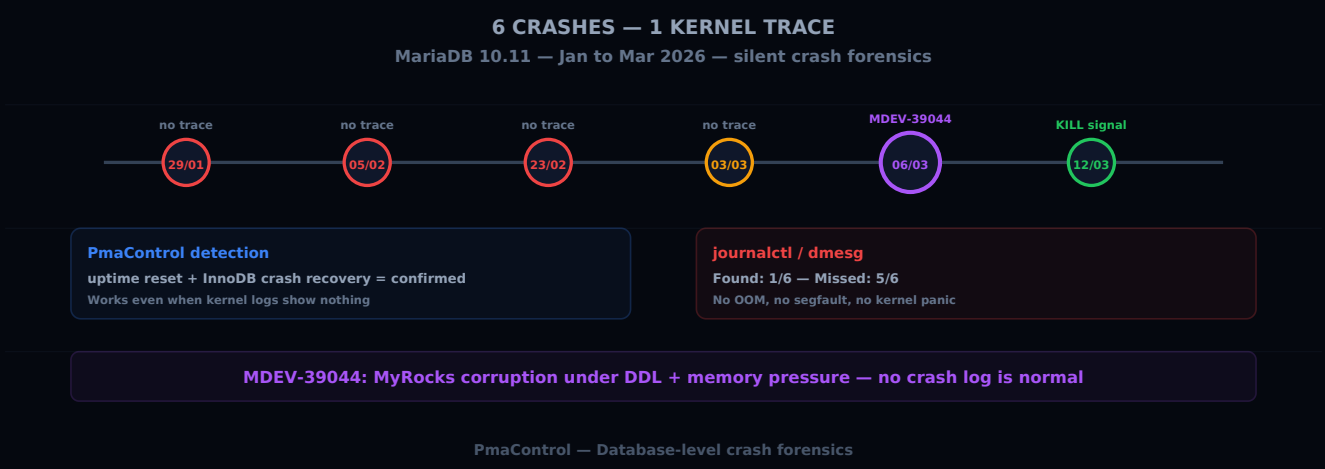
Ciche awarie MariaDB: gdy brak logów maskuje krytyczne problemy

Aurélien LEQUOY · March 12, 2026

MARIADB CRASH-ANALYSIS FORENSICS INCIDENT-RESPONSE ROCKSDB



Problem

Między styczniem a marcem 2026 zaobserwowaliśmy **6 anomalnych resetów uptime** na produkcyjnym serwerze MariaDB 10.11.15 nadzorowanym przez PmaControl. Serwer hostuje wolumenowe tabele RocksDB z partycjonowaniem (metryki szeregów czasowych).

Klasyczny odruch DBA w obliczu awarii: sprawdzić logi systemowe. `journalctl`, `dmesg`, `/var/log/syslog`. Szukamy `OOM`, `Killed process`, `segfault`, `kernel panic`.

Na 6 awarii, tylko jedna miała użyteczny ślad kernela. Pozostałe 5: kompletna cisza po stronie systemu.

Metoda detekcji

Zamiast polegać na logach systemowych, użyliśmy PmaControl do wykrywania awarii poprzez **szereg czasowy** `uptime`:

1. Pobranie wartości `uptime` przez `ts_value_general_int`
2. Filtrowanie anomalnych resetów (uptime spadający do 0)

3. Korelacja z logami MariaDB (`error.log`)
4. Korelacja z `journalctl` w poszukiwaniu sygnatur kernelowych
5. Analiza metryk z poprzedniej godziny (wątki, CPU, pamięć)

To najniezawodniejsze podejście: **reset uptime + sygnatura `InnoDB crash recovery` = potwierdzona awaria**, nawet bez śladu systemowego.

6 zidentyfikowanych awarii

Data	Klasyfikacja	Główna sygnatura
29 sty.	prawdopodobna awaria	`InnoDB crash recovery` + recovery binlog
5 lut.	prawdopodobna awaria	crash recovery + `Event invalid` w binlogu
23 lut.	prawdopodobna awaria	`InnoDB crash recovery` + `Crash table recovery`
3 mar.	prawdopodobna awaria	`Too many connections` potem crash recovery
6 mar.	poważny incydent	**Korupcja MyRocks**: `truncated record body`, `.frm mismatch`
12 mar.	potwierdzona awaria	`systemd: status=9/KILL` + crash recovery

Awaria 6 marca: korelacja MDEV-39044

Najpoważniejszy incydent to ten z 6 marca. Błędy były inne:

```
RocksDB: Error opening instance, Status Code: 2,
  Status: Corruption: truncated record body
Incorrect information in file: './pmacontrol/ts_value_general_int.frm'
Can't init tc log
Aborting
```

Ten wzorzec dokładnie odpowiada zgłoszeniu MariaDB **MDEV-39044**: korupcja MyRocks wyzwolona przez:

- `ALTER TABLE` na wolumenowych tabelach RocksDB z partycjonowaniem
- ciągłe duże obciążenie zapisu
- jednoczesna presja pamięciowa InnoDB

Zgłoszenie wyraźnie potwierdza, że **brak logu awarii lub OOM killera jest normalnym zachowaniem w tym scenariuszu**.

Dlaczego logi systemowe nie wystarczą

Na 6 incydentów `journalctl` znalazł tylko **jeden użyteczny ślad** (`status=9/KILL` z 12 marca).

Dla pozostałych 5:

- brak `Out of memory`
- brak `Killed process`
- brak `segfault`
- brak `kernel panic`

Wnioskowanie jest proste: **brak sygnatury kernelowej nie oznacza braku awarii**. To nawet spójne ze wzorcem MDEV-39044, który dokumentuje awarie bez śladu systemowego.

Co PmaControl wykrywa, czego logi nie pokazują

PmaControl monitoruje `uptime` w sposób ciągły (co 10 sekund). Reset = natychmiastowy alert. Następnie agent automatycznie koreluje:

- metryki z poprzedniej godziny (wątki, pamięć, CPU)
- obecność `crash recovery` w error logu
- błędy metadanych (`.frm mismatch`)

Co pozwala sklasyfikować incydent **nawet bez współpracy jądra systemu**.

Rekomendacje

1. **Nigdy nie polegać wyłącznie na logach systemowych** do wykrywania awarii MariaDB
2. **Monitorować `uptime` jako główny wskaźnik** stabilności
3. **Korelować z error logiem MariaDB**, nie z `journalctl`
4. **Jeśli używasz RocksDB**: ograniczyć DDL na wolumenowych tabelach z partycjonowaniem, szczególnie pod obciążeniem zapisu
5. **Śledzić MDEV-39044** w oczekiwaniu na ewentualną poprawkę MyRocks

Podsumowanie

Serwer MariaDB może ulec awarii **6 razy w 6 tygodni** bez dokumentacji w jakimkolwiek logu systemowym. Tylko dedykowany nadzór baz danych — rozumiejący wewnętrzne sygnatury MariaDB — pozwala wykryć i sklasyfikować te incydenty.

Dokładnie taka jest rola PmaControl.