Instalacja Jenkinsa w Dockerze

Typ: procedura

Dotyczy: Jenkins LTS 2.568.3, obraz jenkins/jenkins:2.568.3-lts-jdk21 (sprawdzone na hub.docker.com dnia 2026-09-28)

Czas: ~30 min

Efekt: kontroler Jenkinsa i jeden agent działające w osobnych kontenerach na `ci01`, wystawione przez reverse proxy pod `jenkins.zielona.example`, gotowe do dodania kolejnych agentów i pierwszego pipeline'u.

Skróty użyte na tej stronie (10)

Skrót	Co znaczy
CI	Continuous Integration: automatyczne budowanie i testowanie kodu po każdej zmianie; w ITIL także configuration item, czyli element konfiguracji: każdy składnik usługi zapisany w CMDB
HTTP	Hypertext Transfer Protocol: protokół stron WWW
IP	Internet Protocol: adres komputera w sieci; w elektryce: stopień ochrony obudowy (Ingress Protection)
JNLP	metoda podłączenia agenta Jenkinsa, w której to agent inicjuje połączenie do kontrolera
LTS	Long Term Support: wersja z długim wsparciem poprawkami
README	Read Me: plik tekstowy z opisem projektu, wykresu lub szablonu, czytany jako pierwszy
RPM	RPM Package Manager: format paczek i narzędzie instalujące oprogramowanie w RHEL/Rocky, dnf korzysta z niego pod spodem
SSH	Secure Shell: szyfrowane zdalne logowanie do serwera
WebSocket	stałe, dwukierunkowe połączenie przeglądarki z serwerem, przez które Home Assistant na żywo odświeża panele; serwer pośredniczący musi przekazywać nagłówki Upgrade i Connection
WWW	World Wide Web: strony i aplikacje otwierane w przeglądarce po HTTP albo HTTPS

Zanim zaczniemy

Z ŻYCIA WZIĘTE

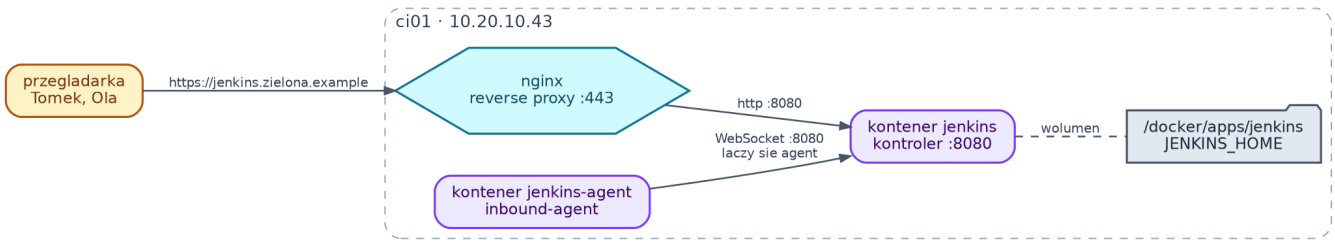
Ola prosi o osobne środowisko CI dla nowego mikroserwisu, żeby jej zespół nie kolejkował się za buildami reszty firmy na wspólnym `ci01`. Zamiast kolejnego agenta SSH stawiamy kontener Jenkinsa obok istniejącej instalacji RPM, żeby sprawdzić migrację do Dockera bez ryzyka dla produkcyjnego kontrolera. Po podłączeniu agenta przez WebSocket build startuje bez otwierania dodatkowego portu na firewallu `ci01`.

Strona Instalacja i aktualizacja Jenkinsa opisuje instalację z pakietu RPM, jako usługę systemd, bez kontenerów. Tutaj stawiamy to samo w Dockerze: dwa kontenery, kontroler i agent, każdy z osobnego obrazu.

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.

Wymaganie	Uwaga
Docker Engine i wtyczka Compose	sprawdzamy `docker compose version`
Wolny port `8080` na hoście	jedyny port, który wystawiamy na zewnątrz kontenera
DNS `jenkins.zielona.example` wskazujący na `ci01`	pod tym adresem stanie reverse proxy
Reverse proxy nginx przed kontenerem	z książki Docker, terminuje TLS tak samo jak dla innych aplikacji na `docker01`

Uważaj: obraz zawsze przypinamy do konkretnej wersji (`2.568.3-lts-jdk21`), nigdy do `lts` czy `latest`. Aktualizacja to świadoma zmiana numeru w `compose.yaml`, opisana na stronie Utrzymanie i diagnostyka, nie przypadkowe pobranie nowszej wersji przy kolejnym `docker compose pull`.



Kontener nginx przyjmuje ruch z przeglądarki, przekazuje go do kontenera jenkins, a agent łączy się z kontrolerem przez WebSocket na tym samym porcie 8080 (kliknięcie otwiera pełny rozmiar)

Przebieg

1. Katalogi zgodnie z konwencją

```
sudo mkdir -p /docker/apps/jenkins /docker/apps/jenkins-agent
sudo mkdir -p /docker/compose/jenkins
```

2. Plik .env

Plik `/docker/compose/jenkins/.env`, obok `compose.yaml`, nie w nim samym. Wartość `JENKINS_AGENT_SECRET` na razie zostawiamy pustym placeholderem, uzupełnimy ją w kroku 6, dopiero po utworzeniu node'a w samym Jenkinsie:

```
JENKINS_AGENT_NAME=agent-docker01
JENKINS_AGENT_SECRET=UzupelnimyPoUtworzeniuNodea
```

3. Plik compose.yaml

```
services:
  jenkins:
    image: jenkins/jenkins:2.568.3-lts-jdk21
    container_name: jenkins
    restart: always
    ports:
      - "8080:8080"
    volumes:
      - /docker/apps/jenkins:/var/jenkins_home
      - /etc/timezone:/etc/timezone:ro
      - /etc/localtime:/etc/localtime:ro

  jenkins-agent:
    image: jenkins/inbound-agent:3391.va_37fa_a_305d6d-3-jdk21
    container_name: jenkins-agent
    restart: always
    depends_on:
      - jenkins
```

```
    environment:
      - JENKINS_URL=http://jenkins:8080/
      - JENKINS_AGENT_NAME=${JENKINS_AGENT_NAME}
      - JENKINS_SECRET=${JENKINS_AGENT_SECRET}
      - JENKINS_WEB_SOCKET=true
    volumes:
      - /docker/apps/jenkins-agent:/home/jenkins/agent
```

> **Pojęcie: WebSocket.** Sposób komunikacji sieciowej, w którym po jednym początkowym żądaniu HTTP obie strony trzymają otwarte to samo połączenie i wysyłają nim dane w obie strony, bez kolejnych osobnych żądań. Dzięki `JENKINS_WEB_SOCKET=true` agent łączy się z kontrolerem tym samym portem `8080`, którego i tak już używa interfejs WWW: nie trzeba wystawiać osobnego portu `50000` na sieć kontenerów ani na firewall hosta, tak jak przy klasycznym połączeniu JNLP opisanym na stronie Architektura Jenkinsa: kontroler, agenci, workspace.

Agent nie ma tu żadnego portu w sekcji `ports`, bo łączy się z kontrolerem sam, poziomo, przez wewnętrzną sieć Dockera utworzoną automatycznie dla tego pliku Compose, adresując go po nazwie usługi `jenkins`, nie po adresie IP.

4. Pierwsze uruchomienie samego kontrolera

Startujemy na razie tylko kontroler, agent poczeka, aż będzie miał czym się połączyć:

```
cd /docker/compose/jenkins
docker compose up -d jenkins
```

5. Odblokowanie instalacji i wtyczki

Hasło startowe wyciągamy z wnętrza kontenera zamiast z dysku hosta:

```
docker exec jenkins cat /var/jenkins_home/secrets/initialAdminPassword
```

Przykład:

```
7f2c4b91ae3d4058b6e1c9a2f0d7883c
```

Dalej kreator, wtyczki sugerowane i pierwszy administrator wyglądają dokładnie tak samo, jak opisuje krok 5 i 6 strony Instalacja i aktualizacja Jenkinsa, niezależnie od tego, czy kontroler stoi w kontenerze, czy z pakietu RPM.

6. Utworzenie node'a agenta i sekret połączenia

W Manage Jenkins → Nodes → New Node zakładamy node typu „Permanent Agent" o nazwie `agent-docker01`, dokładnie takiej, jaką wpisaliśmy w `.env` pod `JENKINS_AGENT_NAME`. W polu „Launch method" wybieramy „Launch agent by connecting it to the controller" (inbound). Po zapisaniu Jenkins pokazuje na stronie node'a wygenerowany sekret jednorazowego użytku.

Wklejamy ten sekret do `/docker/compose/jenkins/.env` w miejsce placeholdera:

```
JENKINS_AGENT_NAME=agent-docker01
JENKINS_AGENT_SECRET=a1b2c3d4e5f60718293a4b5c6d7e8f90
```

7. Uruchomienie agenta

```
docker compose up -d jenkins-agent
```

Sprawdzamy log startu:

```
docker compose logs -f jenkins-agent
```

Przykład:

```
jenkins-agent  | INFO: Connecting to http://jenkins:8080/ via WebSocket
jenkins-agent  | INFO: Remoting version: 3391.v...
jenkins-agent  | INFO: Agent successfully connected and online
```

8. Reverse proxy

Fragment konfiguracji nginksa na `ci01`, według wzoru z książki Docker, z tą samą zasadą co przy instalacji bez kontenerów: bez nagłówka `X-Forwarded-Proto` Jenkins generuje linki z `http://` zamiast `https://`. Przy WebSocket dochodzi jeden dodatkowy wymóg, nagłówki `Upgrade`/`Connection`, inaczej agent nie połączy się przez proxy:

```
server {
    listen 443 ssl;
    server_name jenkins.zielona.example;

    location / {
        proxy_pass http://127.0.0.1:8080;
        proxy_http_version 1.1;
        proxy_set_header Upgrade $http_upgrade;
        proxy_set_header Connection "upgrade";
        proxy_set_header Host $host;
        proxy_set_header X-Real-IP $remote_addr;
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
        proxy_set_header X-Forwarded-Proto $scheme;
    }
}
```

Weryfikacja

```
docker compose ps
```

Przykład:

```
NAME            IMAGE                                              STATUS
jenkins         jenkins/jenkins:2.568.3-lts-jdk21                  Up 3 minutes
jenkins-agent   jenkins/inbound-agent:3391.va_37fa_a_305d6d-3-jdk21   Up 1 minute
```

```
curl -s -o /dev/null -w '%{http_code}\n' http://127.0.0.1:8080/login
```

Przykład:

```
200
```

Status agenta sprawdzamy też z poziomu Manage Jenkins → Nodes: `agent-docker01` powinien świecić się na zielono, bez czerwonego wykrzyknika offline.

□□ ZRZUT EKRANU DO UZUPEŁNIENIA

Zrzut do uzupełnienia: ekran Manage Jenkins → Nodes z listą node'ów, widoczny wiersz `agent-docker01` ze statusem online (zielona ikonka) i liczbą executorów.

Wycofanie zmian

Zatrzymanie i usunięcie kontenerów, bez kasowania danych z wolumenów:

```
docker compose down
```

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

```
docker compose down -v
sudo rm -rf /docker/apps/jenkins /docker/apps/jenkins-agent
```

Gdy coś nie zadziała

Objaw	Przyczyna	Naprawa
Kontener `jenkins` restartuje się w pętli zaraz po starcie	katalog `/docker/apps/jenkins` należy do roota, a proces w obrazie działa jako użytkownik o UID 1000	`sudo chown -R 1000:1000 /docker/apps/jenkins` przed kolejną próbą startu
`jenkins-agent` w logu pokazuje `Failed to connect` w kółko	zły `JENKINS_AGENT_SECRET` (placeholder nieuzupełniony albo sekret z innego node'a)	wracamy do widoku node'a w Manage Jenkins → Nodes i kopiujemy sekret jeszcze raz, dokładnie
Agent łączy się bez proxy, ale nie przez `jenkins.zielona.example` z zewnątrz	reverse proxy bez nagłówków `Upgrade`/`Connection` dla WebSocket	dokładamy `proxy_http_version 1.1` i oba nagłówki z kroku 8
Po restarcie kontenera znikają zainstalowane wtyczki	wolumen `/docker/apps/jenkins` nie był podpięty w `compose.yaml` albo ścieżka się nie zgadza	sprawdzamy `docker inspect jenkins` w sekcji `Mounts`, poprawiamy wpis `volumes`
Logowanie po reverse proxy przekierowuje w pętli	brak nagłówka `X-Forwarded-Proto`	dokładamy nagłówek w konfiguracji nginksa i ustawiamy właściwy adres URL w Manage Jenkins → System

Powiązane: Instalacja i aktualizacja Jenkinsa · Architektura Jenkinsa: kontroler, agenci, workspace · Konwencja Dockera w Zielonej

Źródła: jenkins.io/download/lts · docs.jenkins.io, rozdział „Installing Jenkins with Docker" · hub.docker.com/r/jenkins/jenkins · github.com/jenkinsci/docker-inbound-agent, plik README (zmienne `JENKINS_WEB_SOCKET`, `JENKINS_SECRET`)

Revision #23
Created 2026-09-28 16:57:45 UTC by Tomasz Konieczny
Updated 2026-10-26 14:15:02 UTC by Tomasz Konieczny