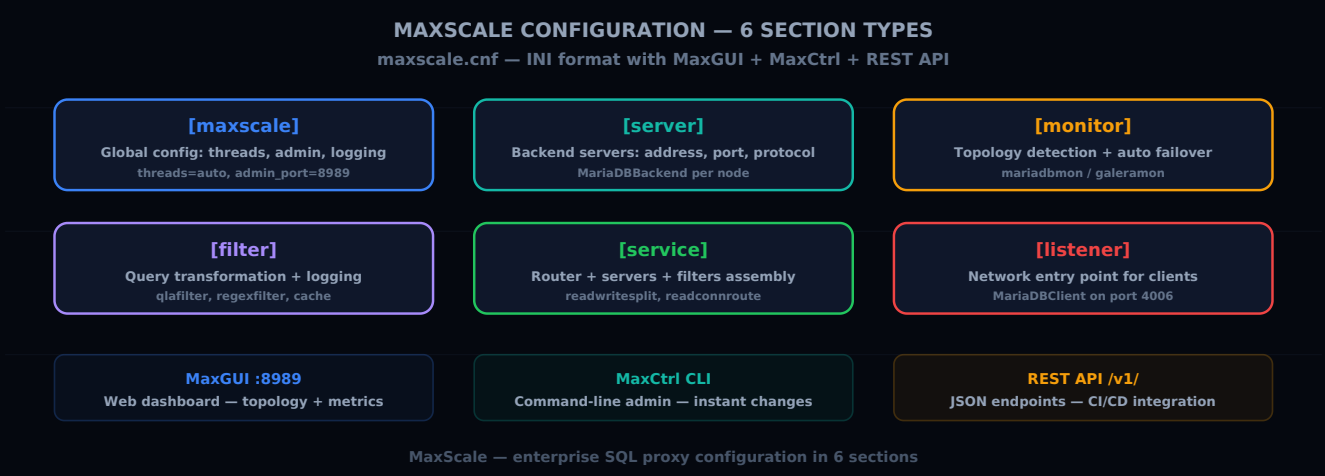
MaxScale: znacznie więcej niż reverse proxy SQL (część 2)

Sylvain ARBAUDIE · September 1, 2025

MAXSCALE MARIADB PROXY CONFIGURATION



Od teorii do praktyki

W pierwszej części zbadaliśmy możliwości architektoniczne MaxScale: jego protokoły, routery, monitory i filtry. Ta druga część to praktyczny przewodnik: jak zainstalować, skonfigurować i administrować MaxScale na co dzień.

Instalacja

MaxScale jest dostępny poprzez oficjalne repozytorium MariaDB. Na Debian/Ubuntu:

```
curl -LsS https://r.mariadb.com/downloads/mariadb_repo_setup \
  | sudo bash -s -- --mariadb-maxscale-version=24.02

sudo apt-get install maxscale
```

Główny plik konfiguracyjny znajduje się w `/etc/maxscale.cnf`. Jest to plik w formacie INI, zorganizowany w wyraźnie zdefiniowane sekcje.

Anatomia pliku konfiguracyjnego

Plik `maxscale.cnf` jest zorganizowany w sześć typów sekcji, z których każda jest identyfikowana nagłówkiem w nawiasach kwadratowych.

Sekcja [maxscale] — Konfiguracja globalna

```
[maxscale]
threads        = auto
log_info       = false
admin_host     = 0.0.0.0
admin_port     = 8989
admin_secure_gui = true
```

- **threads=auto**: MaxScale automatycznie wykrywa liczbę dostępnych rdzeni CPU.
- **admin_host/admin_port**: Aktywuje interfejs webowy MaxGUI i REST API.
- **admin_secure_gui**: Aktywuje HTTPS dla interfejsu administracyjnego.

Sekcja [server] — Definicja serwerów

Każdy serwer backend MariaDB / MySQL jest deklarowany indywidualnie:

```
[db-master]
type     = server
address  = 10.0.1.10
port     = 3306
protocol = MariaDBBackend

[db-slave1]
type     = server
address  = 10.0.1.11
port     = 3306
protocol = MariaDBBackend

[db-slave2]
type     = server
address  = 10.0.1.12
port     = 3306
protocol = MariaDBBackend
```

Każda sekcja `[server]` nosi nazwę logiczną (tutaj `db-master`, `db-slave1`, `db-slave2`), do której będą się odwoływać monitory i usługi.

Sekcja [monitor] — Nadzór topologii

Monitor jest komponentem, który automatycznie wykrywa topologię replikacji i stan każdego węzła:

```
[replication-monitor]
type            = monitor
module          = mariadbmon
servers         = db-master, db-slave1, db-slave2
user            = maxscale_monitor
password        = encrypted_password_here
monitor_interval = 2000
auto_failover    = true
auto_rejoin      = true
```

- **mariadbmon**: Standardowy monitor dla topologii master-slave MariaDB / MySQL.
- **auto_failover**: W przypadku utraty mastera, MaxScale automatycznie promuje slave'a.
- **auto_rejoin**: Były master, który wraca online, jest automatycznie rekonfigurowany jako slave.

Dla Galera należałoby użyć `galeramon` zamiast `mariadbmon`.

Sekcja [filter] — Transformacja zapytań

Filtry przechwytują i modyfikują strumień SQL:

```
[query-log]
type   = filter
module = qlafilter
filebase = /var/log/maxscale/queries
flush   = true

[regex-rewrite]
type    = filter
module  = regexfilter
match   = SELECT.*FROM\s+legacy_table
replace = SELECT * FROM new_table
```

`qlafilter` rejestruje wszystkie zapytania (przydatne do audytu). `regexfilter` pozwala na przepisywanie zapytań w locie, bez modyfikacji aplikacji.

Sekcja [service] — Złożenie komponentów

Usługa łączy serwery, monitory, filtry i router:

```
[rw-service]
type     = service
router   = readwritesplit
servers  = db-master, db-slave1, db-slave2
user     = maxscale_service
password = encrypted_password_here
filters  = query-log | regex-rewrite

max_slave_connections    = 100%
max_slave_replication_lag = 5s
```

- **readwritesplit**: Zapisy trafiają do mastera, odczyty są dystrybuowane na slave'y.
- **max_slave_replication_lag**: Slave opóźniony o więcej niż 5 sekund jest tymczasowo wykluczony z routingu.
- **filters**: Filtry są łączone znakiem pipe `|`.

Sekcja [listener] — Punkt wejścia sieciowego

Listener eksponuje usługę na porcie sieciowym:

```
[rw-listener]
type     = listener
service  = rw-service
protocol = MariaDBClient
port     = 4006
address  = 0.0.0.0
```

Aplikacja łączy się z MaxScale na porcie 4006 tak, jakby był to standardowy serwer MariaDB / MySQL. MaxScale jest przezroczysty z punktu widzenia protokołu.

MaxGUI: interfejs webowy

Od MaxScale 2.5 interfejs webowy MaxGUI oferuje kompletny widok graficzny infrastruktury. Dostępny pod adresem `https://maxscale-host:8989`, pozwala na:

- Wizualizację topologii w czasie rzeczywistym (master, slave'y, stan połączeń)
- Podgląd metryk wydajności (zapytania/sekundę, opóźnienie, aktywne połączenia)
- Zarządzanie serwerami (drain, maintenance, dodawanie)
- Przeglądanie logów i alertów

MaxGUI jest narzędziem nadzoru, nie konfiguracji. Zmiany konfiguracji dokonuje się przez plik `maxscale.cnf` lub przez REST API.

MaxCtrl: narzędzie wiersza poleceń

MaxCtrl jest oficjalnym CLI MaxScale. Komunikuje się z REST API i oferuje intuicyjną składnię:

```
# Wyświetlenie serwerów i ich stanu
maxctrl list servers

# Przełączenie serwera w tryb maintenance
maxctrl set server db-slave1 maintenance

# Wyjście z trybu maintenance
maxctrl clear server db-slave1 maintenance

# Wyświetlenie usług i ich statystyk
maxctrl list services

# Podgląd topologii replikacji
maxctrl show monitor replication-monitor

# Dynamiczne utworzenie filtra
maxctrl create filter my-filter regexfilter \
  match="SELECT 1" replace="SELECT 2"
```

Zaletą MaxCtrl nad bezpośrednią edycją pliku jest natychmiastowe uwzględnienie zmian, bez restartu.

REST API: automatyzacja

REST API MaxScale jest fundamentem automatyzacji. Eksponuje wszystkie funkcjonalności poprzez endpointy HTTP/JSON:

```
# Pobranie listy serwerów
curl -s -u admin:password https://maxscale:8989/v1/servers | jq

# Ręczny failover
curl -X POST -u admin:password \
  https://maxscale:8989/v1/monitors/replication-monitor/failover

# Pobranie metryk usługi
curl -s -u admin:password \
  https://maxscale:8989/v1/services/rw-service | jq '.data.attributes.statistics'
```

REST API pozwala zintegrować MaxScale z pipeline'ami CI/CD, systemami monitoringu (Prometheus, Grafana) oraz narzędziami orkiestracji (Ansible, Terraform).

Dobre praktyki konfiguracji

Oto kilka zaleceń wynikających z doświadczenia terenowego:

1. **Zawsze używaj zaszyfrowanych haseł** w pliku konfiguracyjnym. MaxScale dostarcza narzędzie `maxkeys` do generowania kluczy szyfrowania.
2. **Włącz automatyczny failover** z `auto_failover=true`, ale regularnie testuj go w środowisku przedprodukcyjnym.
3. **Dobierz liczbę wątków** do obciążenia. `threads=auto` jest dobrym ustawieniem domyślnym, ale dla bardzo dużych wolumenów (>50 000 zapytań/sekundę) ręczny tuning może być konieczny.
4. **Monitoruj opóźnienie replikacji** przez `max_slave_replication_lag`. Zbyt permisywna wartość kieruje odczyty na opóźnione slave'y. Zbyt restrykcyjna wartość koncentruje wszystkie odczyty na masterze.
5. **Używaj filtrów z umiarem**. Każdy filtr dodaje warstwę przetwarzania. `regexfilter` w szczególności może mieć znaczący wpływ na wydajność, jeśli wyrażenia regularne są złożone.

Kompletna typowa konfiguracja

Podsumowując, oto typowy plik konfiguracyjny dla topologii master + 2 slave'y:

```
[maxscale]
threads = auto
admin_host = 0.0.0.0
admin_port = 8989

[db-master]
type = server
address = 10.0.1.10
port = 3306
protocol = MariaDBBackend

[db-slave1]
type = server
address = 10.0.1.11
port = 3306
protocol = MariaDBBackend

[db-slave2]
type = server
address = 10.0.1.12
port = 3306
protocol = MariaDBBackend

[replication-monitor]
type = monitor
module = mariadbmon
servers = db-master, db-slave1, db-slave2
user = maxscale_monitor
password = <encrypted>
monitor_interval = 2000
auto_failover = true
auto_rejoin = true

[rw-service]
type = service
router = readwritesplit
servers = db-master, db-slave1, db-slave2
user = maxscale_service
password = <encrypted>
```

```
max_slave_replication_lag = 5s

[rw-listener]
type = listener
service = rw-service
protocol = MariaDBClient
port = 4006
```

Podsumowanie

MaxScale konfiguruje się za pomocą czytelnego i ustrukturyzowanego pliku INI. Sześć typów sekcji wystarczy do opisania kompletnej infrastruktury: konfiguracja globalna, serwery, monitor, filtry, usługa i listener.

Codzienna administracja odbywa się za pomocą trzech komplementarnych narzędzi: MaxGUI do nadzoru wizualnego, MaxCtrl do operacji z wiersza poleceń i REST API do automatyzacji. Razem tworzą kompletny ekosystem do zarządzania proxy SQL klasy enterprise.

Ten artykuł został pierwotnie opublikowany na Medium.