UCM: Fast and Maintainable User-space RDMA Connection Setup

Huijun Shen[1], Jian Yang[2], Zelong Yue[2], Xingyu Guo[1], Xijin Yin[1], Lang An[2], Yulin Chen[2], Jie Ding[2], Hongyu Wu[2], Yong Zhang[2], Jianxi Ye[2], Guo Chen[1]

[1]Hunan University [2]ByteDance

RDMA is widely deployed

- RDMA in production-level applications:
  - Cloud storage, Recommendation system, LLM inference/training…



…

- RDMA connection setup approaches
  - ➢ Socket APIs : out-of-band connection
  - ➢ CM* APIs : in-band connection

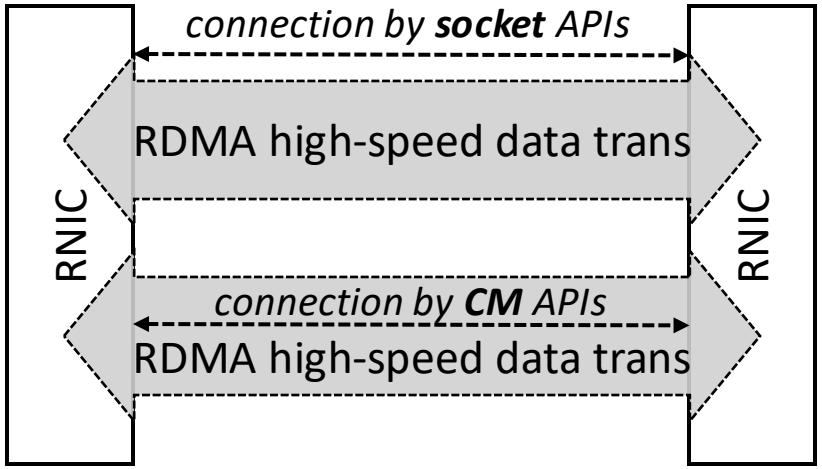
* The **RDMA CM** is a **communication manager** used to setup reliable, connected and unreliable datagram data transfers, and provides standard APIs defined by librdmacm library. *https://man7.org/linux/man-pages/man7/rdma_cm.7.html*

RDMA is widely deployed

- RDMA in production-level applications:
  - Cloud storage, Recommendation system, LLM inference/training…

火山引擎 …

- RDMA connection setup approaches
  - ➢ Socket APIs : out-of-band connection
  - ➢ CM* APIs : in-band connection



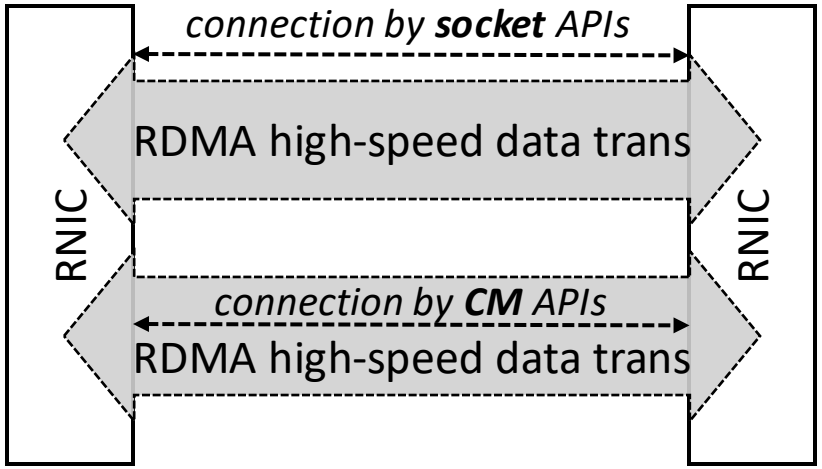
* The **RDMA CM** is a **communication manager** used to setup reliable, connected and unreliable datagram data transfers, and provides standard APIs defined by librdmacm library. *https://man7.org/linux/man-pages/man7/rdma_cm.7.html*

RDMA is widely deployed

- RDMA in production-level applications:
  - Cloud storage, Recommendation system, LLM inference/training…

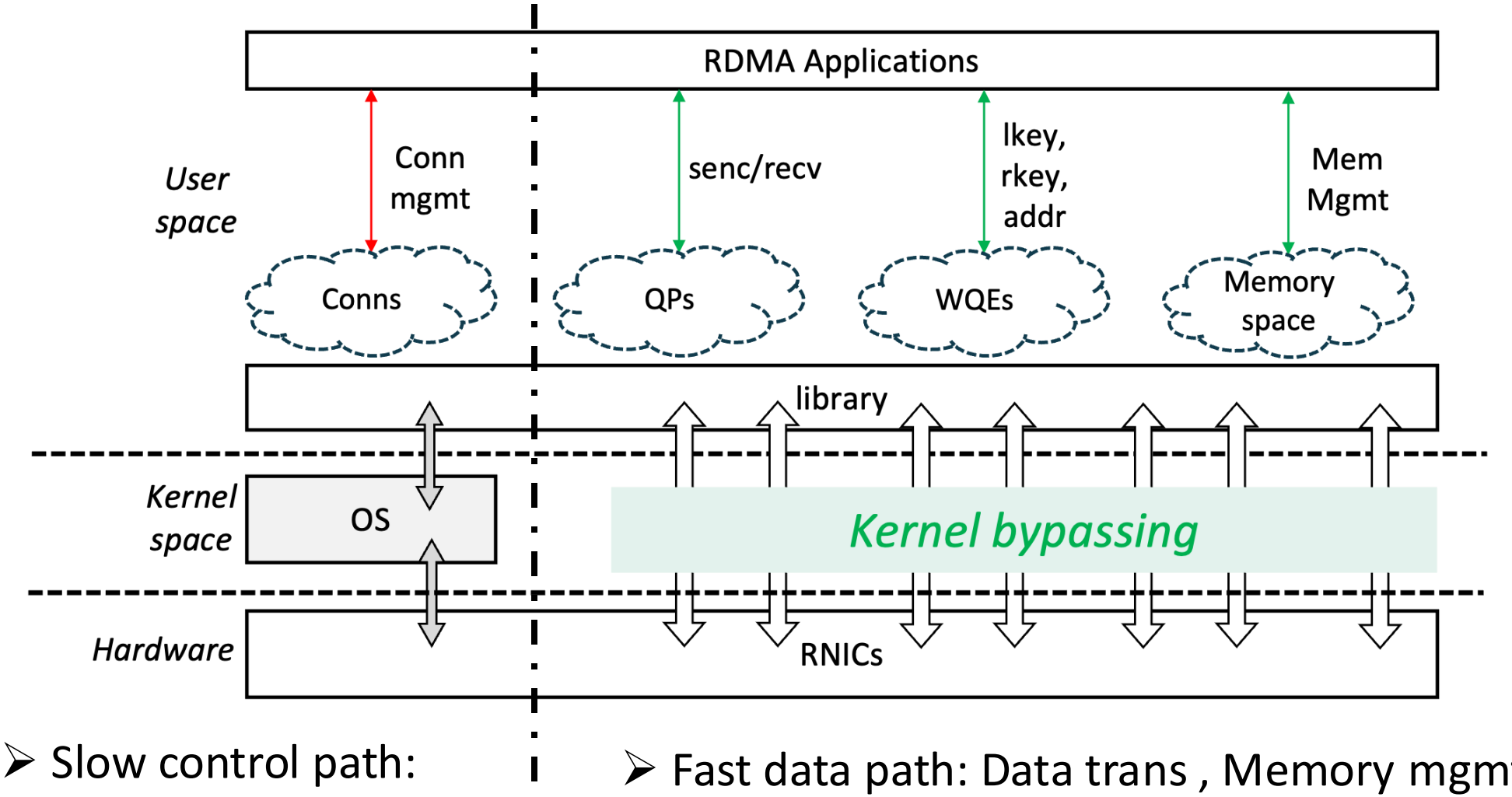
火山引擎 …

- RDMA connection setup approaches
  - ➢ Socket APIs : out-of-band connection
  - ➢ **CM* APIs : in-band connection**



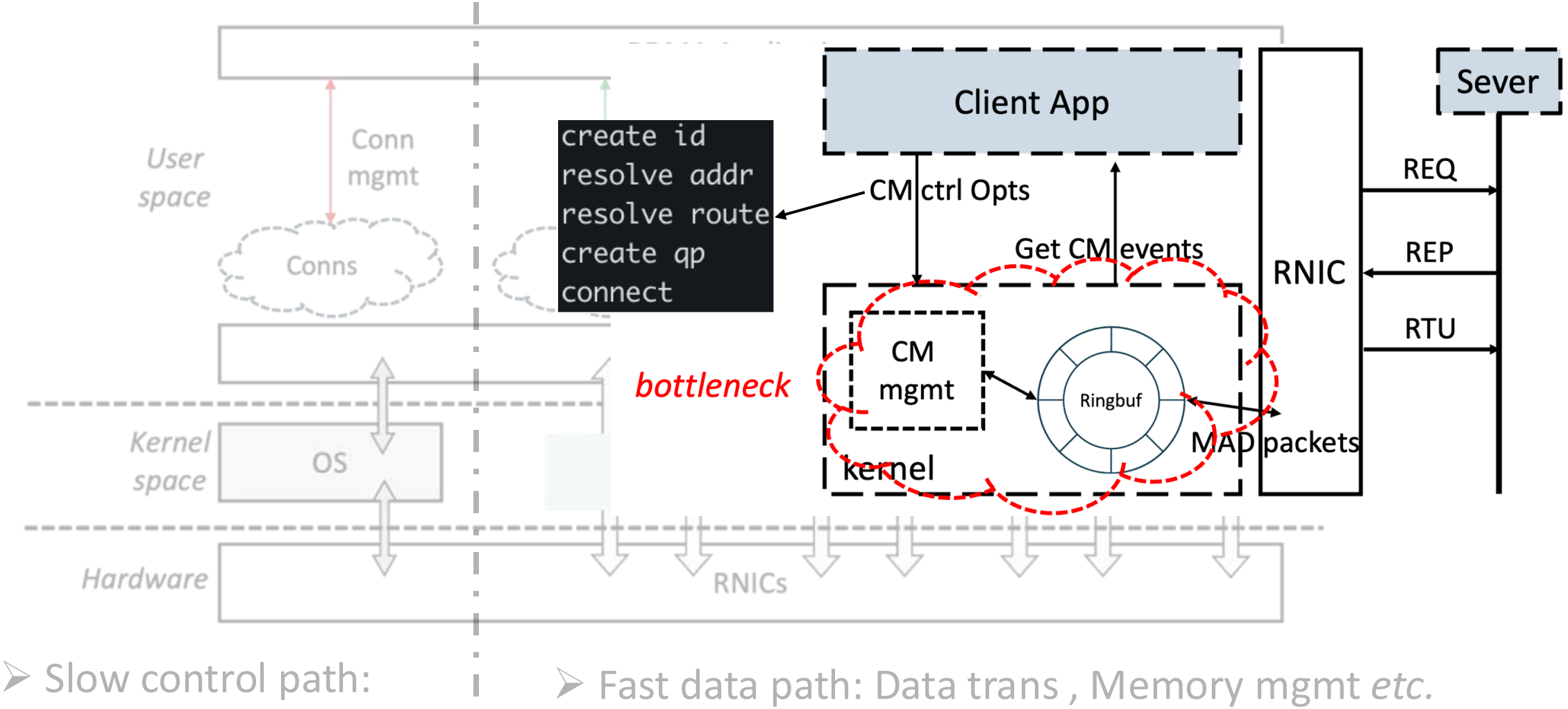
* The **RDMA CM** is a **communication manager** used to setup reliable, connected and unreliable datagram data transfers, and provides standard APIs defined by librdmacm library. *https://man7.org/linux/man-pages/man7/rdma_cm.7.html*

RDMA fast path and slow path



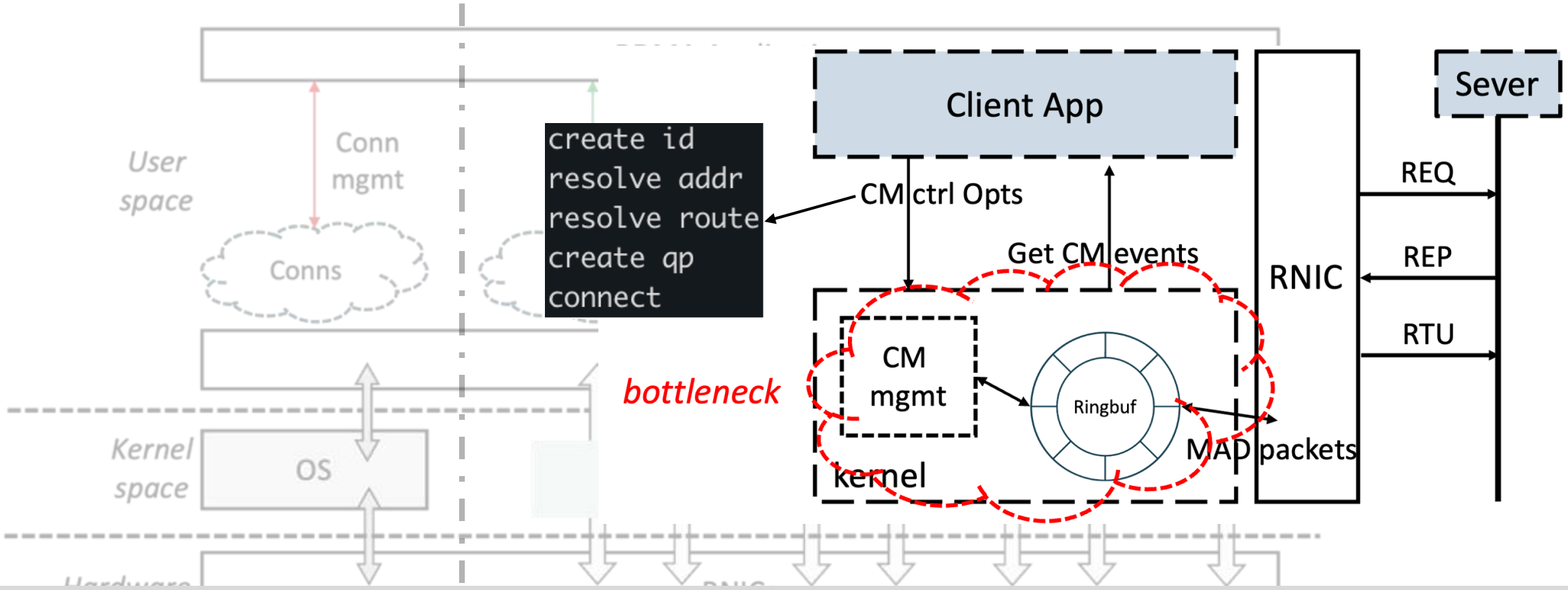
- Slow control path: Connection mgmt, *etc.*
- Fast data path: Data trans , Memory mgmt *etc.*

RDMA control path is slow



- Slow control path: Connection mgmt, *etc.*
- Fast data path: Data trans , Memory mgmt *etc.*

RDMA control path is slow



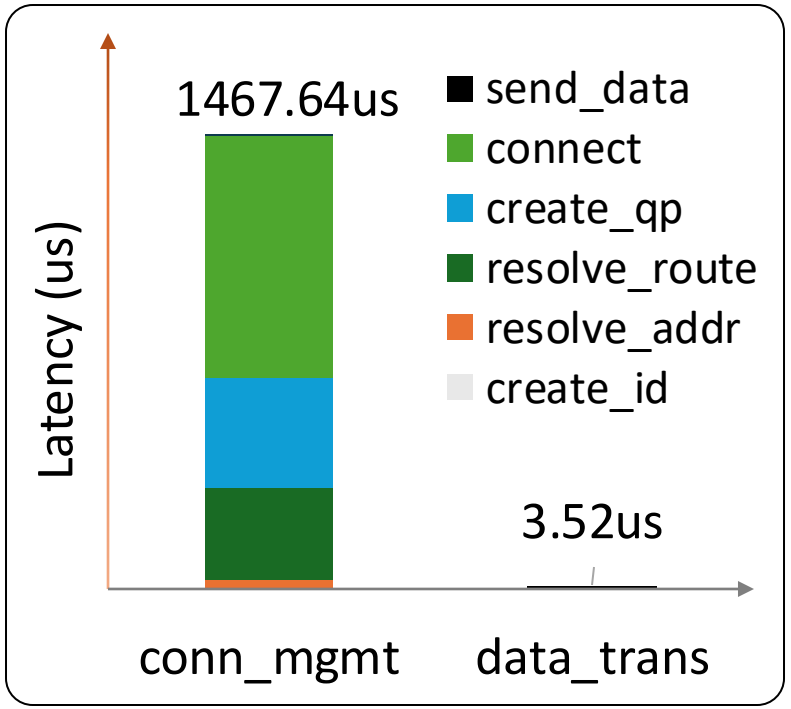
RDMA conn setup process is inefficient and difficult to monitor.

- Slow control path: Connection mgmt, *etc.*
- Fast data path: Data trans , Memory mgmt *etc.*

Problems of RDMA connection setup

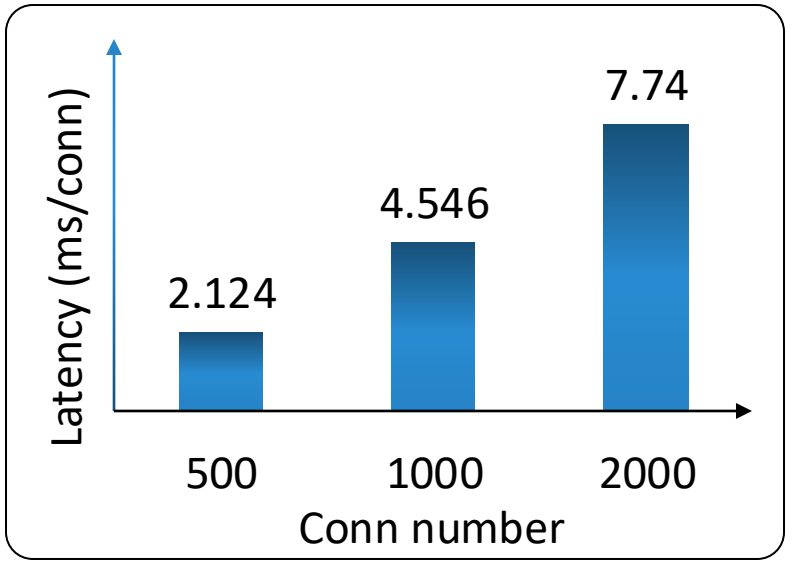
- RDMA connection management (CM) is **Inefficient.**

Overheads in one RDMA transmission



➢ **High cost**: 1-2ms/conn.

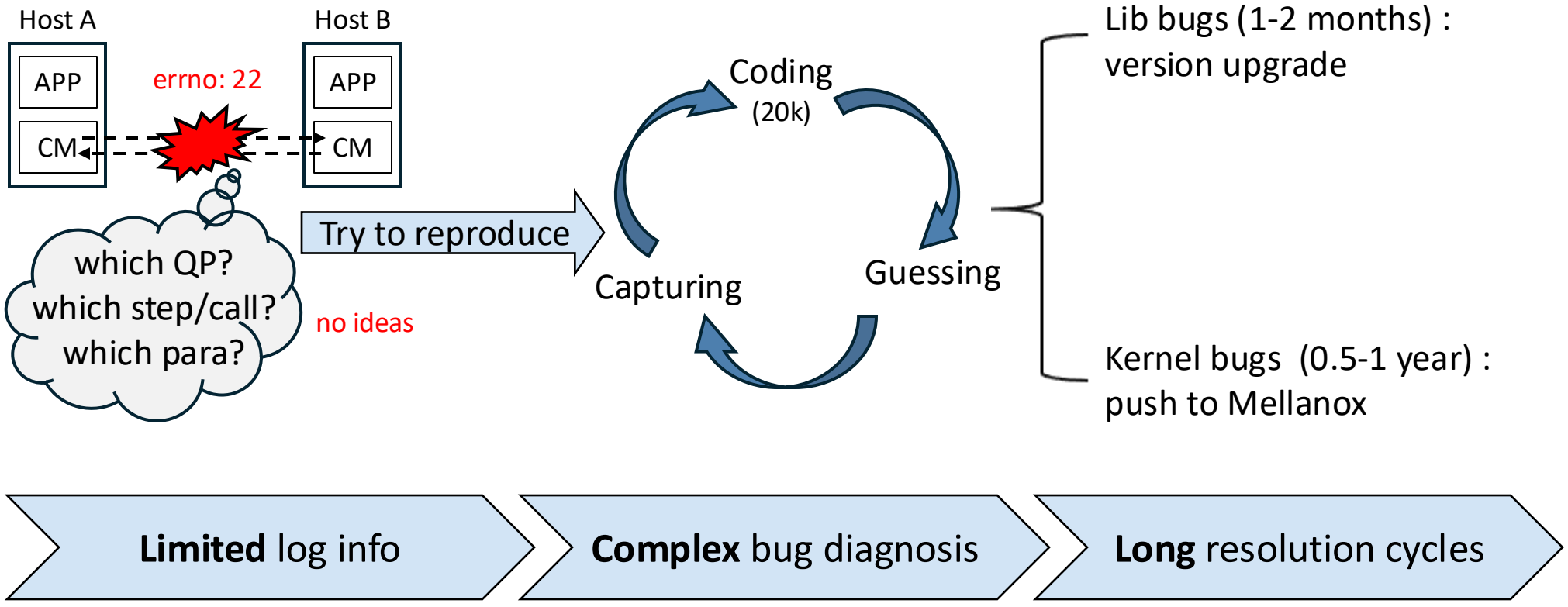
Overheads in large-scale connections



➢ **Bad scalability**: Connection setup efficiency further decreases as scale increases.

Problems of RDMA connection setup

- Production Deployment Practices for RDMA Connections



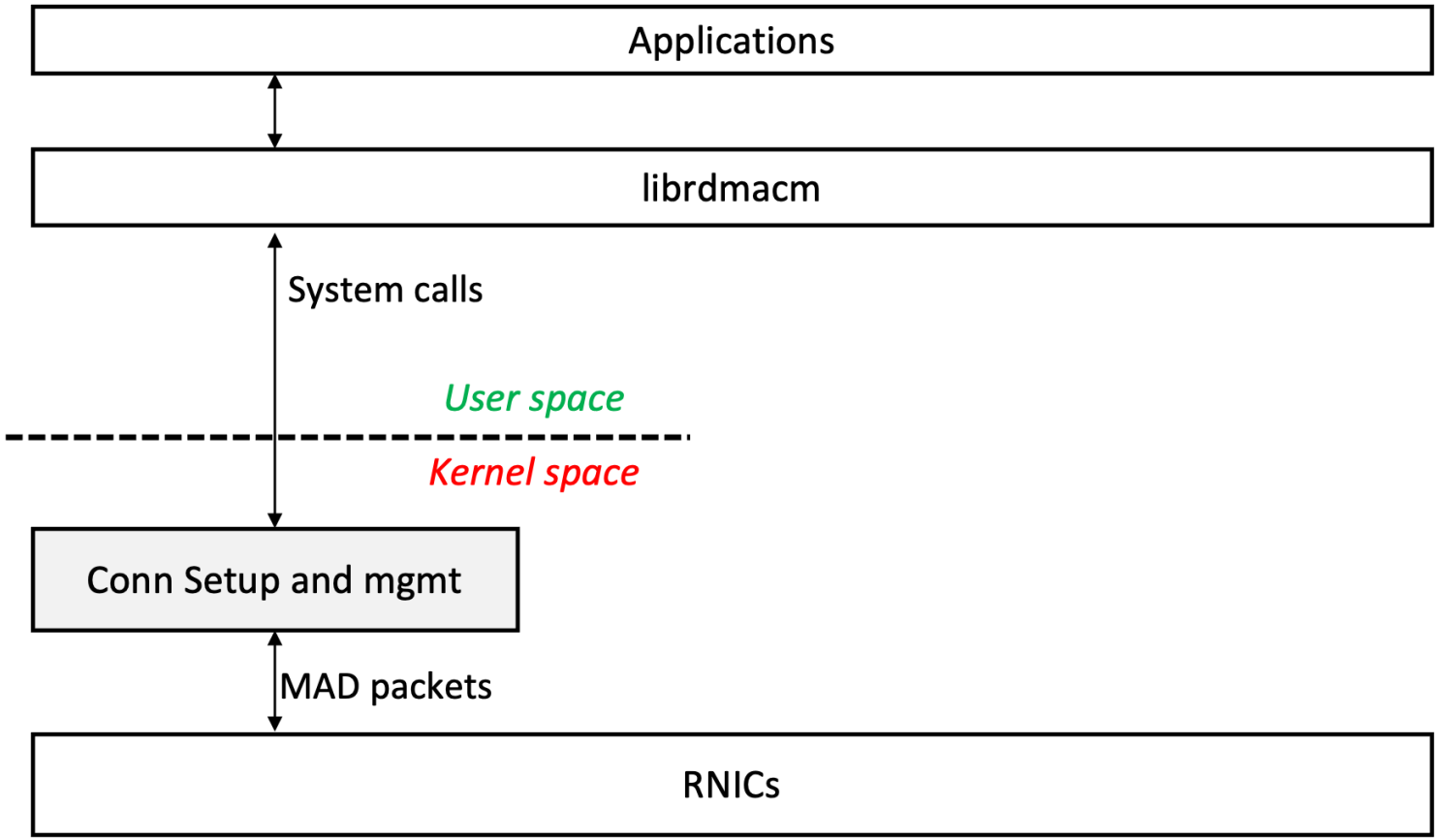
Goals

- Production Deployment Practices for RDMA Connections

Our goal is to develop an new User-space RDMA CM setup approach

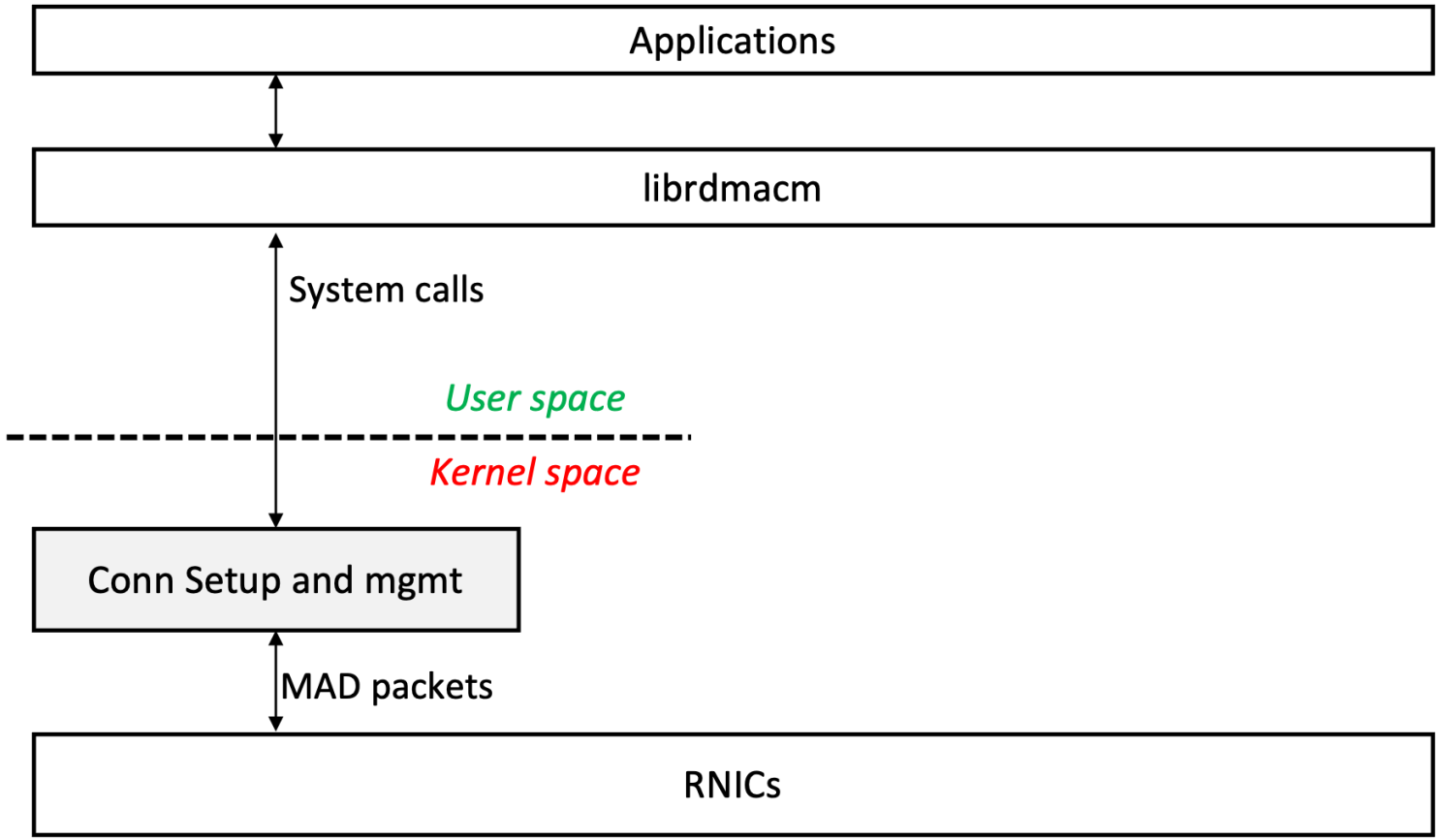
From Kernel to User-space

- **Original RDMA connection management in Linux kernel**



From Kernel to User-space

- **Original RDMA connection management in Linux kernel**

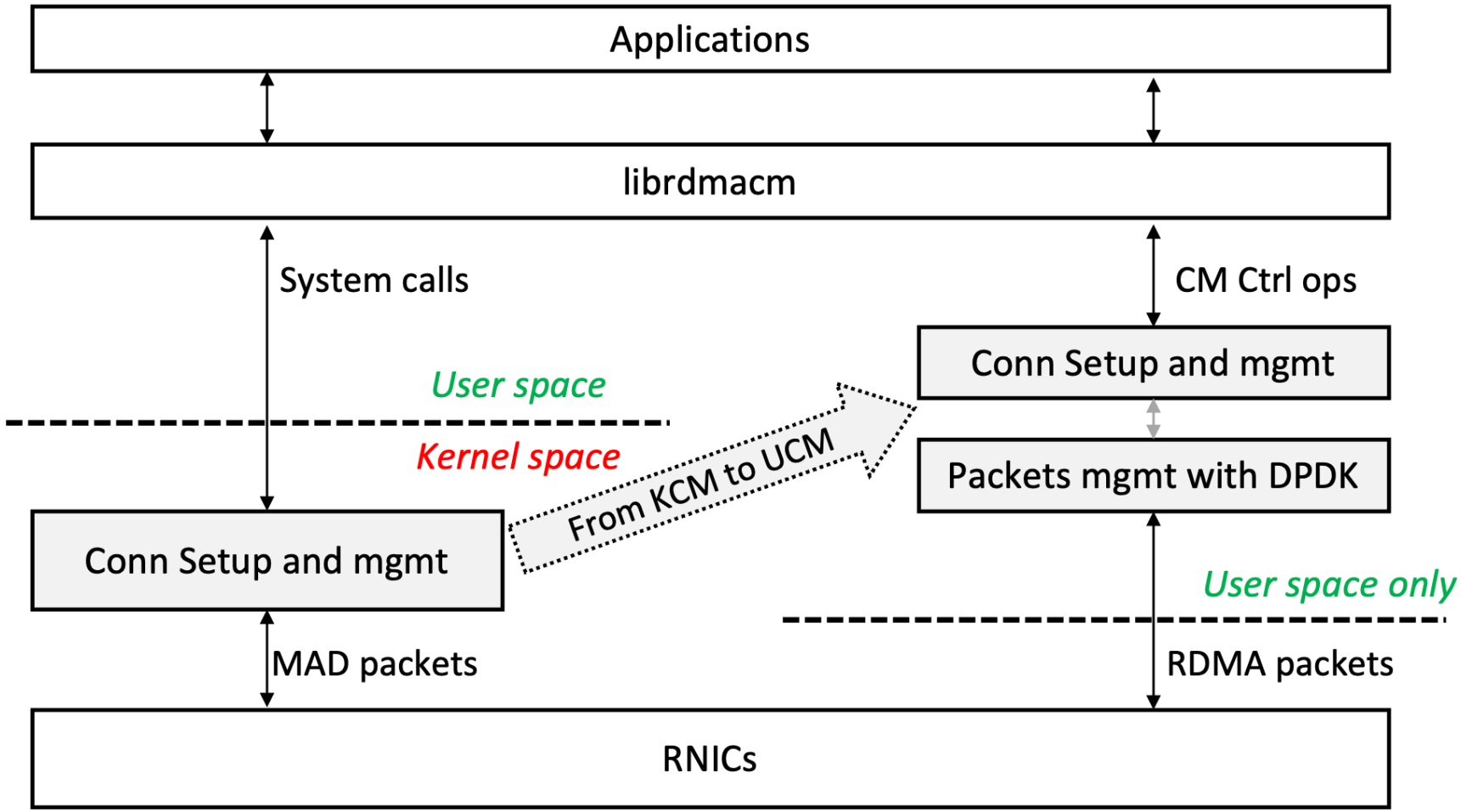


Insights:

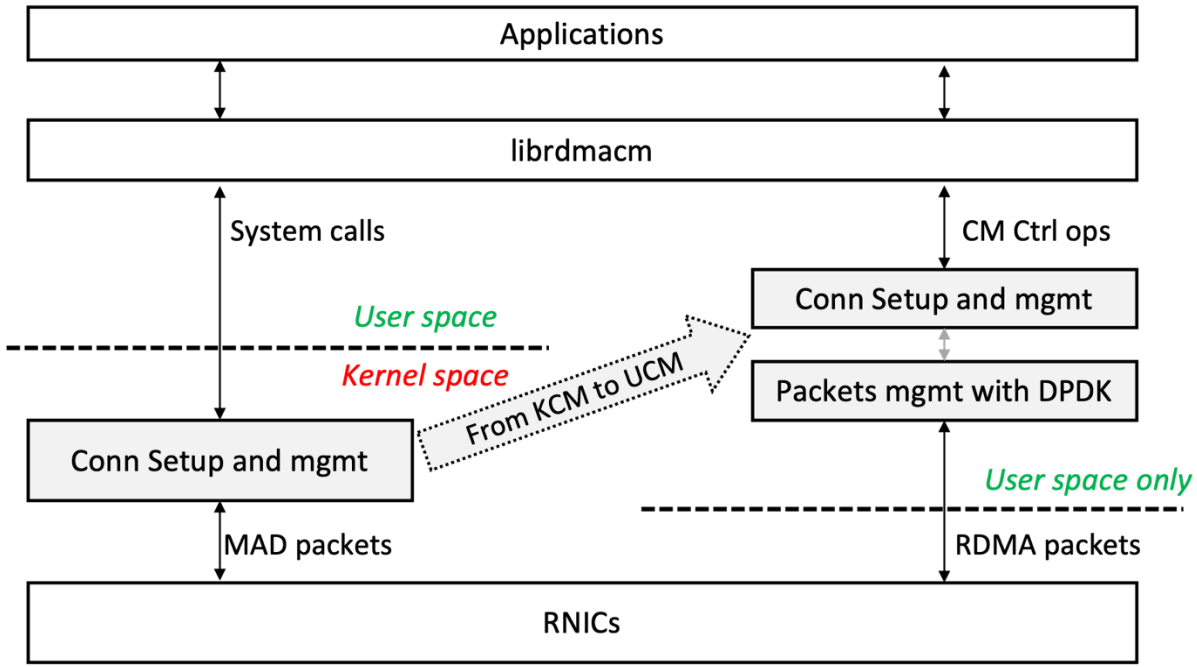
- Choosing CM conn
  - unified APIs
  - in-band path
  - path detection
- Recent User-Space Tech
  - DPDK
  - LibOS

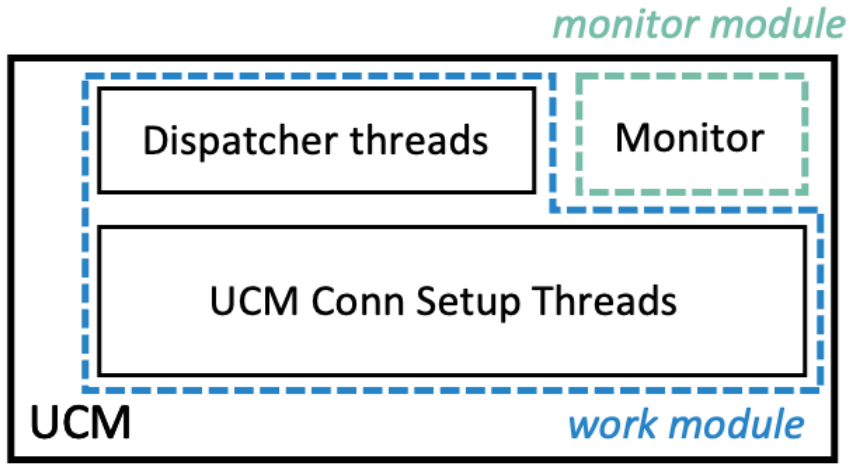
UCM: **U**ser-space RDMA **C**onnection **M**anagement

- **Our idea to bypass kernel for better performance**



UCM overview

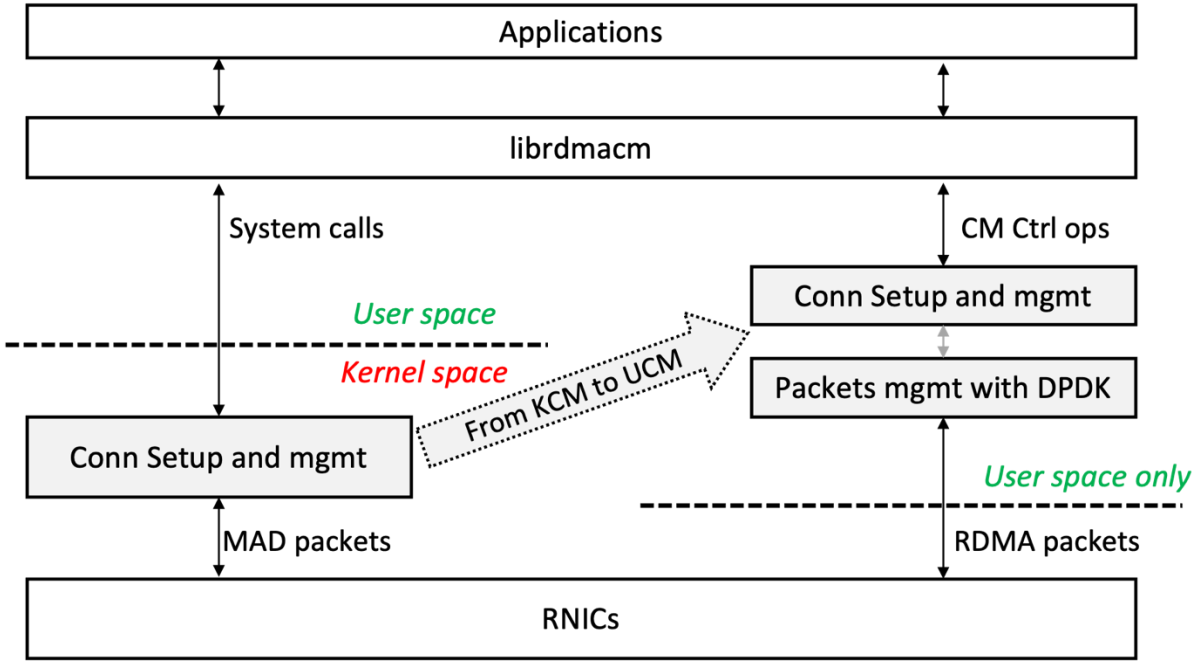




- UCM Framework
  - Work module: setup and manage RDMA connections in user-space
  - Monitor module: offer multi-method monitoring approaches for developers

UCM design --- *UCM work module*

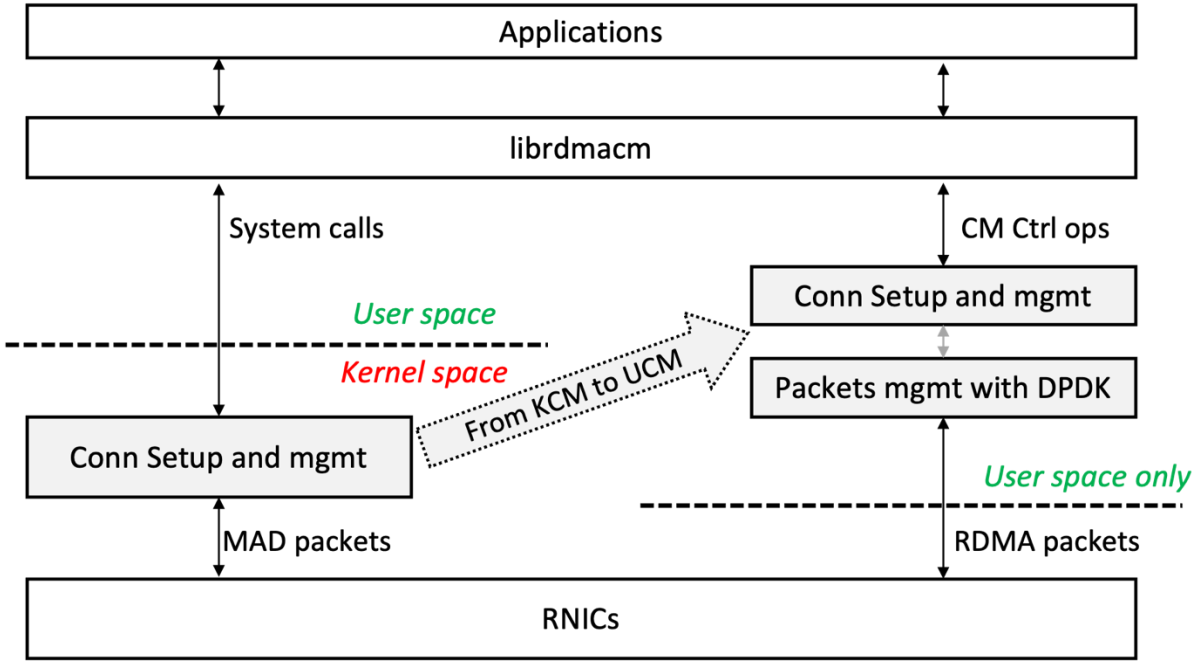
- How to bypass kernel?



- To support muti-thread, how to deal with thread communication?

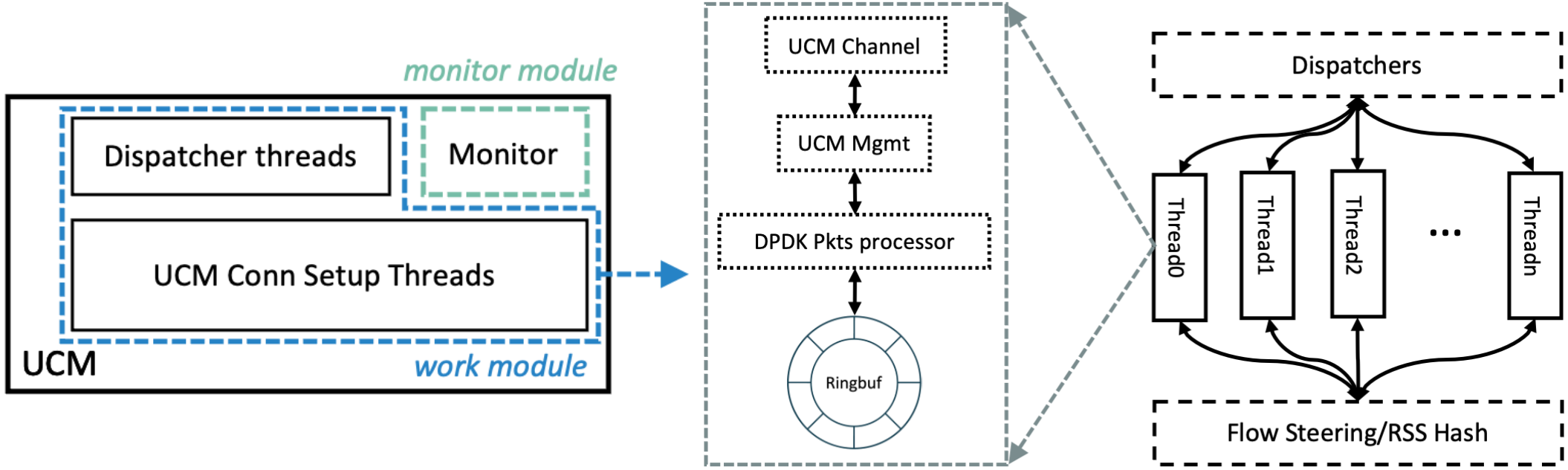
UCM design --- *UCM work module*

- How to bypass kernel?
  - Conn Mgmt in LibOS
  - on-loading packets processing with DPDK



- To support muti-thread, how to deal with thread communication?

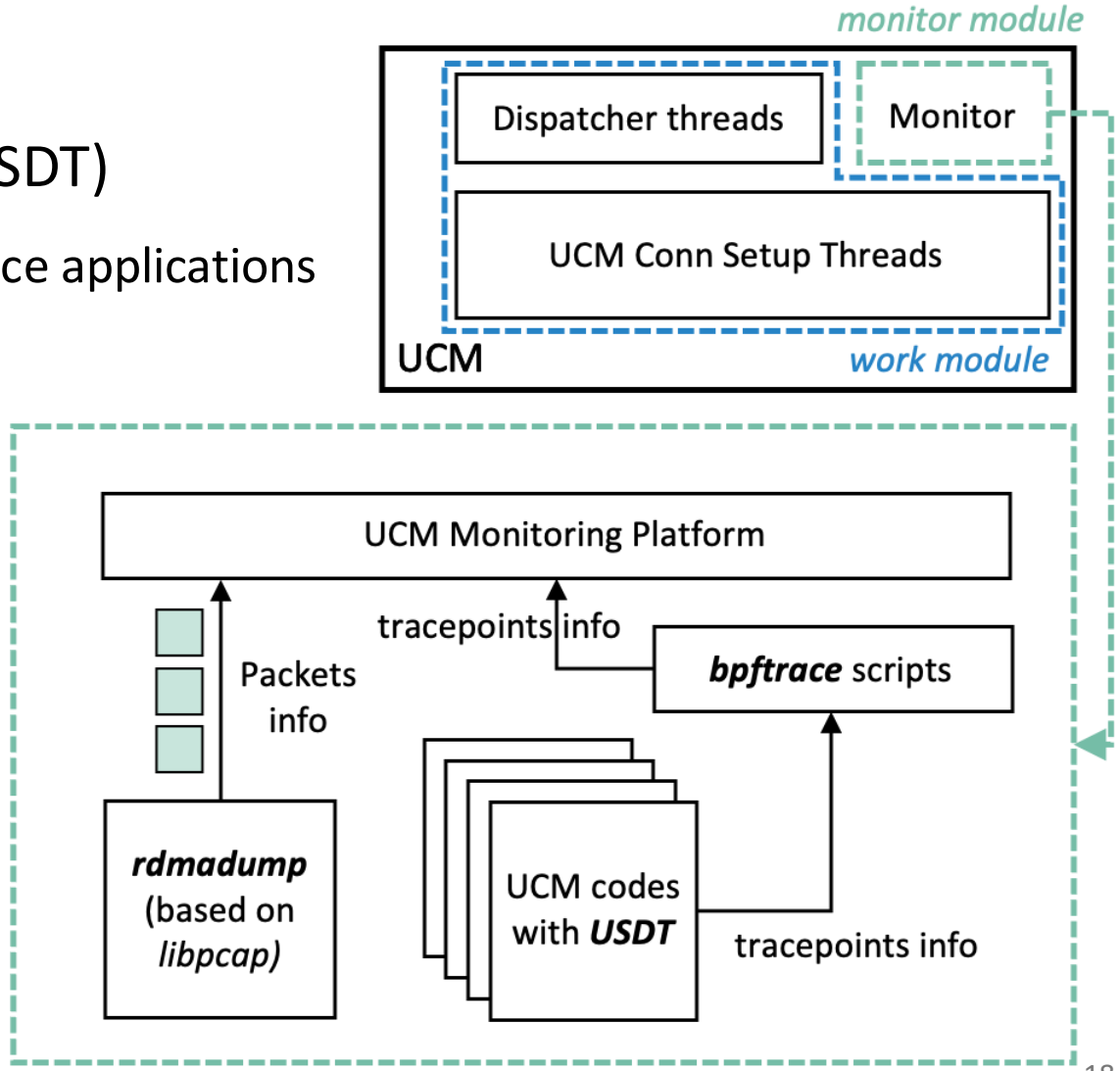
UCM design --- *UCM work module*



- To support muti-thread, UCM leverages multi-thread lock-free management
  - NIC features : Flow Steering, RSS hash
  - Make sure that each connection's related information is accessed and managed by only one thread

UCM design --- *UCM monitor module*

- Monitoring approaches
  - ➢ User Statically-Defined Tracing (USDT)
    - Add dynamic *tracepoints* to user-space applications
    - Dynamic Monitoring with USDT
  - ➢ Self-defined capturing tool
    - Based on *libpcap*



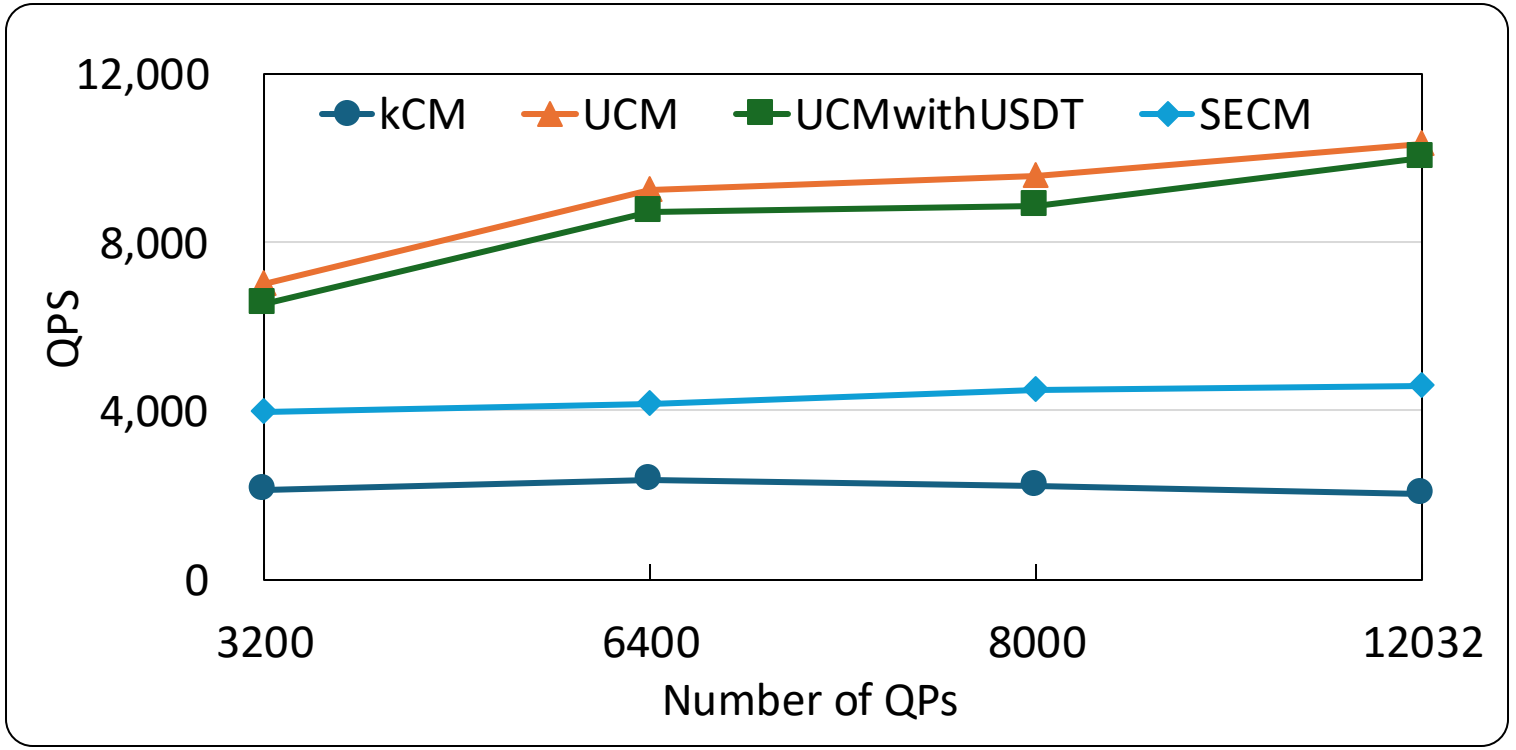
Evaluation

- End-to-end Testbed
  - Hardware: Mellanox ConnectX-6 Dx EN
  - Software: cmtime, perftest, Mellanox OFED 5.8 driver,
  - Comparisons: **UCM**, kCM (original RDMA), SECM

- Goals
  - ➢ Compare UCM's connection setup speed under different scenarios with the *sota* approaches (single-threaded, multi-threaded, and extreme application)
  - ➢ Evaluate the impact of UCM's maintenance overheads on production applications
  - ➢ Showcase successful maintenance experiences with UCM

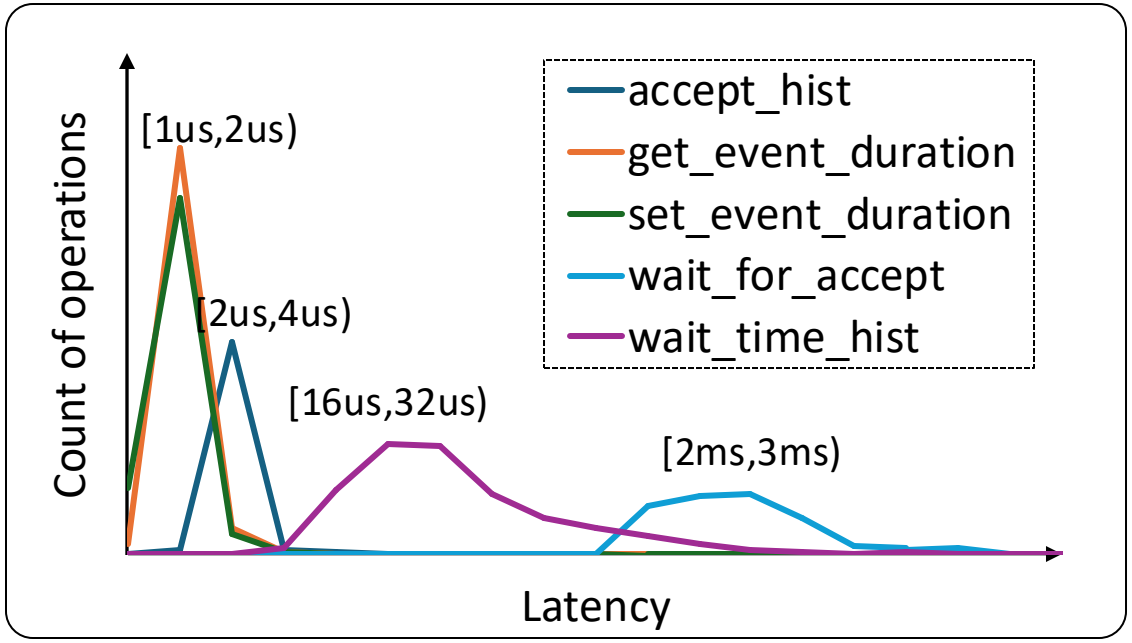
Evaluation#1--- *UCM work performance (RPC apps)*

- Deploy UCM with an enterprise RPC framework in ByteDance
  - UCM's QPS (number of QPs per second) performs 3.3-5.1x that of kCM and 1.8-2.2x of SECM.
  - The extra overhead introduced by enabling USDT is only 3.2%-7%.

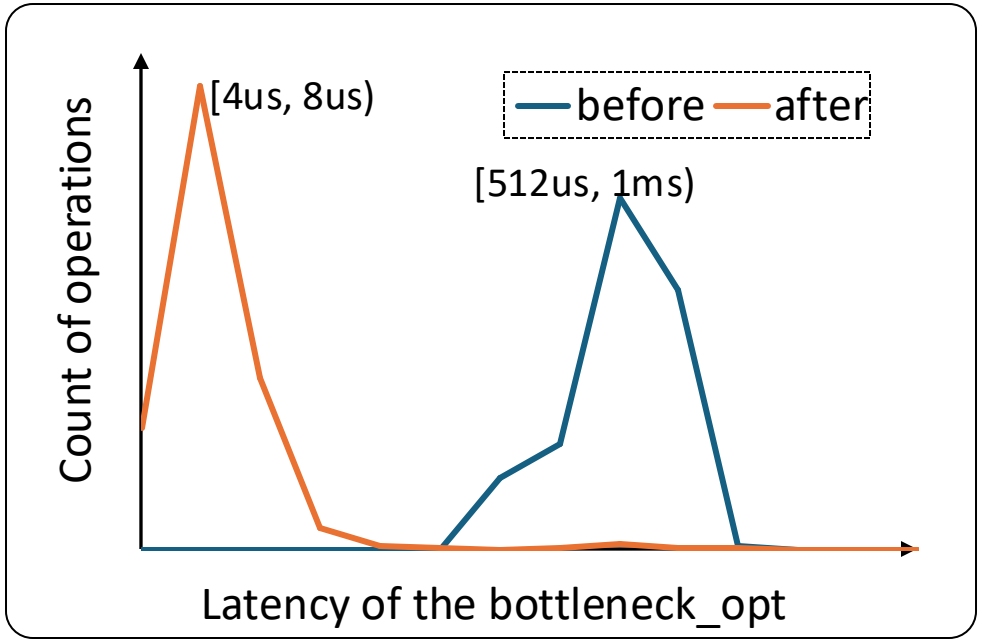


Evaluation#2--- *UCM monitoring effort*

- Monitoring for optimization
  - latency of every operation in online connections
  - performance after optimization



Latency Monitoring for Conn Setup Steps



Latency Comparison for a Single Step

Conclusion & Future work

- Related work
  - KRCore[1] : a shared connection pool
  - SECM[2] : parallel connection setup

Dependency	KRCore	SECM	UCM
User app	✓	✓	
RDMA library	✓	✓	✓
RNIC	✓		

- **UCM: The fist pure user-space RDMA connection management framework.**
  - ✓ Dramatically accelerated RDMA CM setup efficiency
  - ✓ Better observability for production operations

- We hope UCM will inspire more new possibilities for optimizing the RDMA protocol stack at the software (user space) level.

[1] KRCORE: a microsecond-scale RDMA control plane for elastic computing. (ATC 2022)
[2] SECM: Securely and efficiently connections setup using RDMA-CM. Computer Networks 250 (2024)

Thanks!

Q&A

Contact : shj@hnu.edu.cn