PgBouncer in Parallel: Bringing Multithreading to the Trusted Postgres Pooler

Bloomberg

Engineering

PGConf 2025
September 29, 2025

Guanqun Yang, Bloomberg
Beihao Zhou

TechAtBloomberg.com

Agenda

1. Why use PgBouncer

1. Architecture and Core Principles of PgBouncer

1. Why do we multithread PgBouncer?

1. Multithreaded PgBouncer: Implementation and Usage Tips

1. Takeaways

Why use PgBouncer

TechAtBloomberg.com

Bloomberg

Engineering

PgBouncer: The Classic Postgres Pooler

- First released in **2007**, widely adopted across startups → hyperscalers
- Lightweight C implementation, proven in production for 15+ years

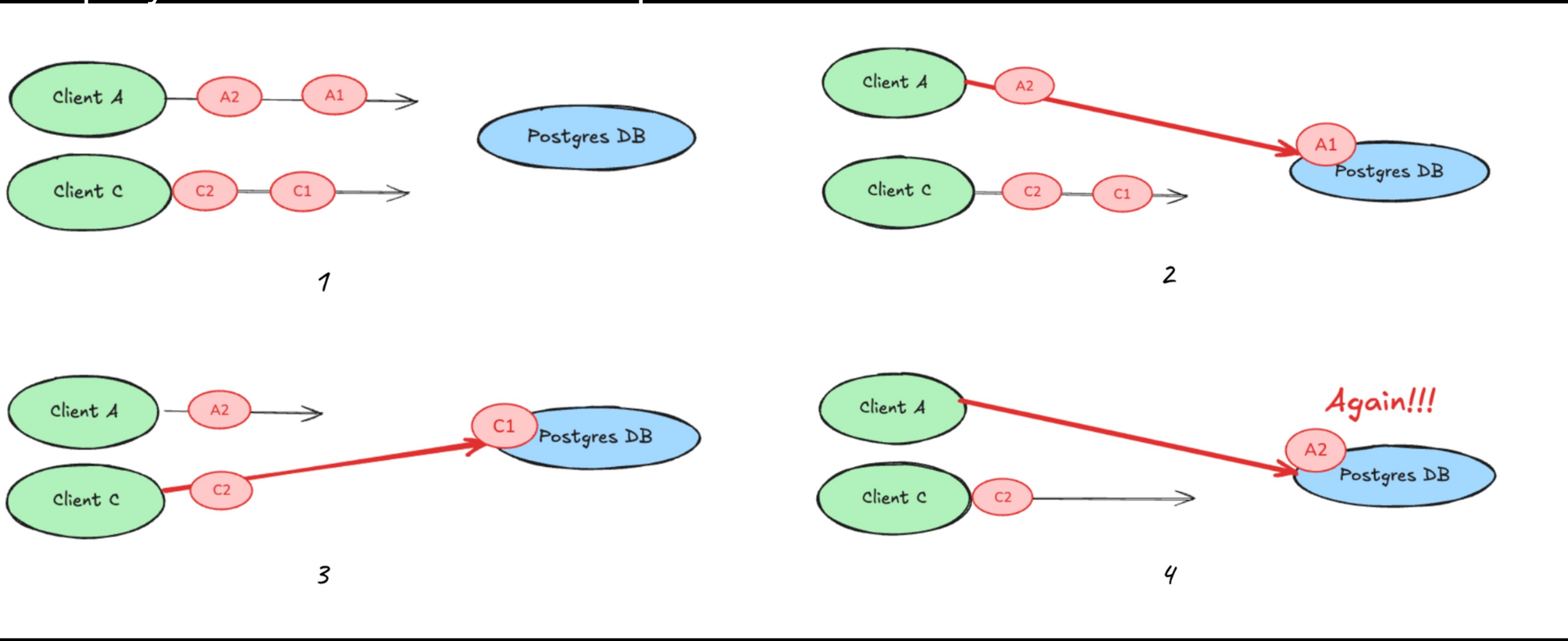
TechAtBloomberg.com

Bloomberg

Engineering

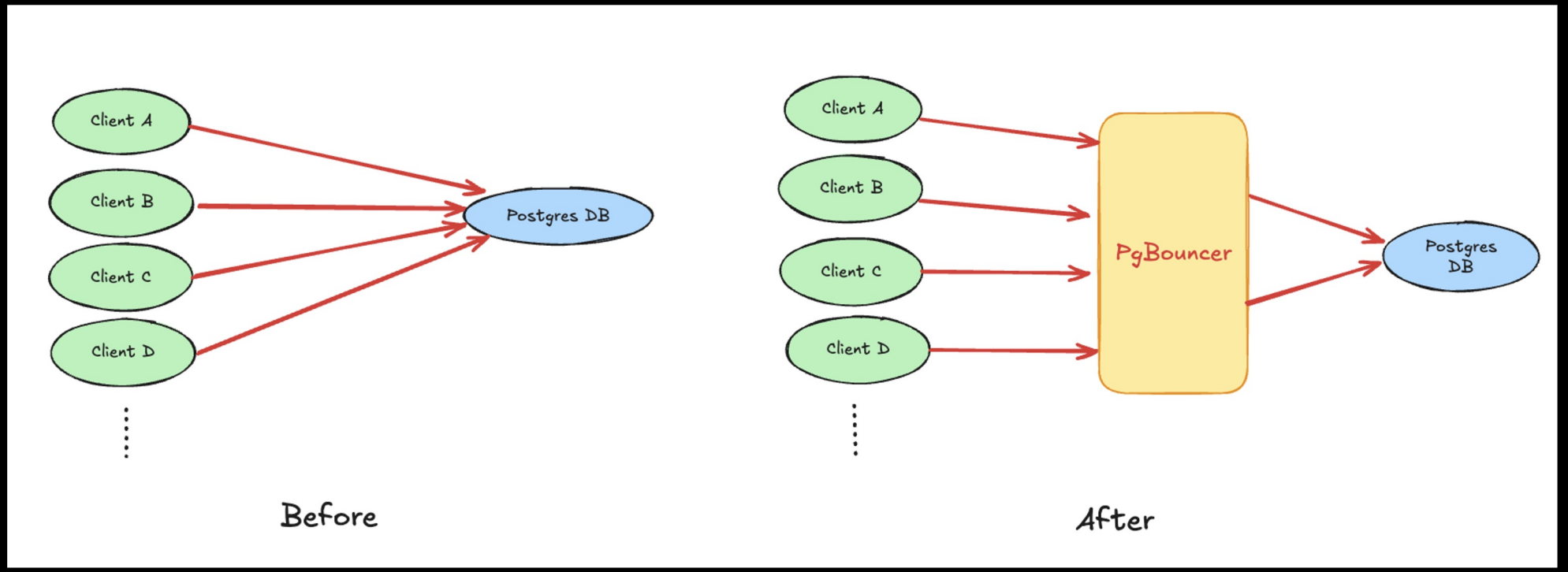
Why Connection Pooling Matters

Rapidly churn connections → frequent creation/teardown overhead



Why Connection Pooling Matters

- **Maintain many persistent connections**
  a. Less idle back-end connections
  b. Cuts connection setup time



Bloomberg
Engineering

How is PgBouncer deployed?

Deployment Options for PgBouncer

- **Client-side / Server-side**
  - ✅ No extra network overhead
  - ❌ Consumes client or server CPU & memory resources

- **Middlebox**
  - ✅ Offloads resource usage from client/server
  - ✅ Enables more flexible functionality
  - ❌ Adds network latency and bandwidth cost

Architecture and Core Principles of PgBouncer

TechAtBloomberg.com

Bloomberg

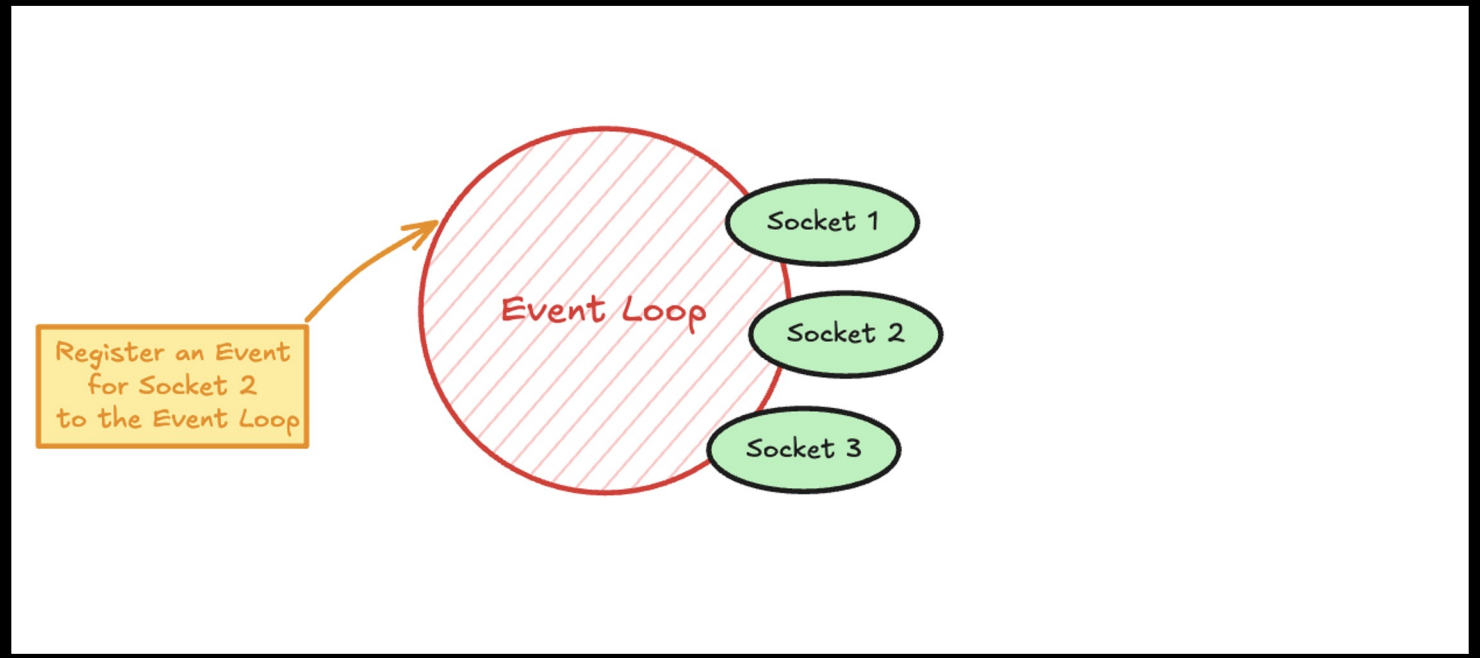
Engineering

libevent at a Glance

- **Events:** The fundamental building blocks representing I/O readiness (read/write), timeouts, or signals
- **Callbacks:** Functions that libevent calls when an event becomes active
- **Event Loop:** The main loop that continuously checks for and dispatches active events to their respective callbacks

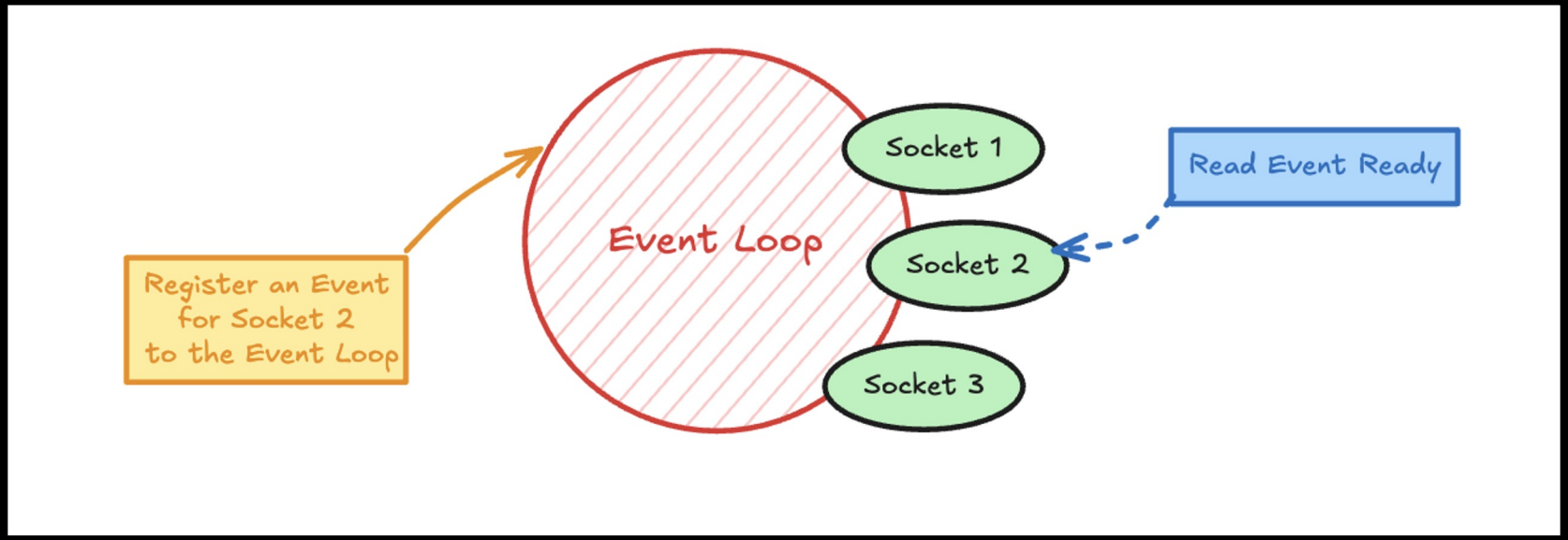
PgBouncer Architecture: Event Execution

1. Registration: PgBouncer register events. libevent starts monitoring that socket for readiness



PgBouncer Architecture: Event Execution

2. Kernel signals readiness:

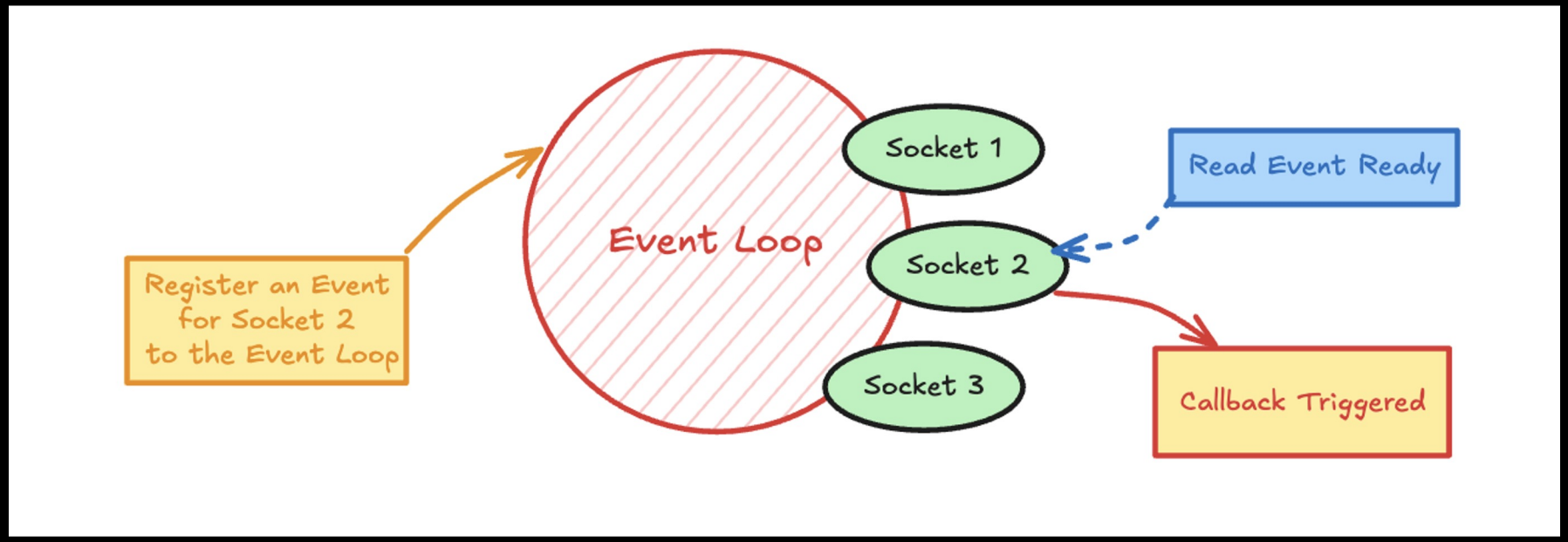


Bloomberg

Engineering

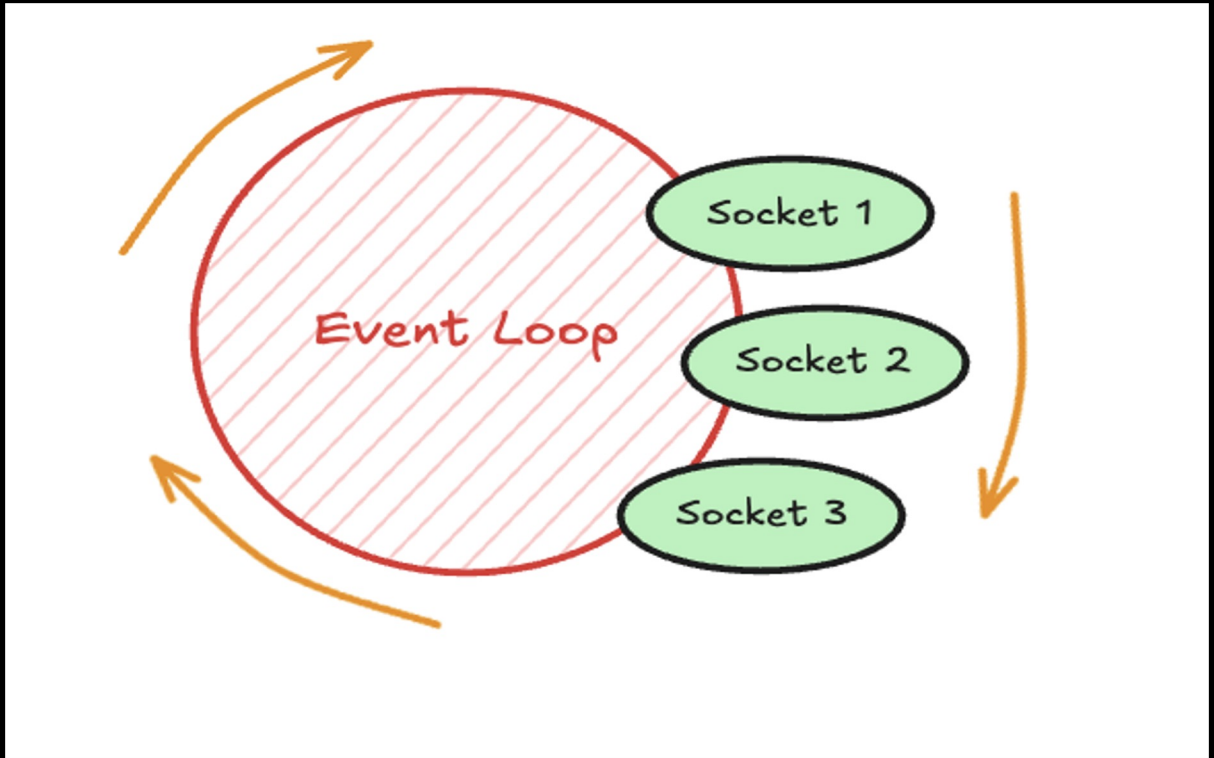
PgBouncer Architecture: Event Execution

3. libevent dispatches + Callback runs quickly: Event Loop triggers the corresponding callback (e.g. updates the state of sockets)



PgBouncer Architecture: Event Execution

4. Event Loop continues: monitors all events and triggers callbacks for whichever socket ready first



TechAtBloomberg.com

Bloomberg

Engineering

In PgBouncer, Everything is an Event!

- **Timed events**
  - Stats
  - Full maintenance
- **Signal events**
  - SIGUSR1 reload
  - SIGINT / SIGTERM handling…
- **Dynamic events**
  - New connections
  - Connected sockets (Connection handled as a state machine driven by events)

Why Event-Driven Design Matters

- **No busy-waiting**

- One thread can handle **thousands of sockets**, because it only wakes up when real work exists

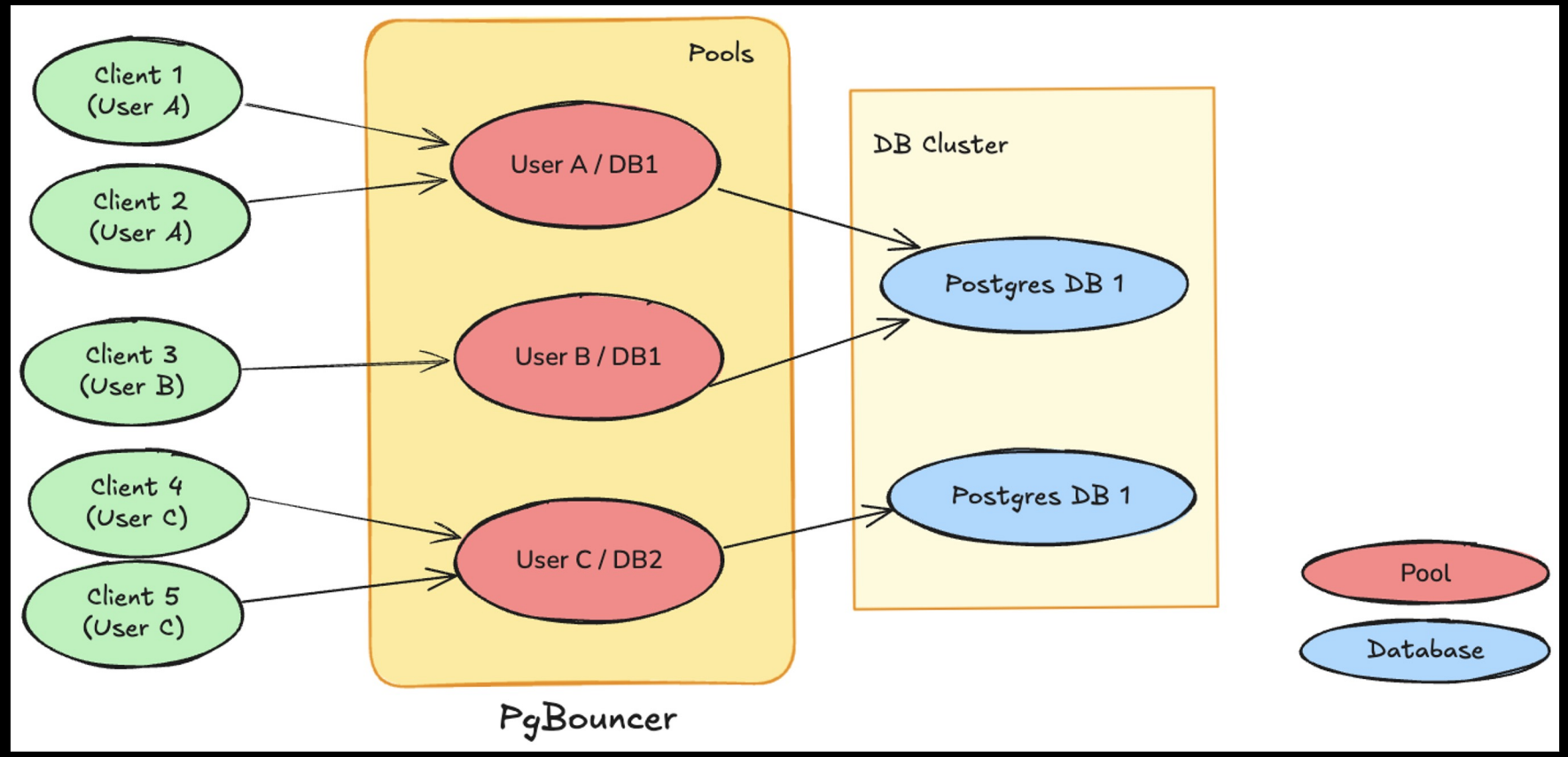
TechAtBloomberg.com

Bloomberg

Engineering

Client-Pool-Server Architecture

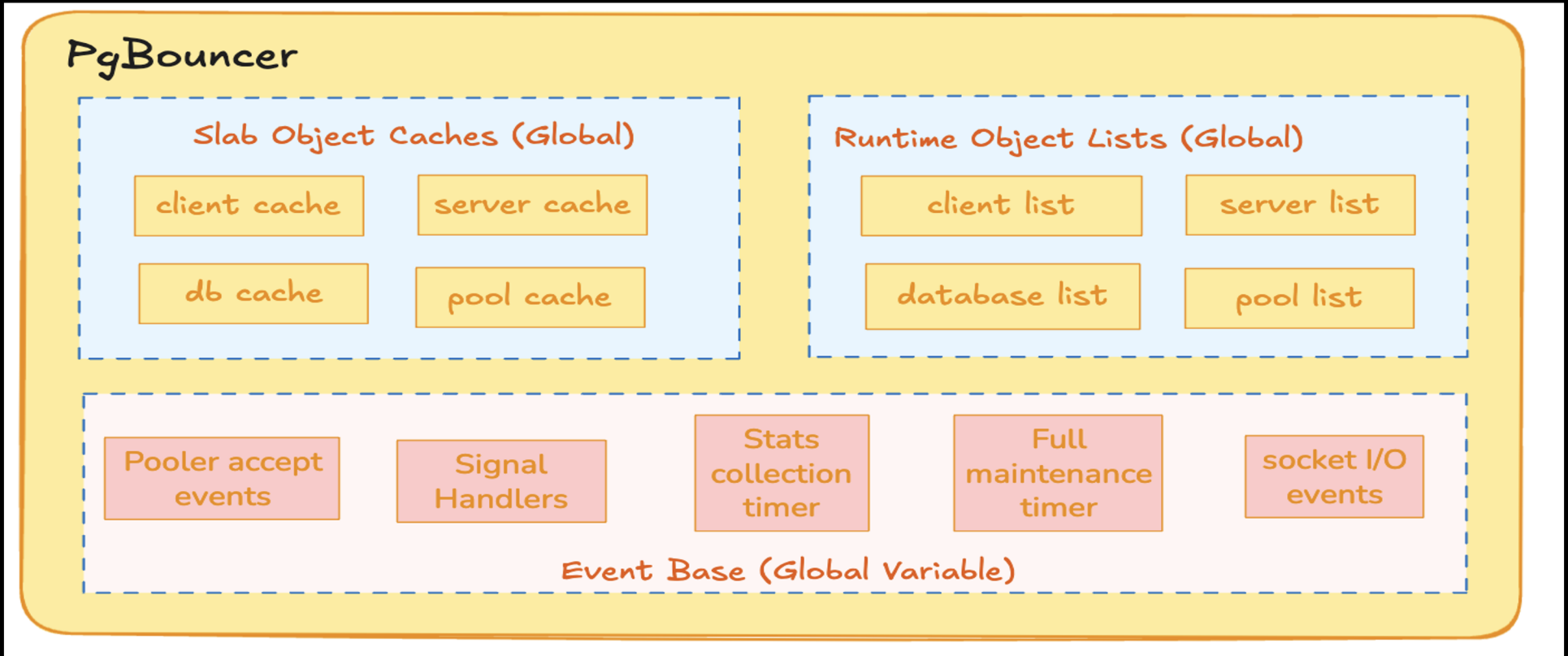
- Each pool maintains its client and server sockets (active / idle)



Bloomberg

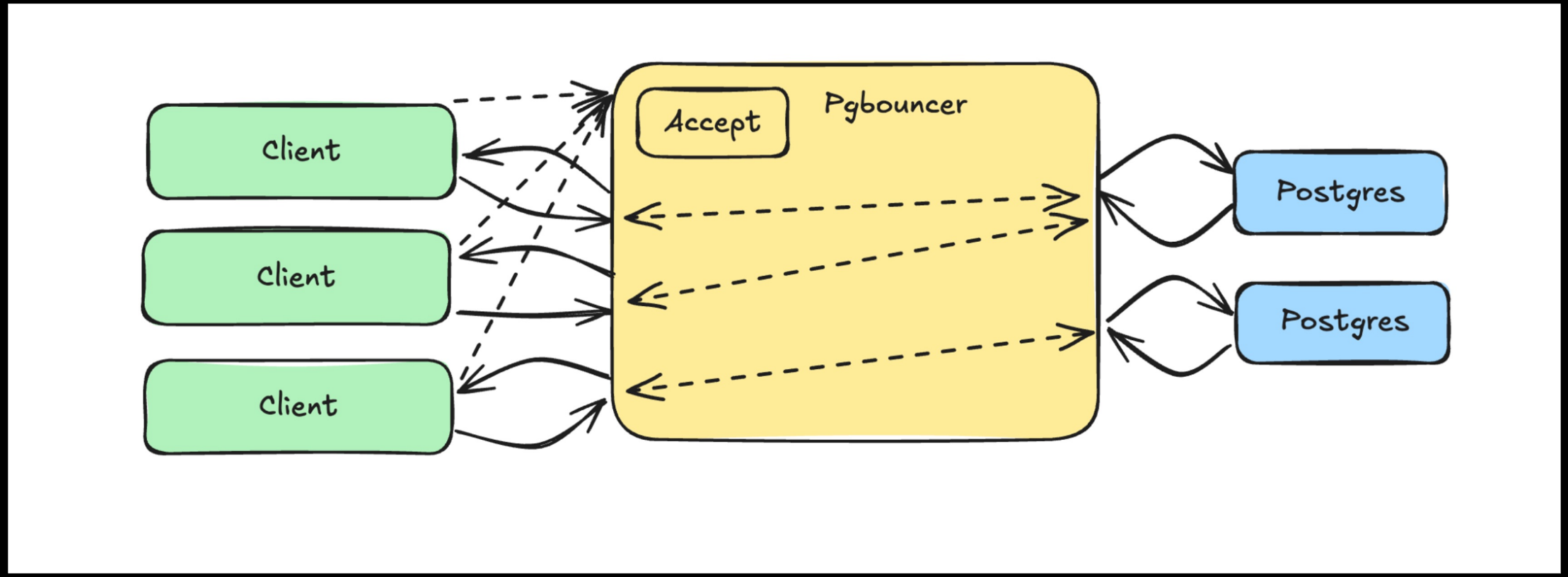
Engineering

PgBouncer Memory Management



- **List:** Collections of live objects, implemented as linked lists. PgBouncer periodically needs to scan all objects
- **Cache:** Provide memory pool for objects

How Single Thread Pgbouncer handle connections



TechAtBloomberg.com

Bloomberg

Engineering

How Admin Works in PgBouncer

- Users connect to **PgBouncer's admin console** via the special "pgbouncer" database
- Allows users to query **PgBouncer's internal status** and execute Admin commands
- Admin requests are handled like normal requests
- Admin connections do **not** establish a back-end connection to PostgreSQL

Why do we multithread PgBouncer?

- Performance
- Desired feature by the community

TechAtBloomberg.com

Bloomberg

Engineering

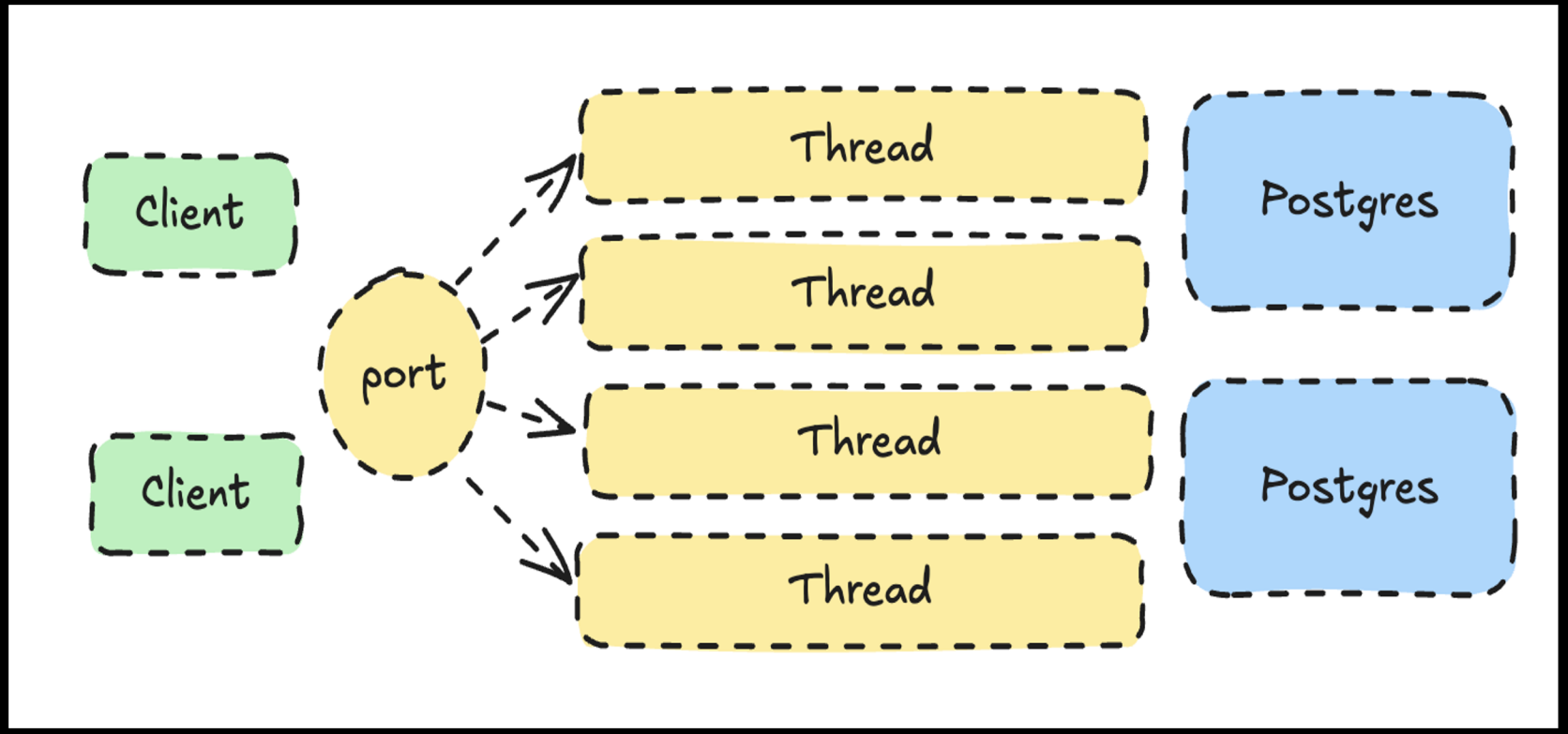
Multi-Threaded PgBouncer is Coming!

TechAtBloomberg.com

Bloomberg

Engineering

Worker Threads Listen on the Port (peer with shared memory)



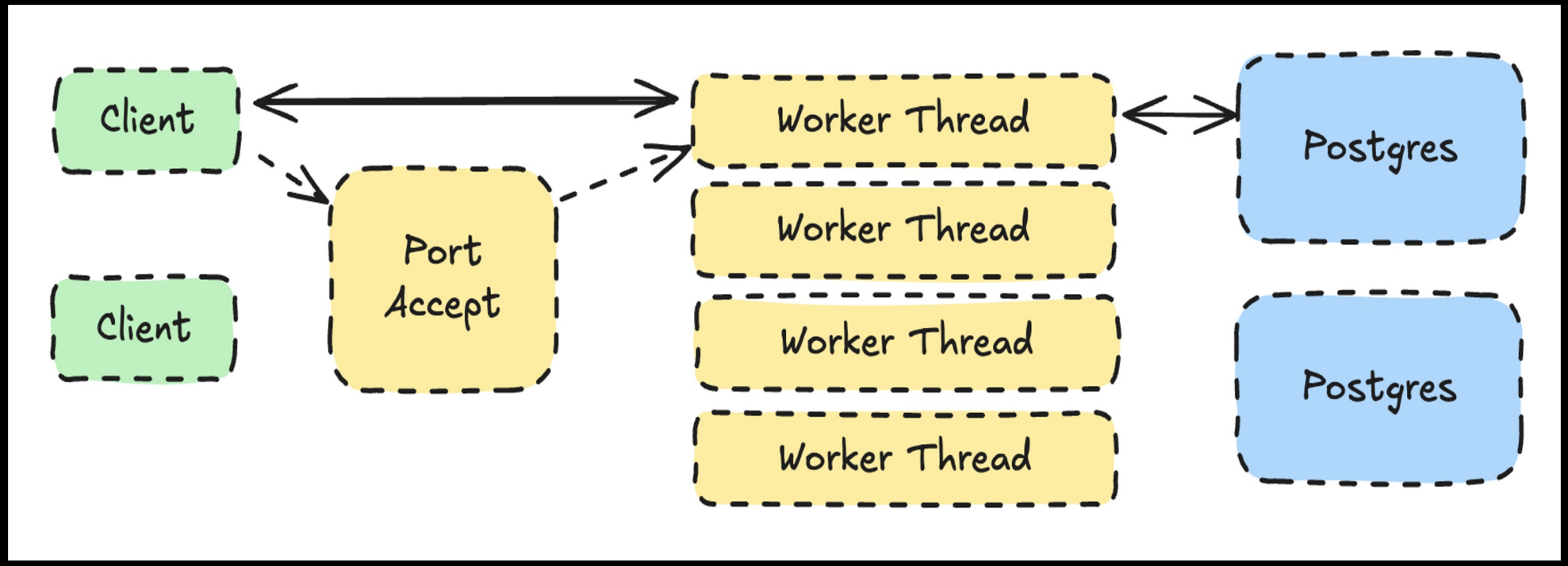
- Kernel distributes the connections
- SO_REUSEPORT on different platforms

Bloomberg

Engineering

Connection-Thread Affinity

- Each connection’s requests are always handled by a single thread
- Each thread dispatches its own event base



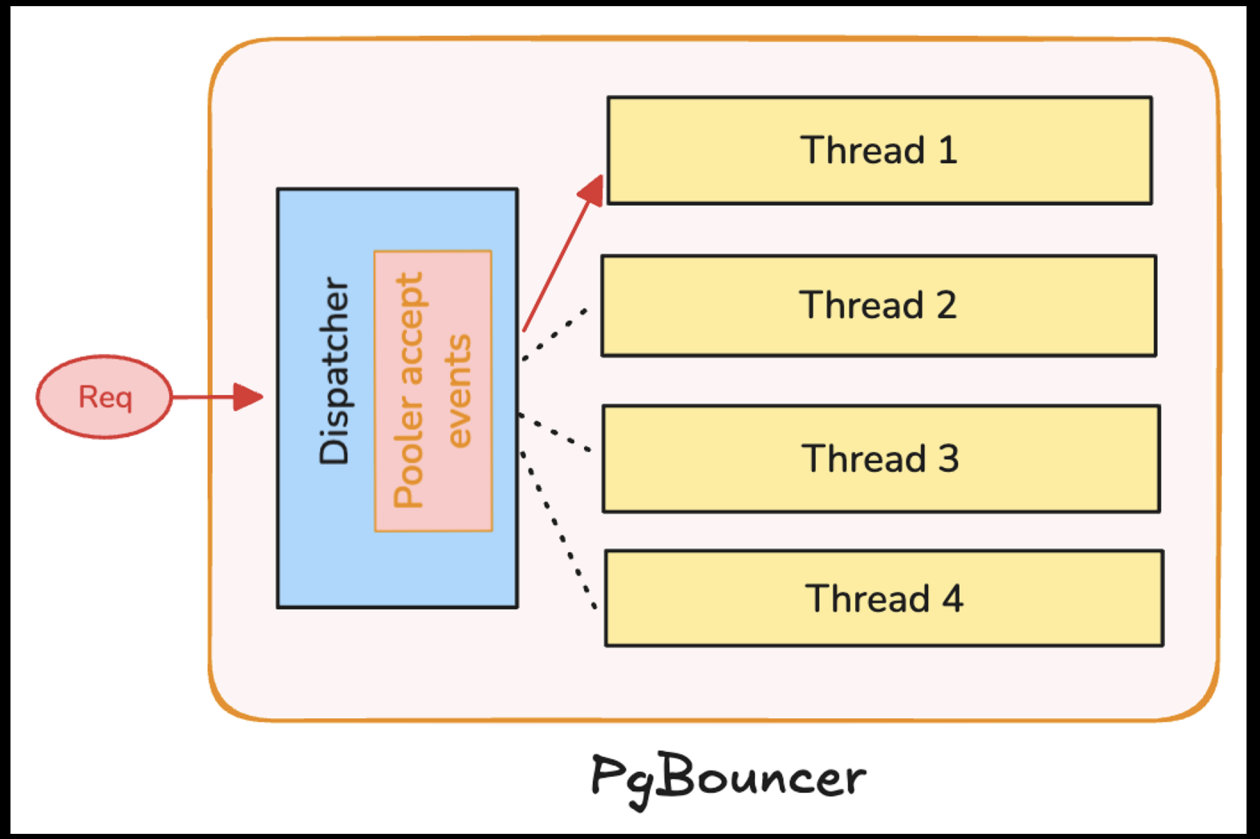
Bloomberg

Engineering

Multi-Threaded PgBouncer: Thread Model

Main thread:

- Accepts new client connections from TCP/Unix sockets
- *Dispatches* file descriptors to worker threads



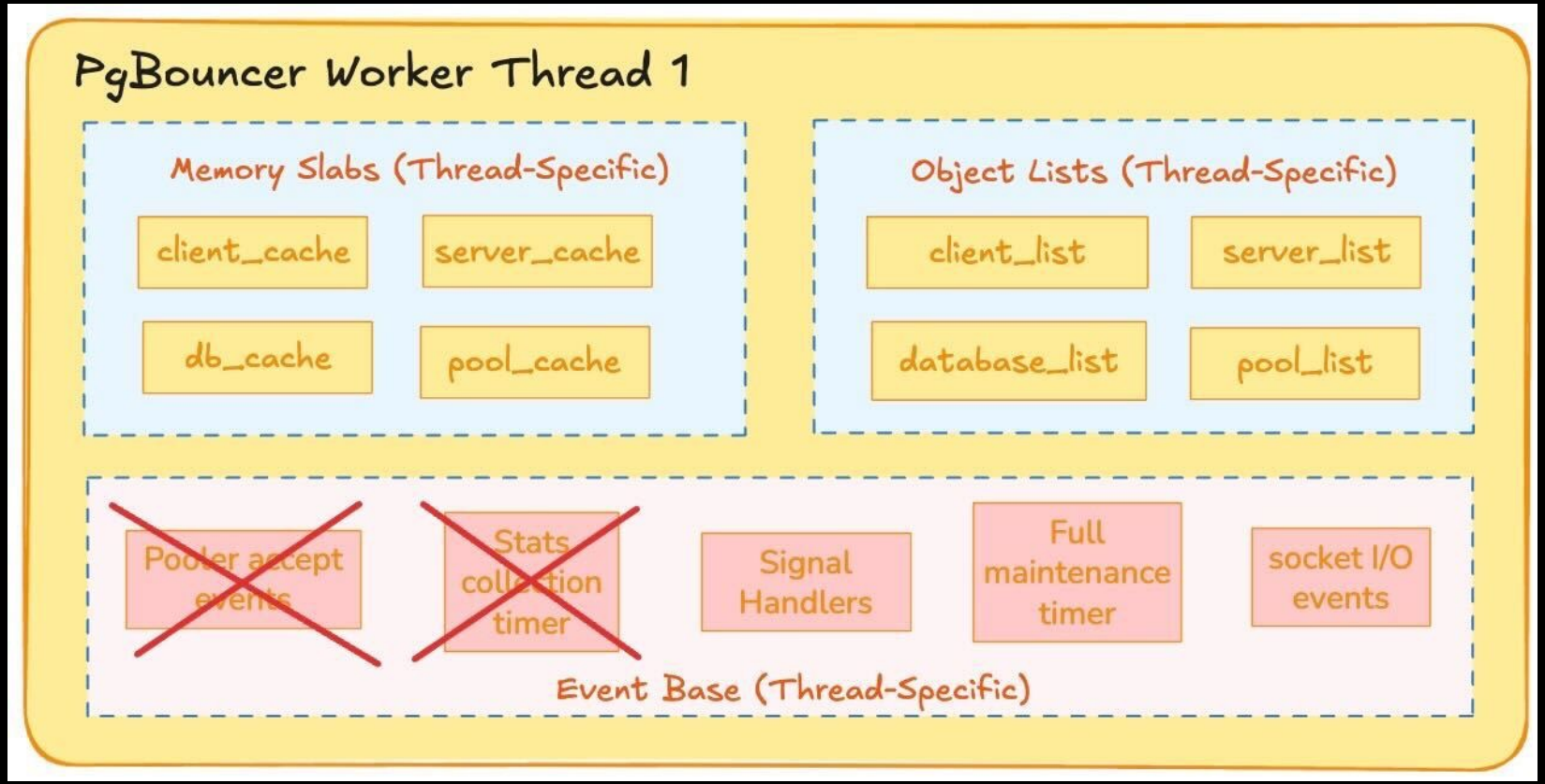
Bloomberg

Engineering

Multi-Threaded PgBouncer: Thread Model

Worker thread:

- Each worker maintains its own event loop, connection pools, and memory slabs



Bloomberg

Engineering

Implementation Challenges

1. Short Critical Sections

2. Deadlocks

3. Cross-thread operations

TechAtBloomberg.com

Bloomberg

Engineering

How Close to Lock-Free?

- Stats
- Admin (KILL Client, KILL db …)
- Limits (max_db_connections …)
- Prepared Statements

Thread Safety

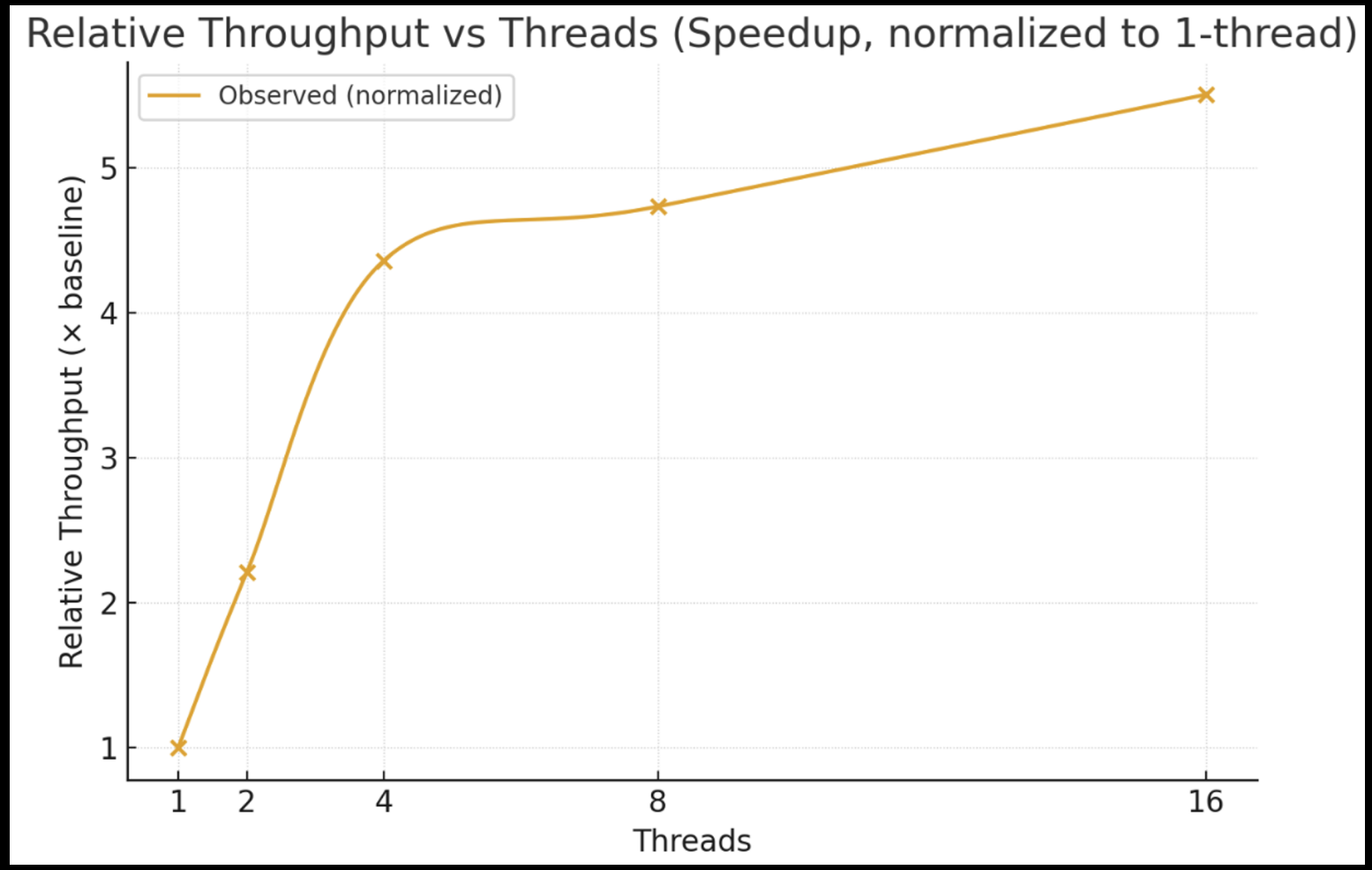
- Locking
- Thread Pausing

TechAtBloomberg.com

Bloomberg

Engineering

Benchmarking - PgBouncer on a dedicated server

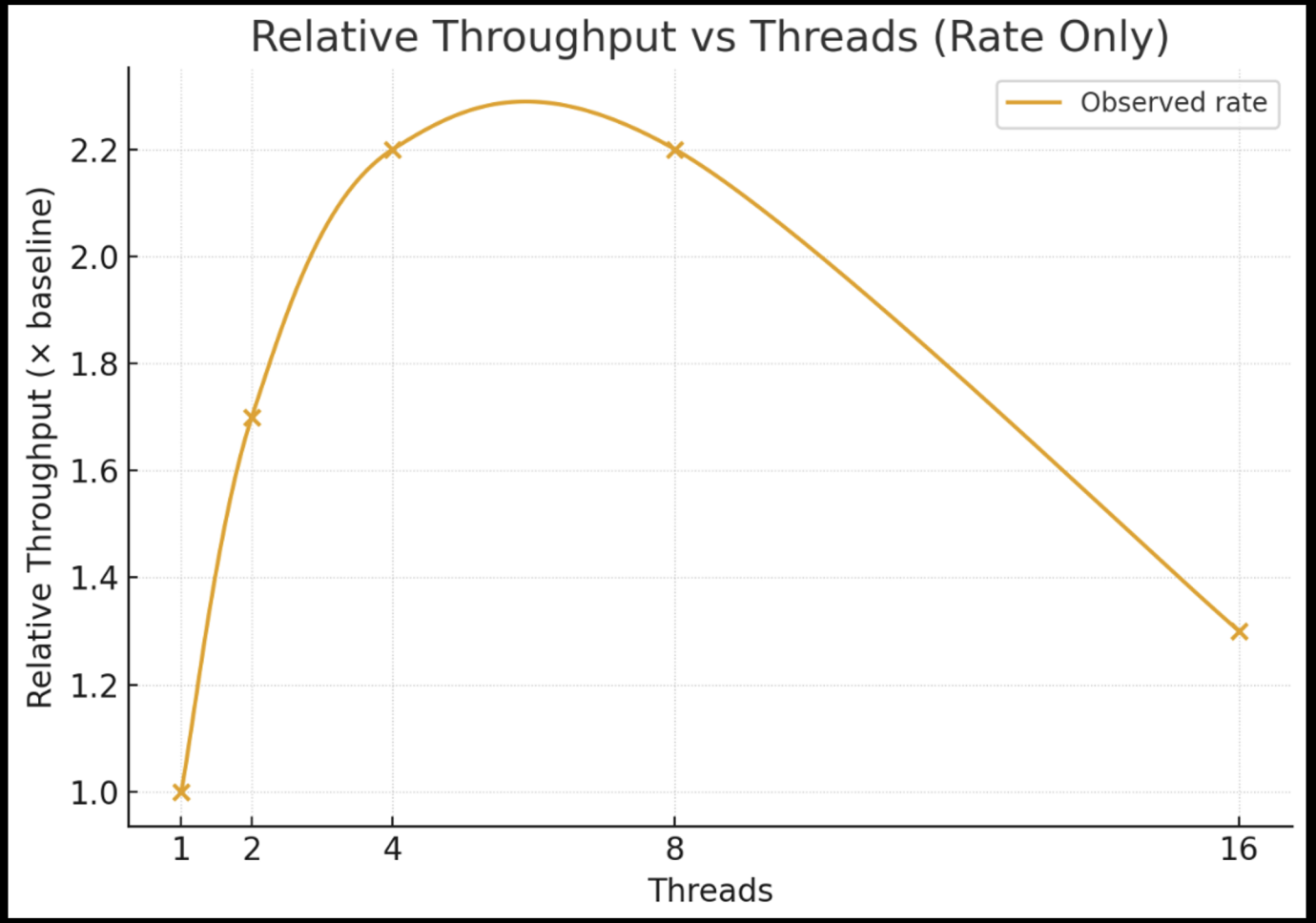


TechAtBloomberg.com

Bloomberg

Engineering

Benchmarking - PgBouncer on a Postgres server

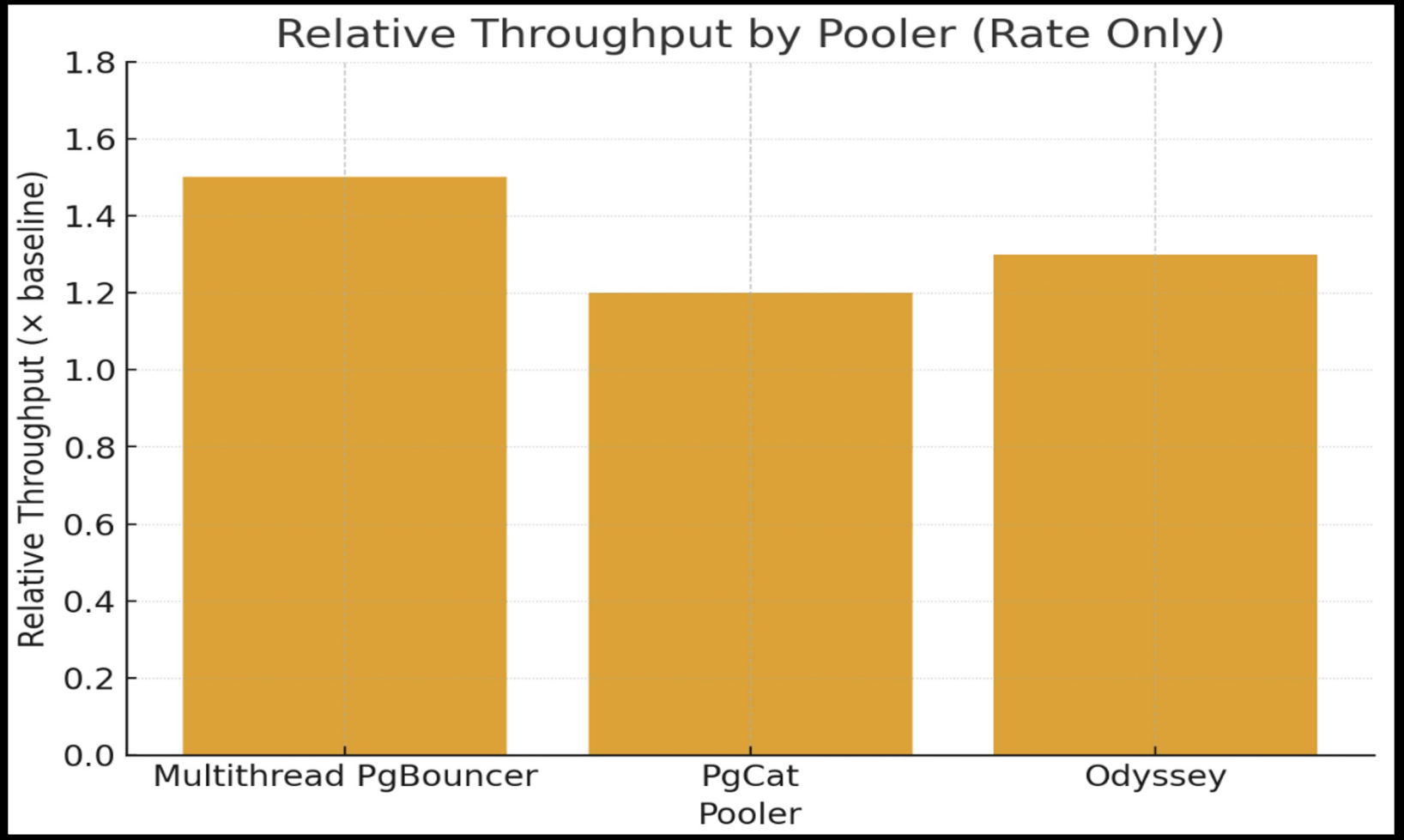


TechAtBloomberg.com

Bloomberg

Engineering

Benchmarking - Poolers on a Postgres server



TechAtBloomberg.com

Bloomberg

Engineering

Squeeze More from Threads

- Avoid frequent stats collection; longer stat period
- Avoid frequent automated admin actions
- Finding the Right PgBouncer Settings

TechAtBloomberg.com

Bloomberg

Engineering

What’s Different: Connection Limits in Multithreaded Mode

- **Less strict accuracy:** Limits are not as precise as in single-threaded mode
- **Bounded deviation:** Actual connections may exceed the limit slightly, but never more than *limit* + *thread count*

Rollout Plan!

PR: https://github.com/pgbouncer/pgbouncer/pull/1386

How to Use

- **No changes:** If you don’t change anything, PgBouncer behaves exactly as before (single-threaded)
- **Opt-in multithread:** Enable by thread_number in config
- **Dual mode during rollout:** We’ll keep the single-threaded mode until bugs and performance issues are fully resolved
- **Same binary:** Use the **same binary** to benchmark and test both modes

Acknowledgement

Bloomberg’s Databases Guild ❤️ Open Source Software!

TechAtBloomberg.com

Bloomberg

Engineering

Thank you!

https://TechAtBloomberg.com/

Contact me: gyang193@bloomberg.net

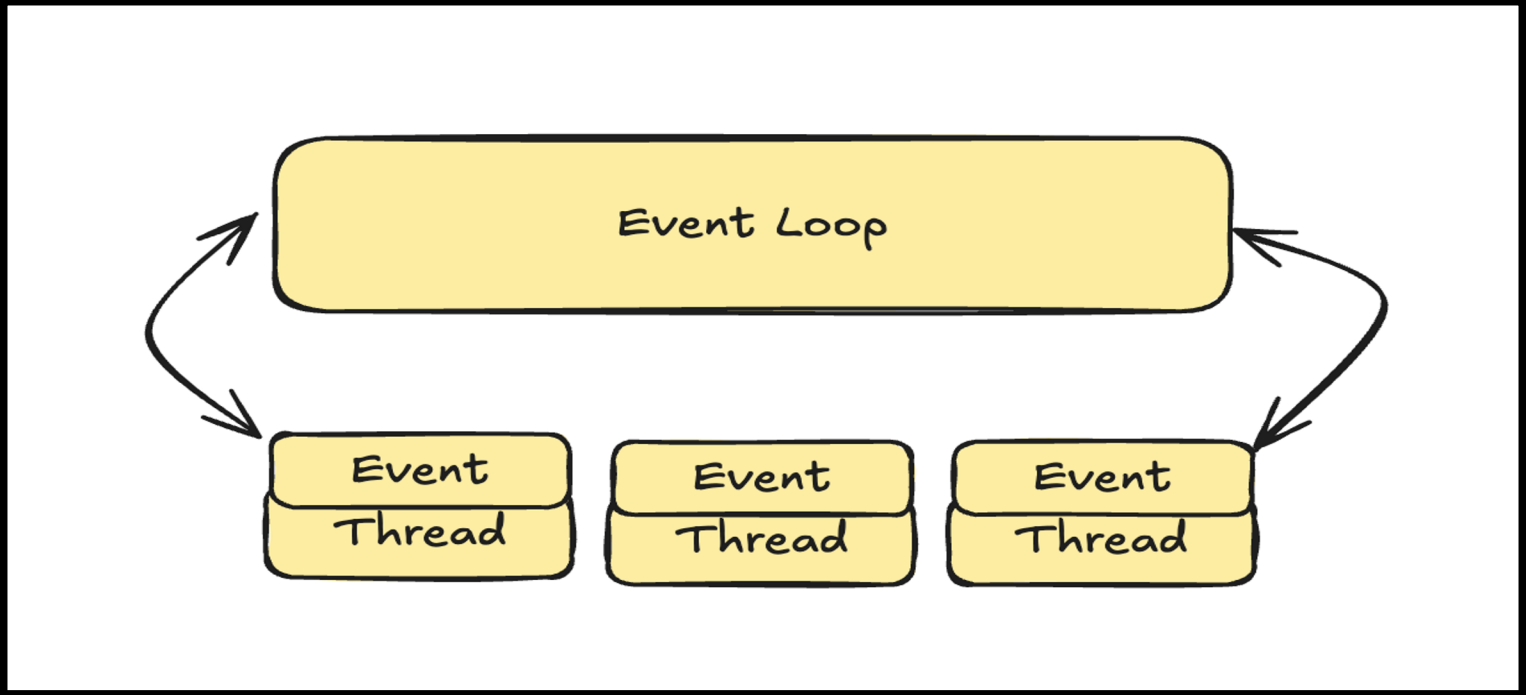
Bloomberg

Engineering

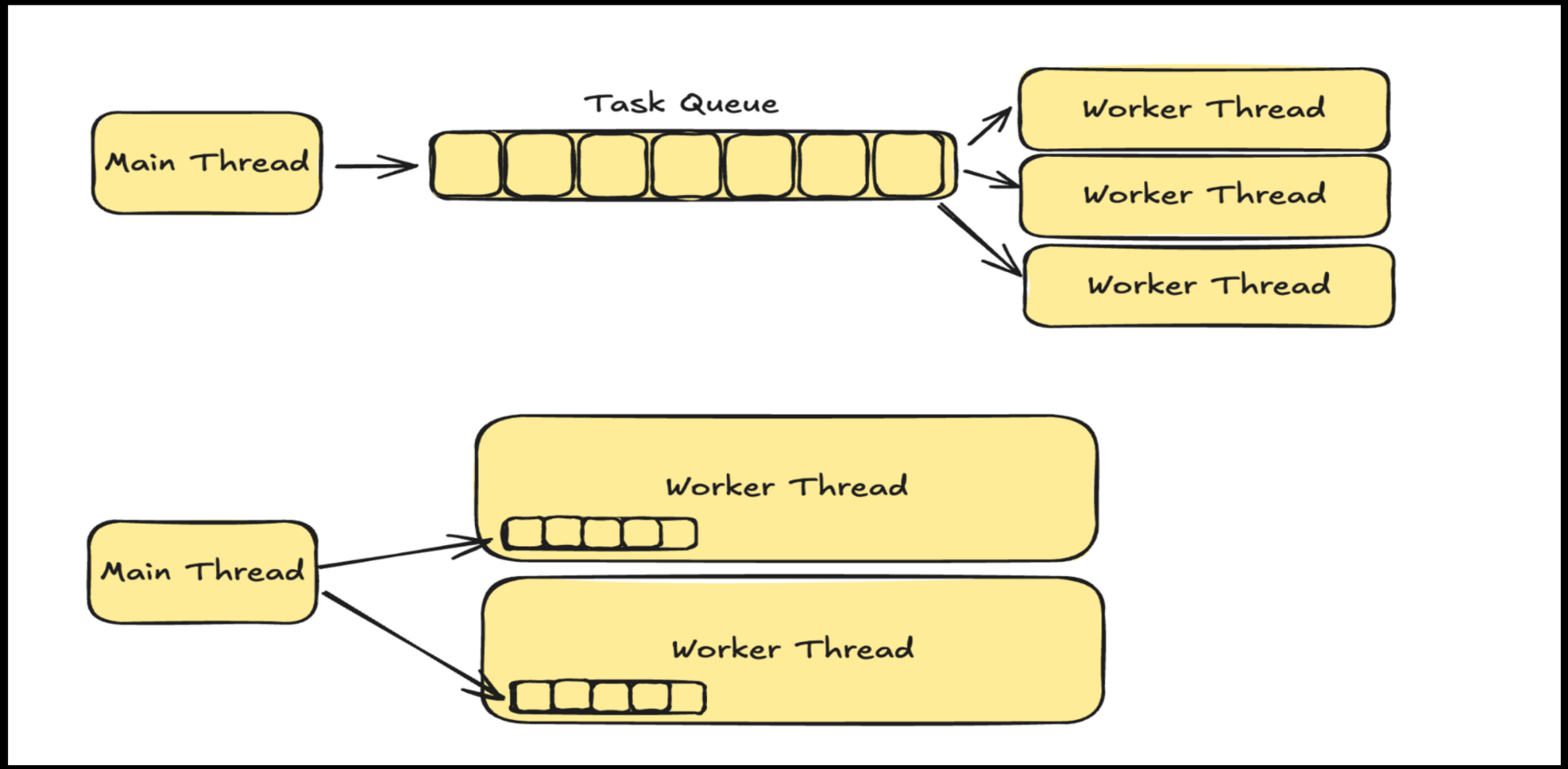
TechAtBloomberg.com

Single Event Loop vs. Multiple Event Loops

- Multiple threads dispatch a single event base, but this would force all threads to share all data structures, requiring excessive locking



Pull vs. Push



Bloomberg

Engineering

Tuning the Accept/Consume Ratio

- Ratio is difficult to tune, influenced by many factors
- Current implementation shows no obvious backlog
- A switch to **limit accept rate** was considered