Docker

Docker ist eine freie Software zur Isolierung von Anwendungen mit Hilfe von Containervirtualisierung. Docker vereinfacht die Bereitstellung von Anwendungen, weil sich Container, die alle nötigen Pakete enthalten, leicht als Dateien transportieren und installieren lassen

- Installation Docker
- Docker Container
    - Dockge
    - Zammad
    - Portainer
    - UniFi Network Application
- Admin Passwort zurücksetzen
- Portainer per Docker Compose bereitstellen
- Knowledge Base
    - Docker Engine unter Debian/Ubuntu komplett entfernen

Installation Docker

Einleitung

In diesem Artikel zeigen wir den Ablauf einer Docker CE (Community Edition) Installation auf einem Debian basierenden Linux auf. Diese Dokumentation zeigt eine vereinfachte Installation auf und sollte bei produktiven Server individuell überdacht werden.

Installation mit apt

Bevor wir die Docker Engine auf dem System installieren können. Müssen wir sicherstellen, dass wir die Repository aktualisieren, denn Docker ist in der allgemeinen Repo nicht enthalten.

1. Einrichtung der Docker Repository

Hinzufügen von Docker's offiziellem GNU-Privatsphärenschutz:

```
sudo apt-get update
sudo apt-get install ca-certificates curl
sudo install -m 0755 -d /etc/apt/keyrings
sudo curl -fsSL https://download.docker.com/linux/debian/gpg -o /etc/apt/keyrings/docker.asc
sudo chmod a+r /etc/apt/keyrings/docker.asc
```

Hinzfügen der Docker Repository:

```
echo \
  "deb [arch=$(dpkg --print-architecture) signed-by=/etc/apt/keyrings/docker.asc]
https://download.docker.com/linux/debian \
  $(. /etc/os-release && echo "$VERSION_CODENAME") stable" | \

sudo tee /etc/apt/sources.list.d/docker.list > /dev/null
sudo apt-get update
```

2. Installation der Docker Pakete Einrichtung der Docker Repository

Aktuelle Version

Um die aktuellste Version zu installieren, führe folgenden Befehl aus:

```
sudo apt-get install docker-ce docker-ce-cli containerd.io docker-buildx-plugin docker-compose-plugin
```

Spezifische Version

```
# List the available versions:
apt-cache madison docker-ce | awk '{ print $3 }'

5:27.0.3-1~debian.12~bookworm
5:27.0.2-1~debian.12~bookworm
...
```

Packetinstallation

- Installation mittels Packet
- Skript Installation

Docker Container

Docker Container

Dockge

How to Install Requirements:

Docker 20+ / Podman (Podman only) podman-docker (Debian: apt install podman-docker) OS: Major Linux distros that can run Docker/Podman such as: Ubuntu Debian (Bullseye or newer) Raspbian (Bullseye or newer) CentOS Fedora ArchLinux Debian/Raspbian Buster or lower is not supported Windows (Will be supported later) Arch: armv7, arm64, amd64 (a.k.a x86_64) Basic Default Stacks Directory: /opt/stacks Default Port: 5001

Create directories that store your stacks and stores Dockge's stack

mkdir -p /opt/stacks /opt/dockge cd /opt/dockge

Download the compose.yaml

curl https://raw.githubusercontent.com/louislam/dockge/master/compose.yaml --output compose.yaml

Start the server

docker compose up -d

If you are using docker-compose V1 or Podman

docker-compose up -d

Dockge is now running on http://localhost:5001

Advanced If you want to store your stacks in another directory, you can generate your compose.yaml file by using the following URL with custom query strings.

Download your compose.yaml

curl "https://dockge.kuma.pet/compose.yaml?port=5001&stacksPath=/opt/stacks" --output compose.yaml port=5001 stacksPath=/opt/stacks Interactive compose.yaml generator is available on: https://dockge.kuma.pet

How to Update cd /opt/dockge docker compose pull && docker compose up -d

Docker Container

Zammad

dsdfdsf
sdfsddfdsfdsfdsf

Systemvorrausetzungen

- Docker Compose muss auf dem System installiert werden
- 4GB RAM
- Systemeinstellungen müssen wie folgt angepasst werden

```
sysctl -w vm.max_map_count=262144
```

Installation mit Docker Compose

GitHub Repo klonen

```
git clone https://github.com/zammad/zammad-docker-compose.git
```

Adjust Environment as Needed

In some cases our default environment is not what a docker-compose user is looking for. See Docker Environment Variables for details on which settings can be configured.

> If you want to use a `.env` file, you can use the provided `.env.dist` file and copy it to `.env`. That way it will be picked up by Docker-Compose automatically and not overwritten during updates.

Zammad runs on port `8080` by default. If you want to use another port, you can set it via the variable `NGINX_EXPOSE_PORT`.

Portainer

Portainer Community Edition (CE)

Installation

```
docker run -d -p 8000:8000 -p 9443:9443 --name portainer --restart=always -v /var/run/docker.sock:/var/run/docker.sock -v portainer_data:/data portainer/portainer-ce
```

Update

1. Stop Docker Container

   ```
   docker stop portainer
   ```

2. Remove Docker Container

   ```
   docker rm portainer
   ```

3. Update Docker Image

   ```
   docker pull portainer/portainer-ce
   ```

Docker Container

UniFi Network Application

docker-compose.yaml

```
services:
  # Define the services to run
  db:
    image: mongo
    volumes:
      - ./db:/data/db # MongoDB data persistence
      - ./db/init-mongo.js:/docker-entrypoint-initdb.d/init-mongo.js:ro
    ports:
      - 27017:27017 # MongoDB default port
    restart: always # Restart policy
    networks:
      backend: null
  app:
    image: lscr.io/linuxserver/unifi-network-application # Docker image to use
    environment:
      # Environmental variables for the container
      - PUID=1000 # User ID
      - PGID=1000 # Group ID
      - TZ=Europe/Zurich # Timezone
      - MONGO_HOST=db # MongoDB host
      - MONGO_USER= ${DB_USER} # MongoDB username
      - MONGO_PASS= ${DB_PW} # MongoDB password
      - MONGO_PORT=27017 # MongoDB port
      - MONGO_DBNAME= ${DB_NAME} # MongoDB database name
      - MEM_LIMIT=1024 #optional  # Memory limit for the container
      - MEM_STARTUP=1024 #optional  # Memory to allocate on container startup
    volumes:
      # Volumes to mount in the container
      - ./web:/config # Map host directory to container directory
    ports:
      # Ports to expose and forward
      - 8443:8443 # HTTPS portal
```

```
      - 3478:3478/udp # STUN service
      - 10001:10001/udp # UniFi AP discovery
      - 8080:8080 # HTTP portal
      - 6789:6789 #optional  # Mobile speed test port
      - 161:161/udp #SNMP
    restart: always # Restart policy for the container
    depends_on:
      - db
    networks:
      frontend: null
      backend: null
networks:
  frontend: null
  backend: null
```

.env

```
DB_USER=unifi
DB_PW=password
DB_NAME=unifi_db
```

init-mongo.js

```
db.getSiblingDB("unifi_db").createUser({user: "unifi", pwd: "password", roles: [{role:
"dbOwner", db: "unifi_db"}]});
db.getSiblingDB("unifi_db_stat").createUser({user: "unifi", pwd: "password", roles: [{role:
"dbOwner", db: "unifi_db_stat"}]});
```

Admin Passwort zurücksetzen

You would typically use this method if you run the Portainer Server on Docker Standalone.

First, go to our reset password container helper in GitHub, then stop the Portainer container by running this command:

```
docker stop "id-portainer-container"
```

Next, run the helper using the following command (you'll need to mount the Portainer data volume):

If your Portainer data volume has a different name than `portainer_data` or you are using a bind mount for your data volume, you will need to adjust the mount in the below `docker run` command to suit your path.

```
docker pull portainer/helper-reset-password
docker run --rm -v portainer_data:/data portainer/helper-reset-password
```

If successful, the output should look like this:

```
2020/06/04 00:13:58 Password successfully updated for user: admin
2020/06/04 00:13:58 Use the following password to login: &_4#\3^5V8vLTd)E"NWiJBs26G*9HPl1
```

If the helper is unable to find an admin user to update, it will create a new one for you. If the username `admin` is already used, it will create a user named `admin-[randomstring]`:

```
2022/08/10 07:36:33 [WARN] Unable to retrieve user with ID 1, will try to create, err: object
not found inside the database
2022/08/10 07:36:33 Admin user admin-u0512b3f0v4dqk7o successfully created
2022/08/10 07:36:33 Use the following password to login: Sr#]YL_6D0k8Pd{pA9^|}F32j5J4I=av
```

Finally, use this command to start the Portainer container then try logging in with the new password:

Portainer per Docker Compose bereitstellen

Portainer Agent

```
services:
  app:
    container_name: portainer-agent #Name kann beliebig angepasst werden
    image: portainer/agent:lts #LTS Version
    ports:
      - "9001:9001"
    volumes:
      # Mount the host's Docker socket into the container
      - /var/run/docker.sock:/var/run/docker.sock
      # Mount the host's Docker volumes into the container
      - /var/lib/docker/volumes:/var/lib/docker/volumes
    restart: always
    network_mode: bridge
    deploy:
      resources:
        limits:
          cpus: '0.5'
          memory: 512M
```

Knowledge Base

Aktive technische Knowledge-Base-Einträge aus Obsidian.

Docker Engine unter Debian/Ubuntu komplett entfernen

Docker Engine unter Debian/Ubuntu komplett entfernen

Voraussetzungen

- Debian- oder Ubuntu-basierte Distribution (apt-Paketmanager). Auf Fedora/RHEL/openSUSE andere Paketnamen und `dnf`/`zypper` verwenden.
- Diese Schritte sind **destruktiv**: Schritt 3 löscht alle Container, Images, Volumes und Netzwerke **endgültig**. Vorher Backup sichern, wenn Daten gebraucht werden.

Schritt 1: Installierte Docker-Pakete identifizieren

```
dpkg -l | grep -i docker
```

Zusätzlich prüfen, ob Container laufen:

```
docker ps -a
```

Schritt 2: Pakete entfernen (nicht destruktiv für Daten)

```
sudo apt-get purge -y docker-engine docker docker.io docker-ce docker-ce-cli \
  docker-compose-plugin containerd.io docker-buildx-plugin
sudo apt-get autoremove -y --purge
```

Schritt 3: Daten und Konfiguration löschen (endgültig!)

> **warning:** Alle Images, Container, Volumes und Netzwerke werden unwiderruflich gelöscht.

```
sudo rm -rf /var/lib/docker /etc/docker
sudo rm /etc/apparmor.d/docker
sudo groupdel docker
sudo rm -f /var/run/docker.sock
sudo rm -rf /var/lib/containerd
sudo rm -rf ~/.docker
```

Schritt 4: Kontrolle

```
dpkg -l | grep -i docker        # keine Pakete mehr
ls /var/lib/docker              # sollte fehlen
docker --version                # command not found
```

Quellen

- Docker Docs — Uninstall Docker Engine: https://docs.docker.com/engine/install/uninstall/
- Docker Docs — Install on Ubuntu: https://docs.docker.com/engine/install/ubuntu/

Quelle: Obsidian `Knowledge Base/Linux/Remove Docker.md` · Stand: 2026-10-12 · Automatisiert publiziert durch Hermes.