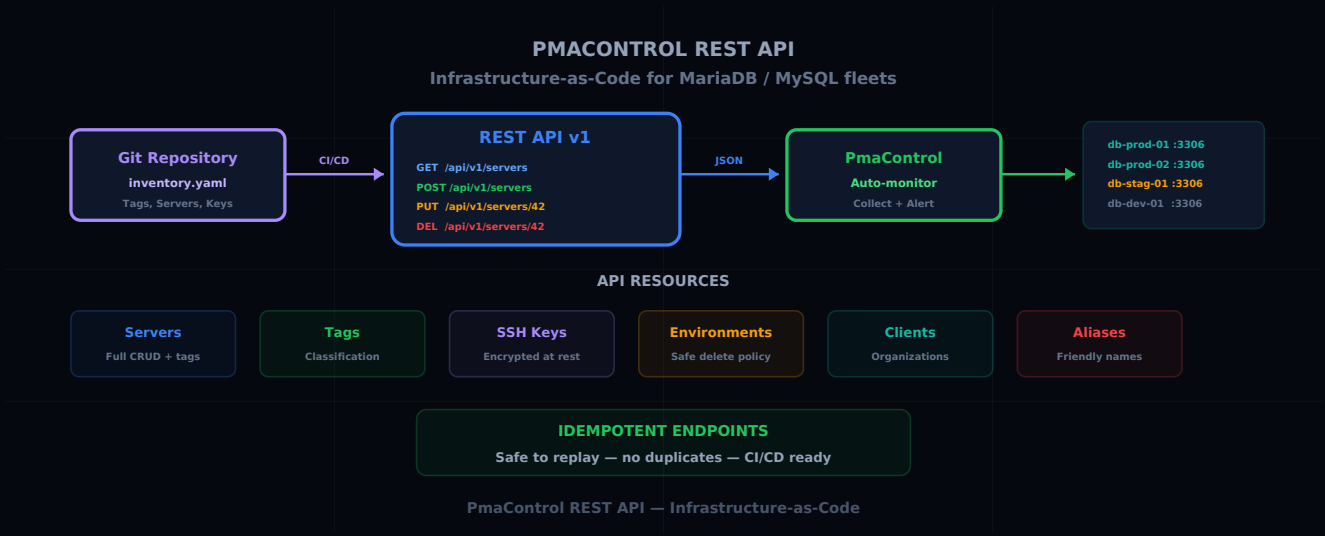
PmaControl REST API: Infrastructure-as-Code dla MariaDB / MySQL

Aurélien LEQUOY · March 15, 2026

PMACONTROL REST-API INFRASTRUCTURE-AS-CODE DEVOPS AUTOMATION



Dlaczego REST API?

PmaControl zaczynał jako narzędzie z interfejsem webowym. Dodajesz serwer MariaDB / MySQL klikając przyciski, konfigurujesz tagi myszką, zarządzasz środowiskami przez formularze.

To działa przy 10 serwerach. Przy 50 jest uciążliwe. Przy 200 jest niemożliwe.

REST API PmaControl przekształca narzędzie w **platformę programowalną**. Każda akcja dostępna w interfejsie webowym jest dostępna przez endpoint JSON. Otwiera to drzwi do Infrastructure-as-Code: opisywanie inwentarza baz danych w plikach YAML lub JSON i automatyczne stosowanie go.

Uwierzytelnianie

API wykorzystuje uwierzytelnianie przez **użytkownika webserwisu**. To dedykowane konto PmaControl, bez dostępu do interfejsu webowego, którego token służy jako klucz API.

```
curl -H "Authorization: Bearer <token>" \
     -H "Content-Type: application/json" \
     https://pmacontrol.example.com/api/v1/servers
```

Użytkownik webserwisu dziedziczy uprawnienia ACL ze swojego profilu. Profil "read-only" pozwala jedynie na listowanie zasobów. Profil "admin" umożliwia tworzenie, modyfikowanie i usuwanie.

Zasoby API

Tagi

Tagi to system klasyfikacji PmaControl. Każdy serwer może mieć jeden lub więcej tagów (production, staging, client-acme, dc-paris itp.).

```
# Listuj wszystkie tagi
GET /api/v1/tags

# Utwórz tag
POST /api/v1/tags
{"name": "production", "color": "#22c55e"}

# Zmodyfikuj tag
PUT /api/v1/tags/42
{"name": "prod", "color": "#22c55e"}

# Usuń tag
DELETE /api/v1/tags/42
```

Tagi są podstawowym elementem organizacji. Pozwalają filtrować dashboardy, ograniczać zakres alertów i kontrolować dostęp przez profile.

Klienci

Klienci reprezentują organizacje lub projekty posiadające serwery.

```
# Listuj klientów
GET /api/v1/clients

# Utwórz klienta
```

```
POST /api/v1/clients
{"name": "Acme Corp", "contact_email": "dba@acme.com"}

# Zmodyfikuj klienta
PUT /api/v1/clients/7
{"name": "Acme Corporation"}

# Usuń klienta
DELETE /api/v1/clients/7
```

Środowiska

Środowiska (production, staging, development itp.) mają specjalną politykę usuwania: **gdy środowisko jest usuwane, jego serwery nie są niszczone, lecz przypisywane** do domyślnego środowiska.

```
# Listuj środowiska
GET /api/v1/environments

# Utwórz środowisko
POST /api/v1/environments
{"name": "staging", "description": "Pre-production environment"}

# Usuń — serwery zostaną przeniesione
DELETE /api/v1/environments/3
# Odpowiedź: {"reassigned_servers": 12, "target_environment": "default"}
```

Ta polityka zapobiega przypadkowemu usuwaniu serwerów podczas reorganizacji środowisk.

Aliasy

Aliasy umożliwiają odwoływanie się do serwera przez czytelną nazwę zamiast adresu IP lub wewnętrznej nazwy hosta.

```
# Listuj aliasy
GET /api/v1/aliases

# Utwórz alias
POST /api/v1/aliases
{"alias": "db-writer-prod", "server_id": 42}
```

Storage Areas

Storage areas reprezentują strefy przechowywania (centra danych, regiony chmurowe itp.).

```
# Listuj storage areas
GET /api/v1/storage-areas

# Utwórz storage area
POST /api/v1/storage-areas
{"name": "dc-paris-1", "provider": "OVH", "location": "Paris, France"}
```

Klucze SSH

PmaControl używa kluczy SSH do łączenia się z serwerami i zbierania metryk. API pozwala zarządzać cyklem życia kluczy.

```
# Listuj klucze SSH
GET /api/v1/ssh-keys

# Utwórz klucz SSH
POST /api/v1/ssh-keys
{
  "name": "pmacontrol-collector-2026",
  "public_key": "ssh-ed25519 AAAA...",
  "private_key": "-----BEGIN OPENSSH PRIVATE KEY-----\n..."
}

# Usuń klucz
DELETE /api/v1/ssh-keys/5
```

Uwaga dotycząca bezpieczeństwa: klucze prywatne są szyfrowane w spoczynku w bazie PmaControl. API nigdy nie zwraca klucza prywatnego w odpowiedzi na zapytanie GET.

Serwery

Główny zasób. Pełny CRUD dla nadzorowanych instancji MariaDB / MySQL.

```
# Listuj wszystkie serwery
GET /api/v1/servers

# Listuj z filtrami
```

```
GET /api/v1/servers?tag=production&environment=staging

# Szczegóły serwera
GET /api/v1/servers/42

# Utwórz serwer
POST /api/v1/servers
{
  "hostname": "db-prod-01.acme.com",
  "ip": "10.0.1.10",
  "port": 3306,
  "ssh_key_id": 5,
  "client_id": 7,
  "environment_id": 1,
  "tags": ["production", "galera", "dc-paris"]
}

# Zmodyfikuj serwer
PUT /api/v1/servers/42
{"port": 3307}

# Przypisz tagi
POST /api/v1/servers/42/tags
{"tags": ["production", "critical"]}

# Przypisz klucz SSH
PUT /api/v1/servers/42/ssh-key
{"ssh_key_id": 5}

# Usuń serwer
DELETE /api/v1/servers/42
```

Przepływy Infrastructure-as-Code

Główną zaletą API jest możliwość opisania kompletnego inwentarza w wersjonowanym pliku i automatycznego zastosowania go.

Przykład: plik inwentarza YAML

```yaml
# pmacontrol-inventory.yaml
tags:
  - name: production
    color: "#22c55e"
  - name: staging
    color: "#f59e0b"
  - name: galera
    color: "#14b8a6"

environments:
  - name: production
  - name: staging
  - name: development

ssh_keys:
  - name: collector-2026
    public_key_file: ./keys/collector-2026.pub

servers:
  - hostname: db-prod-01.acme.com
    ip: 10.0.1.10
    port: 3306
    ssh_key: collector-2026
    environment: production
    tags: [production, galera]

  - hostname: db-prod-02.acme.com
    ip: 10.0.1.11
    port: 3306
    ssh_key: collector-2026
    environment: production
    tags: [production, galera]

  - hostname: db-staging-01.acme.com
    ip: 10.0.2.10
    port: 3306
    ssh_key: collector-2026
    environment: staging
    tags: [staging]
```

Skrypt aplikujący

Skrypt Python lub Bash odczytuje ten plik i wywołuje API PmaControl, aby zsynchronizować stan:

```
#!/bin/bash
API="https://pmacontrol.example.com/api/v1"
TOKEN="your-webservice-token"

# Utwórz tagi
for tag in production staging galera; do
  curl -s -X POST "$API/tags" \
    -H "Authorization: Bearer $TOKEN" \
    -H "Content-Type: application/json" \
    -d "{\"name\": \"$tag\"}"
done

# Utwórz serwery
curl -s -X POST "$API/servers" \
  -H "Authorization: Bearer $TOKEN" \
  -H "Content-Type: application/json" \
  -d @server-prod-01.json
```

Integracja CI/CD

Naturalnym wzorcem jest zintegrowanie tego z pipeline CI/CD:

1. DBA modyfikuje plik `pmacontrol-inventory.yaml` w Git
2. Merge request jest recenzowany przez zespół
3. Pipeline CI waliduje składnię i ograniczenia (brak zduplikowanych IP, prawidłowe klucze SSH)
4. Pipeline CD stosuje zmiany przez API PmaControl
5. PmaControl automatycznie zaczyna nadzorować nowe serwery

Idempotencja

Wszystkie endpointy tworzenia są **idempotentne**: jeśli zasób już istnieje (ten sam hostname, ten sam IP), API zwraca istniejący zasób zamiast tworzyć duplikat. Pozwala to ponownie uruchomić skrypt inwentarza bez efektów ubocznych.

```
# Pierwsze wywołanie: tworzy serwer, zwraca 201
POST /api/v1/servers  → 201 Created
```

```
# Drugie identyczne wywołanie: zwraca istniejący serwer, 200
POST /api/v1/servers  → 200 OK (existing)
```

Kody odpowiedzi

Kod	Znaczenie
200	Sukces (odczyt lub aktualizacja)
201	Zasób utworzony
204	Pomyślne usunięcie
400	Nieprawidłowy payload
401	Brakujący lub nieprawidłowy token
403	Niewystarczające uprawnienia
404	Zasób nie znaleziony
409	Konflikt (ograniczenie unikalności)

Obecne ograniczenia

API v1 obejmuje operacje CRUD na inwentarzu. Nie obejmuje jeszcze:

- **Metryk** (odczyt time-series) — planowane na v2
- **Alertów** (konfiguracja progów) — planowane na v2
- **Backupów** (uruchamianie i status) — planowane na v2
- **Eksportu konfiguracji** (my.cnf, reguły ProxySQL) — w dyskusji

Podsumowanie

REST API PmaControl przekształca zarządzanie flotą MariaDB / MySQL z procesu ręcznego w programowalny i wersjonowany przepływ pracy.

Opisz swój inwentarz w Git. Zastosuj go przez API. Pozwól PmaControl automatycznie nadzorować. To Infrastructure-as-Code zastosowane do monitoringu baz danych.