Upgrade silnika bazy danych

Wprowadzenie:

> Aktualizacja silnika MongoDB między wersjami głównymi (np. 5.0 → 6.0) to kontrolowany proces w kilku krokach: najpierw upewniamy się, że poziom kompatybilności (FCV — Feature Compatibility Version) odpowiada wersji obecnej, potem podmieniamy silnik, a dopiero po sprawdzeniu, że nowa wersja działa, podnosimy FCV. Taka kolejność pozwala w razie problemów wrócić do poprzedniej wersji. Instrukcja opisuje **jeden skok** między sąsiednimi wersjami głównymi dla wdrożenia opartego na Docker Compose. Wariant baremetal różni się wyłącznie krokiem podmiany silnika — zaznaczono to w notkach. W przykładach `NAZWAKONTENERA` i `SERVICENAME` zastąp wartościami z własnego środowiska.

1. Zasada pojedynczego skoku

MongoDB pozwala aktualizować silnik tylko o **jedną wersję główną naraz**. Nie da się przeskoczyć np. z 5.0 od razu do 7.0 — trzeba przejść przez każdą wersję pośrednią:

```
5.0 → 6.0 → 7.0 → 8.0
```

Każdy skok to powtórzenie całej procedury opisanej poniżej. Dodatkowo FCV przed aktualizacją musi odpowiadać wersji obecnej — dlatego krok 1 to jego ustawienie/weryfikacja.

> **Najpierw kopia zapasowa.** Przed jakąkolwiek aktualizacją wykonaj `mongodump` (zob. instrukcja „Podstawowe komendy mongosh", sekcja 21). Aktualizacja zmienia format danych na dysku i bez backupu rollback bywa niemożliwy.

2. Krok 1 – ustawienie FCV na wersję obecną

Wejdź do powłoki mongosh w działającym kontenerze:

```
docker exec -it NAZWAKONTENERA mongosh
```

Sprawdź bieżący poziom kompatybilności:

```
db.adminCommand({ getParameter: 1, featureCompatibilityVersion: 1 })
```

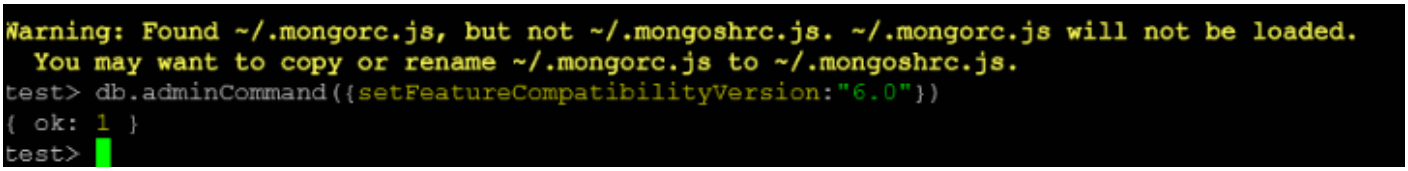
Następnie jawnie ustaw FCV na wersję, na której obecnie pracujesz (tu 5.0). To gwarantuje, że dane są w pełni zgodne z bieżącą wersją, zanim podmienisz silnik:

```
db.adminCommand({ setFeatureCompatibilityVersion: "5.0" })
```

Oczekiwany wynik:

```
{ "ok" : 1 }
```

> **Baremetal:** kroki FCV są identyczne — wchodzisz do powłoki po prostu poleceniem `mongosh` (bez `docker exec`).



3. Krok 2 – zmiana obrazu w `docker-compose.yaml`

W pliku `docker-compose.yaml` podnieś tag obrazu usługi bazy o jedną wersję główną:

```
SERVICENAME:
  image: mongo:6.0
  container_name: NAZWAKONTENERA
```

4. Krok 3 – restart kontenerów

```
docker compose down
docker compose pull
docker compose up -d
```

`down` zatrzymuje i usuwa kontenery (dane pozostają na woluminie), `pull` pobiera nowy obraz w zadeklarowanej wersji, a `up -d` uruchamia kontenery w tle na nowym silniku. Wolumin z danymi nie jest kasowany, więc baza startuje z dotychczasową zawartością.

> **Baremetal:** odpowiednikiem kroków 2–3 jest aktualizacja pakietu z repozytorium MongoDB. W tym samym miejscu procedury (po ustawieniu FCV na wersję obecną i wykonaniu kopii, a przed podniesieniem FCV) przełączasz repozytorium MongoDB na nową wersję główną, zatrzymujesz usługę bazy, instalujesz nowy pakiet i uruchamiasz usługę ponownie. Dane i konfiguracja na dysku pozostają nietknięte — podmieniasz wyłącznie binaria.

> **Screen:** log z `docker compose down` / `pull` / `up -d`.

5. Krok 4 – weryfikacja uruchomienia

Sprawdź, że kontenery wstały poprawnie:

```
docker compose ps
```

Następnie potwierdź, że baza działa już na nowej wersji silnika:

```
docker exec -it NAZWAKONTENERA mongosh --quiet --eval 'db.version()'
```

```
~$: sudo docker exec -it clickstack-db-1 mongosh --quiet --eval 'db.version()'
8.2.9
```

Na tym etapie warto pozwolić bazie popracować chwilę („burn-in"), aby upewnić się, że nowa wersja działa stabilnie — dopóki nie podniesiesz FCV, możesz jeszcze wrócić do poprzedniego silnika.

>

> **Baremetal:** stan sprawdzasz przez menedżer usług systemu zamiast `docker compose ps`; wersję — poleceniem `mongosh --quiet --eval 'db.version()'`.

> **Screen:** wynik `docker compose ps` ze statusem *Up*.

6. Krok 5 – podniesienie FCV na nową wersję

Gdy nowa wersja działa poprawnie, wejdź ponownie do powłoki i podnieś FCV do wersji docelowej — to odblokowuje funkcje nowej wersji:

```
db.adminCommand({ setFeatureCompatibilityVersion: "6.0" })
```

Po tej operacji aktualizacja pojedynczego skoku jest zakończona. Jeśli planujesz kolejny skok, wróć do kroku 1 z nowymi wartościami wersji.

> **Od MongoDB 7.0 wymagane jest pole `confirm: true`** — bez niego komenda zwróci błąd. Dla skoków do 7.0 i wyższych użyj formy:

```
db.adminCommand({ setFeatureCompatibilityVersion: "7.0", confirm: true })
```

> Przy skokach do 5.0 i 6.0 parametr `confirm` nie jest potrzebny (i na starszych binariach może zostać odrzucony jako nieznane pole).

> **Rollback:** aby cofnąć się do poprzedniej wersji silnika, najpierw obniż FCV do wersji wcześniejszej (`setFeatureCompatibilityVersion` ze starszą wartością), a dopiero potem podmień obraz / pakiet z powrotem. Starsza wersja nie wystartuje, jeśli FCV wskazuje wersję nowszą.

Podsumowanie praktyczne

Pełny cykl jednego skoku (na przykładzie 5.0 → 6.0):

```
# 1. FCV na wersję obecną (wewnątrz mongosh)
docker exec -it NAZWAKONTENERA mongosh --eval 'db.adminCommand({ setFeatureCompatibilityVersion: "5.0"
```

```
})'

# 2. docker-compose.yaml:  image: mongo:6.0
#    (baremetal: przełącz repozytorium MongoDB na 6.0 i zaktualizuj pakiet)

# 3. restart na nowy silnik
docker compose down
docker compose pull
docker compose up -d

# 4. weryfikacja
docker compose ps
docker exec -it NAZWAKONTENERA mongosh --quiet --eval 'db.version()'

# 5. FCV na wersję nową (confirm:true od 7.0 wzwyż)
docker exec -it NAZWAKONTENERA mongosh --eval 'db.adminCommand({ setFeatureCompatibilityVersion: "6.0"
})'
```

Revision #3
Created 9 June 2026 09:50:33 by Tomasz Konieczny
Updated 9 June 2026 10:12:42 by Tomasz Konieczny