#SRE_BUILD="23.2.0-3610795"
#OCI_REPO="xxx.azurecr.io"
#OCI_REPO_USERNAME=xxx
#OCI_REPO_PASSWORD="xxx"
#COSIGN_PRIVATE_KEY="-----BEGIN ENCRYPTED SIGSTORE PRIVATE KEY-----
#...
#-----END ENCRYPTED SIGSTORE PRIVATE KEY-----"
#COSIGN_PUBLIC_KEY="-----BEGIN PUBLIC KEY-----
#...
#-----END PUBLIC KEY-----"


export COSIGN_PASSWORD=""

signArtifact () {
```

```
  cosign sign --yes --key env://COSIGN_PRIVATE_KEY \
    --registry-username="${OCI_REPO_USERNAME}" \
    --registry-password="${OCI_REPO_PASSWORD}" \
    ${OCI_REPO}/$1
}

verifyArtifact(){
  cosign verify --key env://COSIGN_PUBLIC_KEY \
    --registry-username="${OCI_REPO_USERNAME}" \
    --registry-password="${OCI_REPO_PASSWORD}" \
    ${OCI_REPO}/$1 >/dev/null
}

source images.src
source charts.src

images=(${images[@]})
charts=(${charts[@]})

requiredBins=(
  "cosign"
)

for bin in "${requiredBins[@]}"; do
  if ! command -v ${bin} &> /dev/null
  then
    echo "Required command ${bin} could not be found, please install and re-run"
    exit
  fi
done

echo
echo "=================="
echo "SIGNING CHARTS"
echo "=================="
for chart in "${charts[@]}"; do
  chartName=`echo ${chart}|sed 's/:.*//'`
  chartVersion=`echo ${chart}|sed -e 's/.*://'`
  echo "ChartName: ${chartName}"
  echo "ChartVersion: ${chartVersion}"
  echo "------------------"
  signArtifact charts/${chartName}:${chartVersion}
  verifyArtifact charts/${chartName}:${chartVersion}
  echo "------------------"
done

echo
echo "=================="
echo "SIGNING IMAGES"
echo "=================="
for image in "${images[@]}"; do
  imageNameAndVersion=`echo $image |sed 's:[^/]*/\(.*\):\1:'`
  imageName=`echo ${imageNameAndVersion}|sed -e 's/:.*//'`
  imageVersion=`echo ${imageNameAndVersion}|sed -e 's/.*://'`
  echo "Image Name: ${imageName}"
  echo "Image Version: ${imageVersion}"
  echo "------------------"
  signArtifact ${imageName}:${imageVersion}
  verifyArtifact ${imageName}:${imageVersion}
  echo "------------------"
done


echo ""
echo "DONE!"
```

images.src

```
images=(
  pkg/fluent-bit:$BUILD"
)
```

charts.src

```
charts=(
  "mychart:$BUILD"
  "ckaf/kafka/rocky8:8.4.2-7.3.1-7486"
)
```

Helpful Scripts

Install latest helm

```
https://raw.githubusercontent.com/helm/helm/main/scripts/get-helm-3 | bash
```

Login to Helm Registry

```
helm registry login ncydacrinprogress.azurecr.io --username ncydacrinprogress --password xxx
```

Push Chart

```
helm push ssh-server-0.1.0.tgz oci://ncydacrinprogress.azurecr.io/charts
```

List artifacts in Azure Container Registry

```
az acr login --name ncydacrinprogress.azurecr.io -u ncydacrinprogress -p xxxx

az acr repository list --name ncydacrinprogress.azurecr.io -u ncydacrinprogress -p xxxx -o tsv
```

Output

```
acm/agent
acm/block-provider
acm/frontend
acm/maria-db
acm/mockserver
acm/server
atlassian/jira-software
azuremonitor/containerinsights/ciprod
bats/bats
bitnami/nginx
...
```

Using ORAS

https://oras.land

Install

```
brew install oras
```

Login

```
oras login ${OCI_REPO} --username ${OCI_REPO_USERNAME} --password ${OCI_REPO_PASSWORD}
```

List Artifacts

```
oras repo ls ${OCI_REPO}/
```

Get Latest Tag

```
oras repo tags ${OCI_REPO}/${artifact}  | tail -n 1
```

References

Reference	URL
Cosign Quickstart	https://docs.sigstore.dev/signing/quickstart/
Prove the Authenticity of OCI Artifacts	https://fluxcd.io/blog/2022/10/prove-the-authenticity-of-oci-artifacts/
Signing and Verifying OCI Artifacts	https://fluxcd.io/flux/cheatsheets/oci-artifacts/#signing-and-verification
Security: Image Provenance	https://fluxcd.io/blog/2022/02/security-image-provenance/
Sign and Verify Container Images with Docker, Cosign, and Kyverno: A Complete Guide	https://medium.com/@seifeddinerajhi/sign-and-verify-container-images-with-cosign-and-kyverno-a-complete-guide-b32b1f6e6264
Sigstore	https://kyverno.io/docs/writing-policies/verify-images/sigstore/