NetBricks: Taking the V out of NFV

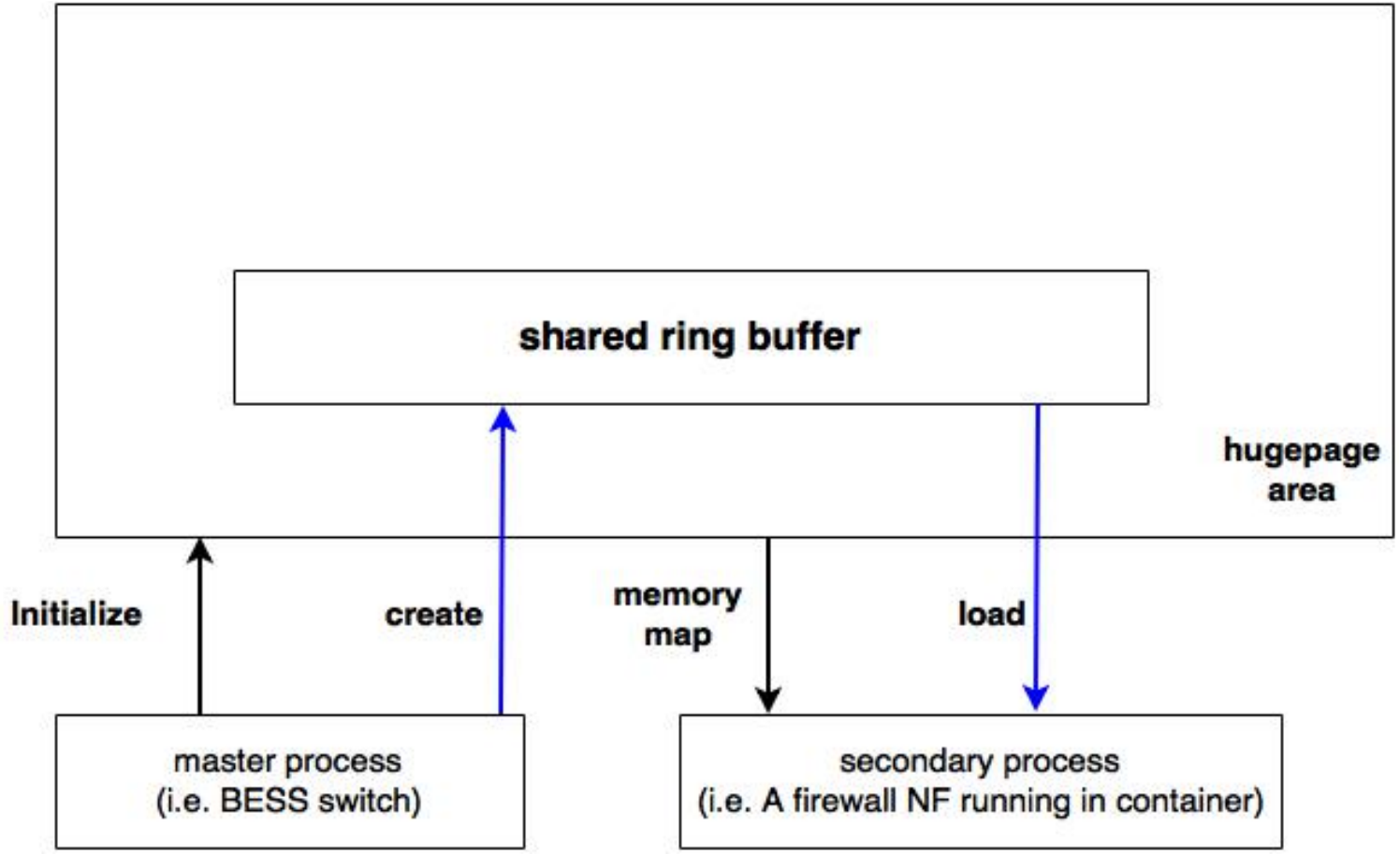
Aurojit Panda, Sangjin Han, Keon Jang, Melvin Walls, Sylvia Ratnasamy,
Scott Shenker UC Berkeley, Google, ICSI

Overview of NetBricks

- Providing isolation (Primary contribution)
  - Memory isolation
  - Packet isolation
  - Performance isolation

- An abstraction for buidling NF (Secondary contribution)
  - Packet processing
  - Control flow
  - Byte stream
  - Shared state management

DPDK Hugepage Memory Management



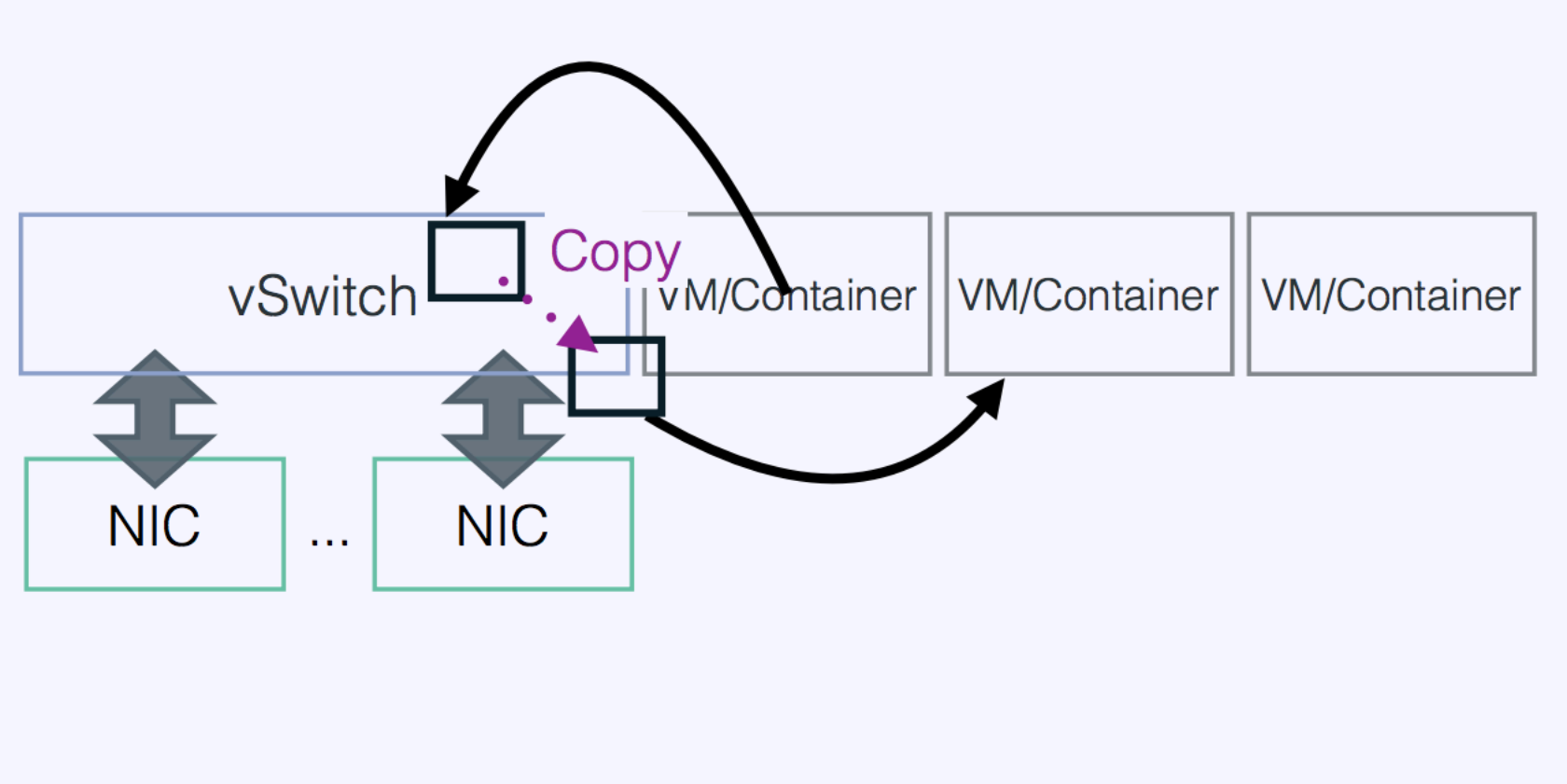
Performance is good

- Packets are passed around as pointers.
- Can achieve very high throughput

Isolation is bad

- Packets are allocated on shared memory area.
- Pointer to a packet is still valid after the packet is sent out.
- Bad intentioned NF may use this to disturb the NFV system.

How to ensure isolation?

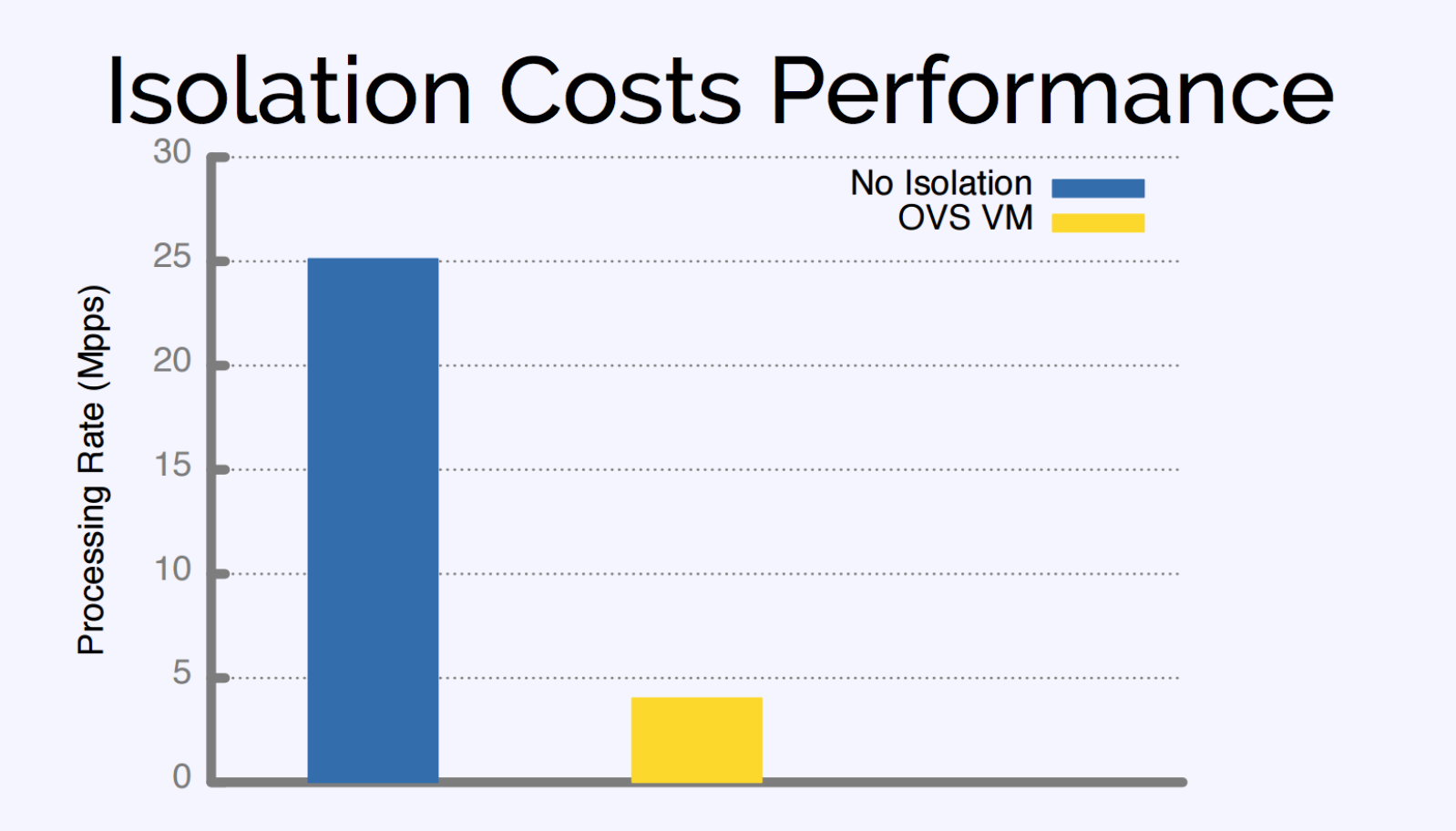


Isolation through packet copy

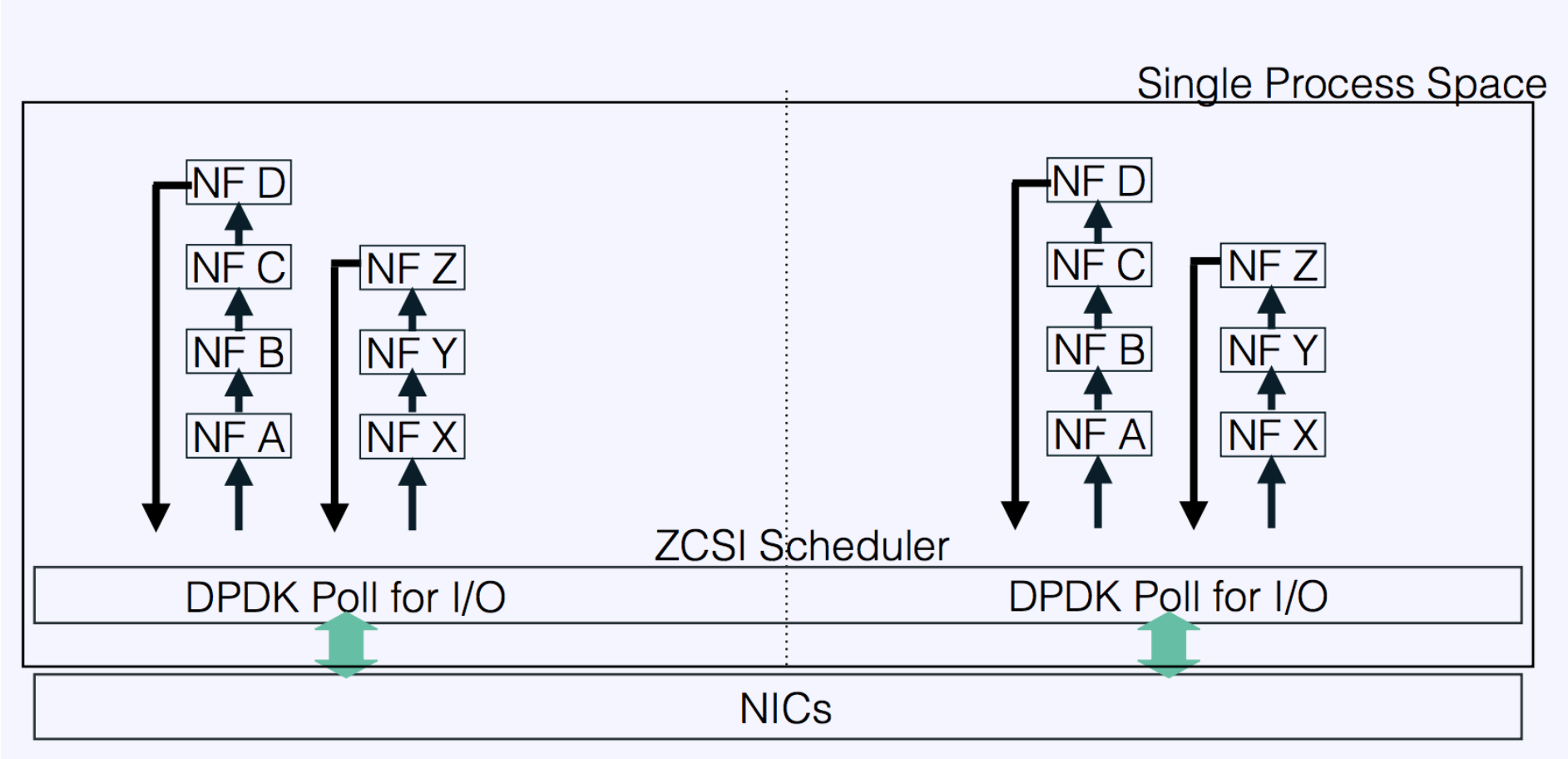
- OpenVSwitch copies the packet and send the pointer to the copied packet to the next NF

- However, performance is bad

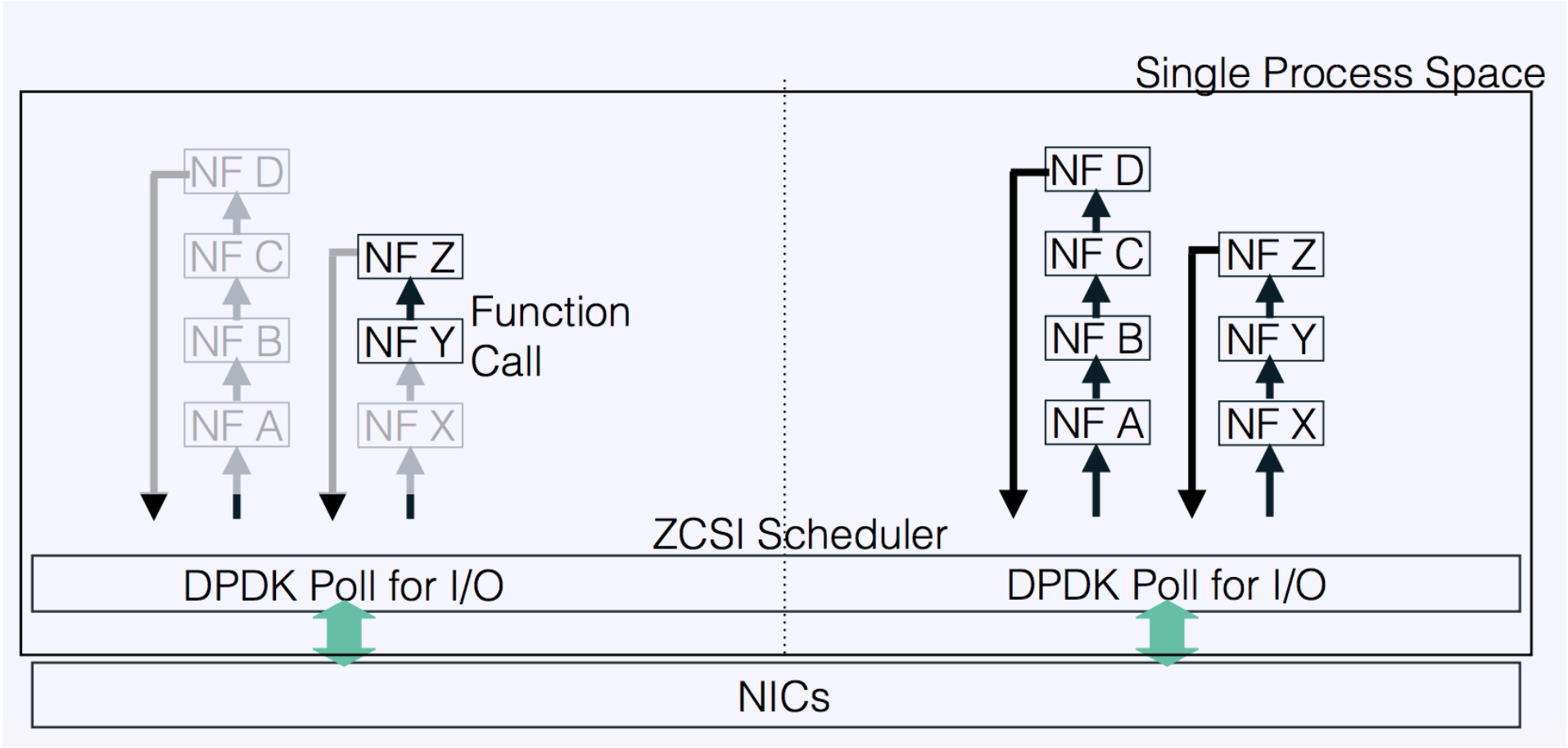
Isolation through packet copy



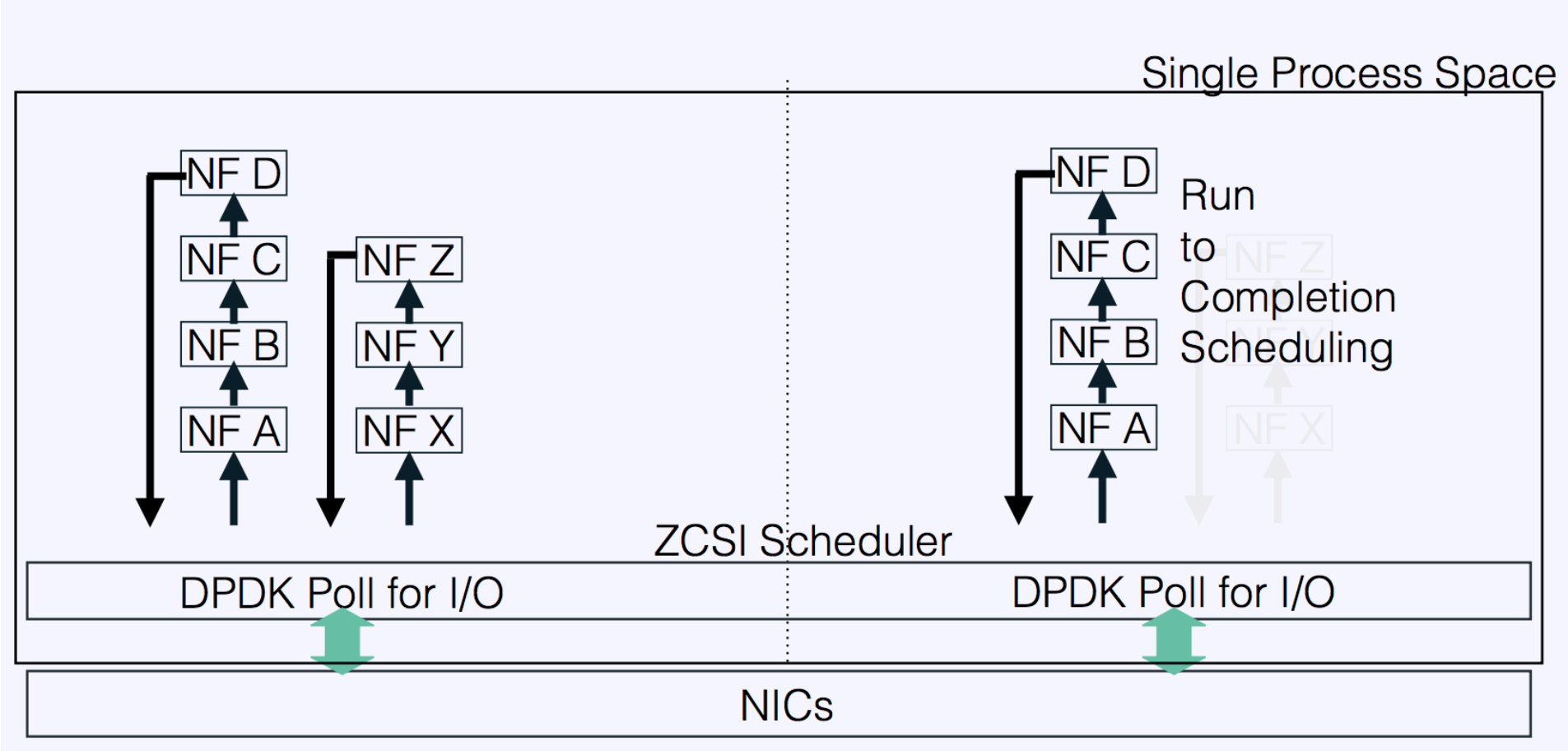
NetBricks: Zero Copy Soft Isolation



NetBricks: Zero Copy Soft Isolation



NetBricks: Zero Copy Soft Isolation



NetBricks: Zero Copy Soft Isolation

- Netbricks build on Rust, a “high-level ”system programming language that intends to replace C/C++/
- No pointer algorithmic in Netbricks. (Can’t manipulate pointers to reference arbitrary memory localtion.)
- Compile time type checking (Rust has a more powerful type system than C++) and runtime array bound checking

NetBricks: Zero Copy Soft Isolation

- Reference to packet is automatically destroyed after it is passed to another NF

- Only a single NF has the reference to the packet at any given time.

Overhead of runtime array bound checks

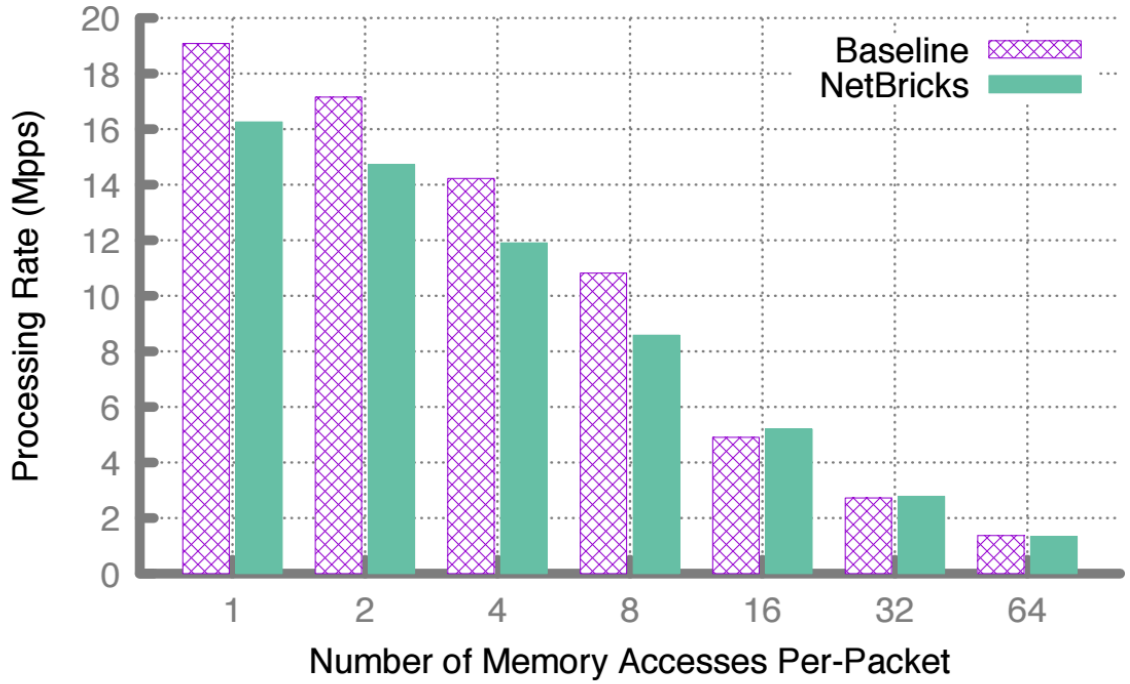


Figure 1: Throughput achieved by a NetBricks NF and an NF written in C using DPDK as the number of memory accesses in a large array grows.

Throughput when providing full packet isolation

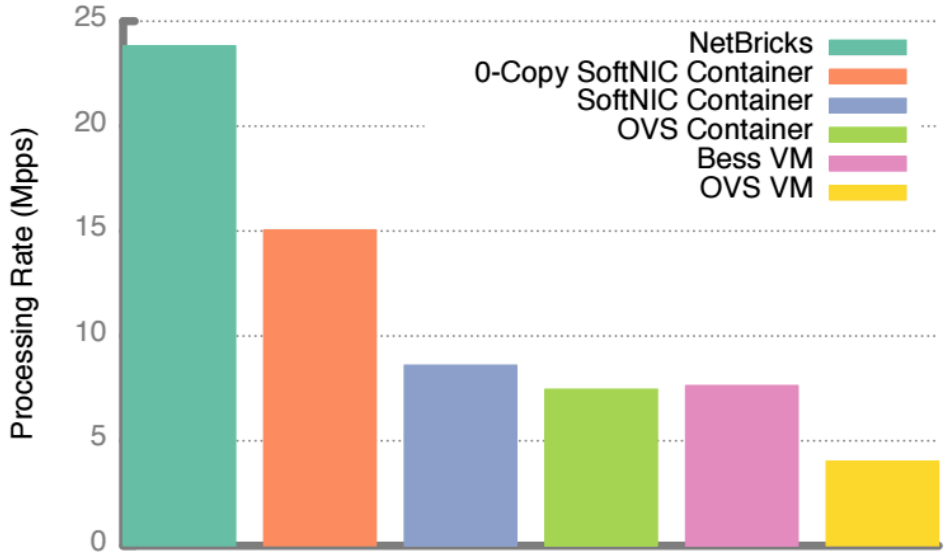
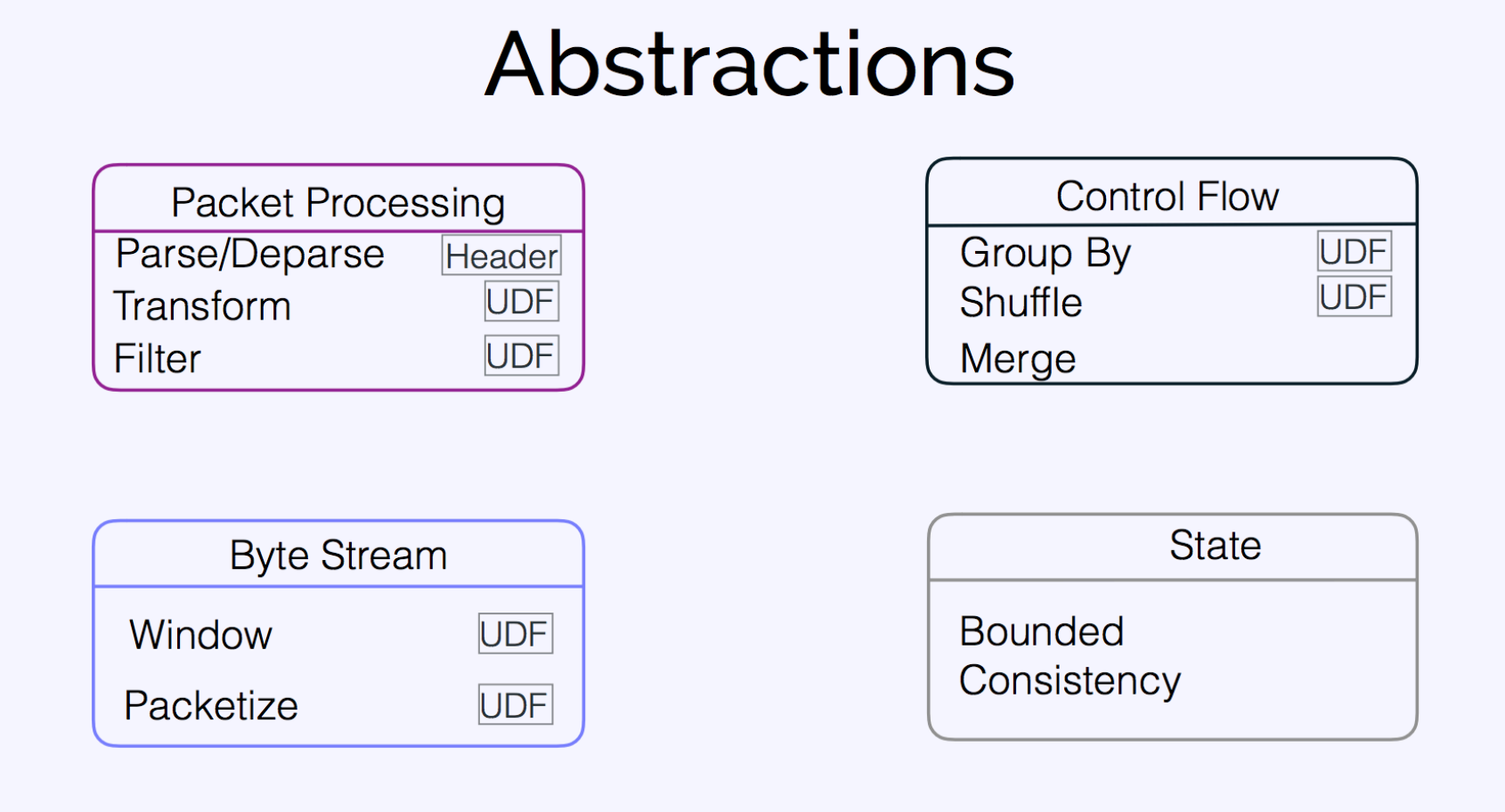


Figure 4: Throughput achieved using a single NF running under different isolation environments.

Summary of Zero Copy Soft Isolation

- Small runtime overhead compared to bare performance
- Excellent throughput compared with existing approaches for providing full packet isolation

New abstractions for building NFs.



Thoughts on NF Building

- Trend:
  - Improve performance and simplify management:
    - Get rid of switching. Use function call to pass packets
    - Get rid of chaining through container/VM. Chaining through NF software modules.

  - Use high-level programming language:
    - C/C++ is fast, but not secure and not expressive
    - Rust seems to be a good substitute.

    - How about PL with garbage-collection
      - Paper claims that garbage collection incurs unpredicable latency spikes.
      - May not be true with PL of ML family.