

Nachfolgend soll die Installation und Konfiguration von Harbor beschrieben werden.

<blockquote>

Harbor Registry ist eine der bekanntesten und am weitesten verbreiteten Open Source-Lösungen für das Speichern und Verwalten von Docker Images. Dabei geht der Funktionsumfang von Harbor deutlich über den Umfang herkömmlicher Registries, wie zum Beispiel Docker Hub hinaus. (Quelle: bnerd.com)

</blockquote>

1 - Vorbereitungen

Es sind mehrere vorbereitende Schritte notwendig.

1.1 - SSL-Zertifikate

Der Harbor bringt selbst keine Zertifikate mit. Diese müssen manuell erstellt oder von Drittanbietern besorgt werden. Eine Methode zum Erstellen eines eigenen Zertifikats ist [hier](#) beschrieben.

1.2 - Docker

Harbor läuft als Docker-Container. Dieses Programm muss als Voraussetzung installiert werden. Docker kann aus den Repositories installiert werden, sofern die Docker-Repositories eingebunden wurden:

```
~# apt install docker-ce docker-ce-cli containerd.io docker-buildx-plugin  
docker-compose-plugin
```

1.3 - Harbor

Harbor kann von der Seite „goharbor.io“ (dort dann oben auf „Getting Started“ klicken und auf dieser Seite dann unter Punkt 2 auf „Download the Harbor Installer“). Die verschiedenen Versionen sind auch auf der Release-Seite „<https://github.com/goharbor/harbor/releases>“ zu finden. Nachfolgend wird der Offline-Installer genutzt. Zum Zeitpunkt der Dokumentation war Version 2.14.1 aktuell.

Schritt 1: Herunterladen

```
~# wget
```

```

https://github.com/goharbor/harbor/releases/download/v2.14.1/harbor-offline-
installer-v2.14.1.tgz
--2025-12-18 22:14:12--
https://github.com/goharbor/harbor/releases/download/v2.14.1/harbor-offline-
installer-v2.14.1.tgz\
Auflösen des Hostnamens github.com (github.com)... 140.82.121.3\
Verbindungsaufbau zu github.com (github.com)|140.82.121.3|:443 ... verbunden.\
HTTP-Anforderung gesendet, auf Antwort wird gewartet ... 302 Found
...
Verbindungsaufbau zu release-assets.githubusercontent.com (release-
assets.githubusercontent.com)|185.199.111.133|:443 ... verbunden.\
HTTP-Anforderung gesendet, auf Antwort wird gewartet ... 200 OK\
Länge: 680961237 (649M) [application/octet-stream]\
Wird in »harbor-offline-installer-v2.14.1.tgz« gespeichert.

harbor-offline-installer-v2.14.1.tgz
100%[=====
==>] 649,42M  18,9MB/s    in 35s
2025-12-18 22:14:47 (18,6 MB/s) - »harbor-offline-installer-v2.14.1.tgz«
gespeichert [680961237/680961237]
~#

```

Schritt 2: Entpacken

```

~# tar xzvf harbor-offline-installer-v2.14.1.tgz
harbor/harbor.v2.14.1.tar.gz
harbor/prepare
harbor/LICENSE
harbor/install.sh
harbor/common.sh
harbor/harbor.yml.tpl
~#

```

1.4 - Zertifikat

Das zum Server erstellte Zertifikat muss für Harbor und Docker bereitgestellt werden.

1.4.1 - Harbor

Für Harbor wird unter „/data“ ein entsprechendes Verzeichnis „cert/“ erstellt, wohin alle benötigten Dateien kopiert werden.

Schritt 1: Erstellen der Verzeichnisstruktur:

```

~# mkdir -pv /opt/harbor/certs
mkdir: Verzeichnis '/opt/harbor/certs' angelegt

```

```
~#
```

Schritt 2: Kopieren der Zertifikatsdateien:

```
~# cp -v slxharbor.fritz.box.crt /opt/harbor/certs/ && cp -v  
slxharbor.fritz.box.key /opt/harbor/certs/  
'slxharbor.fritz.box.crt' -> '/opt/harbor/certs/slxharbor.fritz.box.crt'\  
'slxharbor.fritz.box.key' -> '/opt/harbor/certs/slxharbor.fritz.box.key'\  
~#
```

1.4.2 - Docker

Docker interpretiert „*.crt“-Dateien als CA-Dateien und „*.cert“-Dateien als Server-Zertifikate. Das Server-Zertifikat muss daher einmal umgewandelt werden.

Schritt 1: Konvertieren der Datei:

```
~# openssl x509 -inform PEM -in slxharbor.fritz.box.crt -out  
slxharbor.fritz.box.cert
```

Schritt 2: Erstellen der Verzeichnisstruktur:

```
~# mkdir -pv /etc/docker/certs.d/slxharbor.fritz.box  
mkdir: Verzeichnis '/etc/docker' angelegt\  
mkdir: Verzeichnis '/etc/docker/certs.d' angelegt\  
mkdir: Verzeichnis '/etc/docker/certs.d/slxharbor.fritz.box' angelegt\  
~#
```

Schritt 3: Kopieren der Zertifikate:

```
~# cp -v slxharbor.fritz.box.cert /etc/docker/certs.d/slxharbor.fritz.box/  
&& cp -v slxharbor.fritz.box.key /etc/docker/certs.d/slxharbor.fritz.box/ &&  
cp -v ca.crt /etc/docker/certs.d/slxharbor.fritz.box/\  
'slxharbor.fritz.box.cert' ->  
'/etc/docker/certs.d/slxharbor.fritz.box/slxharbor.fritz.box.cert'\  
'slxharbor.fritz.box.key' ->  
'/etc/docker/certs.d/slxharbor.fritz.box/slxharbor.fritz.box.key'\  
'ca.crt' -> '/etc/docker/certs.d/slxharbor.fritz.box/ca.crt'  
~#
```

Schritt 4: Docker neu starten:

```
~# systemctl restart docker  
~#
```

2 - YAML-Datei

Vor der eigentlichen Installation muss die mitgelieferte YAML-Datei noch bearbeitet werden. Dazu muss die Vorlage „harbor.yml.tpl“ nach „harbor.yml“ kopiert werden:

```
~# cp -v harbor.yml.tpl harbor.yml\  
'harbor.yml.tpl' -> 'harbor.yml'  
~#
```

Jetzt kann die Datei angepasst werden. Folgende Parameter wurden neu gesetzt:

```
hostname: slxharbor.fritz.box  
...  
https:  
  certificate: /opt/harbor/certs/slxharbor.fritz.box.crt\  
  private_key: /opt/harbor/certs/slxharbor.fritz.box.key\  
...  
harbor_admin_password: *****  
...  
database:  
  password: *****  
...  
data_volume: /data/harbor
```

Hinweis: Die gesetzten Kennwörter sollten irgendwo sicher notiert werden (zum Beispiel in Keepass).

3 - Installation

Jetzt kann das Installationsskript ausgeführt werden:

```
~# ./install.sh  
[Step 0]: checking if docker is installed ...  
  
Note: docker version: 29.1.3  
  
[Step 1]: checking docker-compose is installed ...  
  
Note: Docker Compose version v5.0.0  
  
[Step 2]: loading Harbor images ...  
Loaded image: goharbor/trivy-adapter-photon:v2.14.1  
Loaded image: goharbor/harbor-exporter:v2.14.1  
Loaded image: goharbor/harbor-registryctl:v2.14.1  
Loaded image: goharbor/prepare:v2.14.1  
Loaded image: goharbor/harbor-log:v2.14.1  
Loaded image: goharbor/harbor-jobservice:v2.14.1
```

```
Loaded image: goharbor/redis-photon:v2.14.1
Loaded image: goharbor/nginx-photon:v2.14.1
Loaded image: goharbor/registry-photon:v2.14.1
Loaded image: goharbor/harbor-portal:v2.14.1
Loaded image: goharbor/harbor-core:v2.14.1
Loaded image: goharbor/harbor-db:v2.14.1
```

[Step 3]: preparing environment ...

[Step 4]: preparing harbor configs ...

```
prepare base dir is set to /tmp/harbor
Clearing the configuration file: /config/log/rsyslog_docker.conf
Clearing the configuration file: /config/log/logrotate.conf
Clearing the configuration file: /config/portal/nginx.conf
Generated configuration file: /config/portal/nginx.conf
Generated configuration file: /config/log/logrotate.conf
Generated configuration file: /config/log/rsyslog_docker.conf
Generated configuration file: /config/nginx/nginx.conf
Generated configuration file: /config/core/env
Generated configuration file: /config/core/app.conf
Generated configuration file: /config/registry/config.yml
Generated configuration file: /config/registryctl/env
Generated configuration file: /config/registryctl/config.yml
Generated configuration file: /config/db/env
Generated configuration file: /config/jobservice/env
Generated configuration file: /config/jobservice/config.yml
copy /data/secret/tls/harbor_internal_ca.crt to shared trust ca dir as name
harbor_internal_ca.crt ...
ca file /hostfs/data/secret/tls/harbor_internal_ca.crt is not exist
copy to shared trust ca dir as name storage_ca_bundle.crt ...
copy None to shared trust ca dir as name redis_tls_ca.crt ...
Generated and saved secret to file: /data/secret/keys/secretkey
Successfully called func: create_root_cert
Generated configuration file: /compose_location/docker-compose.yml
Clean up the input dir
```

Note: stopping existing Harbor instance ...

[Step 5]: starting Harbor ...

```
[+] up 10/10
  ✓ Network harbor_harbor      Created
0.0s
  ✓ Container harbor-log      Created
0.1s
  ✓ Container harbor-db       Created
0.1s
  ✓ Container registry        Created
0.1s
  ✓ Container harbor-portal   Created
0.1s
  ✓ Container registryctl     Created
```

```

0.1s
✓ Container redis          Created
0.1s
✓ Container harbor-core    Created
0.1s
✓ Container harbor-jobservice Created
0.1s
✓ Container nginx          Created
0.1s
✓ ----Harbor has been installed and started successfully.----
~#

```

4 - Kontrolle

Am Ende der Installation wurde angezeigt, dass die Installation erfolgreich war und Harbor gestartet wurde. Dies lässt sich jetzt auf mehrere Arten kontrollieren.

4.1 - Docker-Container

Es kann geprüft werden, welche Docker-Container gerade laufen:

```

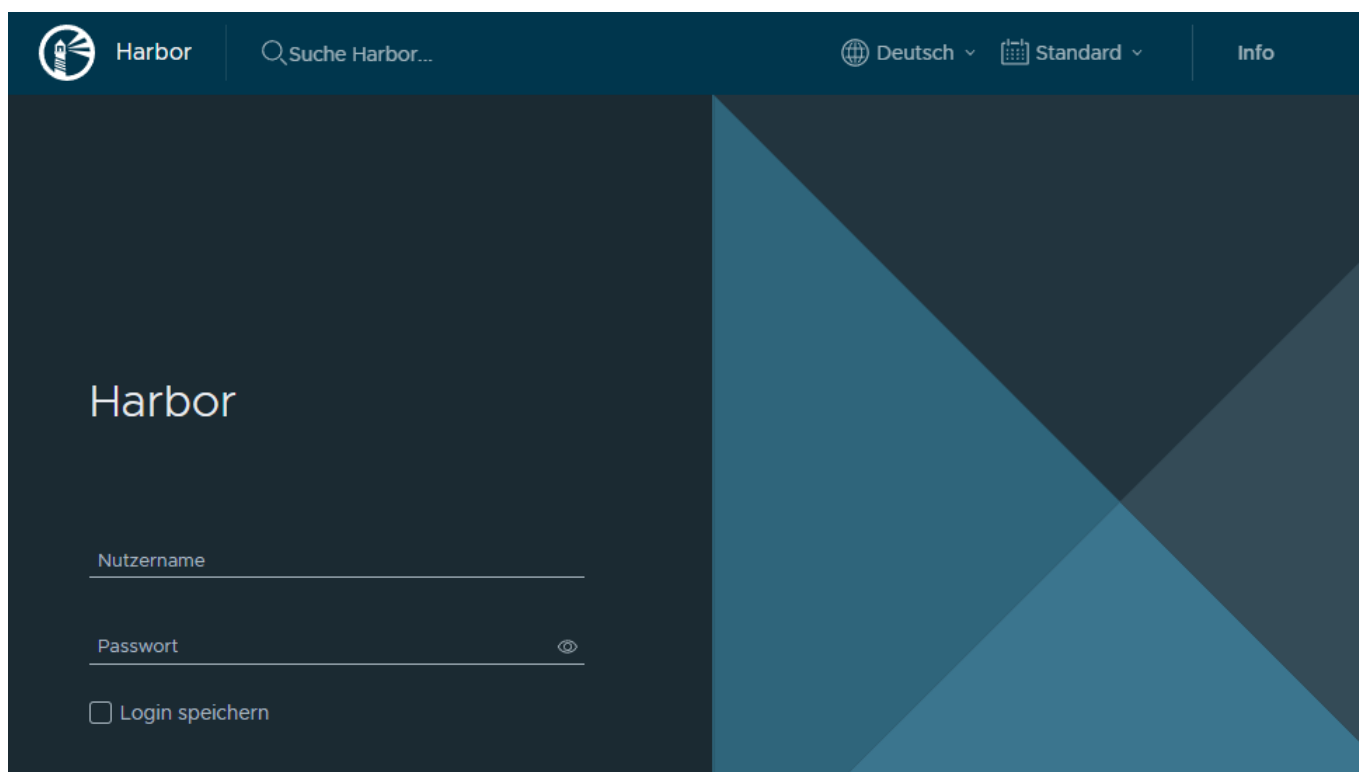
~# docker ps\
CONTAINER ID   IMAGE                                PORTS          COMMAND
CREATED       STATUS
NAMES\
5a9ec57ad6c9   goharbor/nginx-photon:v2.14.1      "nginx -g 'daemon of..."
2 minutes ago Up 2 minutes (healthy)  0.0.0.0:80->8080/tcp,
[::]:80->8080/tcp, 0.0.0.0:443->8443/tcp, [::]:443->8443/tcp  nginx\
b2a1dfeaa584   goharbor/harbor-jobservice:v2.14.1 "/harbor/entrypoint..."
2 minutes ago Up 2 minutes (healthy)
harbor-jobservice\
0b74687b2dae   goharbor/harbor-core:v2.14.1       "/harbor/entrypoint..."
2 minutes ago Up 2 minutes (healthy)
harbor-core\
321d88d3ebcf   goharbor/redis-photon:v2.14.1      "redis-server /etc/r..."
2 minutes ago Up 2 minutes (healthy)
redis\
e65496140ab3   goharbor/registry-photon:v2.14.1   "/home/harbor/entryp..."
2 minutes ago Up 2 minutes (healthy)
registry\
6ebb82ec2e0a   goharbor/harbor-registryctl:v2.14.1 "/home/harbor/start..."
2 minutes ago Up 2 minutes (healthy)
registryctl\
98b15e84535f   goharbor/harbor-portal:v2.14.1     "nginx -g 'daemon of..."
2 minutes ago Up 2 minutes (healthy)
harbor-portal\

```

```
1ac95fd29a67    goharbor/harbor-db:v2.14.1    "/docker-entrypoint..."
2 minutes ago   Up 2 minutes (healthy)
harbor-db\
5254d72818a9    goharbor/harbor-log:v2.14.1    "/bin/sh -c /usr/loc..."
2 minutes ago   Up 2 minutes (healthy)    127.0.0.1:1514->10514/tcp
harbor-log
~#
```

4.2 - Webseite

Auch die Webseite von Harbor sollte jetzt erreichbar sein: <https://slxharbor.fritz.box/>

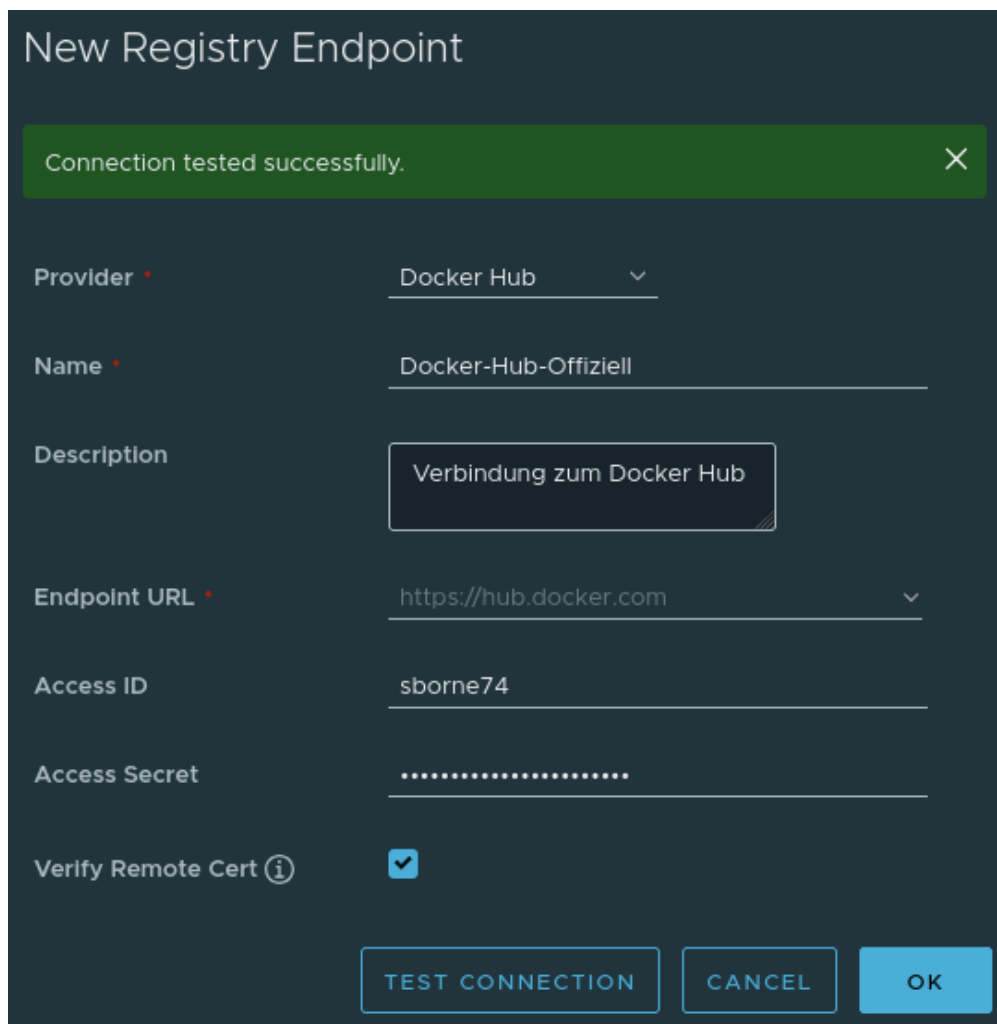


5 - Wartung

Die Wartung von Harbor wird zum entsprechenden Zeitpunkt nachgepflegt.

6 - Registrierung einrichten

Am Beispiel von [Docker Hub](#) soll die Registrierung demonstriert werden. Die Informationen wurden teilweise von der Webseite „[blogs.vmware.com](#)“ entnommen. Auf der Docker-Webseite wurde ein Account erstellt, welcher zur Verbindung genutzt wird. Im Harbor wird eine neue Registrierung unter dem Menüpunkt „Administration“ »> „Registries“ erstellt:



New Registry Endpoint

Connection tested successfully. X

Provider * Docker Hub ▾

Name * Docker-Hub-Offiziell

Description

Verbindung zum Docker Hub

Endpoint URL * https://hub.docker.com ▾

Access ID sborne74

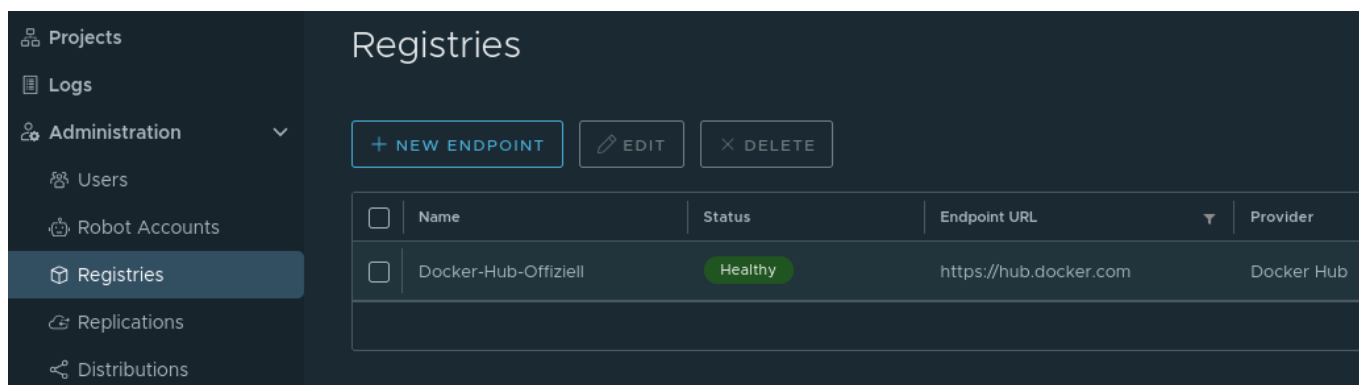
Access Secret

Verify Remote Cert ⓘ ☒

TEST CONNECTION CANCEL OK

Mit Klick auf „TEST CONNECTION“ kann die Verbindung überprüft werden. Mit „OK“ wird die Registrierung dann hinzugefügt.

So sollte es dann hinterher aussehen:



Registries				
+ NEW ENDPOINT EDIT DELETE				
☐	Name	Status	Endpoint URL	Provider
☐	Docker-Hub-Offiziell	Healthy	https://hub.docker.com	Docker Hub

\\

.Ende des Dokuments

From:

<https://looper.de/wiki/> - **Linux4Ever**

Permanent link:

<https://looper.de/wiki/doku.php?id=container:harbor:start>

Last update: **2025/12/22 13:05**

