Contents

Setup Docker Registry2

https://docs.docker.com/registry/deploying/
https://docs.docker.com/registry/spec/api/
https://www.exoscale.com/syslog/setup-private-docker-registry/

A registry can be considered private if pulling requires authentication

Docker command:
docker run -d -p 80:5000 --restart=always -e REGISTRY_STORAGE_DELETE_ENABLED=true --name registry2 registry:2

Using volumes:
docker run -d -p 80:5000 --restart=always --name registry2 -v temp-docker-repo:/var/lib/registry registry:2

Compare daemon.json
Docker expects a secured channel by default, and that's naturally a very good thing.
Configuring Docker to accept connections to unsecure registries depends on your OS, but it's quite straightforward.

In order to push to insecure registries, we need to edit **daemon.json** file at **/etc/docker/daemon.json** and add the below content.
Once edited, restart your docker daemon (**systemctl restart docker**)

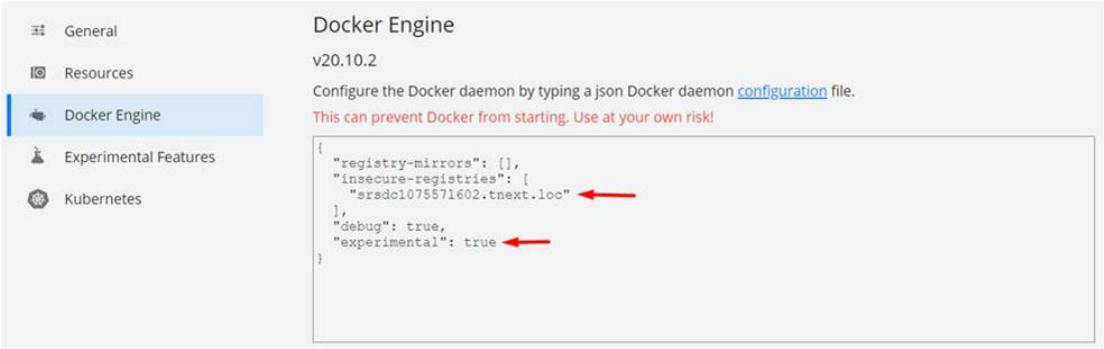
```
{
"insecure-registries" : ["Repo-URL"]
}
```

In Linux, Edit **/etc/docker/daemon.json** and add "insecure-registries" : ["srsdc107557l602.tnext.loc"]

```
{
    "tls": true,
    "tlscacert": "/root/.docker/ca.pem",
    "tlscert": "/root/.docker/cert.pem",
    "tlskey": "/root/.docker/key.pem",
    "tlsverify": true,
    "insecure-registries" : ["srsdc107557l602.tnext.loc"]
}
```

On macOS you do it using the user interface, and the changes will automatically restart the daemon:

- Click on the Docker icon
- Select Preferences… in the menu
- Select the Daemon tab
- Check the checkbox named Experimental features
- In the first list box, enter the address (URL or IP) of the unsecure registry e.g. 127.0.0.1:5000
- Wait a bit for the Docker daemon to restart, then push again to the registry. This time, it should be a success:



Retag image
docker image tag <image-name> <repo-url>/<image-name>
docker image tag nginx:1.0 13.125.88.177:5000/nginx:1.0

Login to repo if needed (Not needed for registry2)
docker login -u <user-name> -p <password> <Repo-URL>

Push the image
docker push <repo-url>/<image-name>
docker push 13.125.88.177:5000/nginx:1.0

APIs for registry2 container:

- Get list of images: curl -X GET http://localhost:5000/v2/_catalog
- Get list of image tags: curl -X GET http://localhost:5000/v2/<image-name>/tags/list
- curl -X GET http://15.206.89.202:80/v2/nginx/tags/list
- Delete an image tag:
  First Get SHA ID for that specific tag
  curl -v --silent -H "Accept: application/vnd.docker.distribution.manifest.v2+json" -X GET http://localhost:5000/v2/<image-name>/manifests/latest 2>&1 | grep Docker-Content-Digest | awk '{print($3)}'

  Use the SHA ID to delete the image tag
  curl -v --silent -H "Accept: application/vnd.docker.distribution.manifest.v2+json" -X DELETE http://127.0.0.1:5000/v2/my-ubuntu/manifests/sha256:f2557f94cac1cc4509d0483cb6e302da841ecd6f82eb2e91dc7ba6cfd0c580ab
- If you get delete errors, then enabled deletion environment variable while starting the container
  docker run -d -p 5000:5000 --restart=always -v $PWD:/var/lib/registry -e REGISTRY_STORAGE_DELETE_ENABLED=true --name registry2 registry:2

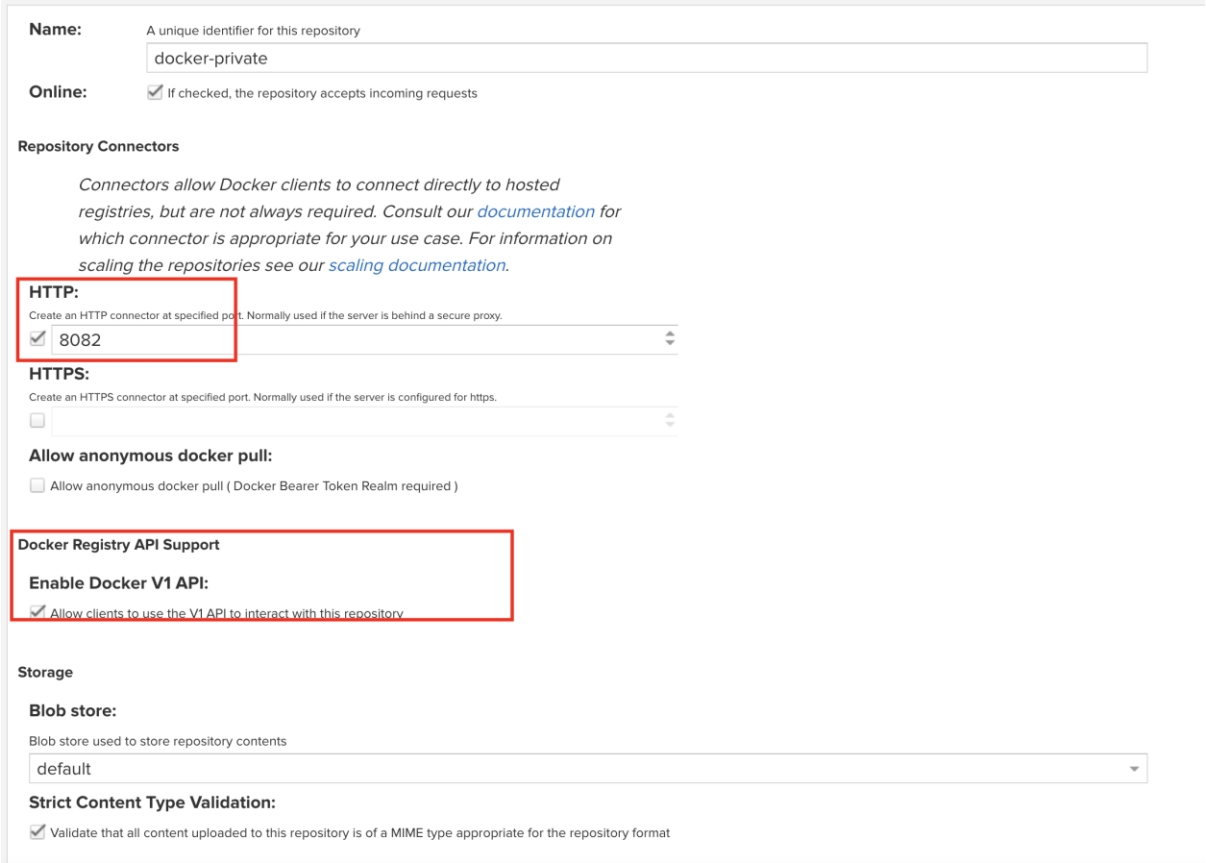
  Expected error:
  [root@ip-172-31-7-249 ec2-user]# curl -v --silent -H "Accept: application/vnd.docker.distribution.manifest.v2+json" -X DELETE http://15.206.89.202/v2/nginx/manifests/sha256:5e95e5eb8be4322e3b3652d737371705e56809ed8b307ad68ec59ddebaaf60e4
  * Trying 15.206.89.202:80...
  * Connected to 15.206.89.202 (15.206.89.202) port 80 (#0)
  > DELETE /v2/nginx/manifests/sha256:5e95e5eb8be4322e3b3652d737371705e56809ed8b307ad68ec59ddebaaf60e4 HTTP/1.1
  > Host: 15.206.89.202
  > User-Agent: curl/7.76.1
  > Accept: application/vnd.docker.distribution.manifest.v2+json
  >
  * Mark bundle as not supporting multiuse
  < HTTP/1.1 405 Method Not Allowed
  < Content-Type: application/json; charset=utf-8
  < Docker-Distribution-Api-Version: registry/2.0
  < X-Content-Type-Options: nosniff
  < Date: Wed, 18 Aug 2021 21:11:48 GMT
  < Content-Length: 78
  <
  {"errors":[{"code":"UNSUPPORTED","message":"The operation is unsupported."}]}
  * Connection #0 to host 15.206.89.202 left intact

Setup Docker Nexus Repository

https://medium.com/hackernoon/deploy-private-docker-registry-on-gcp-with-nexus-terraform-and-packer-1af2b6a2a9c9

https://dzone.com/articles/how-to-publish-docker-images-on-private-nexus-repo

- Run the Nexus container
  docker run -d -p 8081:8081 -p 8082:8082 --name nexus sonatype/nexus3
  It can take some time (2-3 minutes) for the service to launch in a new container. You can tail the log to determine once Nexus is ready
- docker ps -a
- docker logs -f nexus
- Access UI at: http://<ec2-public-ip>:8081 and login
- Default credentials are: **admin** and password file at **/nexus-data/admin.password** after docker exec -it nexus bash
- Go to Repositories and Create a docker hosted repo. Give a http port(like 8082)



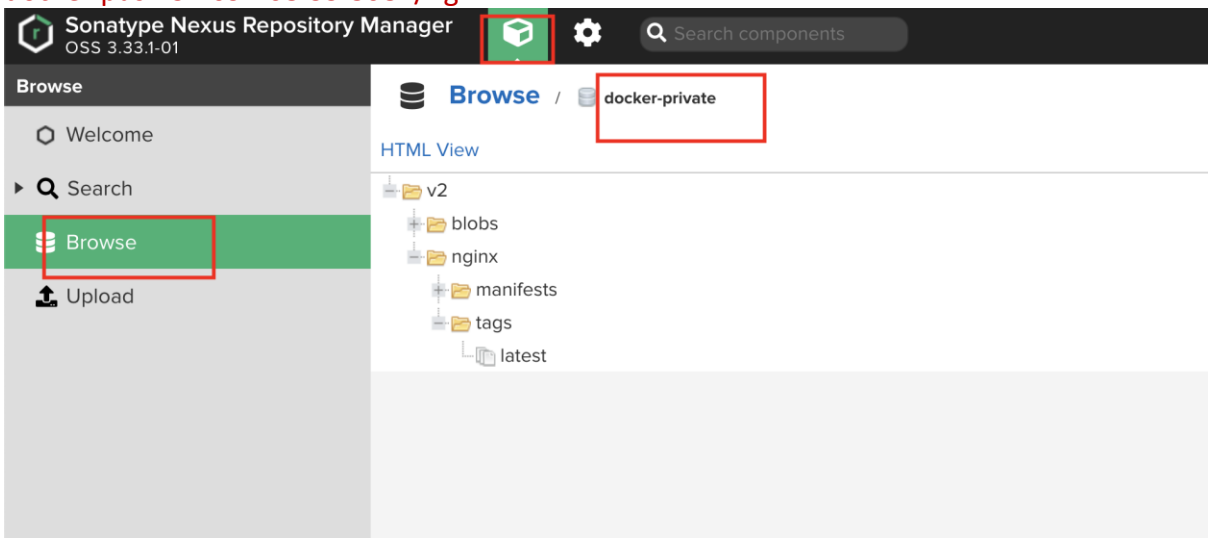
- Retag local images with **<ec2-public-ip>:8082/<image-name>** in order to push it to the Nexus repository
- Update **daemon.json** to allow insecure registries
  root@ip-172-31-4-142 ec2-user]# cat /etc/docker/daemon.json
  {

  "insecure-registries" : ["3.109.108.33:8082"]
  }
- systemctl restart docker

- docker login -u admin -p admin123 3.109.108.33:8082
  [root@ip-172-31-4-142 ec2-user]# docker login -u admin -p admin123 3.109.108.33:8082/
  WARNING! Using --password via the CLI is insecure. Use --password-stdin.
  WARNING! Your password will be stored unencrypted in /root/.docker/config.json.
  Configure a credential helper to remove this warning. See
  https://docs.docker.com/engine/reference/commandline/login/#credentials-store

  Login Succeeded
  [root@ip-172-31-4-142 ec2-user]#

- docker push 3.109.108.33:8082/nginx



Setup ECR Repository

https://docs.aws.amazon.com/cli/latest/userguide/
https://docs.aws.amazon.com/cli/latest/userguide/welcome-examples.html

https://aws.amazon.com/ecr/
https://aws.amazon.com/ecr/resources/
https://docs.aws.amazon.com/AmazonECR/latest/userguide/what-is-ecr.html
https://docs.aws.amazon.com/AmazonECR/latest/userguide/get-set-up-for-amazon-ecr.html
https://docs.aws.amazon.com/AmazonECR/latest/userguide/getting-started-cli.html

Setup CLI

- Download the CLI, AMI linux comes by default
  https://docs.aws.amazon.com/cli/latest/userguide/
- Configure CLI
  aws configure
  https://docs.aws.amazon.com/cli/latest/userguide/welcome-examples.html

```
vikram@vikram-digital-twin MINGW64 /c/Program Files/Terminus
$ aws configure
AWS Access Key ID [None]: [redacted]B4C3WDUF
AWS Secret Access Key [None]: [redacted]
Default region name [None]: ap-south-1
Default output format [None]:
```

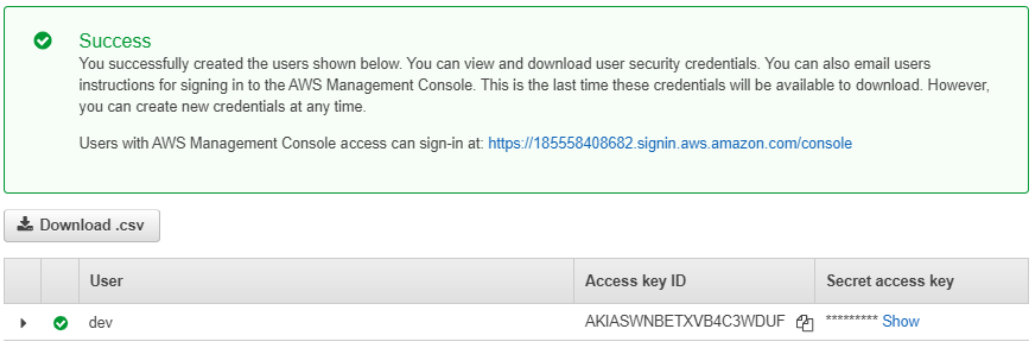
```
[root@ip-172-31-5-247 ec2-user]# ls -al ~/
total 20
dr-xr-x---  4 root root 115 Aug 22 16:45 .
dr-xr-xr-x 18 root root 257 Aug 22 16:30 ..
drwxr-xr-x  2 root root  39 Aug 22 16:45 .aws
-rw-r--r--  1 root root  18 Oct 18  2017 .bash_logout
-rw-r--r--  1 root root 176 Oct 18  2017 .bash_profile
-rw-r--r--  1 root root 176 Oct 18  2017 .bashrc
-rw-r--r--  1 root root 100 Oct 18  2017 .cshrc
drwx------  2 root root  29 Aug 22 16:30 .ssh
-rw-r--r--  1 root root 129 Oct 18  2017 .tcshrc
[root@ip-172-31-5-247 ec2-user]# ls -al ~/.aws/
total 8
drwxr-xr-x 2 root root  39 Aug 22 16:45 .
dr-xr-x--- 4 root root 115 Aug 22 16:45 ..
-rw------- 1 root root  30 Aug 22 16:45 config
-rw------- 1 root root 116 Aug 22 16:45 credentials
```

- aws --version

```
vikram@vikram-digital-twin MINGW64 /c/Program Files/Terminus
$ aws --version
aws-cli/2.2.31 Python/3.8.8 Windows/10 exe/AMD64 prompt/off
```

We cannot use our root account credentials to configure aws cli due to security considerations. So preferred way is to create IAM user and give it necessary credentials to programmatically access AWS resources. If this user's details are compromised, we can simply delete the user or revoke his access key

- **Create an IAM user:** Create an IAM user with programmatic access, and then add the user to an IAM group with administrative permissions or grant this user administrative permissions/service specific permissions like AdministrativeAccess/ AmazonEC2FullAccess etc by attaching the existing policies.
  Once done, download the access keys for the user
  You can then access AWS using a special URL and the credentials for the IAM user. Sign in URL looks like
  https://your_aws_account_id.signin.aws.amazon.com/console/



Policies > AdministratorAccess

Summary

Policy ARN	arn:aws:iam::aws:policy/AdministratorAccess
Description	Provides full access to AWS services and resources.

Permissions | Policy usage | Policy versions | Access Advisor

Policy summary | { } JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "*",
            "Resource": "*"
        }
    ]
}
```

- **Create repo using CLI**. The same can be done from the UI itself

```
aws ecr create-repository \
  --repository-name nginx \
  --image-scanning-configuration scanOnPush=true \
  --region ap-south-1
```

```
aws ecr create-repository \
    --repository-name hello-world \
    --image-scanning-configuration scanOnPush=true \
    --region us-east-1
```

```
[root@ip-172-31-5-247 ec2-user]# aws ecr create-repository \
>     --repository-name nginx \
>     --image-scanning-configuration scanOnPush=true \
>     --region ap-south-1
{
    "repository": {
        "repositoryUri": "185558408682.dkr.ecr.ap-south-1.amazonaws.com/nginx",
        "imageScanningConfiguration": {
            "scanOnPush": true
        },
        "encryptionConfiguration": {
            "encryptionType": "AES256"
        },
        "registryId": "185558408682",
        "imageTagMutability": "MUTABLE",
        "repositoryArn": "arn:aws:ecr:ap-south-1:185558408682:repository/nginx",
        "repositoryName": "nginx",
        "createdAt": 1629651210.0
    }
}
```

ECR > Repositories

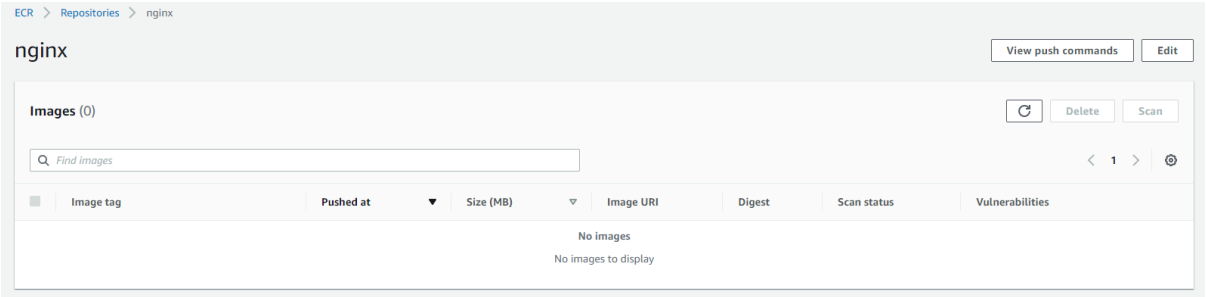
Private | Public

Private repositories (1)

View push commands | Delete | Edit | Create repository

Find repositories

Repository name	URI	Created at	Tag immutability	Scan on push	Encryption type
nginx	185558408682.dkr.ecr.ap-south-1.amazonaws.com/nginx	August 22, 2021, 22:23:30 (UTC+05.5)	Disabled	Enabled	AES-256



- **Install docker if needed**
  yum update -y && yum install docker -y && systemctl enable --now docker && docker pull nginx
- **Tag the image with repo url**
  docker tag nginx:latest **aws_account_id.dkr.ecr.us-east-1.amazonaws.com/nginx:latest**
  docker tag nginx:latest **185558408682.dkr.ecr.ap-south-1.amazonaws.com/nginx:latest**
- **Authenticate to the registry (Docker login)**
  After you have installed and configured the AWS CLI, authenticate the Docker CLI to your default registry. That way, the docker command can push and pull images with Amazon ECR. The AWS CLI provides a get-login-password command to simplify the authentication process.

  The get-login-password is the preferred method for authenticating to an Amazon ECR private registry when using the AWS CLI. Ensure that you have configured your AWS CLI to interact with AWS

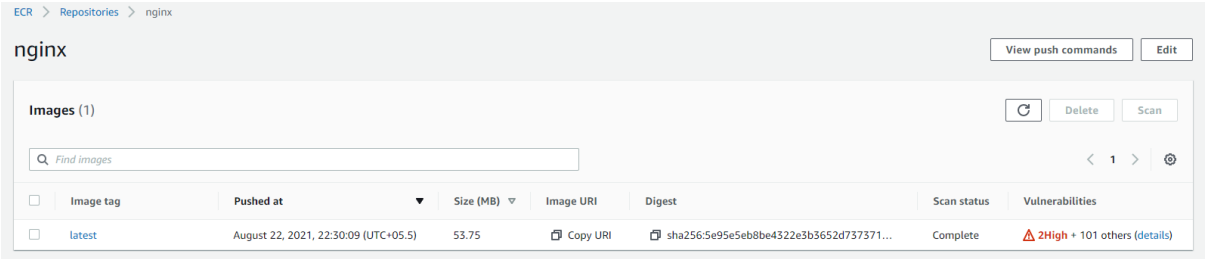
  aws ecr get-login-password --region region | docker login --username AWS --password-stdin aws_account_id.dkr.ecr.region.amazonaws.com

  aws ecr get-login-password --region ap-south-1 | docker login --username AWS --password-stdin **185558408682.dkr.ecr.ap-south-1.amazonaws.com**

```
[root@ip-172-31-5-247 ec2-user]# aws ecr get-login-password --region ap-south-1 | docker login --username AWS --password-stdin 185558408682.dkr.ecr
.ap-south-1.amazonaws.com
WARNING! Your password will be stored unencrypted in /root/.docker/config.json.
Configure a credential helper to remove this warning. See
https://docs.docker.com/engine/reference/commandline/login/#credentials-store
```

- **Push the image**
  docker push 185558408682.dkr.ecr.ap-south-1.amazonaws.com/nginx

```
[root@ip-172-31-5-247 ec2-user]# docker push 185558408682.dkr.ecr.ap-south-1.amazonaws.com/nginx
Using default tag: latest
The push refers to repository [185558408682.dkr.ecr.ap-south-1.amazonaws.com/nginx]
fb04ab8effa8: Pushed
8f736d52032f: Pushed
009f1d338b57: Pushed
678bbd796838: Pushed
d1279c519351: Pushed
f68ef921efae: Pushed
latest: digest: sha256:5e95e5eb8be4322e3b3652d737371705e56809ed8b307ad68ec59ddebaaf60e4 size: 1570
[root@ip-172-31-5-247 ec2-user]#
```



- For pull/delete, refer https://docs.aws.amazon.com/AmazonECR/latest/userguide/getting-started-cli.html