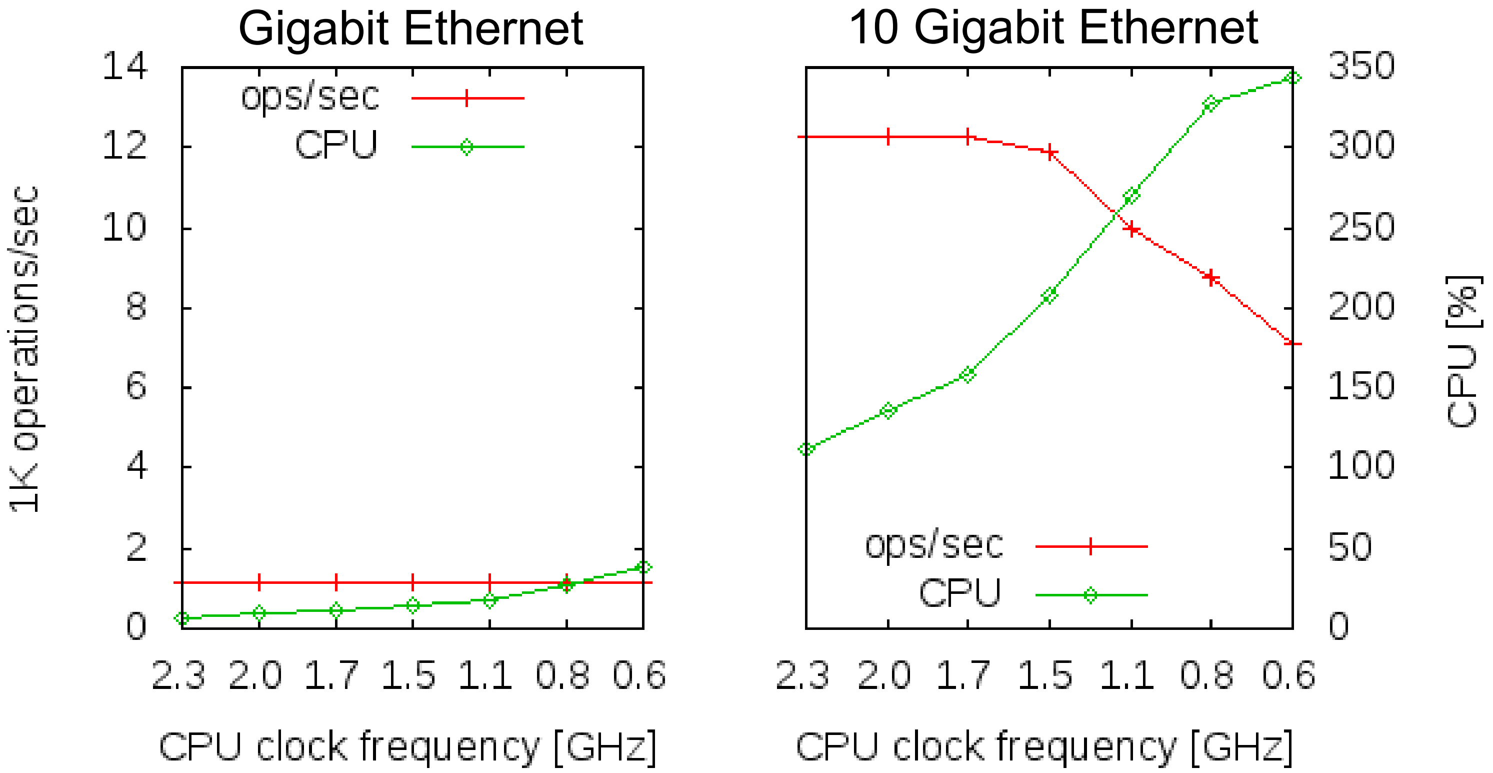
Wimpy Nodes with 10GbE: Leveraging One-Sided Operations in Soft-RDMA to Boost Memcached

Patrick Stuedi,
Animesh Trivedi, Bernard Metzler
IBM Research Zurich

Introduction

- Known: key/value stores and low-power CPUs/cores go well together
    - Workload typically not compute-heavy
    - Slower CPU clock sufficient
    - Easy to parallelize, distribute load over many low-power cores
    - Examples: FAWN, Facebook/Tilera
- What are the implications of attaching 10 GbE NICs to the low-power key/value storage nodes?
    - Improved latency
    - What about the CPU load?

Memcached/GET: 1GbE vs 10GbE



- Setup: 100K GET requests, 6 clients
- High CPU usage limits performance (ops/sec)

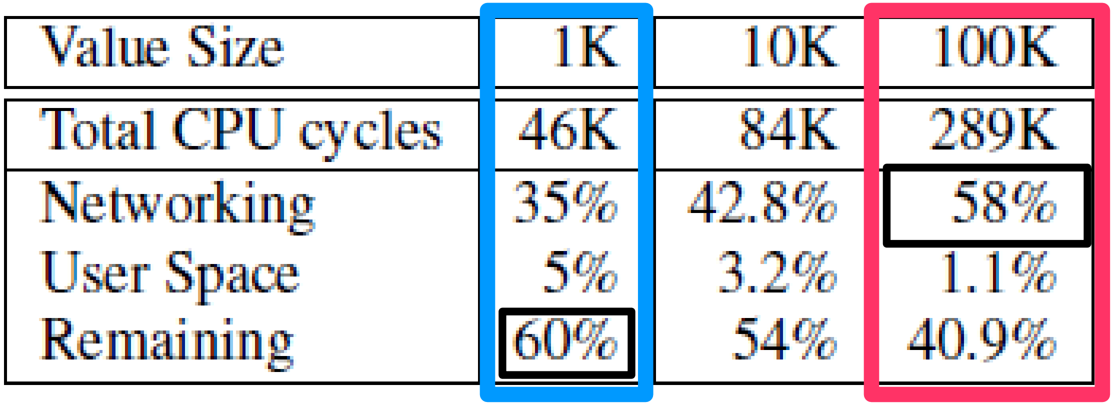
Does multicore help?

- Throughput performance
  - Yes, considering the scaling limitations of Memcached
- Efficiency
  - More cores consume more energy

- Efficient processing on one-core will hopefully translate to efficient multi-core setup.

How is the CPU being used?

- Depends on the size of the value
  - Small values (~1K): 60% of CPU usage scattered across many OS functions (e.g. context switching, etc.)
  - Larger values (~100K): 60% of CPU usage inside network stack



Value Size	1K	10K	100K
Total CPU cycles	46K	84K	289K
Networking	35%	42.8%	58%
User Space	5%	3.2%	1.1%
Remaining	60%	54%	40.9%

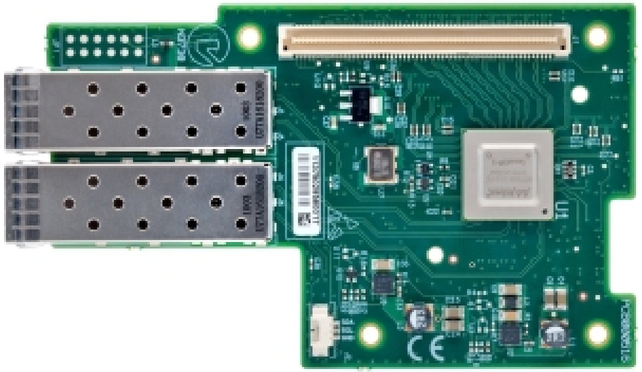
CPU
at 1.1
GHz

Using RDMA

- RDMA: Remote Direct Memory Access
  - Efficient remote memory access
  - Zero-copy (inside the end hosts), low latency, low CPU usage
  - Great! 😁

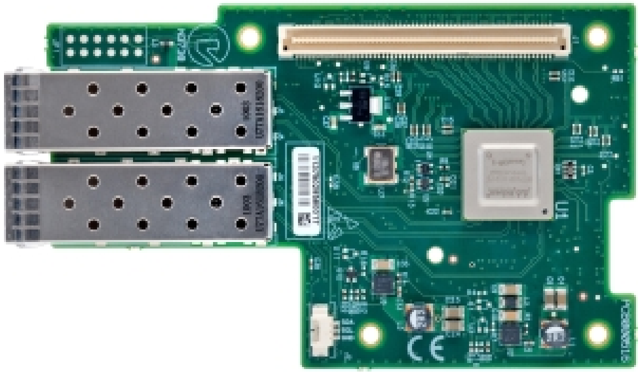
Using RDMA

- RDMA: Remote Direct Memory Access
  - Efficient remote memory access
  - Zero-copy (inside the end hosts), low latency, low CPU usage
  - Great! 😁
- Who has RDMA capable NICs deployed?



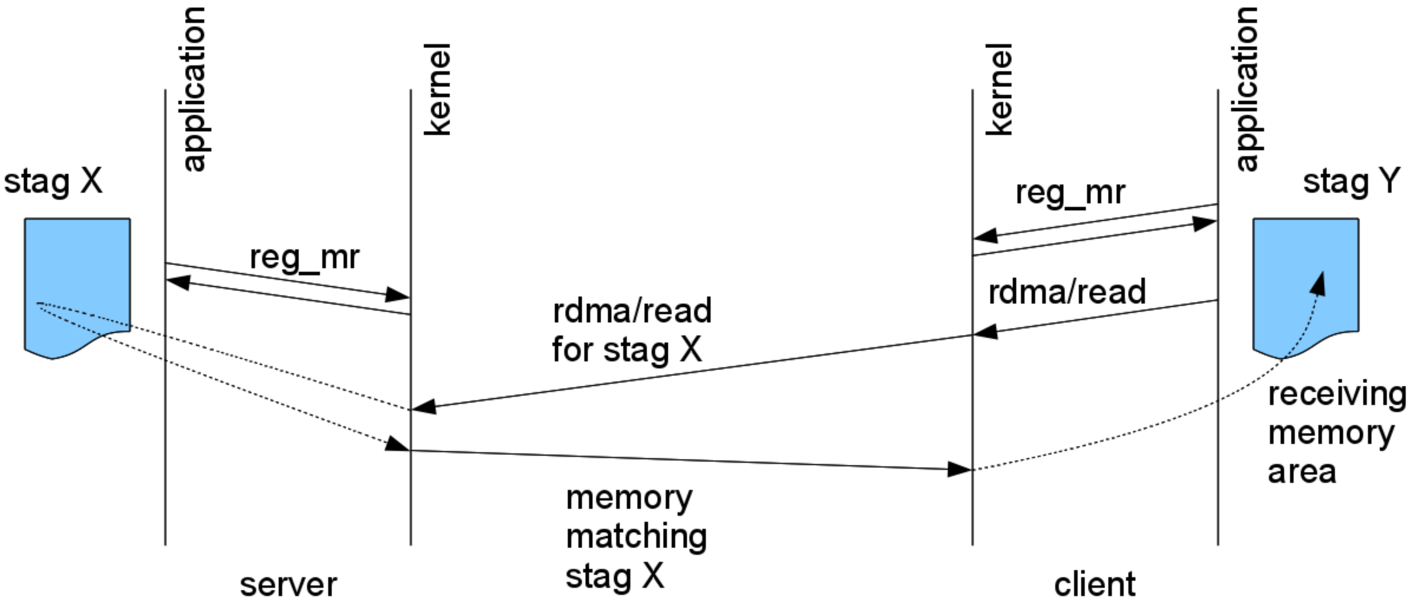
Using RDMA

- RDMA: Remote Direct Memory Access
  - Efficient remote memory access
  - Zero-copy (inside the end hosts), low latency, low CPU usage
  - Great! 😁
- Who has RDMA capable NICs deployed?
  - HPC: Yes
  - Commodity data centers? 😟



RDMA in Software

- No hardware acceleration, runs on Ethernet
- But still RDMA semantics
- Example: One-sided RDMA read in SoftiWARP
  - Zero copy, no context switching, low CPU footprint, etc.



Memcached/RDMA

Modified Memcached Architecture:

1) Memory management: register server-side memory chunks with RDMA

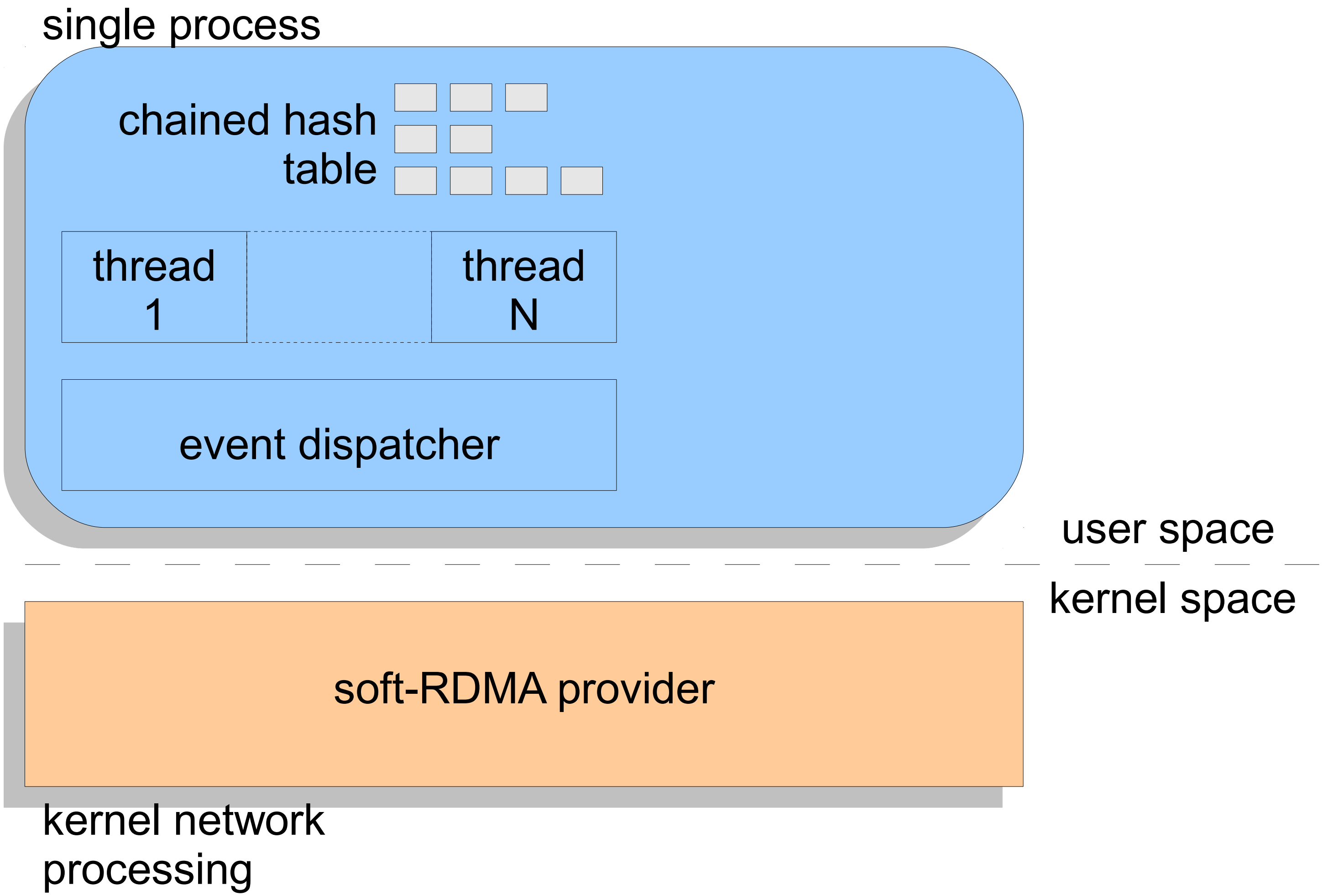
2) SET operation:

   - New value: get new chunk, store key/value pair, return stag
   - Update value: get new chunk, store store key/value pair, swap stags with old chunk

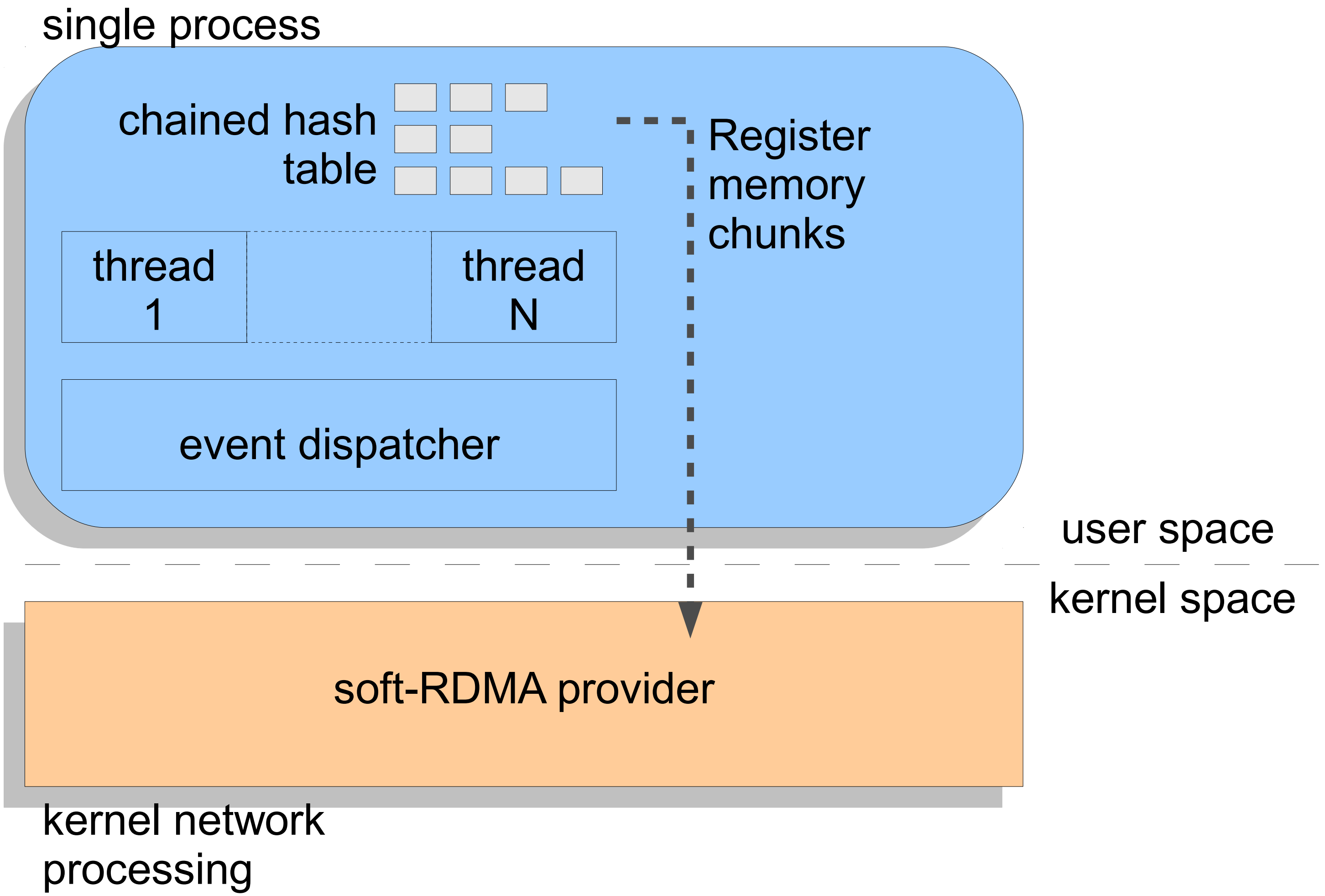
3) GET operation: client uses one-sided RDMA read to retrieve entire chunk

   - Zero copy
   - No context switch
   - Move parts of server processing to the client (e.g., request parsing)

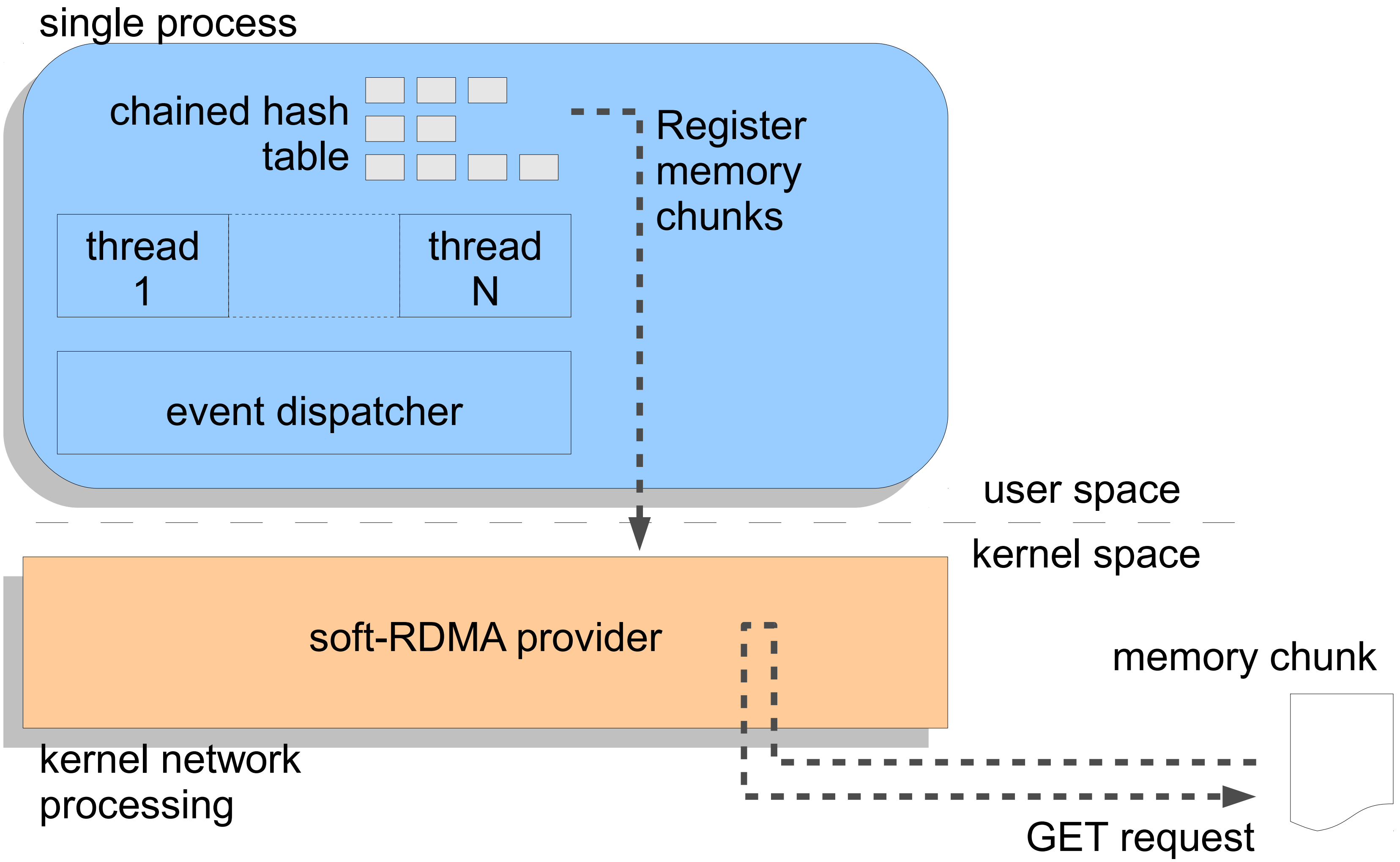
Memcached/RDMA (2)



Memcached/RDMA (2)



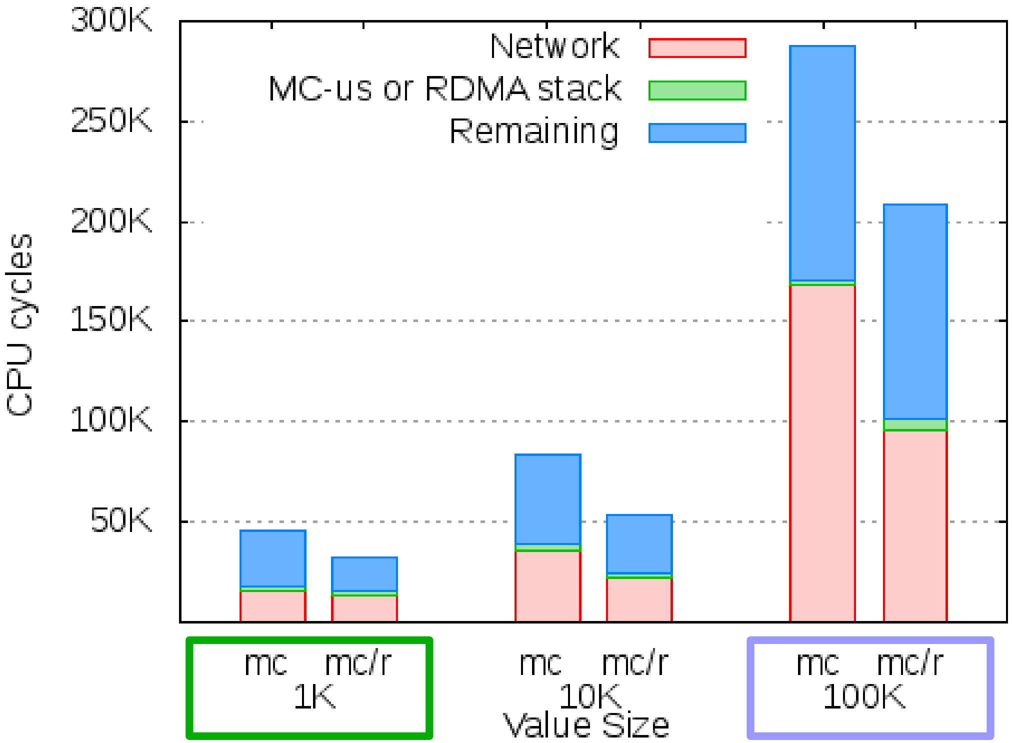
Memcached/RDMA (2)



Implementation & Benchmarks

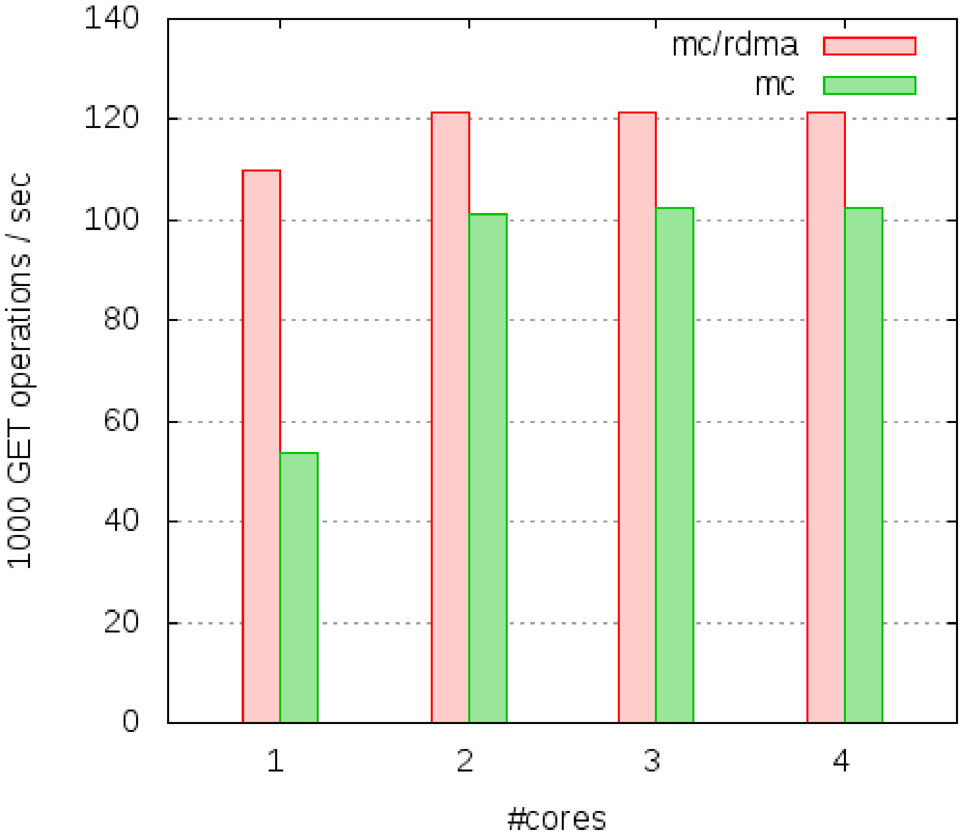
- Implementation Memcached/RDMA
  - Standalone prototype: server/client
  - Uses Memcached data types (e.g., item for storing key/value pairs)
- Benchmark Setup
  - 1 Server, 6 Clients
  - 4 core Intel Xeon E5345, 10GbE
  - Server CPU clock frequency: 1.1 Ghz
  - 1000 pre-insterted key/value pairs
  - OProfile to measure CPU load

CPU Efficiency



- Memcached/RDMA consumes less CPU
  - For **small packets:** less OS overhead (excluding network stack)
  - For **large packets**: less network stack overhead

Multicore Performance



- Memcached/RDMA with one core performs like Memcached with 4 cores

Conclusion

- Memcached/RDMA is a more CPU efficient Memcached based on Software RDMA
  - Zero copy, zero context switching for re-occurring GET requests, memory chunk parsing moved to client side
  - No special hardware required
- Architecture also suitable for SSD based key/value stores
  - Any combination of high bandwidth storage and fast network will put pressure on the CPU
- Outlook: multicore, latency, etc.

Thanks! Questions?

http://gitorious.org/softiwarp