Instalacja NetBoksa w Dockerze

Typ: procedura

Dotyczy: netbox-docker v5.1.0, obraz netboxcommunity/netbox:v4.7-5.1.0 (sprawdzono w dokumentacji projektu 28.09.2026)

Czas: ~45 min

Efekt: działający NetBox na `docker01`, dostępny pod `netbox.zielona.example`, z kontem administratora i pierwszymi wpisanymi danymi: dwiema lokalizacjami Zielonej i dwoma urządzeniami.

Skróty użyte na tej stronie (3)

Skrót	Co znaczy
API	Application Programming Interface: sposób, w jaki programy rozmawiają ze sobą
DCIM	Data Center Infrastructure Management: dokumentacja tego, co fizycznie stoi w serwerowni, czyli szafy, urządzenia i okablowanie
TLS	Transport Layer Security: szyfrowanie połączeń, np. w HTTPS

Zanim zaczniemy

> **Z ŻYCIA WZIĘTE**
>
> Tomek zakłada w NetBoksie pierwszą szafę i pierwsze urządzenie, ale najpierw musi w ogóle postawić samą aplikację. Znajduje w internecie dziesiątki poradników z różnymi wersjami pliku Compose, każdy z inną nazwą usługi bazy danych. Sięga po oficjalny projekt `netbox-community/netbox-docker`, zamiast składać własny plik od zera z fragmentów znalezionych w sieci.

Instalację NetBoksa opieramy na oficjalnym projekcie `netbox-community/netbox-docker`, osobnym od samego NetBoksa, ale utrzymywanym przez tę samą społeczność. Projekt dostarcza gotowy `docker-compose.yml` z przypiętym obrazem, a własne odstępstwa wprowadzamy przez plik `docker-compose.override.yml`, zamiast edytować plik bazowy wprost: dzięki temu aktualizacja do nowszej

wersji repozytorium nie nadpisuje naszych zmian.

> W naszym środowisku dane aplikacji trzymamy w `/docker/apps/<nazwa>`, a plik Compose w `/docker/compose/<nazwa>/compose.yaml`. To konwencja, nie wymóg Dockera: można ją zmienić, byle w całej firmie była jedna. Katalog `/docker` to osobny wolumen `vgDocker/lvDocker`, żeby pełny dysk aplikacji nie zatrzymał systemu.

W odróżnieniu od MailCow, opisanego na stronie Instalacja MailCow na docker01 od zera, repozytorium `netbox-docker` trzyma tylko sam plik Compose i konfigurację, bez katalogu z danymi wymieszanego w środku: dane NetBoksa (baza, media, raporty) domyślnie trafiają do wolumenów zarządzanych przez Dockera, a my przez `docker-compose.override.yml` przekierowujemy je pod `/docker/apps/netbox`, zgodnie ze zwykłą konwencją Zielonej.

Wymaganie	Uwaga
Docker Engine i wtyczka Compose	sprawdzamy `docker compose version`
Wolny port `8000` na hoście (tylko lokalnie)	NetBox stoi za Nginksem, patrz krok reverse proxy
Wariant b: osiągalny `pg01` na porcie `5432`	baza tworzona ręcznie przed pierwszym startem
`SECRET_KEY` wygenerowany osobno dla tej instalacji	nigdy nie kopiujemy klucza z dokumentacji ani z innego środowiska

Uważaj: obraz zawsze przypinamy do konkretnej wersji (`v4.7-5.1.0`), nigdy do `latest`. Aktualizacja to świadoma zmiana tagu w `docker-compose.override.yml`, opisana szerzej na stronie NetBox: logowanie z AD, kopia i aktualizacja.

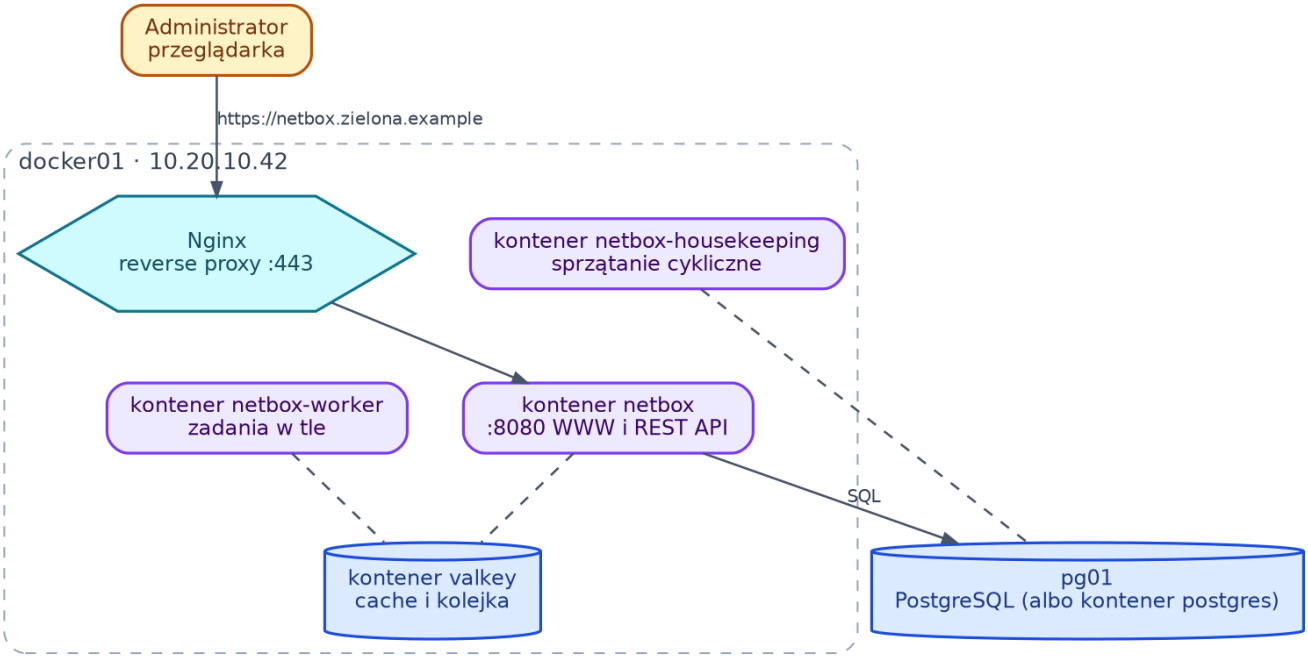
Przebieg

Klonowanie repozytorium: git clone

```
sudo mkdir -p /docker/compose
cd /docker/compose
sudo git clone -b release https://github.com/netbox-community/netbox-docker.git netbox
cd netbox
```

Gałąź `release` śledzi ostatnie stabilne wydanie projektu, zgodnie z zaleceniem samego `netbox-docker`.

Podstawowa architektura instalacji, wspólna dla obu wariantów bazy danych opisanych niżej, wygląda tak:



Architektura NetBoksa w Dockerze: Nginx, kontenery netbox, netbox-worker, netbox-housekeeping, valkey i baza PostgreSQL (kliknięcie otwiera pełny rozmiar)

Wariant a: laboratorium, baza w kontenerze PostgreSQL

Zakładamy katalogi na dane, zgodnie z konwencją ścieżek:

```
sudo mkdir -p /docker/apps/netbox/postgres /docker/apps/netbox/media
/docker/apps/netbox/reports /docker/apps/netbox/scripts
cp docker-compose.override.yml.example docker-compose.override.yml
```

Wygenerowanie klucza `SECRET_KEY`:

```
openssl rand -base64 45
```

Plik `docker-compose.override.yml`, wersja skrócona do zmian względem pliku bazowego:

```
services:
  netbox:
    image: docker.io/netboxcommunity/netbox:v4.7-5.1.0
```

```
    ports:
      - "127.0.0.1:8000:8080"
    environment:
      SECRET_KEY: "WygenerowanyKluczSecretKey"
      SUPERUSER_NAME: admin
      SUPERUSER_EMAIL: it@zielona.example
      SUPERUSER_PASSWORD: "HasloAdministratora"
    volumes:
      - /docker/apps/netbox/media:/opt/netbox/netbox/media:z
      - /docker/apps/netbox/reports:/opt/netbox/netbox/reports:z
      - /docker/apps/netbox/scripts:/opt/netbox/netbox/scripts:z
  netbox-worker:
    image: docker.io/netboxcommunity/netbox:v4.7-5.1.0
  netbox-housekeeping:
    image: docker.io/netboxcommunity/netbox:v4.7-5.1.0
  postgres:
    image: docker.io/postgres:18-alpine
    volumes:
      - /docker/apps/netbox/postgres:/var/lib/postgresql/data
```

(gdzie usługi `netbox-worker` i `netbox-housekeeping` muszą wskazywać ten sam obraz co `netbox`, inaczej mieszają się wersje schematu bazy danych między kontenerami tej samej instalacji)

Wariant b: produkcja Zielonej, baza na zewnętrznym pg01

Zamiast usługi `postgres` w pliku override wskazujemy istniejący serwer `pg01`. Zanim uruchomimy kontenery, zakładamy bazę i użytkownika:

```
psql -h pg01 -U postgres
```

```
CREATE DATABASE netbox;
CREATE USER netbox WITH ENCRYPTED PASSWORD 'HasloDoBazy';
GRANT ALL PRIVILEGES ON DATABASE netbox TO netbox;
ALTER DATABASE netbox OWNER TO netbox;
```

Fragment `docker-compose.override.yml` różniący się od wariantu a:

```
services:
  netbox:
    image: docker.io/netboxcommunity/netbox:v4.7-5.1.0
    environment:
      DB_HOST: pg01
      DB_NAME: netbox
      DB_USER: netbox
      DB_PASSWORD: "HasloDoBazy"
    ports:
      - "127.0.0.1:8000:8080"
    volumes:
      - /docker/apps/netbox/media:/opt/netbox/netbox/media:z
      - /docker/apps/netbox/reports:/opt/netbox/netbox/reports:z
      - /docker/apps/netbox/scripts:/opt/netbox/netbox/scripts:z
```

Usługę `postgres` z pliku bazowego wyłączamy, dopisując w override `postgres: profiles: ["disabled"]`, żeby `docker compose up` jej nie uruchamiał.

Pierwsze uruchomienie: docker compose

```
cd /docker/compose/netbox
sudo docker compose pull
sudo docker compose up -d
```

Reverse proxy: netbox.zielona.example

NetBox nasłuchuje wyłącznie na `127.0.0.1:8000`, więc wystawiamy go na zewnątrz dopiero przez Nginksa, zgodnie ze stroną Reverse proxy i równoważenie obciążenia, z certyfikatem TLS dla `netbox.zielona.example`.

Pierwsze dane: lokalizacje, szafa, urządzenia

Po zalogowaniu kontem administratora wprowadzamy dane w kolejności zgodnej z hierarchią modelu danych NetBoksa, opisaną na stronie NetBox: najpierw lokalizacje, potem szafę, na końcu urządzenia.

```
Organization → Sites → Add: Pamiątkowo (centrala)
Organization → Sites → Add: Swarzędz (magazyn)
Organization → Locations → Add: Serwerownia (site: Pamiątkowo)
DCIM → Racks → Add: R1 (site: Pamiątkowo, location: Serwerownia)
DCIM → Devices → Add: docker01 (site: Pamiątkowo, rack: R1)
DCIM → Devices → Add: fw01 (site: Pamiątkowo, rack: R1)
```

◫ ZRZUT EKRANU DO UZUPEŁNIENIA

Zrzut do uzupełnienia: ekran DCIM → Devices w NetBoksie po dodaniu urządzeń `docker01` i `fw01`, z widoczną kolumną „Site" pokazującą „Pamiątkowo" dla obu wierszy.

Weryfikacja

```
cd /docker/compose/netbox
sudo docker compose ps
```

Przykład:

```
NAME                        IMAGE                                  STATUS
netbox-netbox-1             netboxcommunity/netbox:v4.7-5.1.0      Up 2 minutes (healthy)
netbox-netbox-worker-1      netboxcommunity/netbox:v4.7-5.1.0      Up 2 minutes
netbox-netbox-housekeeping-1  netboxcommunity/netbox:v4.7-5.1.0    Up 2 minutes
netbox-postgres-1           postgres:18-alpine                     Up 2 minutes (healthy)
netbox-valkey-1             valkey/valkey:9.1-alpine               Up 2 minutes (healthy)
```

```
curl -I http://127.0.0.1:8000
```

Przykład:

```
HTTP/1.1 200 OK
```

Sprawdzamy dostęp do API tokenem administratora. Token v2, który zakładamy w profilu konta (zakładka API Tokens), ma postać `nbt_KluczTokenu.TokenAPI` i idzie w nagłówku `Authorization: Bearer`; pełnej wartości nie da się odczytać po jego utworzeniu, więc zapisujemy ją od razu.

```
curl -H "Authorization: Bearer nbt_KluczTokenu.TokenAPI"
https://netbox.zielona.example/api/dcim/devices/
```

Starszy token v1 w nagłówku `Authorization: Token TokenAPI` w NetBoksie 4.x nadal działa, ale od wersji 4.6 jest przestarzały i ma zniknąć w wersji 5.0, więc nowych tokenów w tym formacie nie zakładamy. Szczegóły tokenów opisuje strona Porządek i dokumentacja: patchpanele, etykiety, kolory i NetBox.

Przykład:

```
{"count": 2, "next": null, "previous": null, "results": [...]}
```

Wycofanie zmian

```
cd /docker/compose/netbox
sudo docker compose down
```

Pełne usunięcie razem z danymi, tylko gdy naprawdę zaczynamy od zera:

```
sudo docker compose down -v
sudo rm -rf /docker/apps/netbox
```

Gdy coś nie zadziała

Objaw	Przyczyna	Co robimy
Kontener `netbox` restartuje się w pętli zaraz po starcie	brak albo pusty `SECRET_KEY` w override	uzupełniamy `SECRET_KEY` wygenerowany osobno dla tej instalacji
`docker compose ps` pokazuje `netbox-worker` w innej wersji obrazu niż `netbox`	override nadpisał obraz tylko dla usługi `netbox`, nie dla `netbox-worker` i `netbox-housekeeping`	dopisujemy ten sam tag obrazu dla wszystkich trzech usług
Strona logowania wczytuje się bez stylów CSS	reverse proxy nie przekazuje poprawnie żądań do plików statycznych	sprawdzamy konfigurację Nginksa dla `netbox.zielona.example`, w szczególności nagłówki proxy
`password authentication failed for user netbox` (wariant b)	hasło w override inne niż w bazie na `pg01`	poprawiamy hasło w override albo w bazie przez `ALTER USER netbox WITH PASSWORD`

□□ Ćwiczenia

Ćwiczenie 1: kolejność wprowadzania pierwszych danych

Mamy już założoną lokalizację Pamiątkowo, ale nie mamy jeszcze żadnej szafy ani urządzenia. Chcemy dodać `fw01` do szafy, której jeszcze nie ma w systemie. Ustalamy kolejność kroków.

Odpowiedź

Najpierw zakładamy szafę (rack) przypisaną do lokalizacji Pamiątkowo, dopiero potem dodajemy urządzenie `fw01` wskazując tę już istniejącą szafę i konkretną pozycję w niej. Próba dodania urządzenia przed założeniem szafy nie ma się do czego odnieść, bo formularz urządzenia wymaga wskazania istniejącej szafy.

Powiązane: NetBox · NetBox: źródło prawdy o sieci i serwerowni · Reverse proxy i równoważenie obciążenia · NetBox: logowanie z AD, kopia i aktualizacja

Źródła: github.com/netbox-community/netbox-docker, plik `README.md` i `docker-compose.yml` (gałąź `release`) · github.com/netbox-community/netbox-docker/wiki, strona „Version"

Revision #23
Created 2026-09-28 17:15:38 UTC by Tomasz Konieczny
Updated 2026-10-26 14:14:45 UTC by Tomasz Konieczny