postgresql.conf i pg_hba.conf

Poziom: podstawy **Dotyczy: PostgreSQL 9.6–17** **Czas: ~20 min**

Dwa pliki tekstowe decydują o wszystkim, co instancja robi i kogo do siebie wpuszcza — ta strona pokazuje, gdzie ich szukać, co w nich zmieniamy najczęściej i kiedy zmiana wymaga restartu, a kiedy wystarczy przeładowanie.

Skróty użyte na tej stronie (4)

Skrót	Co znaczy
SIGHUP	sygnał wysyłany, gdy zamykamy terminal, w którym proces wystartował, dawniej oznaczał rozłączenie linii telefonicznej
SQL	Structured Query Language: język zapytań do relacyjnych baz danych
TCP	Transmission Control Protocol: protokół sieciowy z potwierdzaniem dostarczenia
TLS	Transport Layer Security: szyfrowanie połączeń, np. w HTTPS

Fiszka

TYLKO ODCZYT

Sprawdzenie ścieżki pliku postgresql.conf: SHOW config_file

```
SHOW config_file;
```

Pytamy działającą instancję wprost o ścieżkę do jej własnego pliku konfiguracyjnego — pewniejsze niż zgadywanie na podstawie dystrybucji, bo instancja mogła zostać zainstalowana z niestandardowym `PGDATA`, tematem strony Instalacja PostgreSQL z własnym katalogiem danych.

Przykład:

```
               config_file
-----------------------------------------
 /postgres/15/data/postgresql.conf
(1 row)
```

show = FortiOS: wypisuje konfigurację, w podstawowej postaci tylko ustawienia różniące się od domyślnych, a z wariantem full-configuration także wartości domyślne

TYLKO ODCZYT

Wyświetlenie wszystkich parametrów konfiguracyjnych: SHOW ALL

```
SHOW ALL;
```

Zwracamy komplet parametrów wraz z bieżącą wartością i krótkim opisem — wygodne do przeszukania, gdy szukamy parametru po fragmencie nazwy.

Przykład:

```
              name               | setting |                        description
---------------------------------+---------
+------------------------------------------------------------
 max_connections                 | 100     | Sets the maximum number of concurrent
connections.
 shared_buffers                  | 128MB   | Sets the number of shared memory buffers
used by the server.
 work_mem                        | 4MB     | Sets the maximum memory to be used for
query workspaces.
(3 rows)
```

Równoważny wynik, w formie do dalszego przetwarzania w SQL, zwraca zapytanie do widoku katalogowego:

```
SELECT * FROM pg_settings;
```

SQL = Structured Query Language: język zapytań do relacyjnych baz danych · **show** = FortiOS: wypisuje konfigurację, w podstawowej postaci tylko ustawienia różniące się od domyślnych, a z wariantem full-configuration także wartości domyślne

TYLKO ODCZYT

Sprawdzenie wartości jednego parametru: current_setting()

```
SELECT current_setting('max_connections');
```

Pytamy o wartość dokładnie jednego parametru — czytelniejsze w skryptach niż filtrowanie wyniku `SHOW ALL`.

Przykład:

```
 current_setting
-----------------
 100
(1 row)
```

□□ TYLKO ODCZYT

Przeładowanie konfiguracji bez restartu: pg_reload_conf()

```
SELECT pg_reload_conf();
```

Nakazujemy instancji ponownie wczytać `postgresql.conf` i `pg_hba.conf` bez zrywania istniejących połączeń — działa dla większości parametrów, ale nie dla wszystkich, co opisuje tabela niżej. Odpowiednikiem z poziomu systemu operacyjnego, bez logowania do bazy, jest wysłanie sygnału `SIGHUP` do procesu postmastera albo komenda serwisowa:

```
sudo systemctl reload postgresql-15.service
```

Przykład:

```
 pg_reload_conf
----------------
 t
(1 row)
```

□□ **SIGHUP** = sygnał wysyłany, gdy zamykamy terminal, w którym proces wystartował, dawniej oznaczał rozłączenie linii telefonicznej · **systemctl** = system control: sterowanie usługami systemd

Z życia wzięte

Z ŻYCIA WZIĘTE

Zabbix na pg01 zgłasza coraz więcej odrzuconych połączeń tuż przed zamknięciem miesiąca, więc podnosimy `max_connections` w `postgresql.conf`. Przeładowanie sygnałem SIGHUP nic nie zmienia, bo ten konkretny parametr wymaga pełnego restartu instancji. Restart planujemy na wieczór, żeby nie przerywać pani Halinie pracy w księgowości w środku dnia.

Czym jest postgresql.conf

Pojęcie: postgresql.conf. Główny plik konfiguracyjny instancji, leżący w katalogu `PGDATA` opisanym na stronie Architektura PostgreSQL: klaster, baza, schemat, procesy. Każdy parametr to jedna linia `nazwa = wartosc`, z komentarzem `#` opisującym jednostkę i zakres. Plik jest wspólny dla całego klastra — dotyczy wszystkich baz w środku, nie da się mieć osobnego `postgresql.conf` dla jednej bazy (choć część parametrów da się nadpisać na poziomie pojedynczej bazy albo roli komendą `ALTER DATABASE ... SET` czy `ALTER ROLE ... SET`).

Najważniejsze parametry postgresql.conf

Poniższa tabela nie jest pełną listą — `SHOW ALL` zwraca kilkaset pozycji — tylko zestawem, który realnie zmieniamy w codziennej eksploatacji.

Parametr	Do czego służy	Wymaga restartu
`listen_addresses`	interfejsy sieciowe, na których instancja przyjmuje połączenia TCP	tak
`port`	port TCP nasłuchu	tak
`max_connections`	maksymalna liczba jednoczesnych połączeń klienckich	tak
`shared_buffers`	pamięć dzielona na bufor stron danych, zwykle 25% RAM serwera	tak
`work_mem`	pamięć na operację sortowania i haszowania w jednym zapytaniu	nie

Parametr	Do czego służy	Wymaga restartu
`maintenance_work_mem`	pamięć dla operacji utrzymaniowych — `VACUUM`, `CREATE INDEX`, `ALTER TABLE`	nie
`wal_level`	ilość informacji zapisywanej do WAL — `minimal`, `replica`, `logical`	tak
`archive_mode`	włączenie archiwizacji segmentów WAL	tak
`archive_command`	polecenie kopiujące zamknięty segment WAL do archiwum	nie
`checkpoint_timeout`	maksymalny odstęp czasu między checkpointami	nie
`log_min_duration_statement`	próg czasu wykonania, od którego zapytanie trafia do logu	nie
`log_destination`, `log_directory`, `log_filename`	dokąd i w jakich plikach trafiają logi serwera	częściowo (zależnie od parametru)

Uważaj: `work_mem` jest limitem na jedną operację sortującą albo haszującą wewnątrz jednego zapytania, nie na całe zapytanie ani na całą sesję — zapytanie z kilkoma sortowaniami i złączeniami hashowymi może w szczycie zużyć wielokrotność tej wartości, co jest częstym źródłem nieoczekiwanego zużycia pamięci przy zbyt optymistycznie podkręconym `work_mem`.

`shared_buffers`, `wal_level` i `max_connections` należą do parametrów, które zmieniają rozmiar segmentów pamięci dzielonej alokowanych przy starcie procesu — stąd wymóg restartu, nie samego przeładowania.

Czym jest pg_hba.conf

> **Pojęcie: pg_hba.conf.** Plik kontrolujący, kto i w jaki sposób może się uwierzytelnić w klastrze — leży w tym samym katalogu `PGDATA` co `postgresql.conf` i jest odczytywany jeszcze przed sprawdzeniem hasła. Nazwa pochodzi od „host-based authentication". Każda linia to jedna reguła, sprawdzana od góry pliku w dół — pierwsza pasująca reguła decyduje, żadna kolejna linia nie jest już brana pod uwagę, nawet jeśli pasowałaby lepiej.

Format linii to pięć kolumn: typ połączenia (`local` dla gniazda uniksowego, `host` dla TCP, `hostssl` dla TCP wymuszającego TLS), nazwa bazy, nazwa roli, adres klienta (pominięty dla `local`) i metoda uwierzytelniania.

Przykład fragmentu pliku, otwierającego dostęp replikacyjny i dostęp konta monitorującego z sieci lokalnej:

```
# TYPE  DATABASE        USER            ADDRESS         METHOD
local   all             postgres                        peer
host    replication     base_backup_user 127.0.0.1/32   trust
host    all             zbx_monitor     10.20.10.0/24   scram-sha-256
host    all             all             0.0.0.0/0       reject
```

Metody uwierzytelniania w pg_hba.conf

> **Pojęcie: metoda uwierzytelniania.** Sposób, w jaki instancja potwierdza tożsamość klienta pasującego do danej reguły — od całkowitego zaufania po hasło przesyłane w postaci nieodwracalnie zaszyfrowanej.

Metoda	Jak działa	Kiedy używamy
`trust`	wpuszcza bez żadnego sprawdzenia hasła	wyłącznie połączenia lokalne, zaufane, np. narzędzia utrzymaniowe na tym samym hoście — nigdy z sieci publicznej
`peer`	porównuje nazwę konta systemowego z nazwą roli bazodanowej	połączenia lokalne przez gniazdo uniksowe, typowo konto `postgres`
`ident`	jak `peer`, ale przez zewnętrzny serwer ident po sieci	połączenia TCP z zaufanej sieci, rzadko spotykane dziś
`md5`	hasło porównywane po zahaszowaniu MD5	starsze instalacje, kompatybilność wsteczna
`scram-sha-256`	hasło porównywane silniejszym mechanizmem SCRAM, domyślne od PostgreSQL 10+	zalecany standard dla połączeń hasłowych
`reject`	odrzuca połączenie bez sprawdzania czegokolwiek	reguła zamykająca plik, blokująca to, co nie pasowało do żadnej wcześniejszej, bardziej szczegółowej reguły

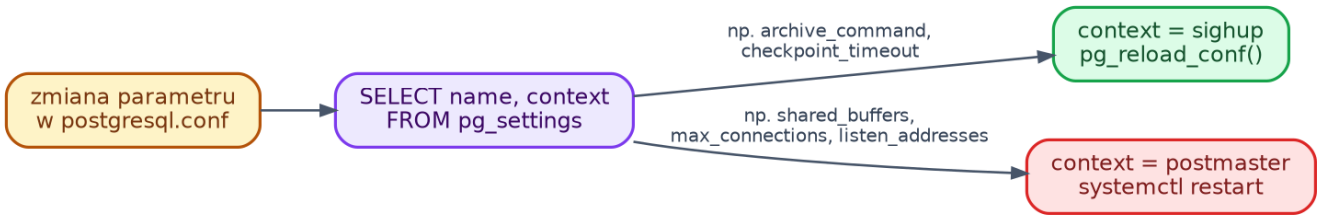
Uważaj: `trust` na regule pasującej do adresu szerszego niż pojedynczy host lokalny (`0.0.0.0/0` albo cała sieć firmowa) oznacza logowanie bez hasła dla każdego, kto potrafi nawiązać połączenie TCP do instancji — to jest błąd konfiguracyjny, nie skrót, nawet jeśli chwilowo wygodny przy pracach serwisowych, jak przy upgrade opisanym na stronie Upgrade silnika PostgreSQL.

Przeładowanie a restart

Zmiana w `pg_hba.conf` zawsze wystarcza przeładować — `pg_reload_conf()` albo `systemctl reload`, bez zrywania istniejących sesji. Zmiana w `postgresql.conf` wymaga przeładowania dla większości parametrów, ale nie dla tych związanych z rozmiarem pamięci dzielonej i topologią nasłuchu, wypisanych w tabeli wyżej — dla nich jedynym sposobem jest pełny restart usługi:

```
sudo systemctl restart postgresql-15.service
```

Sposobem na sprawdzenie z wyprzedzeniem, czy dany parametr wymaga restartu, jest kolumna `context` w `pg_settings`:



Kolumna context w pg_settings rozstrzyga, czy zmiana parametru wystarczy przeładować, czy wymaga pełnego restartu (kliknięcie otwiera pełny rozmiar)

```
SELECT name, context FROM pg_settings WHERE name = 'shared_buffers';
```

Przykład:

```
      name      |  context
----------------+------------
 shared_buffers  | postmaster
(1 row)
```

Wartość `postmaster` w kolumnie `context` oznacza, że parametr jest czytany tylko raz, przy starcie procesu nadzorczego — stąd konieczność restartu. Wartość `sighup` oznacza, że wystarczy przeładowanie sygnałem `SIGHUP`, czyli dokładnie to, co robi `pg_reload_conf()`.

Częste błędy

Edycja postgresql.conf i oczekiwanie efektu bez przeładowania. Sama zmiana pliku na dysku nic nie robi, dopóki instancja go nie wczyta ponownie — `pg_reload_conf()` albo restart, zależnie od parametru.

Zbyt szeroka reguła trust w pg_hba.conf zostawiona po pracach serwisowych. Reguła otwarta tymczasowo na czas upgrade czy migracji, o czym przypomina strona Upgrade silnika PostgreSQL, zapomniana i pozostawiona w produkcji, jest realną luką bezpieczeństwa, nie kosmetycznym niedopatrzeniem.

Powiązane: Architektura PostgreSQL: klaster, baza, schemat, procesy · Użytkownicy, role i uprawnienia · Logi WAL: włączenie i archiwizacja · Upgrade silnika PostgreSQL

Źródła: dokumentacja PostgreSQL, rozdział „Server Configuration" i „Client Authentication"

Revision #43
Created 2026-09-24 23:08:04 UTC by Tomasz Konieczny
Updated 2026-10-26 14:12:20 UTC by Tomasz Konieczny