

# Docker installation

## Beschreibung:

Installation via Docker und Lets encrypt.

Das Image hat den certbot schon drin, so das wir keinen Certbot Conatiner brauchen.

## Instalktion

```
cd /root/
```

Abhängigkeiten installieren

```
apt install docker.io docker-compose apparmor unzip curl
```

## Jitsi zip downloaden

```
LATEST_VERSION=$(curl --silent https://api.github.com/repos/jitsi/docker-jitsi-meet/releases/latest | grep -Po  
"tag_name": "\K.*\d')  
wget https://github.com/jitsi/docker-jitsi-meet/archive/refs/tags/${LATEST_VERSION}.zip
```

Nun unzippen

```
unzip ${LATEST_VERSION}.zip
```

Verzeichnis in jitsi umbenennen

```
mv docker-jitsi-meet-${LATEST_VERSION} jitsi
```

## .env Datei erstellen und passwörter generieren

```
cd jitsi && cp env.example .env  
./gen-passwords.sh
```

Die .env für Letsencrypt etc bearbeiten

```
nano /root/jitsi/.env
```

Diese zeilen anpassen

```
HTTP_PORT=80
HTTPS_PORT=443
TZ=Europe/Berlin
PUBLIC_URL=https://jitsi.example.com # <== Your domain name here
#lets encrypt
ENABLE_LETSENCRYPT=1
LETSENCRYPT_DOMAIN=jitsi.example.com# <== Your domain name here
LETSENCRYPT_EMAIL=mail@example.com
#benutzerauthentifizierungs
# Enable authentication
#ENABLE_AUTH=1

# Enable guest access
ENABLE_GUESTS=1

# Select authentication type: internal, jwt, ldap or matrix
#AUTH_TYPE=internal
```

## Container Starten

```
docker-compose up -d
```

Um den container status zu sehen.

```
docker-compose logs -f <containername im docker compose file>
```

## JWT Token einrichten

Mit JWT Token kann man seine JITSI instanz sichern, ohne Benutzer zu haben.

An die URL wir einfach

```
?jwt=meintoken
```

angegeben.

die .env anpassen

Wir legen folgendes fest:

```
# Enable authentication
ENABLE_AUTH=1

# Enable guest access
ENABLE_GUESTS=1

# Select authentication type: internal, jwt, ldap or matrix
AUTH_TYPE=jwt

# JWT authentication
#

# Application identifier
JWT_APP_ID=unsere__app_id : beispiel jitsi.company

# Application secret known only to your token generator
JWT_APP_SECRET=ghsgfbva54dsgcayAas33s
```

container neustarten

docker-compose up -d

Nun den Token erstellen, dazu nehmen wir ein python script

```
import jwt
import datetime

# Ihr AppSecret
secret = 'IHR_APP_SECRET'

# Die Claims
claims = {
    "context": {
        "user": {
            "name": "Max Mustermann",
            "email": "max@example.com",
```

```

        "avatar": "https://example.com/avatar.jpg"
    }
},
"aud": "jitsi",
"iss": "your_app_id",
"sub": "meet.example.com",
"room": "*",
"exp": datetime.datetime.utcnow() + datetime.timedelta(days=1)
}

# Token generieren
token = jwt.encode(claims, secret, algorithm='HS256')

print(token)

```

Unser angepassten Beispiel:

Was passen wir an:

Unter User :

Name:

Email : lassen wir leer

avatar : lassen wir auch leer, können aber auch ein Logo von der Firma hinterlegen muss eine http oder https url sein

in der Funktion Exp, geben wir das Ablaufdatum in Tagen an. Ich nehme 100 Jahre (36525 Tage) :

"exp": datetime.datetime.utcnow() + datetime.timedelta(days=36525)

```

import jwt
import datetime

# Ihr AppSecret
secret = 'ghsgfbva54dsgcayAas33s'

# Die Claims
claims = {
    "context": {
        "user": {
            "name": "Meeting Leader",
            "email": "test@test.de",
            "avatar": "https://example.com/avatar.jpg"
        }
    }
},

```

```
"aud": "jitsi",  
"iss": "jstis.company",  
"sub": "jitsi.meinedomain.",  
"room": "*",  
"exp": datetime.datetime.utcnow() + datetime.timedelta(days=36525)  
}  
  
# Token generieren  
token = jwt.encode(claims, secret, algorithm='HS256')  
  
print(token)
```

## Benutzer anlegen

---

Version #10

Erstellt: 3 April 2024 13:33:07 von Admin

Zuletzt aktualisiert: 11 Mai 2024 13:50:20 von Admin