Container Image Caching Service at the UChicago AF

Fengping Hu on behalf of the UChicago team
Feb 15 2023

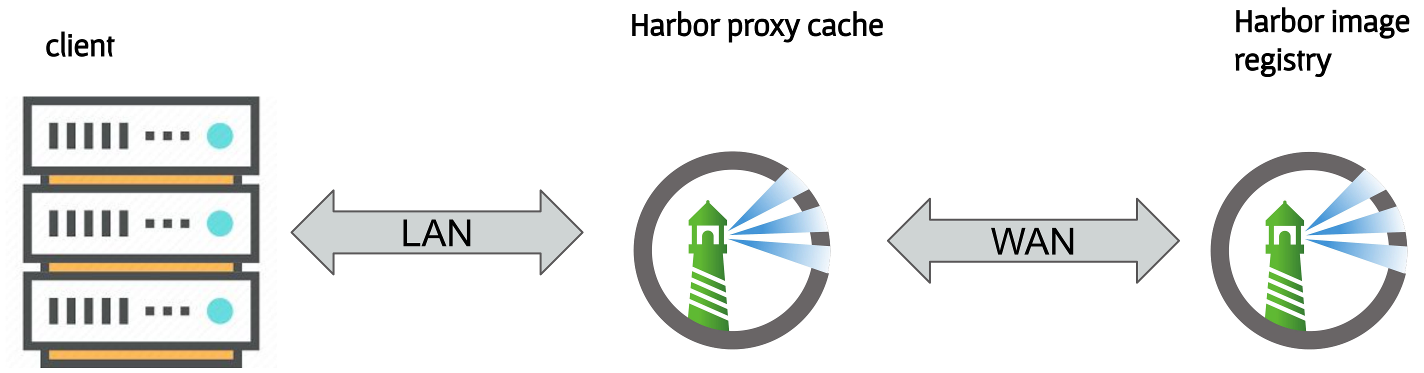
Caching Services at UChicago AF

- Data caching– xcache
- Software distribution caching – cvmfs/squid,varnish
- **Container image caching**
  - **Harbor proxy cache**
  - **Kubernetes policy engine**

Harbor Proxy Cache

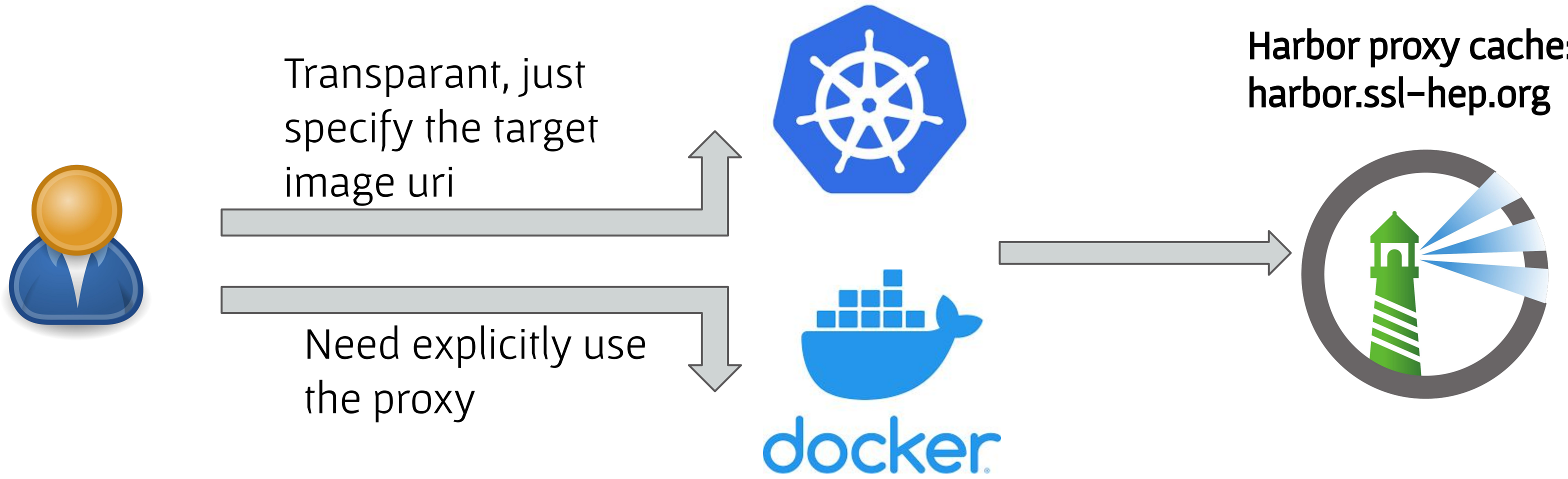
- Cache images from a target public or private registry
- Benefits
    - Enable environment with limited or no access to the internet.
    - Limit the amount of requests made to a public registry, avoiding consuming too much bandwidth or being throttled by the registry server
    - Speed up image pull. Make image pull consistent and less susceptible to WAN congestion.

How Harbor Proxy Cache Works



- If the image has not been updated in the target image registry, the cached image is served from the proxy cache
- If the image has been updated in the target registry, the new image is pulled from the target registry, then served and cached in the proxy cache
- If the target registry is not reachable, the proxy cache serves the cached image.
- If the image is no longer in the target registry, no image is served.

How to Use the Proxy Cache


Transparant, just specify the target image uri
Need explicitly use the proxy
docker
Harbor proxy cache:
harbor.ssl-hep.org

- Kubernetes policy engine enables Kubernetes native workload to use the cache service transparently
- Manually use the proxy
    - docker pull harbor.ssl-hep.org/cernharborproxy/atlas/athena **vs** docker pull registry.cern.ch/atlas/athena

Kubernetes Policy Engine

- Two options
  - OPA Gatekeeper
  - Kyverno(what we pick for now)
- What they do:
  - validate, mutate, generate, or cleanup (remove) any resource
  - verify container images for software supply chain security
  - inspect image metadata
  - ….
- In proxy cache case
  - We use it to replace image registry (filter out private access because the proxy can't pass along authention)

hub.opensciencegrid.org/usatlas/cc-ubuntu:2022.11.16

→

harbor.ssl-hep.org/osgharborproxy/usatlas/cc-ubuntu:2022.11.16

registry.cern.ch/atlas/athena

→

harbor.ssl-hep.org/cernharborproxy/atlas/athena

Gitops Compatibility

- **Flux** (https://fluxcd.io/) – a Continuous Delivery tool to help keep Kubernetes clusters in sync with configuration sources such as Git repositories and automate configuration updates when available
- **Kyverno** (https://kyverno.io/) – a Kubernetes policy engine that we use to modify pod objects (image registry) to use the proxy cache.
- Will there be a conflict – luckily not: Flux detects changes by looking at the dry-run result and comparing it with the cluster state. – which means if the mutation supports dry-run, it will be ok
- Policy failurepolicy – defines the API server behavior if the webhook fails to respond, need to set it to Fail rather than ignore

Registry Types Supported

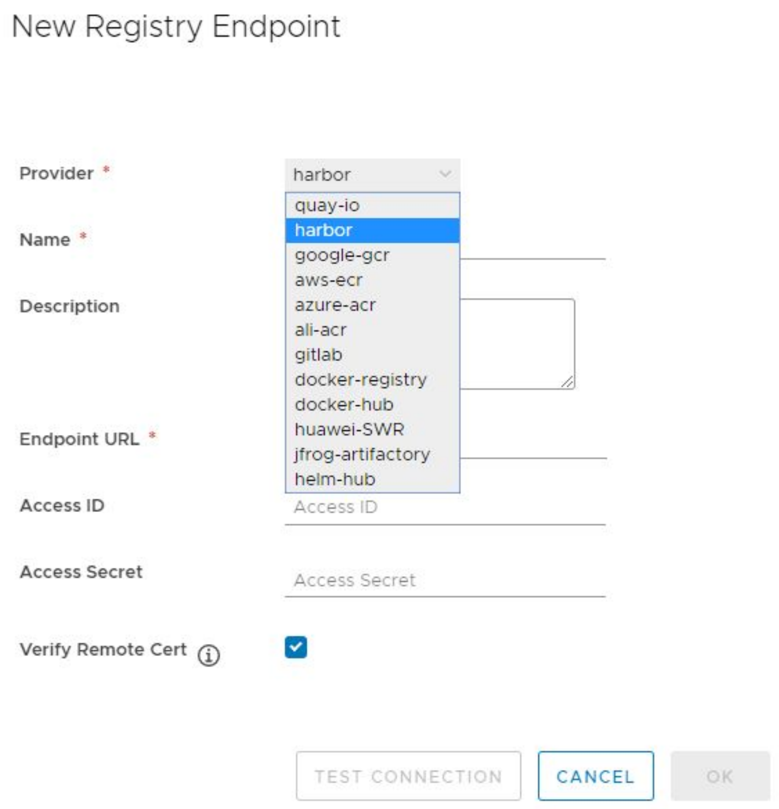
- Harbor (OSG and CERN registries)
- Docker Hub
- Docker registry
- AWS Elastic Container Registry
- Azure Container Registry
- Google Container Registry
- Quay

Deployment

- Deployment is done via Flux
- Both Harbor and Kyverno have Helm charts available
- Harbor service currently deployed on the IRIS-HEP SSL (River cluster)
    - Pgo
    - Ceph object store
- Kubernetes policy engine (Kyverno) is deployed on the Kubernetes cluster where the image policy is needed (in our case the UC Analysis Facility)

Setup Steps – 1

- Creating registry endpoint

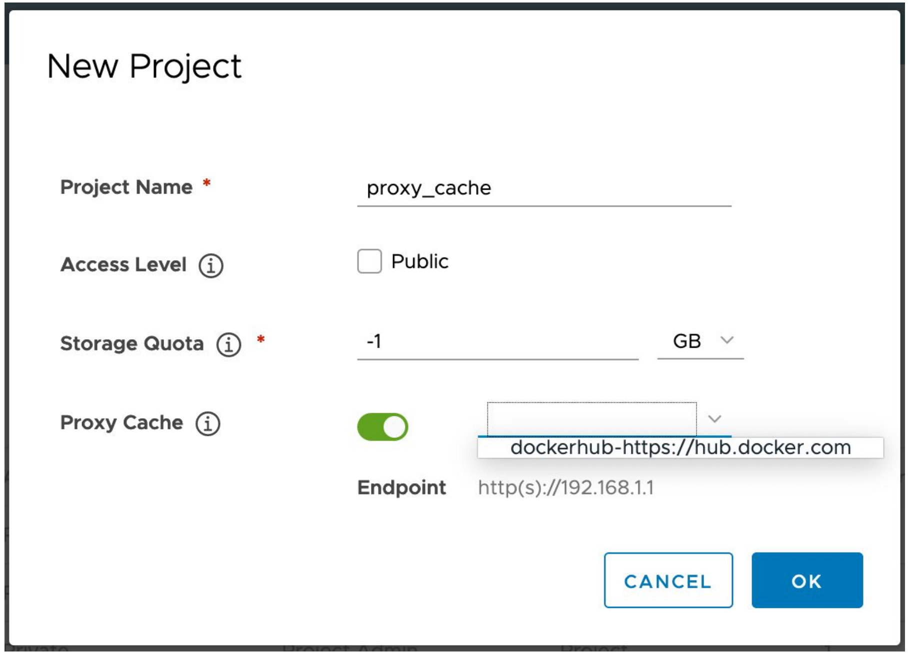

New Registry Endpoint

Provider *	harbor
Name *	
Description	
Endpoint URL *	
Access ID	Access ID
Access Secret	Access Secret
Verify Remote Cert	✓

quay-io
harbor
google-gcr
aws-ecr
azure-acr
ali-acr
gitlab
docker-registry
docker-hub
huawei-SWR
jfrog-artifactory
helm-hub

Setup Steps – 2

- Create proxy cache project – A project in Harbor contains all repositories of an application
    - Same features available to a normal Harbor project, except that you are not able to push images to a proxy cache project



New Project

Project Name *	proxy_cache
Access Level	☐ Public
Storage Quota *	-1 GB
Proxy Cache	✓ dockerhub-https://hub.docker.com
Endpoint	http(s)://192.168.1.1

Setup Steps – 3 Create Registry Replacement Policy

```
kind: ClusterPolicy
metadata:
  name: replace-image-registry
  annotations:
    policies.kyverno.io/title: Replace Image Registry
    policies.kyverno.io/description: >-
      This policy mutates all images to use the proxy
cache service
spec:
  background: false
  failurePolicy: Ignore
  rules:
    - name: replace-image-registry-pod-containers
      match:
        any:
        - resources:
            kinds:
            - Pod
      mutate:
        foreach:
        - list: "request.object.spec.containers"
          patchStrategicMerge:
            spec:
              containers:
              - name: "{{ element.name }}"
                image: "{{
regex_replace_all_literal('hub.opensciencegrid.org',
'{{element.image}}',
'harbor.ssl-hep.org/osgharborproxy' )}}"
      preconditions:
        all:
        - key: '{{
request.object.spec.imagePullSecrets[] || `[]` |
length(@) }}'
          operator: Equals
          value: 0
        - key: "{{
request.object.spec.containers[].image | join(',',@) |
contains(@,'hub.opensciencegrid.org')}}"
          operator: Equals
          value: true
```

Performance Comparison

```
time docker pull
harbor.ssl-hep.org/cernharborpr
oxy/atlas/athena@sha256:9515f
228ea1763f96d190c3c73a347f6
8

4191a69bc89a15e47072728c93
8f2a4

5fb95acba89b: Pull complete

f36f852d5b24: Pull complete

…

real    1m0.295s

user    0m0.215s

sys     0m0.143s
```

```
time docker pull
registry.cern.ch/atlas/athena@s
ha256:9515f228ea1763f96d190
c3c73a347f684191a69bc89a15e
47072728c938f2a4

5fb95acba89b: Pull complete

f36f852d5b24: Pull complete

…

real    4m15.889s

user    0m0.599s

sys     0m0.448s
```

Shown here is just the download time.
Extract time is excluded.

Current Status on the UC AF

- Configured for OSG and CERN Harbor registries
- HTCondor
- JupyterLab
- Coffea-Casa
    - Some parts uses Docker directly, so this needed explicit configurations
- ServiceX
    - Not using the caching yet because the images are on Docker hub