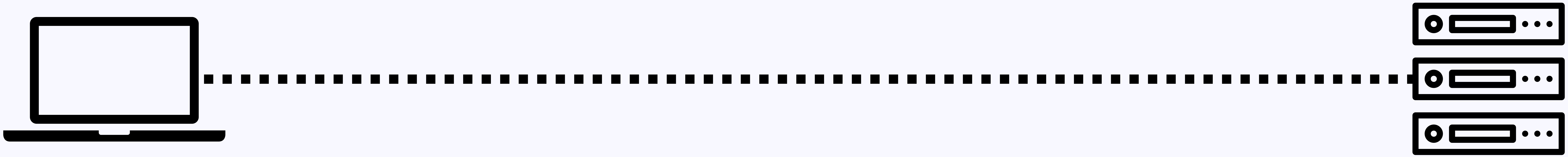
# NetBricks: Taking the V out of NFV

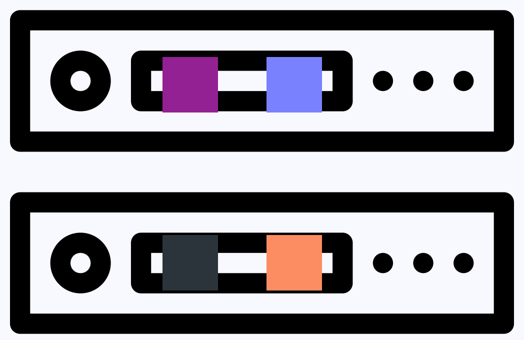
Aurojit Panda, Sangjin Han, Keon Jang, Melvin Walls, Sylvia Ratnasamy, Scott Shenker
UC Berkeley, Google, ICSI

NETSYS

# What the heck is NFV?

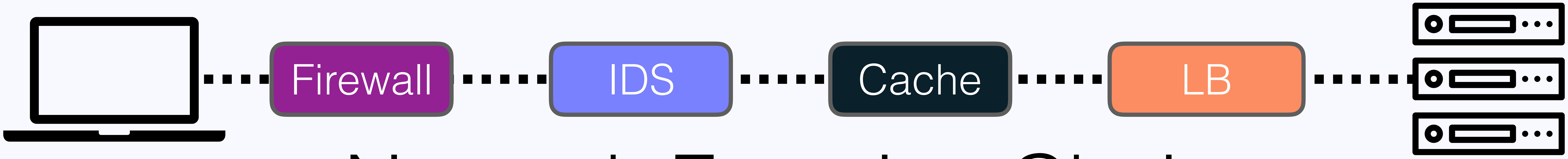
# A Short Introduction to NFV





# A Short Introduction to NFV

# A Short Introduction to NFV




Network Function Chain

# Why NFV?

- Simplifies **adding new functionality**: Deploy new software.

# Why NFV?

- Simplifies **adding new functionality**: Deploy new software.
- Simplifies **developing new functionality**: Write software vs design hardware

# Why NFV?

- Simplifies **adding new functionality**: Deploy new software.
- Simplifies **developing new functionality**: Write software vs design hardware
- Reuse **management tools** from other domains.

# Why NFV?

- Simplifies **adding new functionality**: Deploy new software.
- Simplifies **developing new functionality**: Write software vs design hardware
- Reuse **management tools** from other domains.
- **Consolidation**: Reduce number of hardware boxes in the network.

# Challenges for NFV

# Challenges for NFV

- **Running NFs**
  - **Isolation** and **Performance**

# Challenges for NFV

- **Running NFs**
  - **Isolation** and **Performance**

- **Building NFs**
  - **High-Level Programming** and **Performance**

# Running NFs

# Isolation

- **Memory Isolation**: Each NF’s memory cannot be accessed by other NFs.

# Isolation

- **Memory Isolation**: Each NF's memory cannot be accessed by other NFs.

- **Packet Isolation**: When chained, each NF processes packets in isolation.

# Isolation

- **Memory Isolation**: Each NF’s memory cannot be accessed by other NFs.

- **Packet Isolation**: When chained, each NF processes packets in isolation.

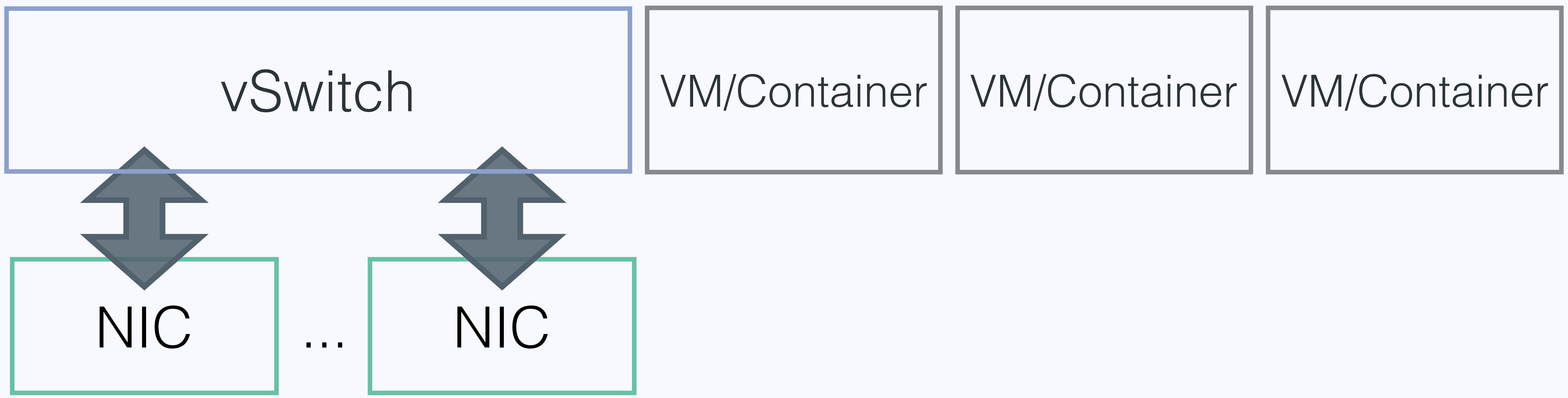
# Isolation

- **Memory Isolation**: Each NF’s memory cannot be accessed by other NFs.
- **Packet Isolation**: When chained, each NF processes packets in isolation.
- **Performance Isolation**: One NF does not affect another’s performance.

# Isolation

- **Memory Isolation**: Each NF's memory cannot be accessed by other NFs.
- **Packet Isolation**: When chained, each NF processes packets in isolation.
- ~~**Performance Isolation**: One NF does not affect another's performance.~~

# Current Solution

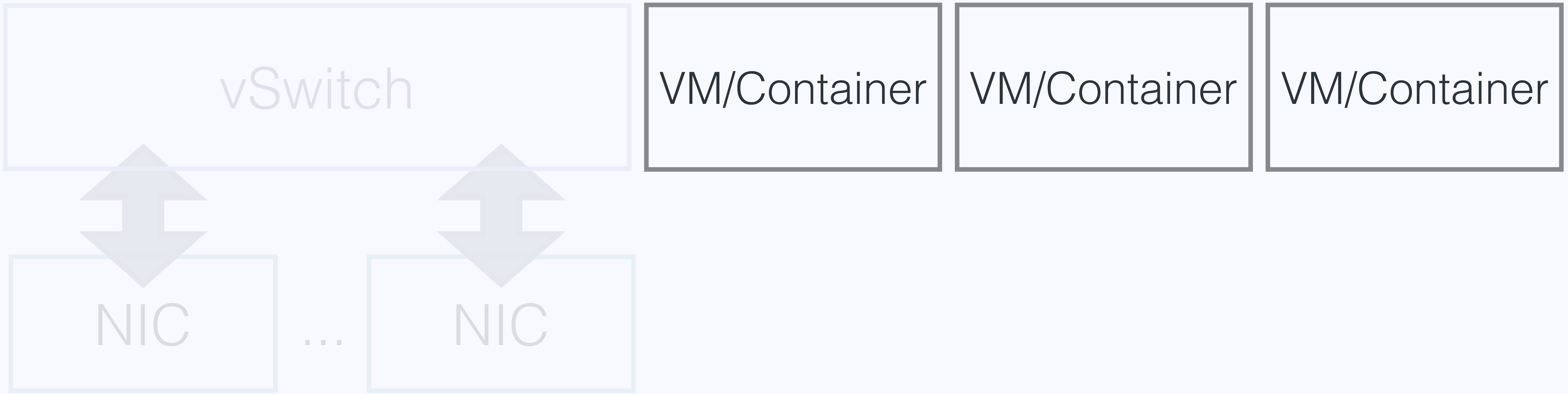



Memory Isolation

Packet Isolation

Performance

# Current Solution

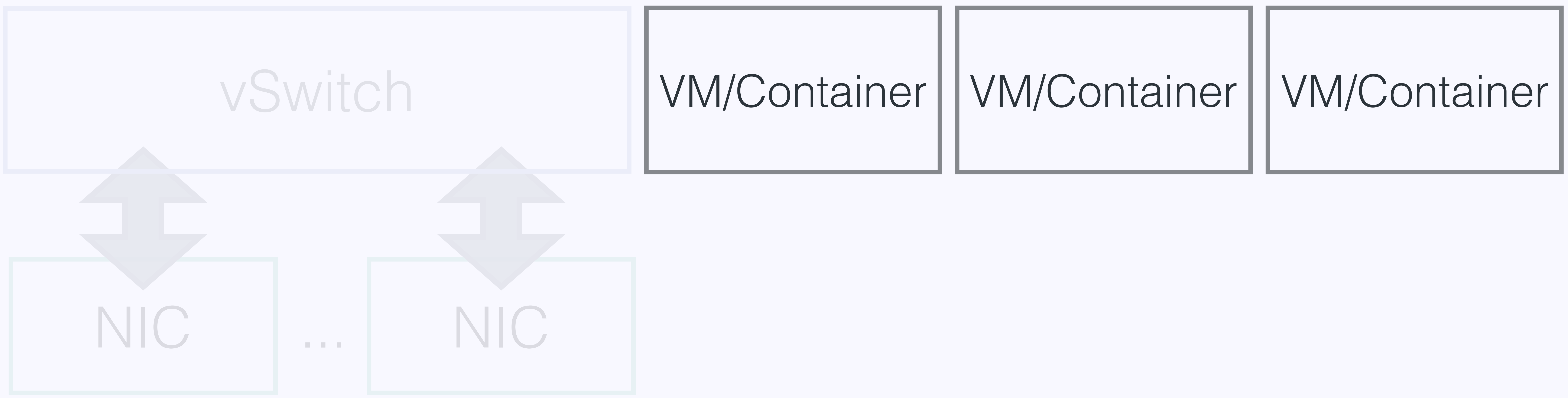



Memory Isolation

Packet Isolation

Performance

# Current Solution

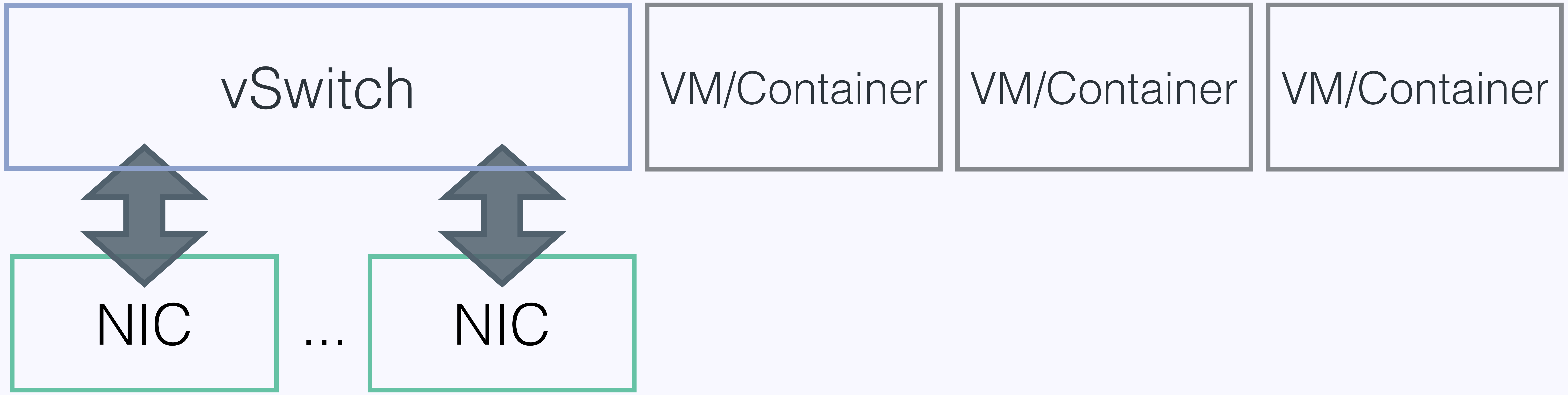



✔Memory Isolation

Packet Isolation

Performance

# Current Solution

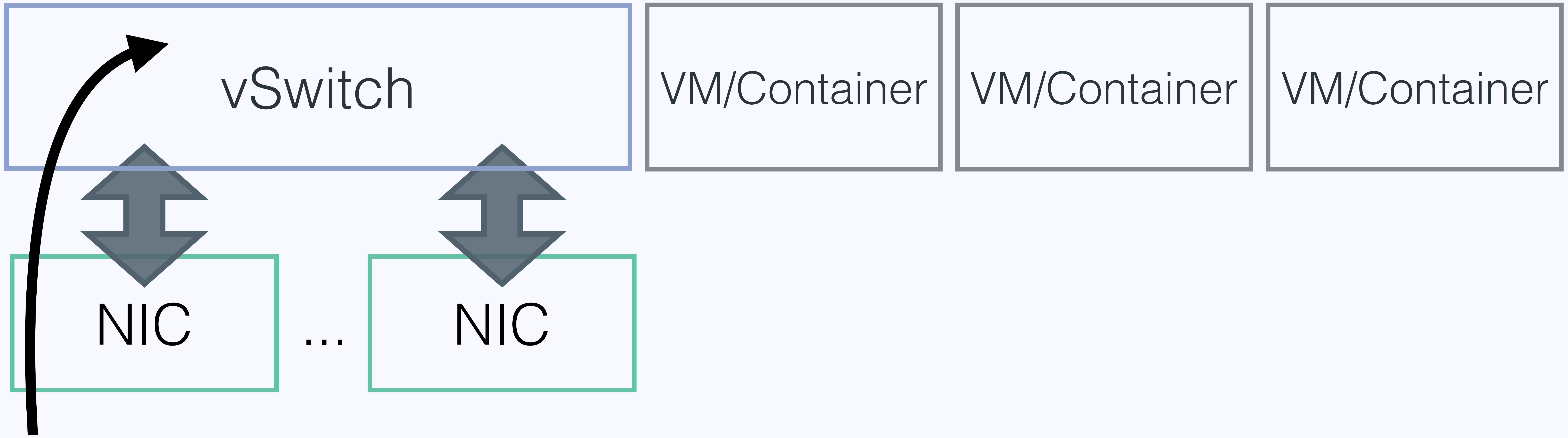



✔Memory Isolation

Packet Isolation

Performance

# Current Solution

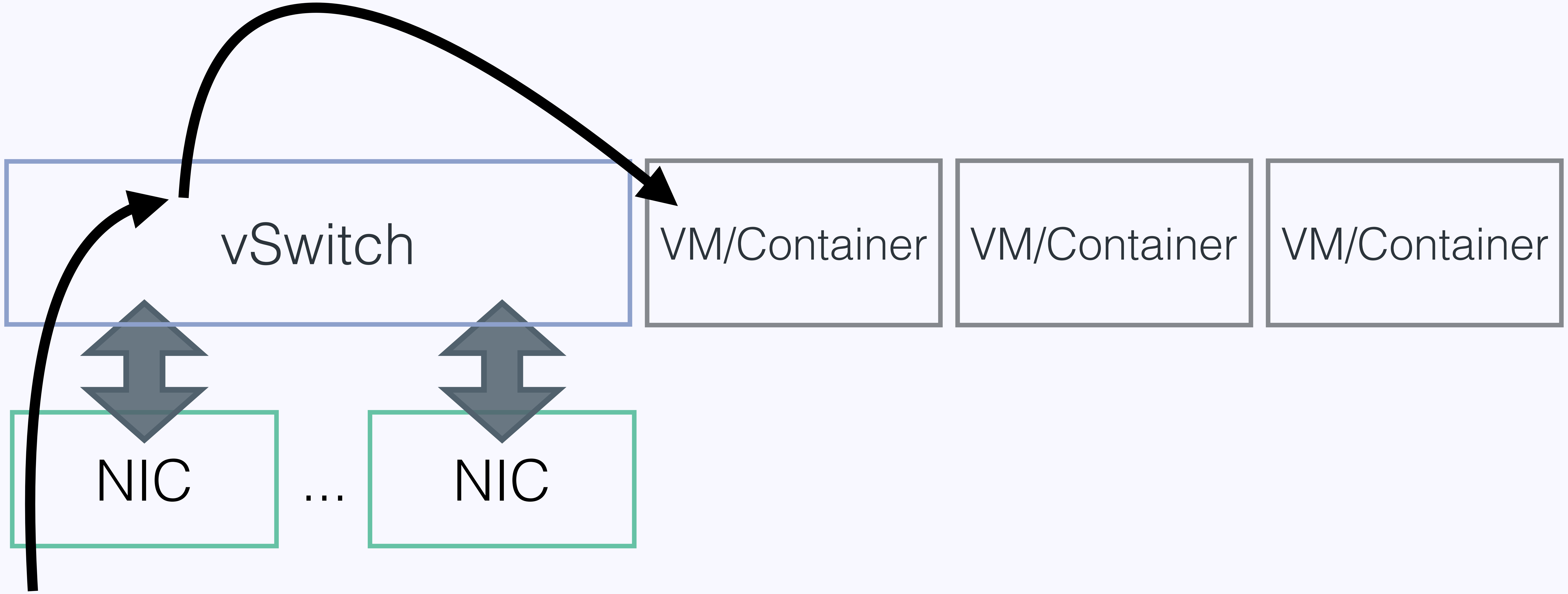



✔Memory Isolation

Packet Isolation

Performance

# Current Solution

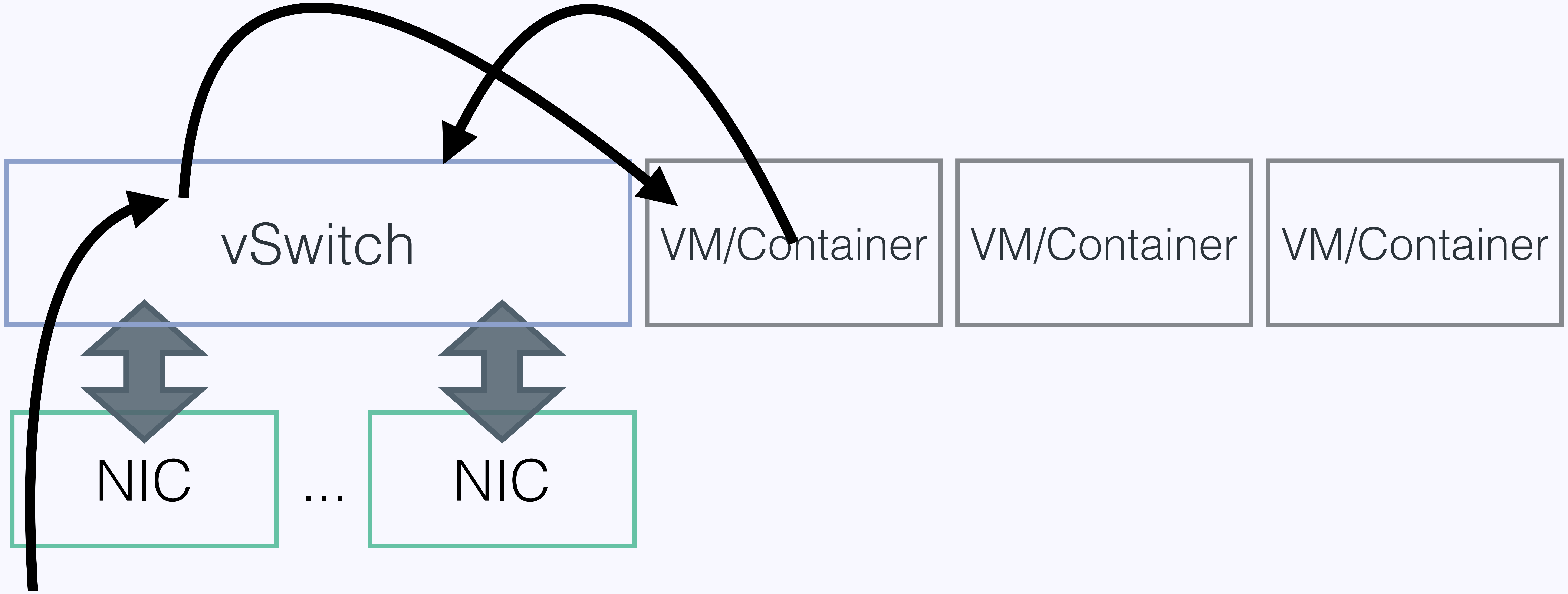



✔Memory Isolation

Packet Isolation

Performance

# Current Solution

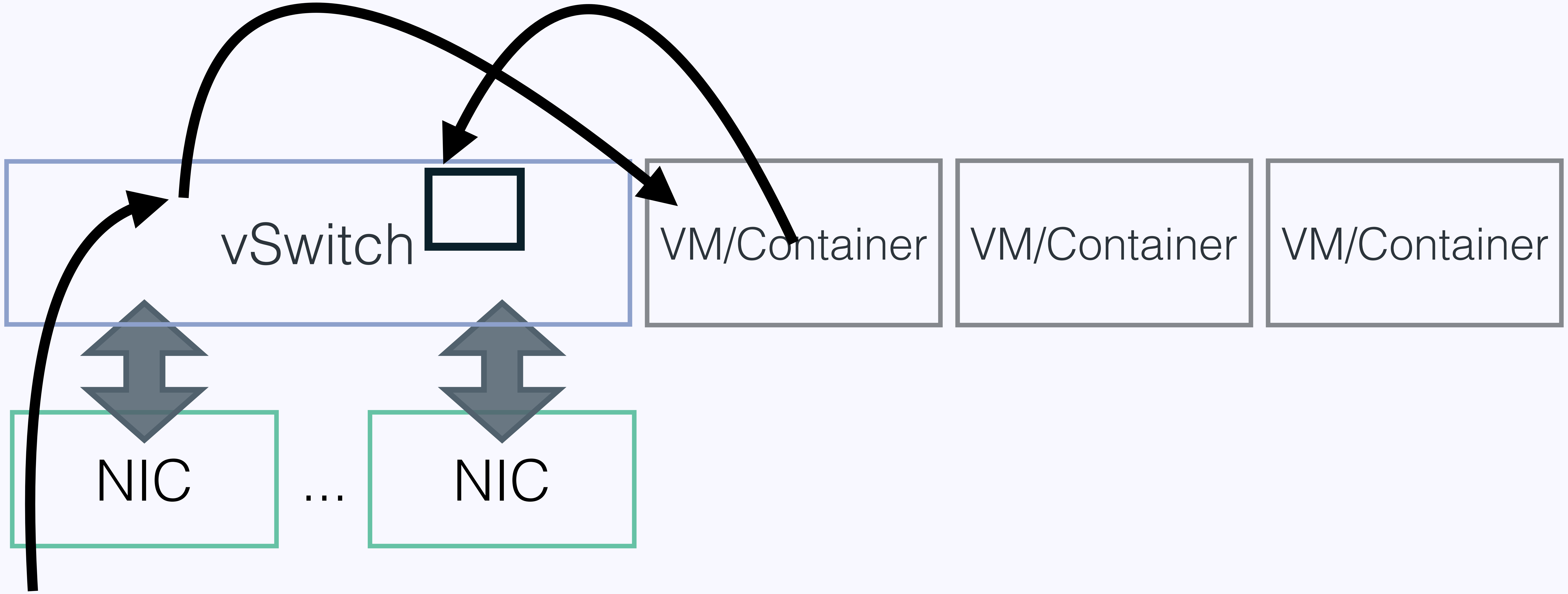



✔ Memory Isolation

Packet Isolation

Performance

# Current Solution

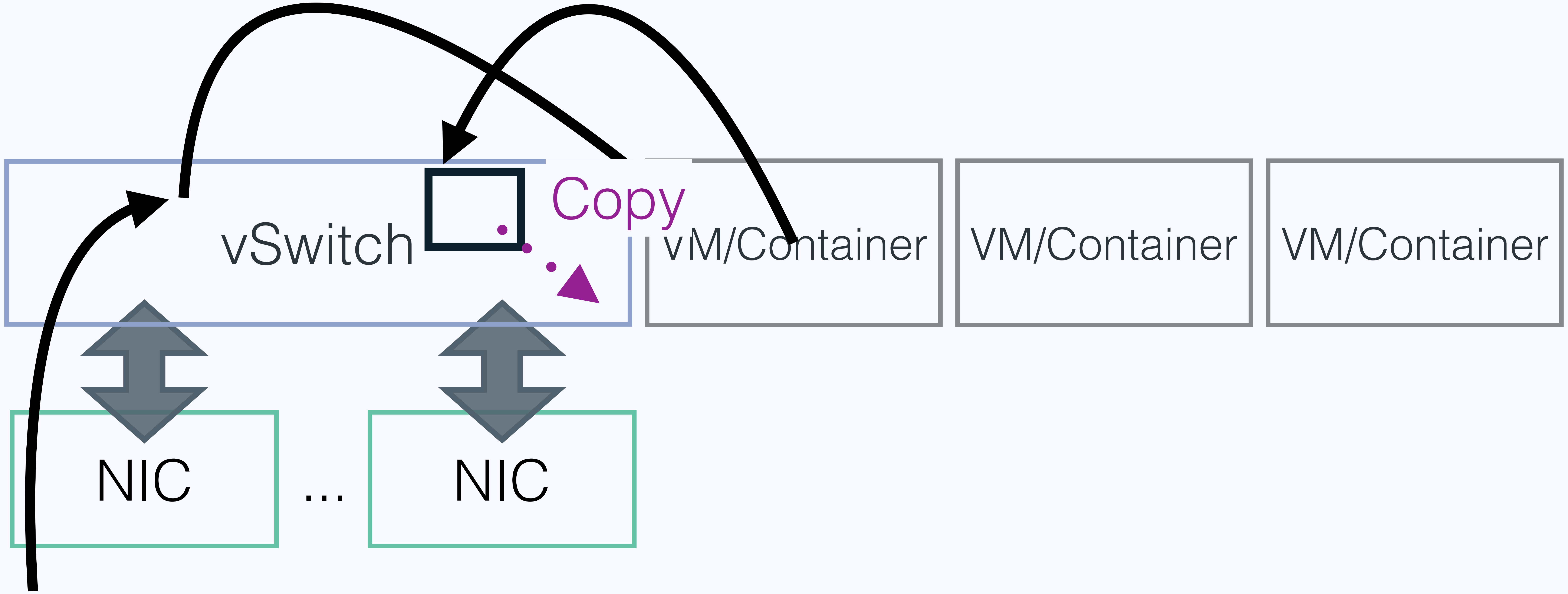



✔Memory Isolation

Packet Isolation

Performance

# Current Solution

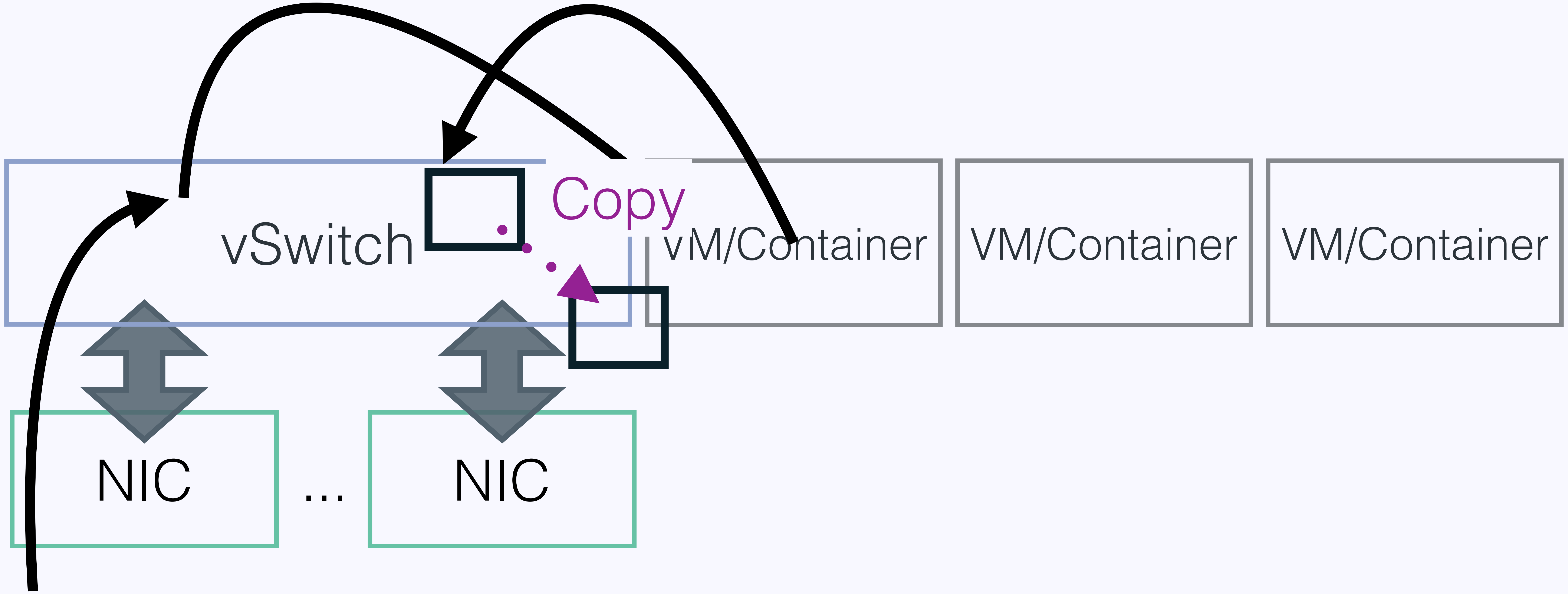



✔Memory Isolation

Packet Isolation

Performance

# Current Solution

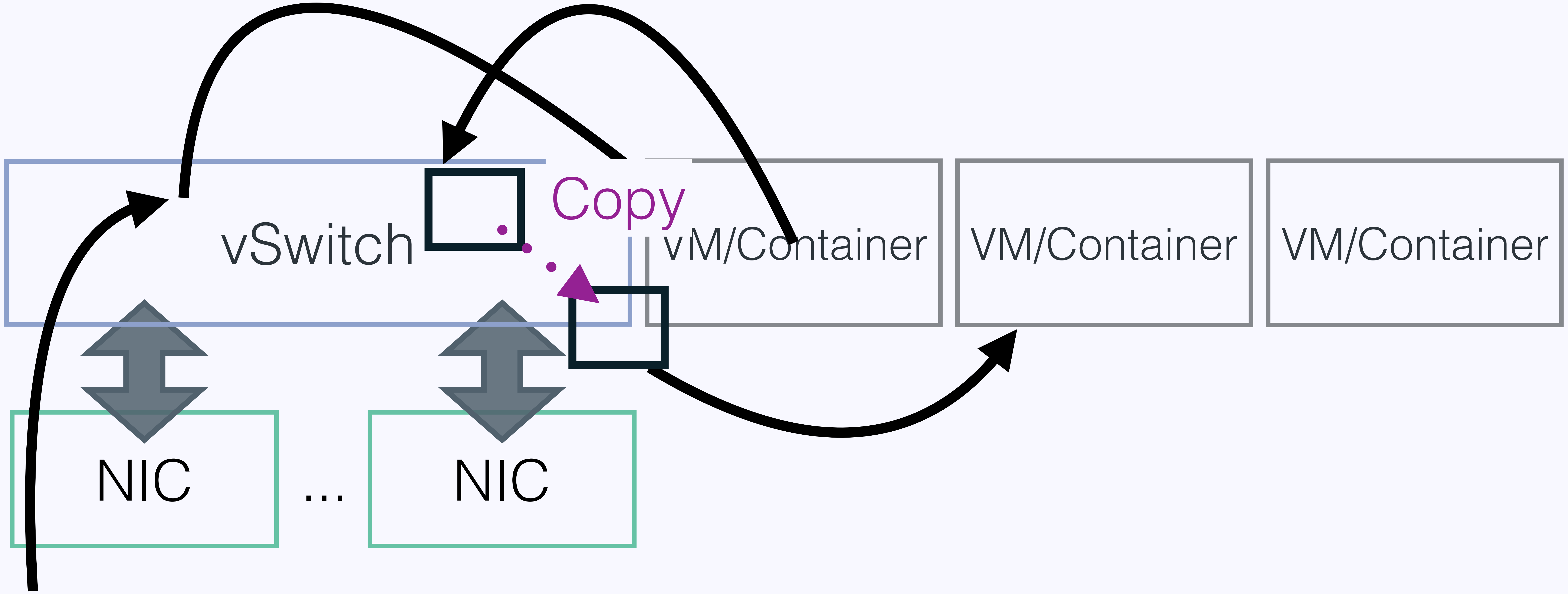



✔Memory Isolation

Packet Isolation

Performance

# Current Solution

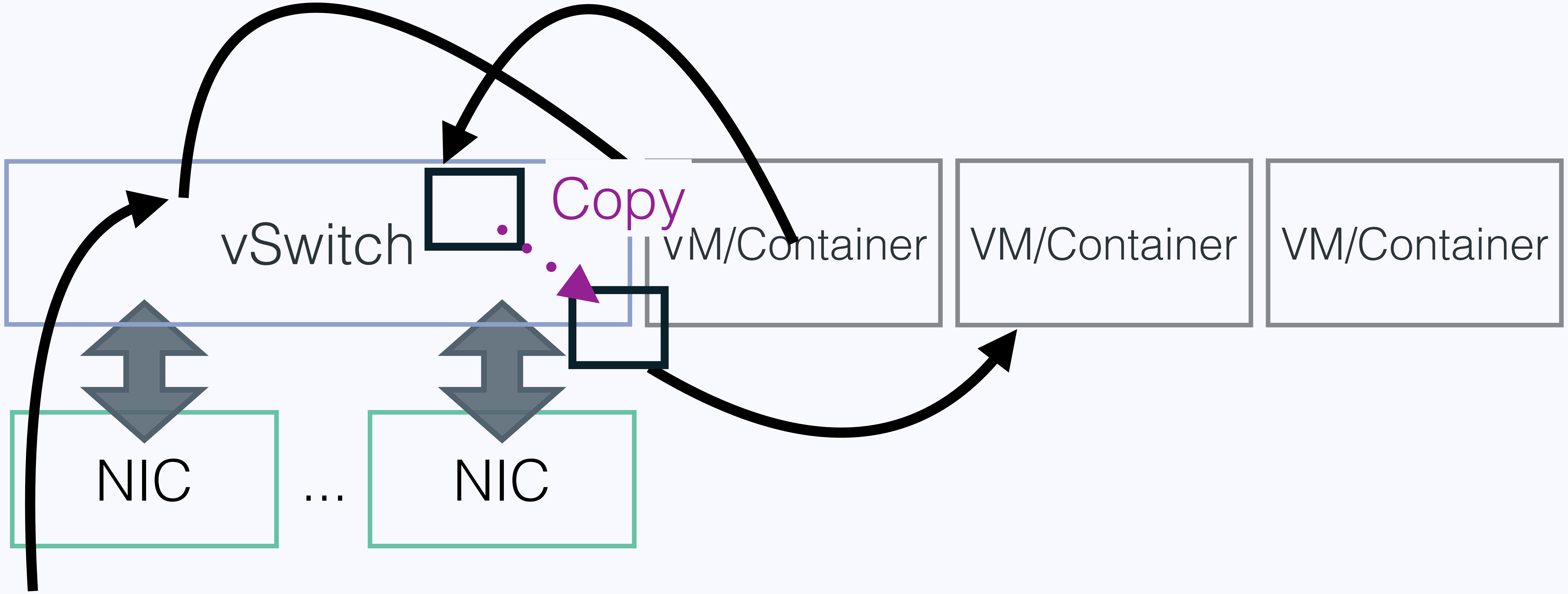



✔Memory Isolation

Packet Isolation

Performance

# Current Solution

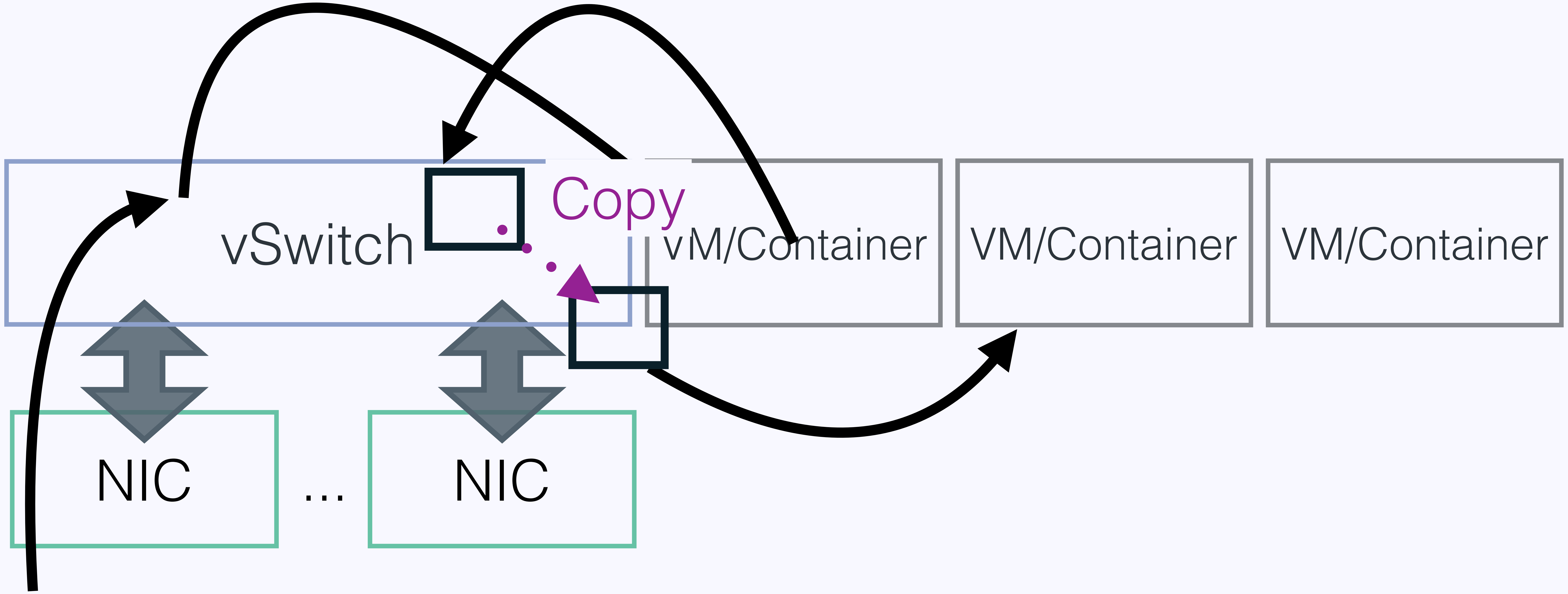



✔Memory Isolation

✔ Packet Isolation

Performance

# Current Solution

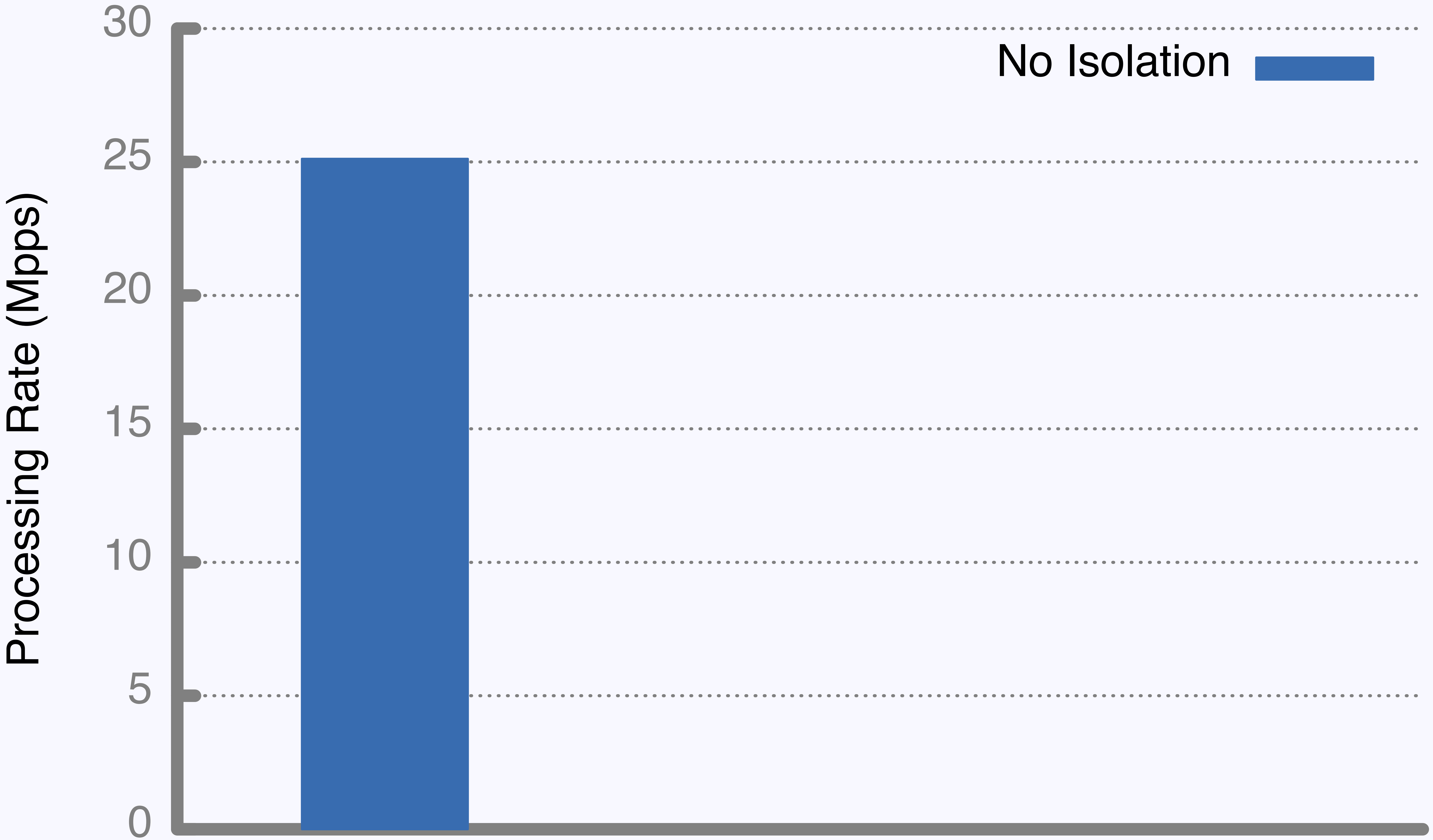



✔ Memory Isolation

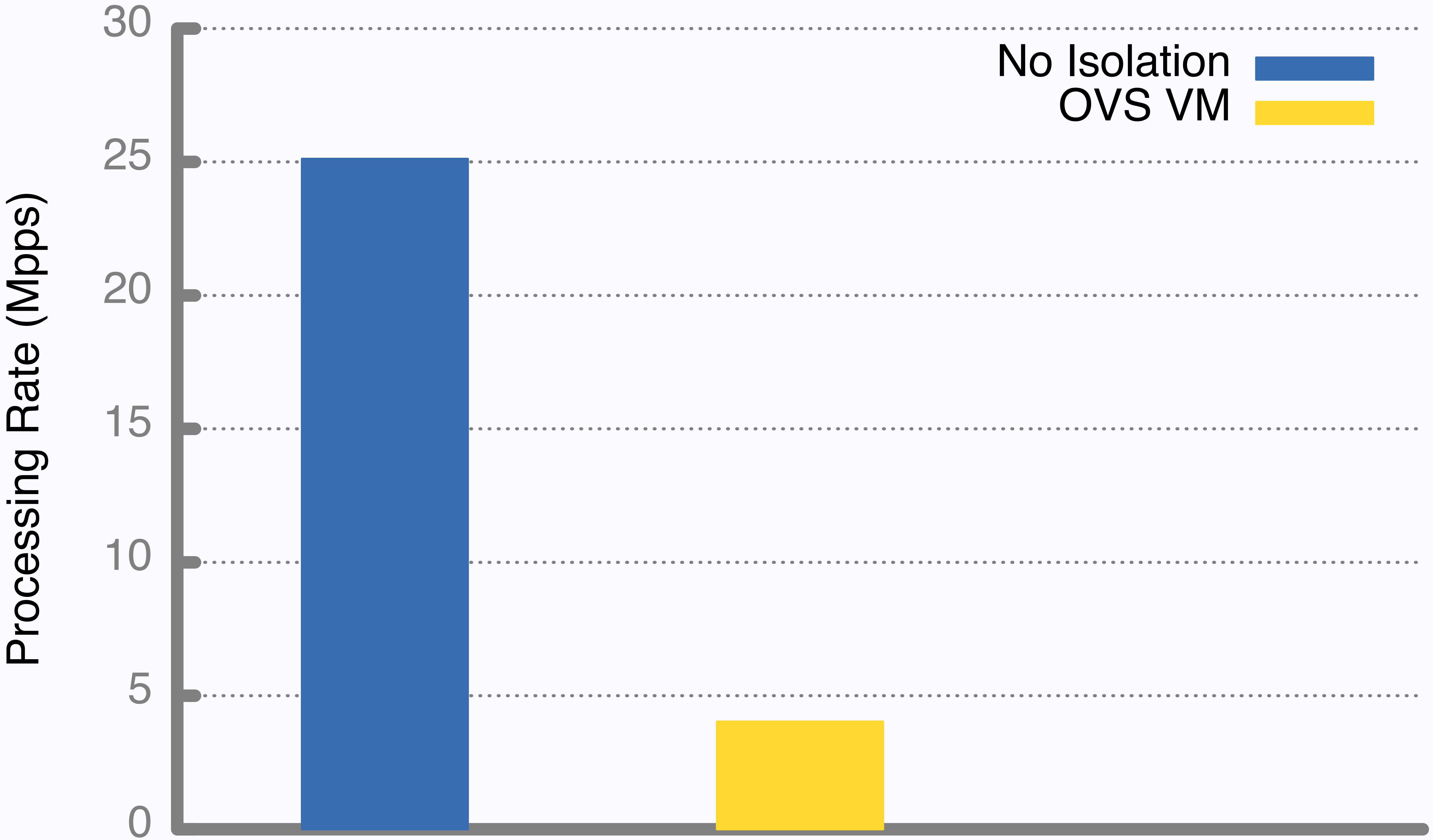
✔ Packet Isolation

✗ Performance

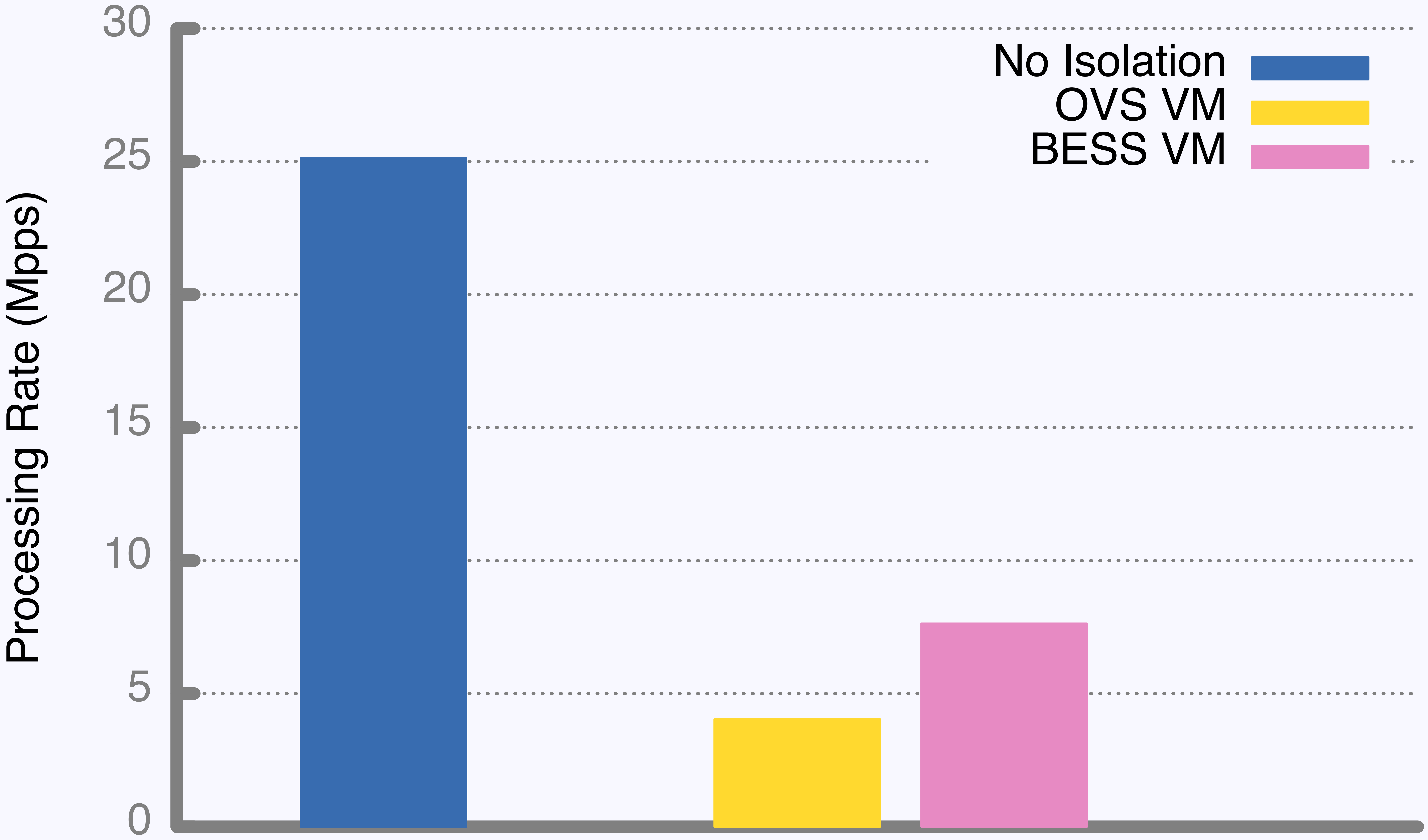
# Isolation Costs Performance



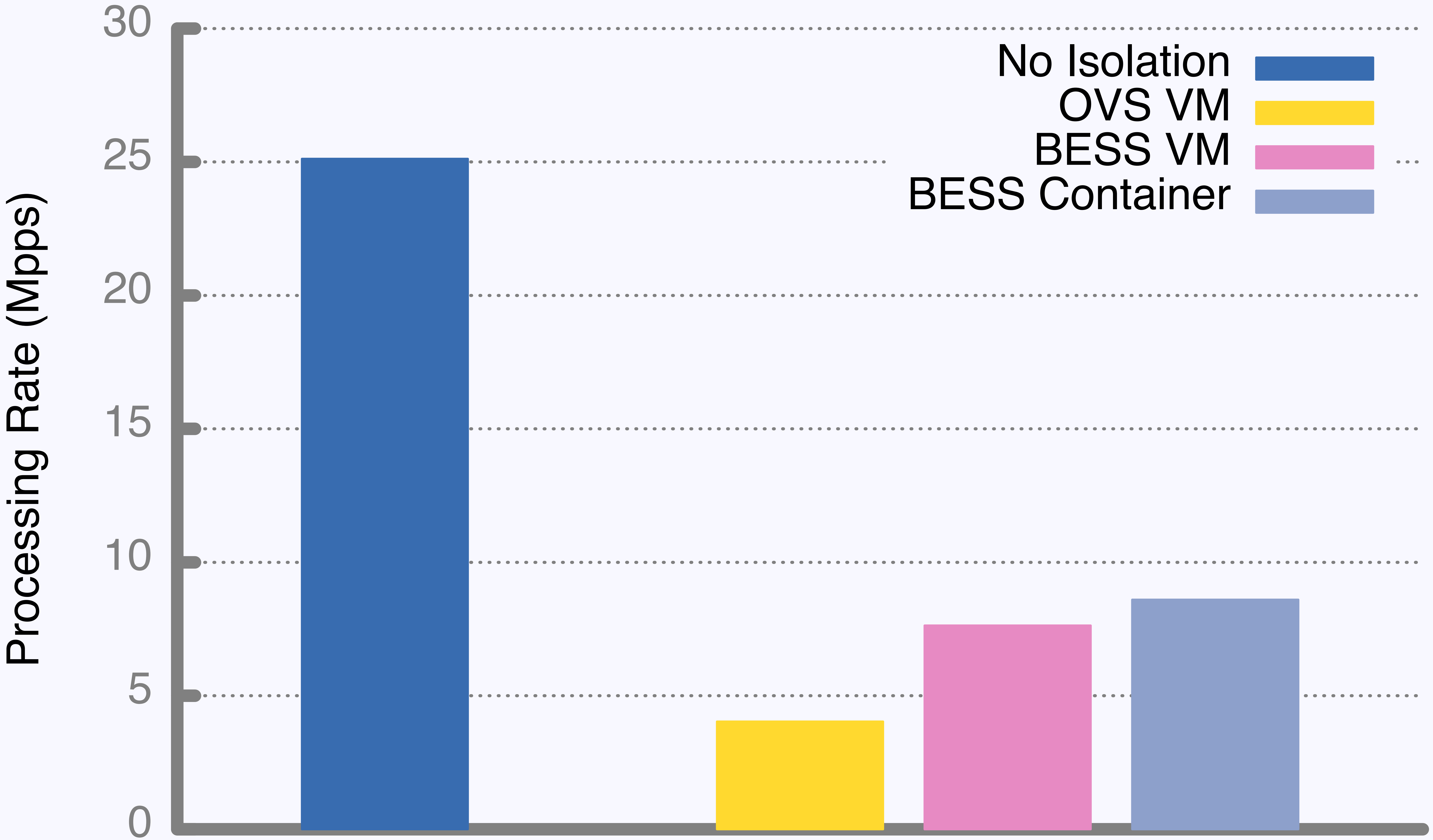
# Isolation Costs Performance



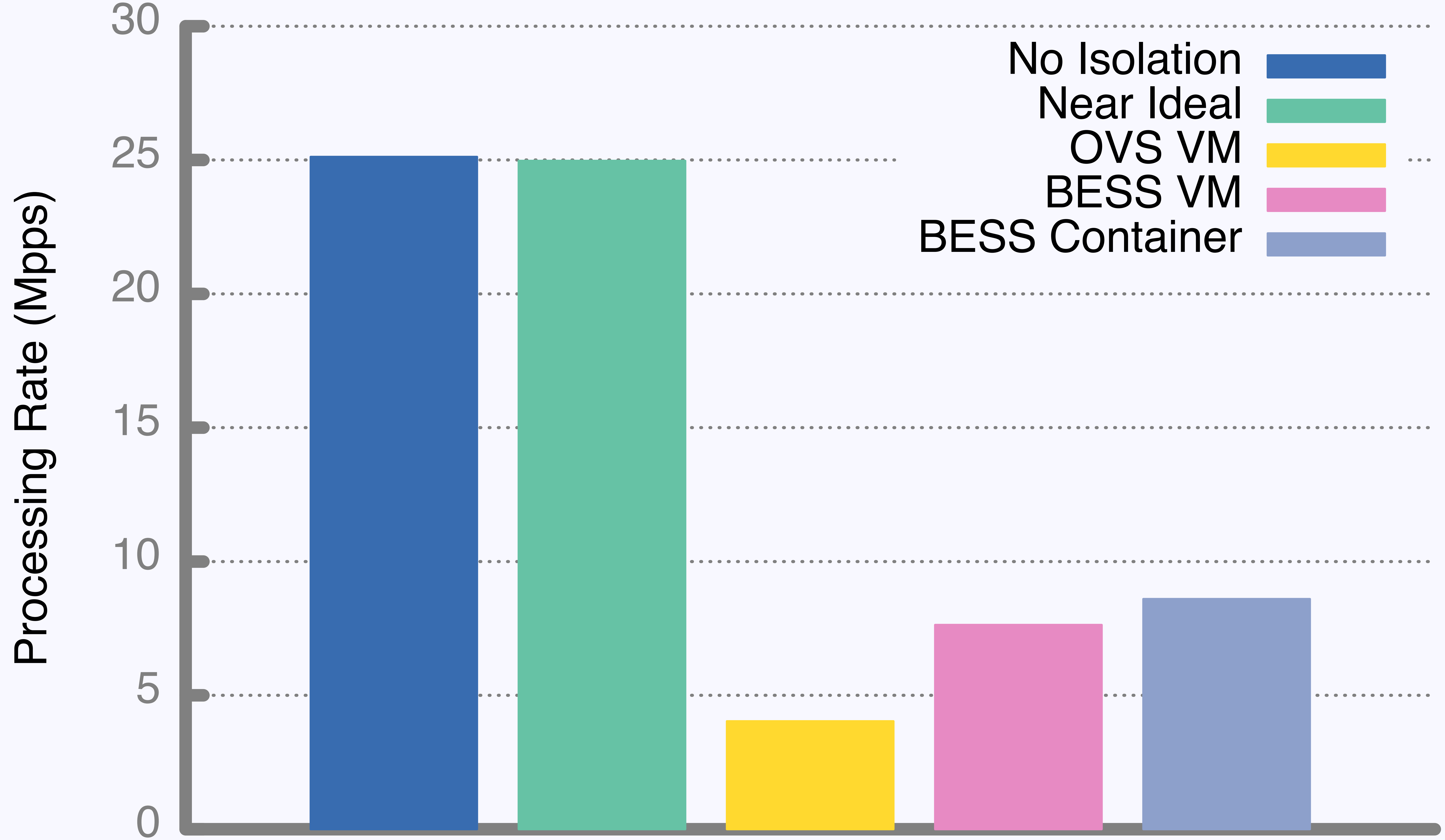
# Isolation Costs Performance



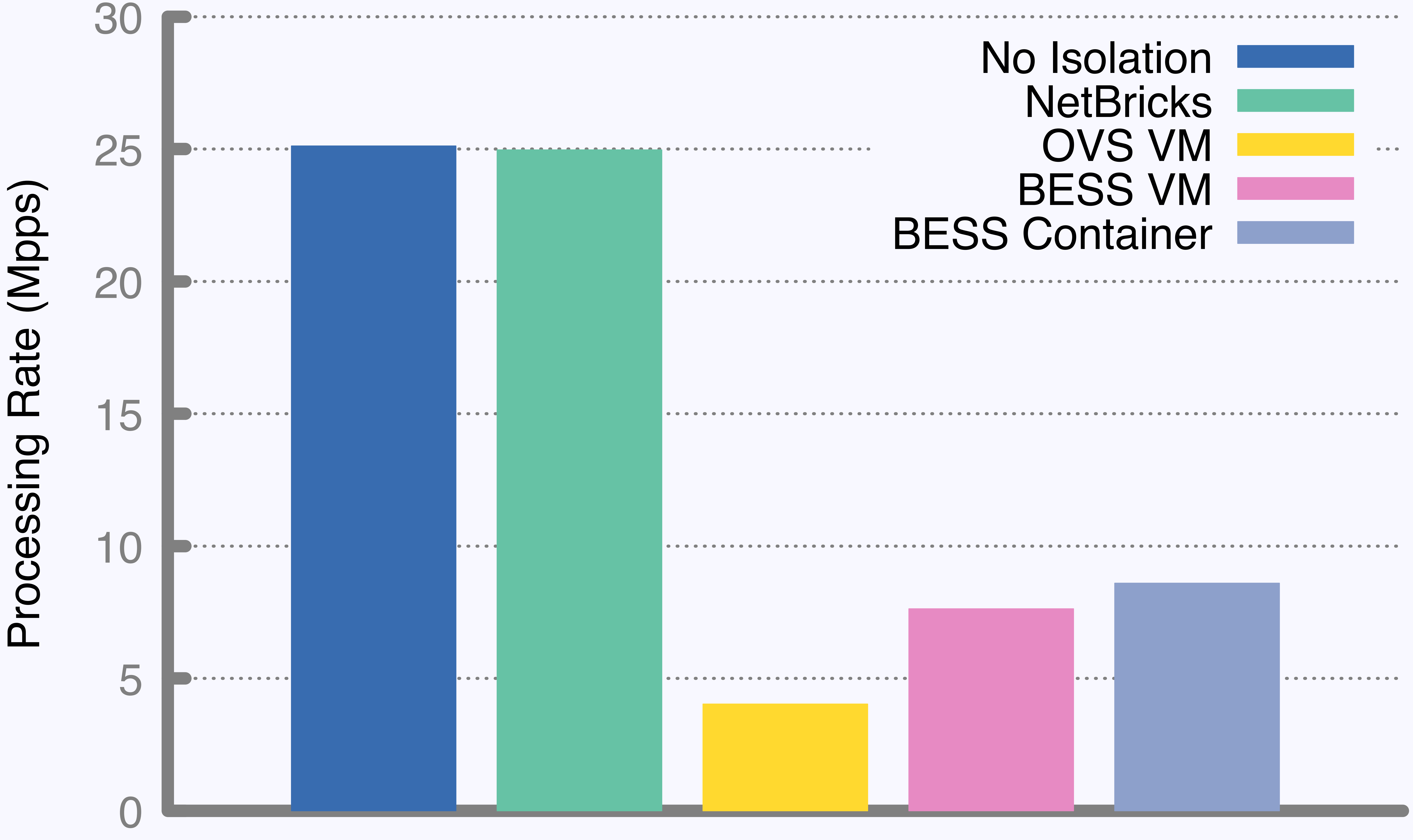
# Isolation Costs Performance



# Isolation Costs Performance



# Isolation Costs Performance



# NetBricks Runtime Architecture

# NetBricks Runtime Architecture

# NetBricks Runtime Architecture

# NetBricks Runtime Architecture

# What about Isolation?

# Provide Isolation through Software

# ZCSI: Zero Copy Soft Isolation

- VMs and containers impose cost on packets crossing isolation boundaries.
  - Frequent operation for many NFs which must support 10s of MPPS.

# ZCSI: Zero Copy Soft Isolation

- VMs and containers impose cost on packets crossing isolation boundaries.
  - Frequent operation for many NFs which must support 10s of MPPS.
- **Insight**: Use type checking (compile time) and runtime checks for isolation.
  - Isolation costs largely paid at compile time (small runtime costs).

# Our Approach

- Disallow pointer arithmetic in NF code: use safe subset of languages.

# Our Approach

- Disallow pointer arithmetic in NF code: use safe subset of languages.
- Type checks + array bounds checking provide **memory isolation**.

# Our Approach

- Disallow pointer arithmetic in NF code: use safe subset of languages.
- Type checks + array bounds checking provide **memory isolation**.
- Build on unique types for **packet isolation**.

# Our Approach

- Disallow pointer arithmetic in NF code: use safe subset of languages.
- Type checks + array bounds checking provide **memory isolation**.
- Build on unique types for **packet isolation**.
  - Unique types ensure references destroyed after certain calls.

# Our Approach

- Disallow pointer arithmetic in NF code: use safe subset of languages.
- Type checks + array bounds checking provide **memory isolation**.
- Build on unique types for **packet isolation**.
    - Unique types ensure references destroyed after certain calls.
    - Ensure only one NF has a reference to a packet.

# Our Approach

- Disallow pointer arithmetic in NF code: use safe subset of languages.
- Type checks + array bounds checking provide **memory isolation**.
- Build on unique types for **packet isolation**.
    - Unique types ensure references destroyed after certain calls.
    - Ensure only one NF has a reference to a packet.
    - Enables zero copy packet I/O.

# Our Approach

- Disallow pointer arithmetic in NF code: use safe subset of languages.
- Type checks + array bounds checking provide **memory isolation**.
- Build on unique types for **packet isolation**.
    - Unique types ensure references destroyed after certain calls.
    - Ensure only one NF has a reference to a packet.
    - Enables zero copy packet I/O.
- All of these features implemented on top of **Rust**.

# Software can provide both Memory and Packet Isolation

# Benefits of Software Isolation

- Enable better consolidation: multiple NFs can share a core.

# Benefits of Software Isolation

- Enable better consolidation: multiple NFs can share a core.
  - Normally hard because of context switch costs (~1μs).

# Benefits of Software Isolation

- Enable better consolidation: multiple NFs can share a core.
  - Normally hard because of context switch costs (~1μs).
  - In our case just a function call (a few cycles at most).

# Benefits of Software Isolation

- Enable better consolidation: multiple NFs can share a core.
  - Normally hard because of context switch costs (~1μs).
  - In our case just a function call (a few cycles at most).
- Reduce memory and cache pressure for NFV deployments.

# Benefits of Software Isolation

- Enable better consolidation: multiple NFs can share a core.
  - Normally hard because of context switch costs (~1μs).
  - In our case just a function call (a few cycles at most).
- Reduce memory and cache pressure for NFV deployments.
  - Zero copy I/O => do not need to copy packets around.

# Challenges for NFV

- **Running NFs**
  - **Isolation** and **Performance**

- **Building NFs**
  - **High-Level Programming** and **Performance**

# How to write NFs?

- **Current**: NF writers concerned about meeting performance targets

# How to write NFs?

- **Current**: NF writers concerned about meeting performance targets
  - Low level abstractions (I/O, cache aware data structures) and low level code.

# How to write NFs?

- **Current**: NF writers concerned about meeting performance targets
  - Low level abstractions (I/O, cache aware data structures) and low level code.
- Spend lots of time optimizing how abstractions are used to get performance.

# How to write NFs?

- **Current**: NF writers concerned about meeting performance targets
  - Low level abstractions (I/O, cache aware data structures) and low level code.
- Spend lots of time optimizing how abstractions are used to get performance.
- **Observation**: NFs exhibit common patterns: abstract and optimize these.

# How to write NFs?

- **Current**: NF writers concerned about meeting performance targets
  - Low level abstractions (I/O, cache aware data structures) and low level code.
- Spend lots of time optimizing how abstractions are used to get performance.
- **Observation**: NFs exhibit common patterns: abstract and optimize these.
- What happened in other areas

# How to write NFs?

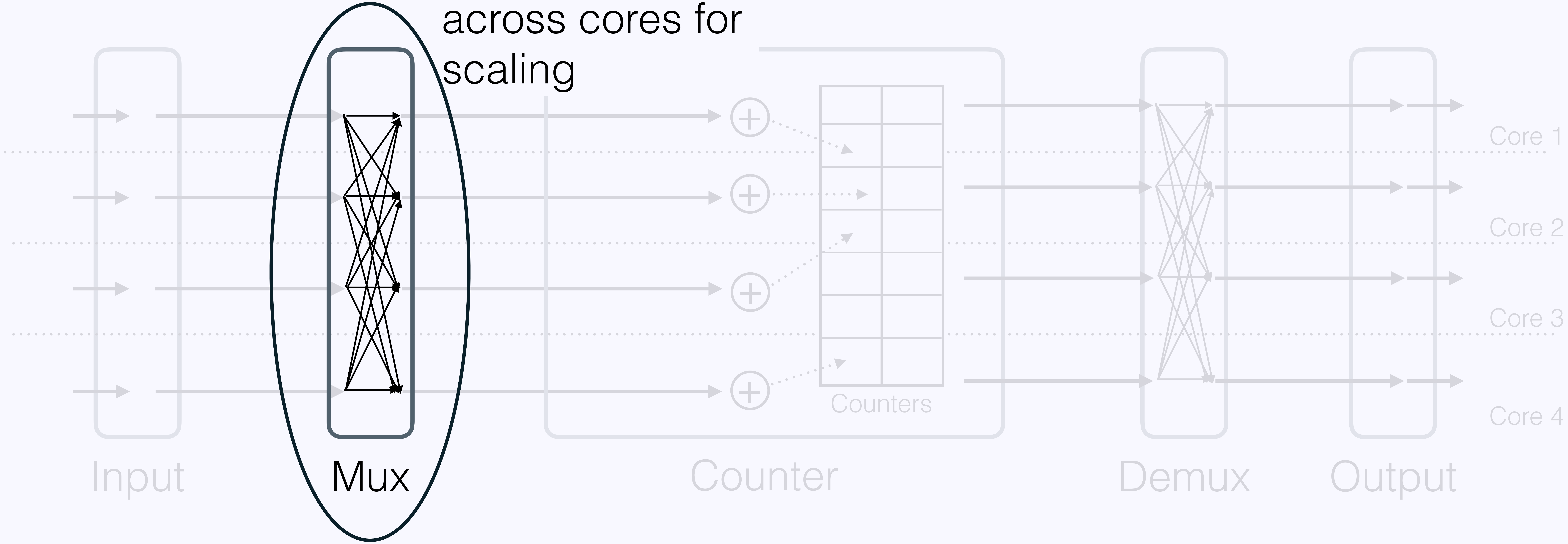
- **Current**: NF writers concerned about meeting performance targets
  - Low level abstractions (I/O, cache aware data structures) and low level code.
- Spend lots of time optimizing how abstractions are used to get performance.
- **Observation**: NFs exhibit common patterns: abstract and optimize these.
- What happened in other areas
  - MPI to Map Reduce, etc.

# Abstractions

| | |
|---|---|
| **Packet Processing Abstractions** | |
| Parse/Deparse | Parse (or undo parsing for) a header from the packet. |
| Transform | Operate on the packet header and payload. |
| Filter | Drop packet whose header or payload meet some criterion. |
| **Byte Stream Processing Abstractions** | |
| Window | Use a sliding window to gather packet payload and call a function. |
| Packetize | Segment a byte array into a sequence of packets, |
| **Control Flow** | |
| Group By | Branch control flow between abstractions. |
| Shuffle | Shuffle packets across processing cores. |
| Merge | Merge control from branches. |
| **State Abstractions** | |
| Bounded Consistency State | State store with tunable consistency specification. |
| **Schedulabe Abstractions** | |
| Invoke | Periodically execute a function. |

# Shuffle Abstraction

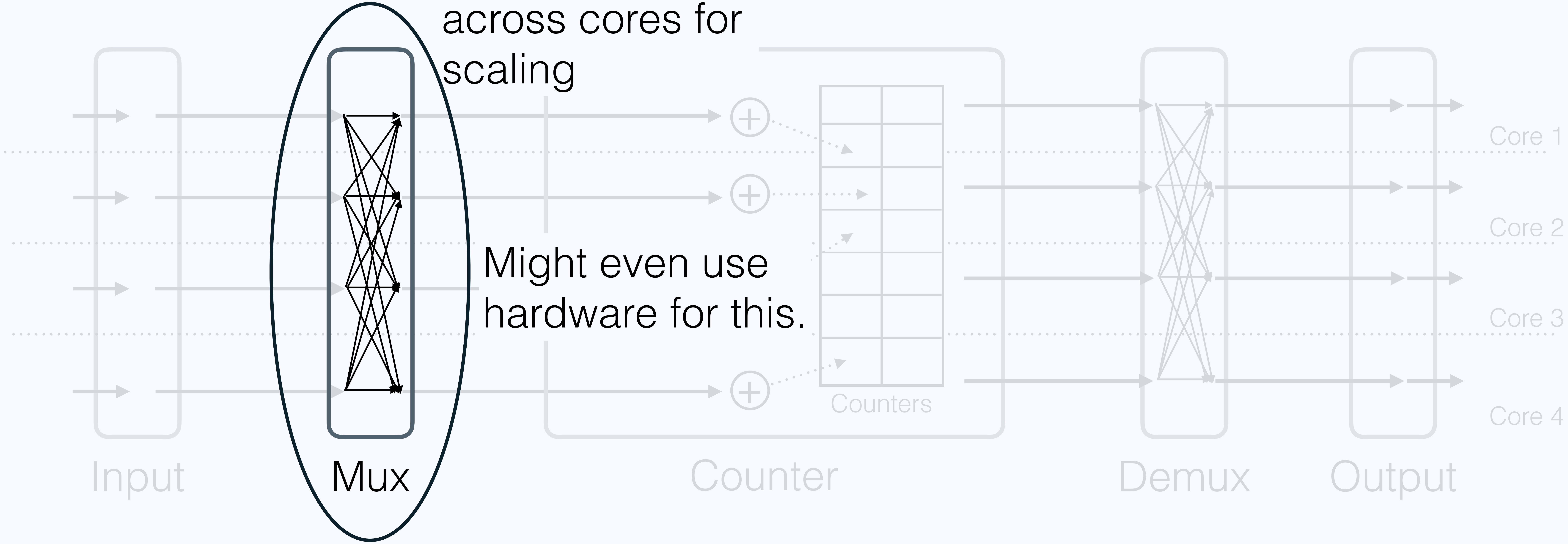
Spread packets across cores for scaling



# Shuffle Abstraction

Spread packets across cores for scaling

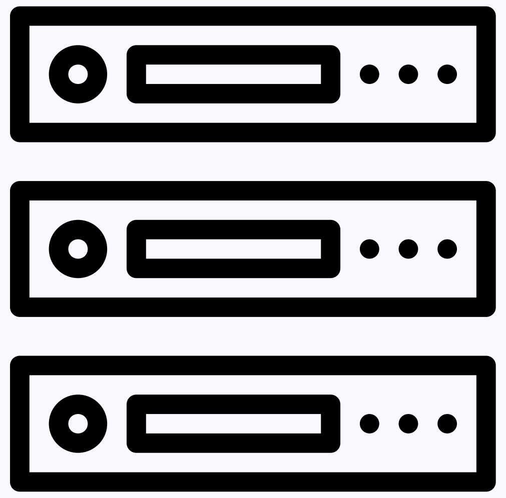
Might even use hardware for this.



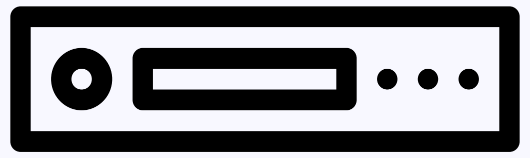
# Example NF: Maglev

- **Maglev**: Load balancer from Google (NSDI'16).
- Main contribution: a **novel consistent hashing algorithm.**
  - Most of the work in common optimization: batching, scaling cross core.
- NetBricks implementation: **105 lines, 2 hours of grad student time**.
- Comparable performance to optimized code

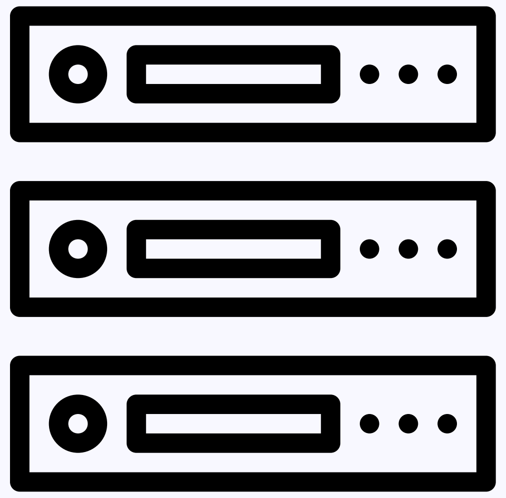
## Managing NFs



## Building and Running NFs



## Managing NFs



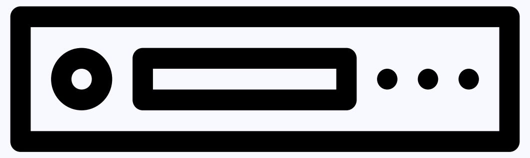
E2 (SOSP’15)

Stratos

FTMB (SIGCOMM ’15)

FlowTags (NSDI ’14)

## Building and Running NFs



## Managing NFs

E2 (SOSP’15)

Stratos

FTMB (SIGCOMM ’15)

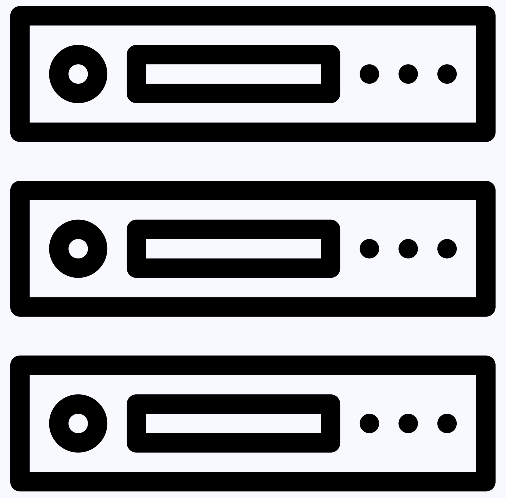
FlowTags (NSDI ’14)

## Building and Running NFs

No Isolation

CoMB (NSDI’12)
xOMB (ANCS’12)

## Managing NFs



E2 (SOSP'15)

Stratos

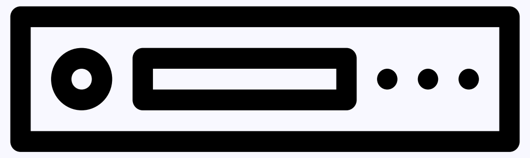
FTMB (SIGCOMM '15)

FlowTags (NSDI '14)

## Building and Running NFs

No Isolation

CoMB (NSDI'12)
xOMB (ANCS'12)



VM Isolation

NetVM (IEEE TNSM)
ClickOS (NSDI'14)
HyperSwitch (ATC'13)
mSwitch (SOSR'15)

## Managing NFs

E2 (SOSP'15)

Stratos

FTMB (SIGCOMM '15)

FlowTags (NSDI '14)

## Building and Running NFs

No Isolation

CoMB (NSDI'12)
xOMB (ANCS'12)

VM Isolation

NetVM (IEEE TNSM) No Packet Isol.

ClickOS (NSDI'14)
HyperSwitch (ATC'13)
mSwitch (SOSR'15)

# Conclusion

- Performance demands for NFV require forwarding 10-100 MPPS.
- Requires **isolation** for consolidation.
  - Software isolation is necessary to meet performance requirements.
- Requires low level optimization, slowing down NF development.
  - Abstract operators + UDF can simplify development without sacrificing performance.

# Conclusion

- Performance demands for NFV require forwarding 10-100 MPPS.
- Requires **isolation** for consolidation.
    - Software isolation is necessary to meet performance requirements.
- Requires low level optimization, slowing down NF development.
    - Abstract operators + UDF can simplify development without sacrificing performance.

Code available at **http://netbricks.io/**

# Backup

# Both Memory Isolation and I/O Induce Overheads