Gitea: częste problemy

Poziom: operacyjny **Dotyczy: Gitea 1.27, Gitea Runner 1.0.0, Nginx jako reverse proxy**

Czas: ~15 min

Zestawienie objawów, na które zespół IT trafia najczęściej przy pracy z Giteą: logowanie, klonowanie, Actions i rejestr pakietów. Tabela niżej daje szybką odpowiedź, rozwinięcie pod nią tłumaczy przyczynę do końca.

Skróty użyte na tej stronie (1)

Skrót	Co znaczy
IT	Information Technology: dział albo praca związana z komputerami i infrastrukturą firmy

Fiszka

TYLKO ODCZYT

Sprawdzenie logu kontenera Gitei: docker compose logs

```
docker compose -f /docker/compose/gitea/compose.yaml logs --tail 100 gitea
```

Pierwsze miejsce diagnozy niemal każdego z objawów niżej.

docker = narzędzie do kontenerów

KLIKAMY W INTERFEJSIE

Sprawdzenie, czy runner żyje i jest zarejestrowany: Site Administration → Actions → Runners

```
Site Administration → Actions → Runners
```

Zielona kropka i status „Idle" albo „Active" przy nazwie runnera znaczy, że łączność z instancją działa. Runner widoczny, ale wyszarzony ze statusem „Offline" znaczy, że kontener

runnera nie żyje albo stracił łączność sieciową.

TYLKO ODCZYT

Sprawdzenie aktualnego ROOT_URL: grep w app.ini

```
docker exec gitea grep ROOT_URL /data/gitea/conf/app.ini
```

Przykład:

```
ROOT_URL = http://localhost:3000/
```

docker = narzędzie do kontenerów

Częste błędy

Objaw	Przyczyna	Co robimy
Klonowanie po SSH pyta o hasło zamiast użyć klucza	zły port (Gitea w Dockerze nasłuchuje SSH na 2222, nie 22) albo klucz publiczny nieprzypięty do konta	w URL klona wskazujemy port `ssh://git@git.zielona.example:2222/...`, sprawdzamy klucz w profilu użytkownika
Błąd `SSL certificate problem: unable to get local issuer certificate` przy `git clone` / `push` na Windows	Git na Windows korzysta z własnego magazynu certyfikatów (OpenSSL), nie z magazynu systemu Windows	`git config --global http.sslbackend schannel`, patrz rozwinięcie niżej
Linki w powiadomieniach e-mail i w ustawieniach webhooków prowadzą na `localhost:3000`	`ROOT_URL` w `app.ini` nie zgadza się z rzeczywistym adresem instancji	ustawiamy `ROOT_URL = https://git.zielona.example/` w `app.ini`, restart kontenera
Push dużego pliku kończy się błędem `413 Request Entity Too Large`	domyślny limit rozmiaru zapytania w Nginx przed Giteą	podnosimy `client_max_body_size` w konfiguracji reverse proxy, patrz rozwinięcie niżej
Runner widoczny jako „Offline" w Site Administration	kontener runnera zatrzymany albo bez łączności sieciowej do instancji	`docker compose ps` i `docker compose logs gitea-runner` w `/docker/compose/gitea-runner/`
Workflow wisi w statusie „Waiting" bez końca	brak runnera obsługującego etykietę z pola `runs-on` danego zadania	Site Administration → Actions → Runners, porównujemy etykiety z tym, czego żąda workflow
`docker login git.zielona.example` odrzucony mimo poprawnego hasła konta	token użyty jako hasło bez uprawnienia `write:package`	generujemy nowy token z zaznaczonym zakresem pakietów

Objaw	Przyczyna	Co robimy
Użytkownik z konta w Active Directory nie może się zalogować do Gitei	filtr wyszukiwania LDAP nie obejmuje konta, konto poza wymaganą grupą albo zablokowane w AD	sprawdzamy kolejno: `--user-filter` źródła LDAP, członkostwo w grupie, stan konta w Active Directory
Po aktualizacji Gitea zgłasza błąd migracji i nie startuje	migracja przerwana w połowie albo niekompatybilna zmiana wymagająca kroku pośredniego, pominiętego przy skoku o kilka wersji naraz	odtwarzamy z kopii sprzed aktualizacji, patrz Kopia zapasowa i aktualizacja Gitea

Błąd certyfikatu SSL przy klonowaniu na Windowsie

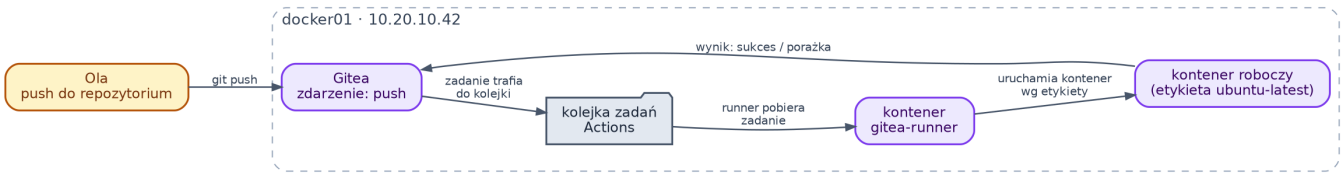
Przyczyna. Domyślny mechanizm weryfikacji certyfikatów w Gicie na Windows korzysta z własnego, dostarczanego razem z Gitem magazynu zaufanych certyfikatów, opartego o OpenSSL, zamiast z systemowego magazynu certyfikatów Windows. Certyfikat wewnętrznego urzędu certyfikacji, zaufany przez system i przez przeglądarkę na tej samej stacji, nie musi być automatycznie zaufany przez ten wbudowany magazyn Gita, stąd błąd mimo pozornie poprawnej konfiguracji serwera.

```
git config --global http.sslbackend schannel
```

Po tej zmianie Git na Windows korzysta z tego samego magazynu zaufanych certyfikatów co system operacyjny, więc certyfikat wewnętrznego urzędu certyfikacji rozpoznawany przez system jest rozpoznawany też przez Gita.

Runner offline albo workflow czeka w kolejce

Oba objawy dotyczą tego samego mechanizmu: zdarzenie trafia do kolejki zadań, ale nic go z niej nie odbiera. Przypominamy sobie drogę, którą normalnie przechodzi zadanie, zanim zaczniemy szukać, na którym odcinku utknęło:



 *Push trafia do kolejki zadań, runner pobiera zadanie i uruchamia kontener roboczy, wynik wraca do Gitei (kliknięcie otwiera pełny rozmiar)*

„Offline" znaczy, że nie żyje ogniwo `runner`: kontener zatrzymany albo bez łączności do instancji. „Waiting" bez runnera offline znaczy, że runner żyje, ale żadna z jego etykiet nie pasuje do tego, czego żąda zadanie w polu `runs-on`, więc kolejka stoi w miejscu tak, jakby runnera w ogóle nie było.

413 przy pushu dużego pliku

Reverse proxy przed Giteą (Nginx) domyślnie odrzuca zapytania powyżej 1 MB, zanim w ogóle dotrą do samej Gitei. Push repozytorium z dużym plikiem binarnym uderza w ten limit, nie w żadne ustawienie samej Gitei.

```
server {
    client_max_body_size 512M;
    ...
}
```

Podnosimy limit w bloku `server` albo `location` obsługującym `git.zielona.example`, przeładowujemy Nginx. Wartość dobieramy do realnych potrzeb repozytoriów, nie ustawiamy od razu bez limitu: bardzo duże pliki binarne w repozytorium Git to zwykle sygnał, że to repozytorium powinno raczej trzymać je w rejestrze pakietów albo w osobnym magazynie plików, nie w historii commitów.

 Ćwiczenia

> Zadania sprawdzające.

1. Handlowiec zgłasza, że link z powiadomienia e-mail od Gitei prowadzi na `localhost:3000` zamiast na adres firmowy. Wskazujemy, co sprawdzić w pierwszej kolejności.

Odpowiedź

Wartość `ROOT_URL` w sekcji `[server]` pliku `app.ini`. Jeśli nie zgadza się z rzeczywistym adresem instancji (`https://git.zielona.example/`), wszystkie generowane linki, nie tylko w e-mailach, wskazują zły adres.

1. Workflow czeka w statusie „Waiting" od kwadransa, w logu runnera cisza. Wymieniamy dwa miejsca, które sprawdzamy, i w jakiej kolejności.

Odpowiedź

Najpierw Site Administration → Actions → Runners: czy jakikolwiek runner jest w ogóle zarejestrowany i widoczny jako aktywny. Jeśli tak, porównujemy jego etykiety z etykietą w polu `runs-on` workflow: cisza w logu runnera zwykle znaczy, że żadna z jego etykiet nie pasuje, więc runner nawet nie próbuje odebrać zadania.

Powiązane: Gitea Actions: runner i pierwszy workflow · Pierwszy kontener: obraz budowany w Gitea i wdrożony na docker01 · Kopia zapasowa i aktualizacja Gitea · Logowanie kontem z Active Directory · Instalacja Gitea w Dockerze

Źródła: dokumentacja Gitea, rozdział „Installation" i „Command Line" · `git help config` (sekcja `http.sslbackend`) · dokumentacja Nginx, dyrektywa `client_max_body_size`

Revision #33
Created 2026-09-27 10:32:15 UTC by Tomasz Konieczny
Updated 2026-10-26 14:17:38 UTC by Tomasz Konieczny