**Project-5 | Private Docker Registry In Linux With SSL**

**Implementing a Private Docker Registry on Debian/Ubuntu**

A **Private Docker Registry** allows you to store and manage container images privately instead of using Docker Hub. This is useful for **security, performance, and better control** over your images in a DevOps workflow.

**📌 Overview**

* **Operating System:** Debian/Ubuntu
* **Registry Access:** Private (secured with authentication)
* **Storage Location:** Local or Cloud
* **Security:** Secured with TLS (SSL certificate)

**🔹 Step 1: Install Docker and Dependencies**

**1.1 Update Your System**

Before installing Docker, it's best to update your package list and upgrade existing packages:

sudo apt update && sudo apt upgrade -y

* sudo apt update: Updates the list of available packages from repositories.
* sudo apt upgrade -y: Upgrades all installed packages to their latest versions.

**1.2 Install Docker and Docker Compose**

sudo apt install -y docker.io

sudo systemctl enable --now docker

* sudo apt install -y docker.io: Installs Docker.
* sudo systemctl enable --now docker: Enables Docker to start on boot and starts it immediately.

**1.3 Verify Docker Installation**

docker --version

This command confirms that Docker has been installed correctly.

sudo usermod -aG docker $USER

newgrp docker

**🔹 Step 2: Run a Local Docker Registry**

Docker provides an official registry image that allows you to host your own private registry.

**2.1 Start the Registry**

docker run -d -p 5000:5000 --name registry --restart always registry:2

* docker run -d: Runs the container in detached mode (background).
* -p 5000:5000: Maps port 5000 on the host to port 5000 inside the container.
* --name registry: Assigns the container the name "registry."
* --restart always: Ensures the container restarts automatically if it stops.
* registry:2: Uses the official Docker registry version 2 image.

**2.2 Verify the Registry is Running**

curl http://localhost:5000/v2/

* This checks if the registry is running by querying its API.
* If successful, it will return {}.

**🔹 Step 3: Push and Pull an Image to the Private Registry**

**3.1 Pull a Sample Image**

docker pull ubuntu

* This downloads the latest Ubuntu image from Docker Hub.

**3.2 Tag the Image for the Private Registry**

docker tag ubuntu localhost:5000/ubuntu

* This renames (tags) the image to associate it with our private registry.

**3.3 Push the Image to the Registry**

docker push localhost:5000/ubuntu

* Uploads the image to our private registry.

**3.4 Check Available Images in Registry**

curl http://localhost:5000/v2/\_catalog

* Lists all images stored in the registry.
* Expected output: {"repositories":["ubuntu"]}.

**3.5 Pull the Image from the Registry**

To simulate pulling the image on another machine or after removing it:

docker rmi ubuntu

docker pull localhost:5000/ubuntu

* docker rmi ubuntu: Removes the local image.
* docker pull localhost:5000/ubuntu: Pulls the image back from the private registry.

**🔹 Step 4: Secure the Registry with Authentication**

By default, anyone can access the registry, so we need authentication.

**🛠 Step 4.1: Create Authentication Credentials**

**Create a Directory for Authentication Files**

sudo mkdir -p /etc/docker/registry

sudo chmod 777 /etc/docker/registry

* mkdir -p /etc/docker/registry: Creates a directory to store credentials.
* sudo chmod 777: Grants full read/write access

**Install Apache Utilities (htpasswd)**

sudo apt update

sudo apt install -y apache2-utils

* Installs htpasswd, which is used to create username/password authentication.

**Generate Credentials**

htpasswd -Bbn gur q123 > /etc/docker/registry/htpasswd

* -B: Uses bcrypt for password encryption.
* -n: Prints credentials instead of saving them.
* myuser mypassword: Replace with your username and strong password.
* > /etc/docker/registry/htpasswd: Saves credentials in a file.

**Step 4.2: Run Registry with Authentication**

**Stop the Current Registry**

docker stop registry && docker rm registry

* Stops and removes the existing registry container.

**Run a New Instance with Authentication**

docker run -d -p 5000:5000 --name registry --restart always \

-v /etc/docker/registry:/auth \

-e "REGISTRY\_AUTH=htpasswd" \

-e "REGISTRY\_AUTH\_HTPASSWD\_REALM=Registry Realm" \

-e "REGISTRY\_AUTH\_HTPASSWD\_PATH=/auth/htpasswd" \

registry:2

* Mounts the authentication file (-v /etc/docker/registry:/auth).
* Enables authentication (REGISTRY\_AUTH=htpasswd).

**Login to the Private Registry**

docker login localhost:5000

* Prompts for username and password.

**🔹 Step 5: Secure the Registry with SSL/TLS**

TLS encrypts traffic between clients and the registry.

# add your ipaddress in your dns like godaddy.com

# <https://www.whatsmydns.net/>

Check your dns name is proper working

**🛠 Step 5.1: Install Certbot for SSL Certificates**

sudo apt install -y certbot

* Installs Certbot to generate free SSL certificates.

**🛠 Step 5.2: Generate an SSL Certificate**

sudo certbot certonly --standalone -d gur.stackdev.live

* certonly: Only generates the certificate (does not modify system configuration).
* --standalone: Uses its own web server.
* -d gur.stackdev.live: Replace with your domain.

Certificates are stored in:

/etc/letsencrypt/live/gur.stackdev.live/

**🛠 Step 5.3: Run Registry with SSL**

**Stop the Running Registry**

docker stop registry && docker rm registry

**Run the Registry with SSL & Authentication**

docker run -d -p 5000:5000 --name registry --restart always \

-v /etc/docker/registry:/auth \

-v /etc/letsencrypt:/certs \

-e "REGISTRY\_AUTH=htpasswd" \

-e "REGISTRY\_AUTH\_HTPASSWD\_REALM=Private Docker Registry" \

-e "REGISTRY\_AUTH\_HTPASSWD\_PATH=/auth/htpasswd" \

-e "REGISTRY\_HTTP\_TLS\_CERTIFICATE=/certs/live/skjptpp.in/fullchain.pem" \

-e "REGISTRY\_HTTP\_TLS\_KEY=/certs/live/skjptpp.in/privkey.pem" \

registry:2

OR

sudo docker run -d \

--restart=always \

--name registry \

-v /etc/letsencrypt/live/gur.stackdev.live/fullchain.pem:/certs/fullchain.pem \

-v /etc/letsencrypt/live/gur.stackdev.live/privkey.pem:/certs/privkey.pem \

-e REGISTRY\_HTTP\_TLS\_CERTIFICATE=/certs/fullchain.pem \

-e REGISTRY\_HTTP\_TLS\_KEY=/certs/privkey.pem \

-p 5000:5000 \

registry:2

* Mounts SSL certificates (-v /etc/letsencrypt:/certs).
* Uses fullchain.pem and privkey.pem for secure communication.

**Test Secure Connection**

sudo chmod -R 755 /etc/letsencrypt/

sudo chmod -R 755 /etc/letsencrypt/live/

sudo chmod -R 644 /etc/letsencrypt/live/gur.stackdev.live /\*

sudo chmod -R 644 /etc/letsencrypt/archive/gur.stackdev.live /\*

sudo chmod 640 /etc/docker/registry/htpasswd

sudo chown root:docker /etc/docker/registry/htpasswd

docker restart registry

**Test Secure Connection**

curl -k -u gur:'q123' https:// gur.stackdev.live:5000/v2/

**Login Securely to the Registry**

docker login gur.stackdev.live:5000

* Uses HTTPS instead of HTTP.

**Pull Images from the Secure Registry**

docker pull alpine

docker tag alpine:latest gur.stackdev.live:5000/alpine:latest

docker push gur.stackdev.live:5000/alpine

docker pull gur.stackdev.live:5000/ubuntu

curl https://gur.stackdev.live:5000/v2/\_catalog

**🔹 Final Checks**

**✅ 1️. Ensure the Registry is Running**

docker ps | grep registry

**✅ 2️. Verify Login Works**

docker login gur.stackdev.live:5000