LIVE PATCHING
DATABASE MANAGEMENT SYSTEMS

UNIVERSITY OF PASSAU
Faculty of Computer Science and Mathematics

Michael Fruth
michael.fruth@uni-passau.de
University of Passau
Passau, Bavaria, Germany

DBMS
Version 5.15.**0**

DBMS
Version 5.15.**1**

MariaDB MDEV-20753

🕐 **Time** | Critical (security) issue found | Patch issue: - if (tables_list && !procedure) + if (tables_list && top_join_tab_count && !procedure) | Release new version

How to update the DBMS as cloud provider? Problems:
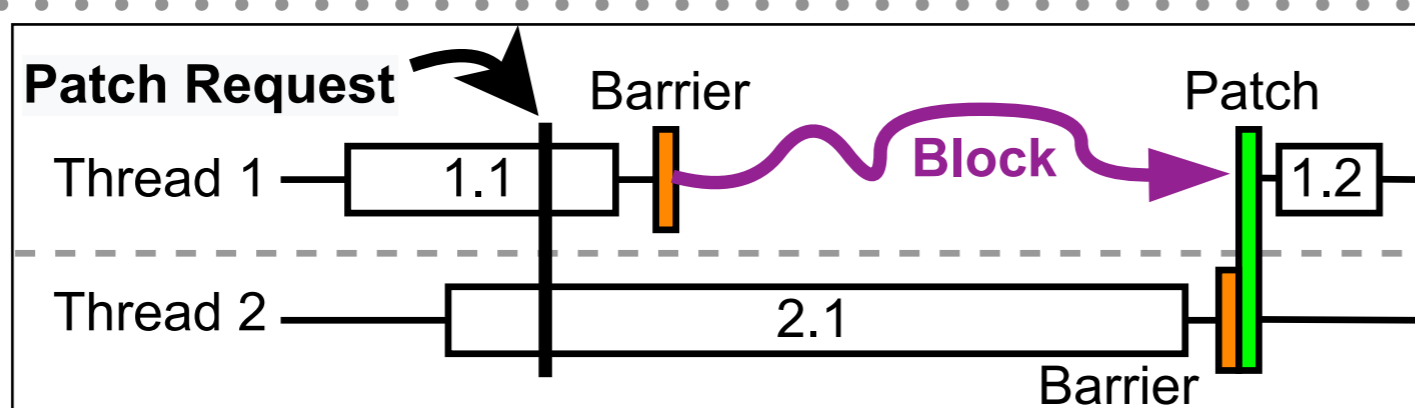
(1) A sense of urgency as security leaks might be exploited.

(2) For self-service products offered to the customer: When are downtimes acceptable to the customer?

(3) Rolling updates or hot standby is very expensive.

Solution:
Dynamic Software Updating (DSU), a.k.a. **live patching**, allows to patch a running process without restart.

Global Quiescence

Patch Request → Barrier | Patch
Thread 1 ... 1.1 ... Barrier ... **Block** → Patch 1.2
Thread 2 ... 2.1 ... Barrier

- **Each** thread needs to reach its barrier before a patch can be applied. Can cause unbounded wait times:
  (1) Thread waits for user input that may never be provided.
  (2) Thread performs a long running task.
  (3) Deadlock can occur with inter-thread dependencies.

Local Quiescence

Patch Request
Thread 1 ... 1.1 ... Barrier&Patch ... 1.2
Thread 2 ... 2.1 ... Barrier&Patch
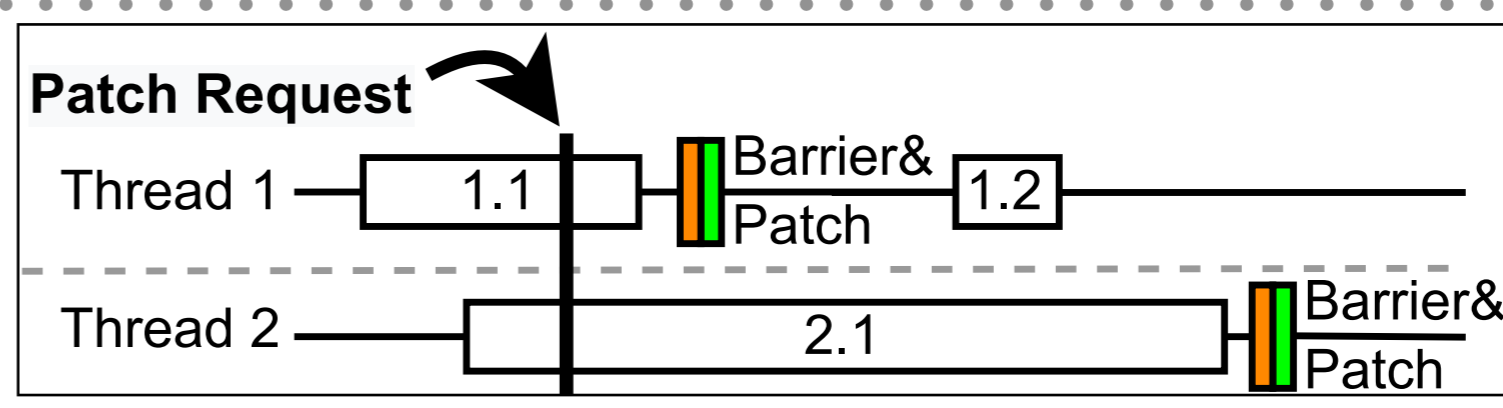
- Each thread is "patched" **individually**, regardless of the status of the other threads.
- WFPATCH [1], a novel approach proposed by Rommel et al., enables local quiescence using a modified linux kernel and quiescence points implemented into the source code of the application to patch.
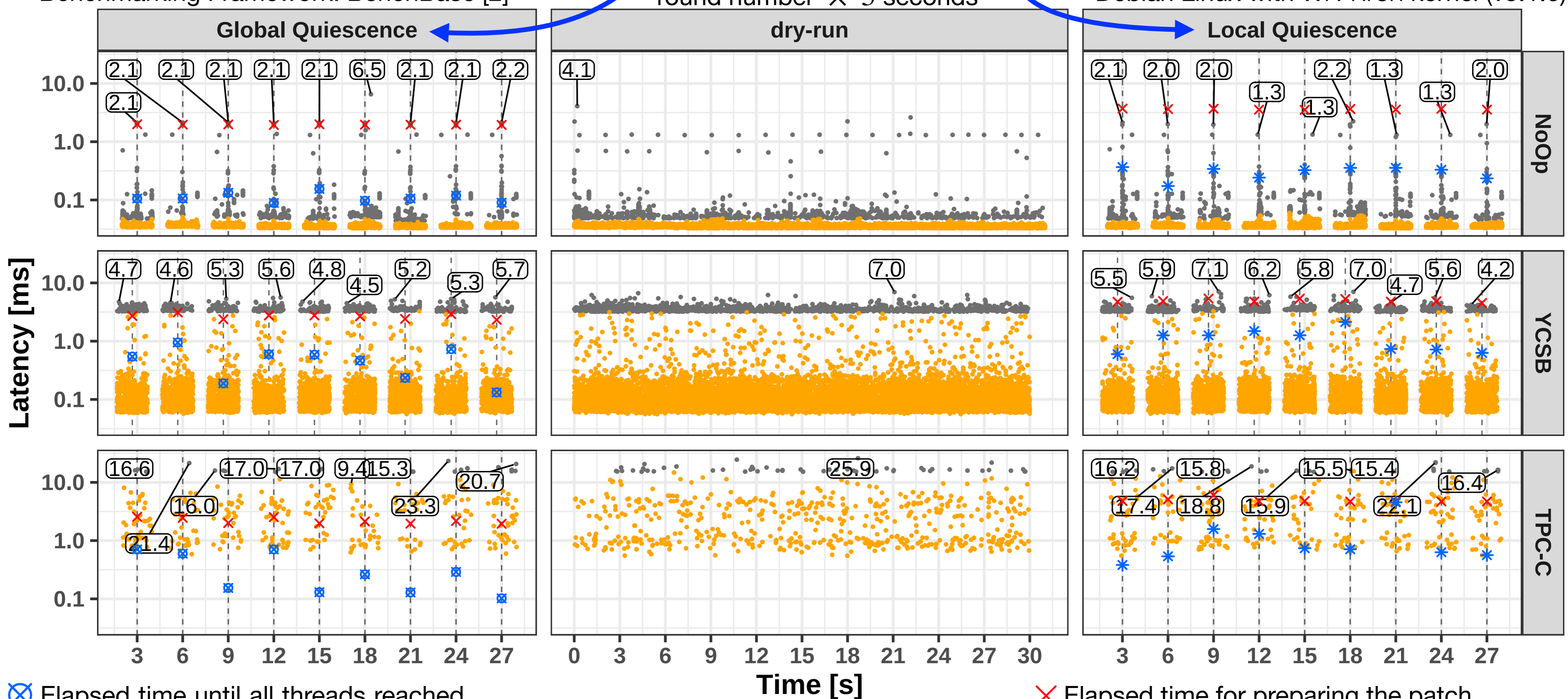
Experiments:
- Benchmarks: NoOp (No Operation), YCSB, TPC-C
- Database: MariaDB (10.5.1)
- Benchmarking Framework: BenchBase [2]

- Patch: MDEV-21665
- 1 dry-run (without applying a patch)
- 9 runs while applying a patch after "round number × 3 seconds"

System Setup:
- Server: DELL R640 (2x Intel Gold 6248R; 384 GB RAM)
- Debian Linux with WFPATCH kernel (v5.1.0)

Latency [ms] — Time [s]

Global Quiescence | dry-run | Local Quiescence

NoOp — Global: 2.1, 2.1, 2.1, 2.1, 2.1, 6.5, 2.1, 2.1, 2.2 / 2.1 ; dry-run: 4.1 ; Local: 2.1, 2.0, 2.0, 2.2, 1.3, 1.3, 1.3, 1.3, 2.0

YCSB — Global: 4.7, 4.6, 5.3, 5.6, 4.8, 4.5, 5.2, 5.3, 5.7 ; dry-run: 7.0 ; Local: 5.5, 5.9, 7.1, 6.2, 5.8, 7.0, 4.7, 5.6, 4.2

TPC-C — Global: 16.6, 16.0, 17.0, 17.0, 9.4, 15.3, 23.3, 20.7, 21.4 ; dry-run: 25.9 ; Local: 16.2, 17.4, 15.8, 18.8, 15.9, 15.5, 15.4, 22.1, 16.4

⊠ Elapsed time until all threads reached their barrier (maximum time a thread blocks and waits for other threads)

⬤ Latency exceeding 99.95th latency percentile
🟠 Down-sampled (0.5%) standard latency

✕ Elapsed time for preparing the patch
✳ Elapsed time until all threads migrated to the patched process version (corresponds to the quiescence time of global quiescence)

References
[1] Florian Rommel, Christian Dietrich, Daniel Friesel, Marcel Köppen, Christoph Borchert, Michael Müller, Olaf Spinczyk, and Daniel Lohmann. 2020. From Global to Local Quiescence: Wait-Free Code Patching of Multi-Threaded Processes. In Proc. OSDI.
[2] Djellel Eddine Difallah, Andrew Pavlo, Carlo Curino, and Philippe Cudré-Mauroux. 2013. OLTP-Bench: An Extensible Testbed for Benchmarking Relational Databases. Proc. VLDB Endow. 7, 4 (2013).

OTH OSTBAYERISCHE TECHNISCHE HOCHSCHULE REGENSBURG | Leibniz Universität Hannover | TUHH Hamburg University of Technology