Utrzymanie serwera Zabbix

Poziom: średnio

Dotyczy: Zabbix server 6.0 LTS, PostgreSQL/MySQL jako baza zapleczowa **Czas: ~25 min**

Instalacja, która działa poprawnie przez pierwszy rok, zaczyna spowalniać w drugim — prawie zawsze dlatego, że nikt nie dopilnował retencji i wielkości tabel historii.

Skróty użyte na tej stronie (3)

Skrót	Co znaczy
CPU	Central Processing Unit: procesor
LTS	Long Term Support: wersja z długim wsparciem poprawkami
RAM	Random Access Memory: pamięć operacyjna

Fiszka

TYLKO ODCZYT

Sprawdzenie liczby wierszy w tabelach historii (PostgreSQL): psql

```
sudo -u postgres psql -d zabbix -c "SELECT relname, n_live_tup FROM pg_stat_user_tables
WHERE relname LIKE 'history%' OR relname LIKE 'trends%' ORDER BY n_live_tup DESC;"
```

Przykład:

```
      relname       | n_live_tup
--------------------+------------
 history_uint       |  184201933
 history            |   92841207
 trends_uint        |    4128933
 trends             |    1802441
(4 rows)
```

□□ TYLKO ODCZYT

Podgląd rozmiaru kolejki: zabbix[queue]

```
zabbix[queue]
```

Podgląd kolejki jest dostępny we frontendzie (Raporty → Kolejka), a jej rozmiar liczbowo pobieramy elementem wewnętrznym.

□□ TYLKO ODCZYT

Sprawdzenie liczby procesów roboczych: grep

```
sudo grep -E '^Start(Pollers|Trappers|DBSyncers)' /etc/zabbix/zabbix_server.conf
```

Przykład:

```
StartPollers=10
StartTrappers=5
StartDBSyncers=4
```

□□ **grep** = global regular expression print: szukaj wzorca w tekście

□□ TYLKO ODCZYT

Wypisanie statystyk wewnętrznych serwera: zabbix_server -R diaginfo

```
sudo zabbix_server -R diaginfo
```

Wypisujemy do logu serwera bieżące statystyki wewnętrzne — wykorzystanie cache'u konfiguracji, cache'u wartości, liczbę elementów w kolejce per typ. Komenda nie zwraca wyniku bezpośrednio w terminalu, tylko dopisuje go do logu serwera.

□□ TYLKO ODCZYT

Podgląd wyniku diaginfo w logu: tail

```
sudo tail -n 40 /var/log/zabbix/zabbix_server.log
```

Przykład:

```
18204:20260922:093011.204 ====== history syncer #2 diagnostic information ======
18204:20260922:093011.204 history values processed: 1420318
18204:20260922:093011.204 values per second: 394.53
```

tail = pokaż koniec pliku, z flagą -f śledzi plik na żywo

ZMIENIA SYSTEM

Wykonanie kopii bazy przed aktualizacją: pg_dump

```
sudo -u postgres pg_dump zabbix > /var/backups/zabbix_przed_aktualizacja.sql
```

Zanim rozpoczynamy aktualizację, wykonujemy kopię bazy danych — sama aktualizacja pakietu serwera automatycznie migruje schemat bazy przy pierwszym starcie po aktualizacji, a ten proces jest jednokierunkowy i nie da się go cofnąć bez przywrócenia bazy z backupu.

ZMIENIA SYSTEM

Aktualizacja pakietów serwera i frontendu (RHEL/Rocky): dnf update

```
sudo dnf update zabbix-server-pgsql zabbix-web-pgsql zabbix-agent2
```

RHEL = Red Hat Enterprise Linux: komercyjna dystrybucja Linuksa · **dnf** = Dandified YUM: menedżer pakietów RHEL/Rocky

ZMIENIA SYSTEM

Restart usługi i podgląd logu migracji schematu: systemctl restart

```
sudo systemctl restart zabbix-server
sudo tail -f /var/log/zabbix/zabbix_server.log
```

Po aktualizacji sprawdzamy log serwera pod kątem migracji schematu bazy przy pierwszym starcie.

Przykład:

```
 21044:20260922:100211.117 current database version (mandatory/optional):
06000000/06000000
 21044:20260922:100211.117 required mandatory version: 06040000
 21044:20260922:100211.117 starting automatic database upgrade
 21044:20260922:100234.809 database upgrade fully completed
```

Uważaj: aktualizacja o więcej niż jedną główną wersję na raz (np. z 5.0 od razu do 6.4) nie jest wspierana wprost — sprawdzamy w dokumentacji wydania, czy wymagane jest przejście przez wersje pośrednie, zanim uruchomimy proces.

systemctl = system control: sterowanie usługami systemd · **tail** = pokaż koniec pliku, z flagą -f śledzi plik na żywo

Z życia wzięte

Z ŻYCIA WZIĘTE

W poniedziałek rano baza pod Zabbixem na `pg01` zapełnia dysk szybciej niż zwykle, a Tomek dostaje o 6:40 alarm o wolnym miejscu. Sprawdza `n_live_tup` na tabelach historii i widzi, że `history_uint` urosło dziesięciokrotnie od ostatniego kwartału, odkąd ktoś podłączył nowy szablon do stu hostów bez skrócenia interwałów. Skraca retencję historii dla mniej krytycznych elementów z 90 do 30 dni, co od razu spowalnia dalszy przyrost, i zapisuje w notatkach zespołu, żeby przy każdym nowym szablonie sprawdzać wpływ na rozmiar bazy.

Housekeeping i retencja

> **Pojęcie: housekeeping.** Wbudowany proces serwera Zabbixa, cyklicznie usuwający z bazy dane starsze niż skonfigurowany okres retencji — bez niego historia rosłaby bez końca, aż do wyczerpania miejsca na dysku pod bazę danych.

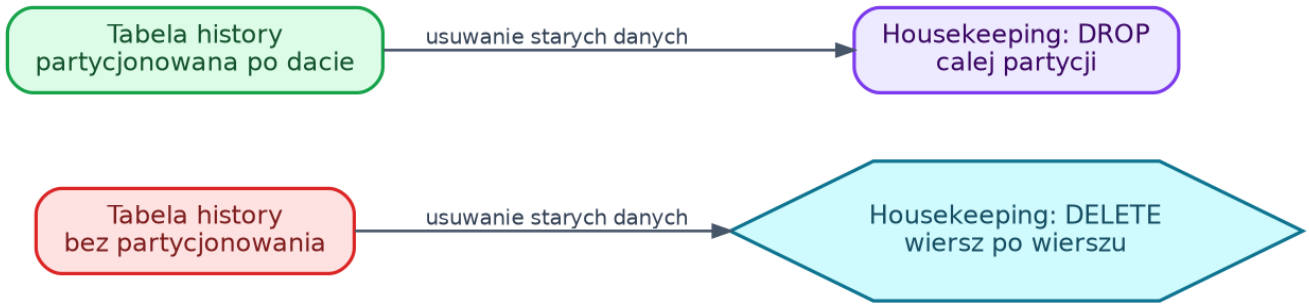
Historia kontra trendy. Retencję konfigurujemy osobno dla dwóch rodzajów danych, opisanych też przy architekturze na stronie Architektura Zabbixa: serwer, proxy, agent, frontend: **historia** to pojedyncze zebrane wartości z pełną rozdzielczością czasową, **trendy** to zagregowane dane godzinowe (min/max/avg), zajmujące ułamek miejsca historii. Typowa praktyka to krótka retencja historii (7–31 dni — wystarczająco długo, żeby zdiagnozować świeży incydent z dokładnością do pojedynczego sprawdzenia) i długa retencja trendów (kilka lat — wystarczająco, żeby porównać obciążenie z tym samym okresem rok wcześniej, już bez dokładności co do sekundy).

Retencję ustawiamy per element danych albo globalnie w konfiguracji housekeepingu — element o krytycznym znaczeniu diagnostycznym (np. log błędów aplikacji) trzymamy dłużej w pełnej historii niż standardowy element obciążenia CPU.

Uważaj: wyłączenie housekeepingu „na chwilę", żeby przyspieszyć bazę pod dużym obciążeniem, jest rozwiązaniem odwrotnym do zamierzonego — bez housekeepingu tabele historii rosną bez ograniczenia, więc obciążenie bazy z czasem tylko się pogłębia.

Partycjonowanie tabel history

> **Pojęcie: partycjonowanie tabeli.** Fizyczny podział jednej dużej tabeli na mniejsze fragmenty (partycje), zwykle po dacie, tak żeby operacja usunięcia starych danych była usunięciem całej partycji, a nie kasowaniem miliardów pojedynczych wierszy poleceniem `DELETE`. Housekeeping działający na niepartycjonowanej tabeli `history` liczącej setki milionów wierszy potrafi trwać godzinami i mocno obciążać bazę operacjami blokowania wierszy — na partycjonowanej tabeli to samo sprowadza się do `DROP` jednej partycji, operacji niemal natychmiastowej.



Housekeeping na tabeli bez partycji usuwa wiersz po wierszu poleceniem DELETE, a na tabeli partycjonowanej po dacie sprowadza się do DROP całej partycji (kliknięcie otwiera pełny rozmiar)

Przy instalacjach powyżej kilku milionów elementów danych wdrożenie natywnego partycjonowania PostgreSQL albo rozszerzenia TimescaleDB (obsługiwanego oficjalnie od Zabbixa 5.0) jest praktycznie obowiązkowe — bez niego housekeeping w pewnym momencie przestaje nadążać z usuwaniem danych szybciej, niż te dane przybywają.

Tryb konserwacji hosta

> **Pojęcie: tryb konserwacji (maintenance).** Okno czasowe zdefiniowane dla wybranych hostów albo grup hostów, w którym Zabbix albo całkowicie zawiesza zbieranie danych, albo (w wariancie „bez zbierania danych") zbiera je nadal, ale tłumi generowanie nowych zdarzeń i powiadomień. Stosujemy tryb konserwacji przed planowaną przerwą serwisową — restartem serwera, aktualizacją, migracją — żeby zaplanowana, świadoma niedostępność nie wygenerowała alarmu i nie obudziła nikogo z dyżuru.

Uważaj: tryb konserwacji ustawiony na zbyt długi okres i zapomniany po zakończeniu prac maskuje realne problemy powstałe już po zaplanowanej przerwie — dobra praktyka to zawsze węższe okno niż szacowany czas prac, z możliwością przedłużenia, zamiast szerokiego okna „na wszelki wypadek".

Kolejka — zabbix[queue]

> **Pojęcie: kolejka (queue).** Lista elementów danych oczekujących na zebranie, których termin sprawdzenia już minął, a serwer albo proxy jeszcze nie zdążył ich obsłużyć. Rosnąca kolejka jest pierwszym sygnałem, że serwer nie nadąża z bieżącym obciążeniem — więcej elementów danych albo krótsze interwały, niż faktycznie jest w stanie obsłużyć liczba skonfigurowanych procesów roboczych (`StartPollers`, `StartTrappers` i pokrewne parametry w `zabbix_server.conf`).

Diagnostyka wydajności serwera

> **Pojęcie: cache wartości (value cache).** Pamięć podręczna w RAM serwera, przechowująca ostatnio odczytane wartości historii, tak żeby ewaluacja wyzwalaczy odwołujących się do funkcji historycznych (`avg()`, `last()` na dłuższym oknie) nie wymagała każdorazowego zapytania do bazy danych. Zbyt mały `ValueCacheSize` w `zabbix_server.conf` objawia się rosnącym obciążeniem samej bazy przy dużej liczbie wyzwalaczy odwołujących się do dłuższych okresów historii.

Częste błędy

Retencja historii ustawiona globalnie i zbyt długo dla wszystkich elementów naraz. Trzymanie 90 dni pełnej historii dla elementu odpytywanego co 10 sekund mnoży rozmiar bazy

dziesięciokrotnie względem retencji 30-dniowej, bez realnej wartości diagnostycznej po pierwszym tygodniu.

Brak partycjonowania przy dużej skali. Instalacja, która wystartowała ze stu hostami i niepartycjonowaną bazą, po dwóch latach i dziesięciu tysiącach hostów ma housekeeping trwający dłużej niż odstęp między jego uruchomieniami — objawia się to permanentnie rosnącą bazą mimo skonfigurowanej krótkiej retencji.

Zapomniany tryb konserwacji. Okno konserwacji ustawione „na tydzień, żeby na pewno starczyło" i nieusunięte po zakończeniu prac maskuje realne alarmy przez cały pozostały czas jego trwania.

□□ Ćwiczenia

> Zadania sprawdzające.

Ćwiczenie 1 — diagnoza rosnącej kolejki

Kolejka Zabbixa (`zabbix[queue]`) rośnie od kilku dni, mimo że liczba hostów się nie zmieniła. Wskazujemy dwa najbardziej prawdopodobne kierunki dalszej diagnostyki.

Odpowiedź

Pierwszy kierunek to sprawdzenie, czy nie przybyło elementów danych albo nie skrócono interwałów (np. po podłączeniu nowego szablonu do istniejących hostów) bez odpowiedniego zwiększenia liczby procesów roboczych. Drugi kierunek to sprawdzenie obciążenia samej bazy danych — rosnąca kolejka bywa objawem tego, że baza nie nadąża z zapisem historii, a nie tego, że serwerowi brakuje mocy obliczeniowej na samo zbieranie.

Powiązane: Architektura Zabbixa: serwer, proxy, agent, frontend · Host, element danych, wyzwalacz, akcja: jak to się składa · Retencja logów

Źródła: dokumentacja Zabbix 6.0, rozdział „Housekeeper", „Maintenance" i „Upgrade procedure"

Revision #36
Created 2026-09-25 07:40:34 UTC by Tomasz Konieczny
Updated 2026-10-26 14:09:50 UTC by Tomasz Konieczny