1-2-4-https-caddy-keycloak

## **1.0 Проверка текущих настроек**

Перед началом настройки необходимо убедиться, что:

* Виртуальная машина работает;
* Все нужные сервисы (OpenWebUI, Keycloak, Caddy) запущены или готовы к запуску;
* Порты не заняты другими сервисами;
* Docker работает корректно внутри виртуальной машины.

❗️**Важно**: где запускать команды:

* 💻 **MacBook (хост)**: терминал на вашем компьютере, где установлен Multipass.
* 🖥️ **ВМ openwebui-test**: Ubuntu внутри виртуальной машины (через multipass shell).
* 🐳 **Docker-контейнер внутри ВМ**: внутри контейнера через docker exec.

### **1.1 Проверить, запущена ли виртуальная машина (на MacBook)**

multipass list

Ожидаем увидеть строку со статусом Running для вашей ВМ:

Name State IPv4 Image  
openwebui-test Running 192.168.64.2 Ubuntu 22.04 LTS

Если не запущена — запустите:

multipass start openwebui-test

### **1.2 Подключиться к ВМ (на MacBook)**

multipass shell openwebui-test

### **1.3 Проверить, что Docker установлен и работает (в ВМ)**

docker info

### **1.4 Посмотреть запущенные контейнеры (в ВМ)**

docker ps

Убедитесь, что видите следующие контейнеры:

CONTAINER ID IMAGE NAMES  
... quay.io/keycloak/keycloak:24... openwebui-keycloak-1  
... caddy openwebui-caddy-1  
... ghcr.io/open-webui/open-webui openwebui-open-webui-1

### **1.5 Проверить занятые порты (в ВМ)**

sudo lsof -i -P -n | grep LISTEN

Проверьте, что нужные порты (например, 80, 443, 3000, 8080, 8443) используются только Docker-контейнерами, связанными с OpenWebUI.

### **1.6 Проверить, что нет конфликтов с Caddy (в ВМ)**

Если раньше был установлен Nginx или другой reverse-proxy, убедитесь, что он остановлен:

sudo systemctl stop nginx  
sudo systemctl disable nginx

### **1.7 Приведение к чистому состоянию (если что-то работает неправильно)**

Если вы хотите полностью сбросить состояние:

# Удаление всех контейнеров  
docker rm -f $(docker ps -aq)  
  
# Удаление всех образов (опционально)  
docker rmi -f $(docker images -aq)  
  
# Удаление всего в папке с openwebui  
cd ~/openwebui  
sudo rm -rf ./\*

После этого вы можете продолжить установку с шага 2.0: > - Markdown: [Настройка Keycloak через CLI](#keycloak-cli-setup) > - HTML: [Настройка Keycloak через CLI](#keycloak-cli-setup) > - По заголовку: [2. Настройка Keycloak (Realm, Client, User) через CLI](#X857ecf6e78ecc9723b454dc8a5fdfded5dbd3c6)

## Продакшн-настройка OpenWebUI с Keycloak и HTTPS

### 🧩 Состав системы

* **OpenWebUI** — веб-интерфейс для LLM, работает на localhost:8080.
* **Keycloak** — система аутентификации, доступна на localhost:8081.
* **Caddy** — reverse proxy и TLS-терминатор, обрабатывает HTTPS-запросы и направляет их на соответствующие сервисы.

## 🔧 Шаги настройки

### 1. Установите Keycloak

Создайте директорию keycloak и файл Dockerfile:

# keycloak/Dockerfile  
FROM quay.io/keycloak/keycloak:24.0.3  
ENV KC\_DB=dev-mem  
ENV KEYCLOAK\_ADMIN=admin  
ENV KEYCLOAK\_ADMIN\_PASSWORD=admin  
ENTRYPOINT ["/opt/keycloak/bin/kc.sh", "start-dev"]

Создайте docker-compose.override.yml или добавьте в основной docker-compose.yml:

services:  
 keycloak:  
 build: ./keycloak  
 container\_name: openwebui-keycloak-1  
 ports:  
 - "8081:8080"  
 networks:  
 - openwebui-network  
  
networks:  
 openwebui-network:  
 external: false

### 2. Настройка Keycloak (Realm, Client, User) через CLI

Создайте скрипт init-keycloak.sh:

#!/bin/bash  
  
KEYCLOAK\_URL=http://openwebui-keycloak-1:8080  
REALM=openwebui  
CLIENT=openwebui-client  
CLIENT\_SECRET=supersecret  
REDIRECT\_URI=http://192.168.64.2/auth/callback  
ADMIN\_USER=admin  
ADMIN\_PASS=admin  
TEST\_USER=user1  
TEST\_PASS=user1pass  
  
# Ждем, пока Keycloak поднимется  
until curl -s $KEYCLOAK\_URL > /dev/null; do  
 echo "⌛ Ждём запуска Keycloak..."  
 sleep 5  
done  
  
# Получение токена администратора  
TOKEN=$(curl -s -X POST "$KEYCLOAK\_URL/realms/master/protocol/openid-connect/token" \  
 -H "Content-Type: application/x-www-form-urlencoded" \  
 -d "username=$ADMIN\_USER" \  
 -d "password=$ADMIN\_PASS" \  
 -d "grant\_type=password" \  
 -d "client\_id=admin-cli" | jq -r .access\_token)  
  
# Создание реалма  
curl -s -X POST "$KEYCLOAK\_URL/admin/realms" \  
 -H "Authorization: Bearer $TOKEN" \  
 -H "Content-Type: application/json" \  
 -d "{\"realm\":\"$REALM\",\"enabled\":true}"  
  
# Создание клиента  
curl -s -X POST "$KEYCLOAK\_URL/admin/realms/$REALM/clients" \  
 -H "Authorization: Bearer $TOKEN" \  
 -H "Content-Type: application/json" \  
 -d "{  
 \"clientId\": \"$CLIENT\",  
 \"enabled\": true,  
 \"protocol\": \"openid-connect\",  
 \"redirectUris\": [\"$REDIRECT\_URI\"],  
 \"publicClient\": false,  
 \"secret\": \"$CLIENT\_SECRET\"  
 }"  
  
# Создание пользователя  
curl -s -X POST "$KEYCLOAK\_URL/admin/realms/$REALM/users" \  
 -H "Authorization: Bearer $TOKEN" \  
 -H "Content-Type: application/json" \  
 -d "{  
 \"username\": \"$TEST\_USER\",  
 \"enabled\": true,  
 \"credentials\": [{  
 \"type\": \"password\",  
 \"value\": \"$TEST\_PASS\",  
 \"temporary\": false  
 }]  
 }"  
  
echo "✅ Keycloak сконфигурирован"

Запуск:

chmod +x init-keycloak.sh  
docker exec -it openwebui-keycloak-1 /bin/sh -c "apk add curl jq && /srv/init-keycloak.sh"

### 3. Подготовьте Caddy

Создайте папку caddy/ и файл Caddyfile:

# caddy/Caddyfile  
  
http://192.168.64.2 {  
 handle\_path /auth/\* {  
 reverse\_proxy openwebui:8080  
 }  
  
 handle\_path /realms/\* {  
 reverse\_proxy openwebui-keycloak-1:8080  
 }  
  
 handle {  
 respond "Not Found" 404  
 }  
}

Добавьте Caddy в docker-compose.yml:

services:  
 caddy:  
 image: caddy:2  
 container\_name: openwebui-caddy  
 ports:  
 - "80:80"  
 volumes:  
 - ./caddy/Caddyfile:/etc/caddy/Caddyfile  
 depends\_on:  
 - openwebui  
 - keycloak  
 networks:  
 - openwebui-network

### 4. Обновите переменные среды OpenWebUI (.env)

OAUTH\_ENABLED=true  
OAUTH\_PROVIDER=oidc  
OAUTH\_CLIENT\_ID=openwebui-client  
OAUTH\_CLIENT\_SECRET=supersecret  
OAUTH\_ISSUER\_URL=http://192.168.64.2/realms/openwebui  
OAUTH\_REDIRECT\_URL=http://192.168.64.2/auth/callback

### 5. Перезапустите все сервисы

docker compose down  
docker compose up -d --build

## ✅ Результат

OpenWebUI доступен по адресу:

http://192.168.64.2/auth

* Авторизация и регистрация через Keycloak Realm openwebui
* После входа пользователи перенаправляются в OpenWebUI

## 📊 ASCII-схема взаимодействия

+------------------------+ +----------------------+  
| Пользователь | <---> | Caddy |  
+------------------------+ +----------------------+  
 |  
 | (reverse proxy)  
 v  
 /auth/\* -------------------------> OpenWebUI:8080  
 /realms/\* -----------------------> Keycloak:8080  
 ^  
 |  
 Docker network