# Working with Containers

Private Registries

## What is a Registry?

- Stateless, server side application that stores and distributes container images
  - Provides control over image storage
  - Provides version control of container images through “tags” assigned to a container image
  - Integrates container image distribution into development workflow

## Examples

- Public Registry
  - Quay.io
  - Registry.redhat.io
  - Docker.io/Docker Hub
- Private Registry
  - Google Container Registry
  - Amazon Elastic Container Registry
  - Red Hat Quay
  - Azure Container Registry
  - Docker Registry

# Why a Private Registry?

- Greater Security
  - Access Control/RBAC
  - Vulnerability Scanning
  - Audit logging
  - Limit connections to public registries
- Privacy
- Locally hosted or remotely (Datacenter, Cloud, etc)
- Full Control of image distribution pipeline
- Mirror public registries

# Configuring a Simple Private Registry

- Requirements
  - RHEL 7
    - Docker Distrubition RPM (Available from base repo)
    - Docker Registry Container from Docker Hub (Optional)
    - Podman or Docker installed
    - Access to CDN or Satellite with rhel7 synced
  - RHEL 8
    - Docker Registry Container from Docker Hub
    - Podman (Comes with the operating system.  Select container-tools during install to installed podman)

# Docker Distribution Method

- Install rpm
  - yum –y install docker-distrubition
- Edit Config to add hostname or ip to addr block in http:
  - vi /etc/docker-distribution/registry/config.yml
  - addr: <ip-or-hostname>:port
- Enable Service
  - systemctl enable docker-distribution
- Optional:
  - Create certificate directories (Optional if using https for registry)
  - Generate certificate request and place in directory
  - Configure "secure mode" for registry startup in /etc/docker-distribution/registry/config.yml
- Start the registry
  - systemctl start docker-distribution
- Open port 5000 is open in firewalld

# Docker Registry Container Method

- Create folders for registry
  - mkdir –p /opt/registry/data
- Use the registry container from docker.io to start registry as a container
  - Podman run --name registry -p 5000:5000 -v /opt/registry/data:/var/lib/registry:z -e REGISTRY_COMPATIBILITY_SCHEMA1_ENABLED=true -d docker.io/library/registry:latest
- Open port 5000 in firewalld on localhost
- Optional:
  - Configure https with certs in directory /opt/registry/certs
  - Configure authentication of registry with httpd-tools in /opt/registry/auth
  - Run container with extra environment statements to mount certs and auth in the container and specify authentication realm

# Red Hat Quay Registry

- Proof-of-concept deployment (Non Production)
- Requirements
  - RHEL8
  - podman or container-tools (Installed with OS if container-tools selected)
  - Access to registry.redhat.io
  - Minimum 2 CPU
  - Minimum 4GB RAM
  - Minimum 30GB Disk
  - Containers:
    - postgresql-10:1
    - redis-5:1
    - quay-rhel8:v3.5.1

# Using the Registry

- Use --tls-verify=false to ignore https when interacting with registries if accessing as insecure
- Login to public registry if required
  - podman login <hostname:port>
- Pull/Obtain image from source
  - podman pull <hostname:port>/path/to/container:tag
- Tag Image
  - podman tag <hostname:port>/path/to/container:tag> <newlocation:port>/path/to/container:tag>
- Push image
  - podman push <newlocation:port>/path/to/container:tag>

# Links

- http://redhatgov.io/workshops/containers_101/exercise1.4/
- https://www.redhat.com/sysadmin/simple-container-registry
- https://access.redhat.com/documentation/en-us/red_hat_quay/3.5/html/deploy_red_hat_quay_for_proof-of-concept_non-production_purposes/index