SwitchNIC: A Hybrid Architecture for Network Functions with Fast and Consistent Shared States

Yiran Lei, Francisco Pereira, Arvind Krishnamurthy, Justine Sherry

Carnegie Mellon University

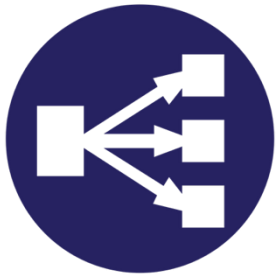
inesc id lisboa

TÉCNICO LISBOA

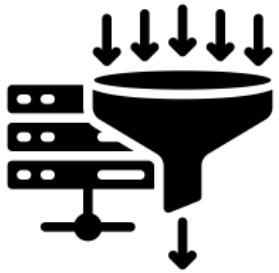
UNIVERSITY of WASHINGTON

Stateful network functions are commonly deployed

Performance impact

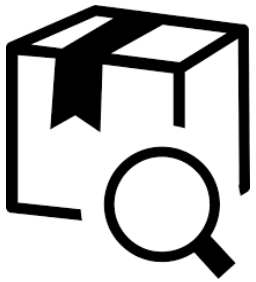


Load Balancer

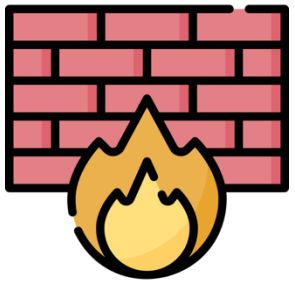


Rate Limiter

Security impact



Packet Inspector



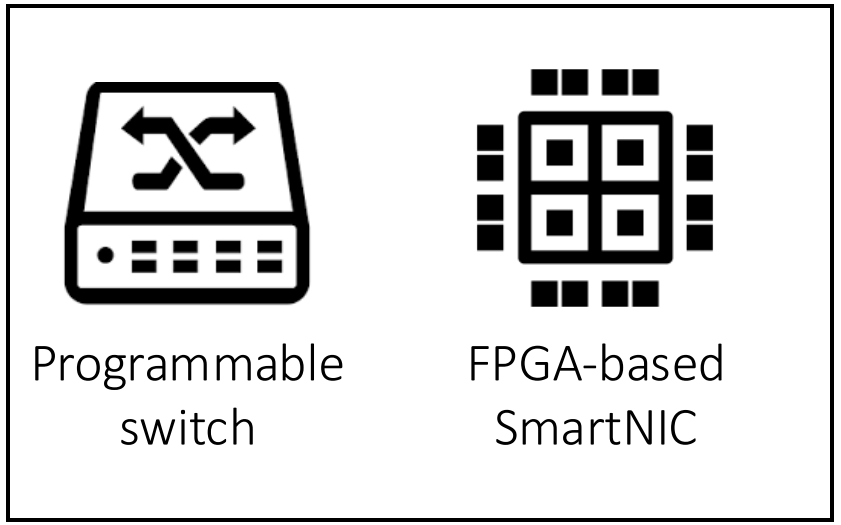
Firewall

Enhancing stateful NFs' performance and efficiency is crucial

State of the art in NF implementation

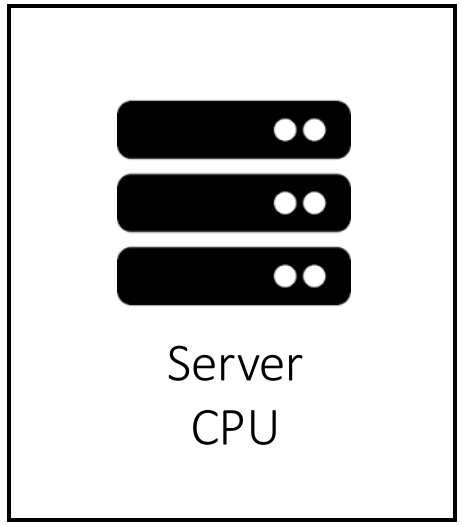
E.g. NetCache [SOSP'17], Gallium [SIGCOMM'20], TEA [SIGCOMM'20], Pigasus [OSDI'20]

High-performance hardware



For common simple operations

Flexible software



For rare complex operations or added memory needs

Consistency is important, but hard

How existing work handles the states?

Example: key-value store for food preference

Programmable switch

Name	Preference
Alice	
Bob	

Server

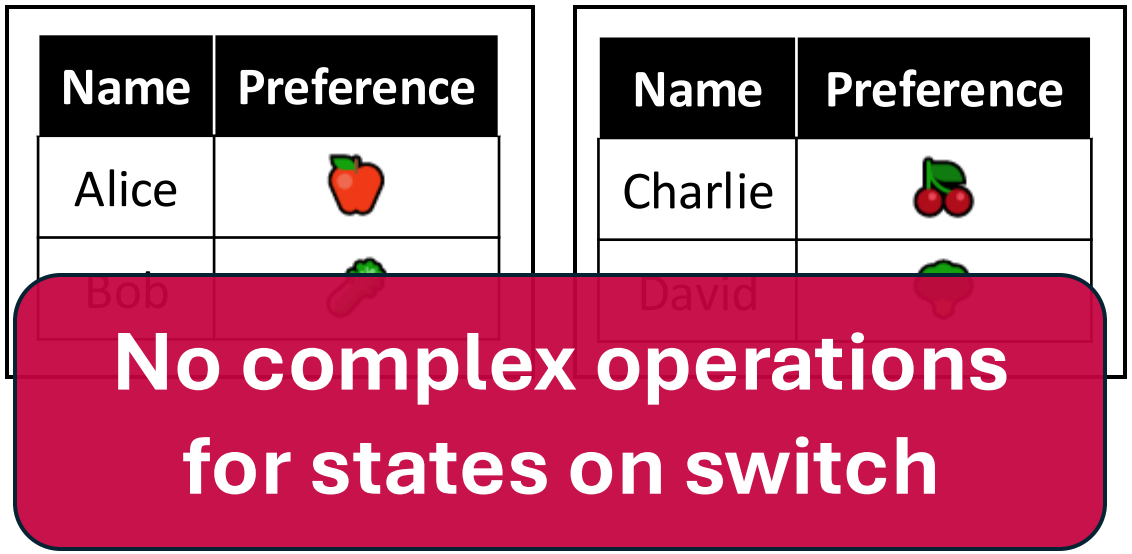
Name	Preference
Alice	
Bob	
Charlie	
David	

How existing work handles the states?

1. No state sharing

- Read+write on both platforms
- E.g. Gallium [SIGCOMM'20]

Programmable switch *Server*

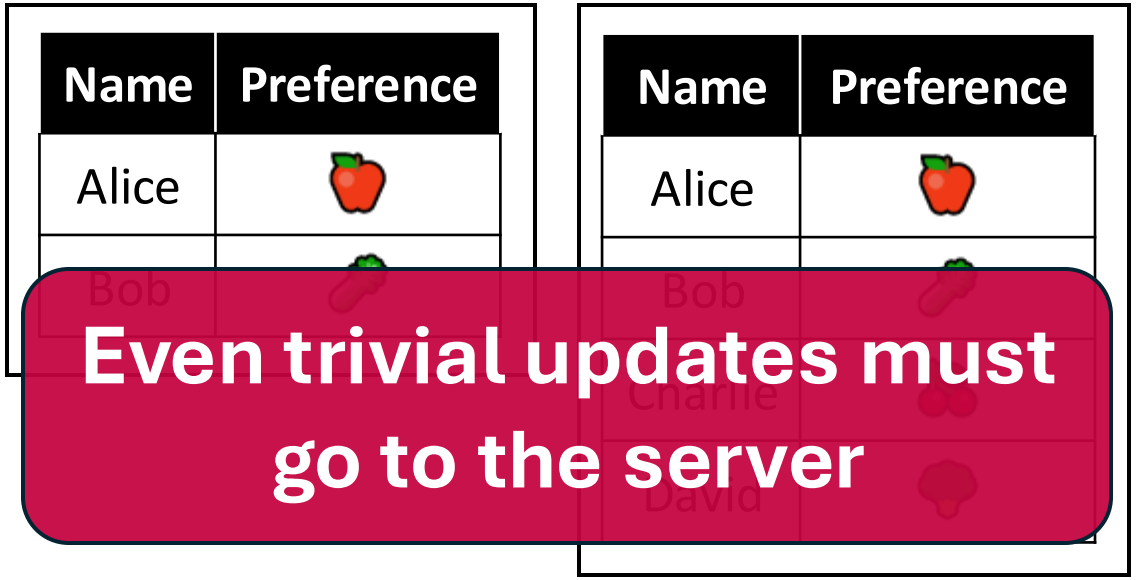


No complex operations for states on switch

2. Shared, but read-only switch state

- Write packets go to server
- E.g. NetCache [SOSP'17], TEA [SIGCOMM'20]

Programmable switch *Server*



Even trivial updates must go to the server

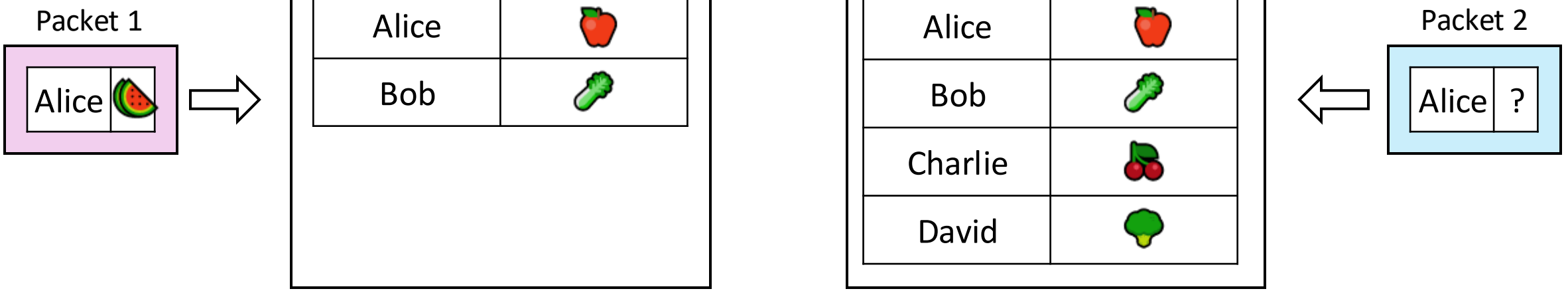
What we really want is for both platforms to ***write shared*** state

For common simple operations

For occasional complex operations

Programmable switch

Server



What we really want is for both platforms to ***write shared*** state

For common simple operations

For occasional complex operations

Programmable switch

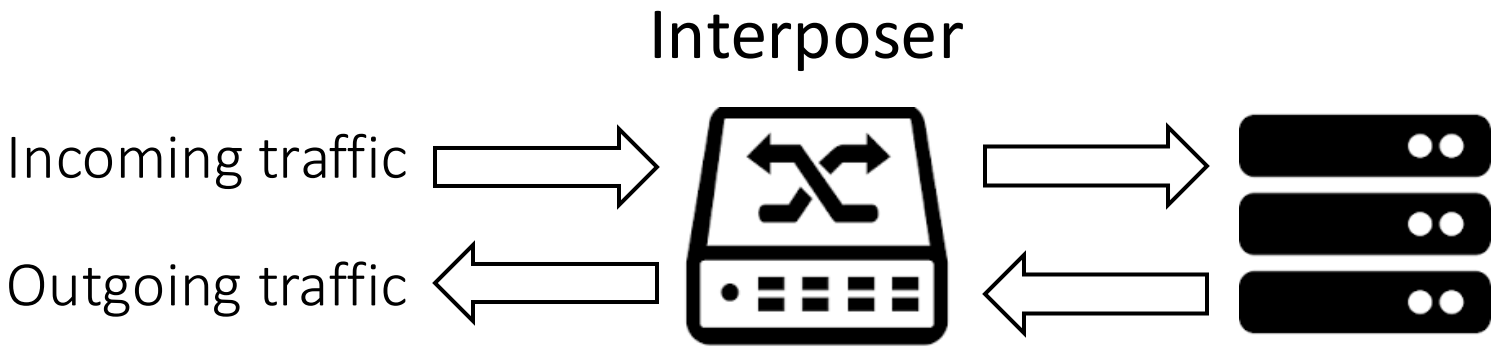
Server

Name	Preference
Alice	
Bob	

Name	Preference
Alice	
Bob	
Charlie	
David	

Inconsistency: What is Alice's preference

Assumption: both income and outgoing traffic of a flow to a server must go through the same switch



- Tracks state locations
- Triggers synchronization when needed
- State piggybacks on the packets that use it

Sounds easy enough, what's the problem?

Realizing this requires:

1. **Strong Consistency Protocol**
   - Latency and lossy communication channel
2. **Fully Mutable Data Plane State Table**
   - Limited programmability of the switch
3. **Caching Algorithm**
   - Based on both Recency and Frequency

We propose SwitchNIC

Tightly integrated heterogeneous architecture

Fast-path programmable switch + flexible slow-path ARM cores

In this talk:

1. Strong Consistency Protocol
2. Fully Mutable Data Plane State Table
3. Caching Algorithm

Strong Consistency Protocol

Managing mutable, shared states in a loss-free network:

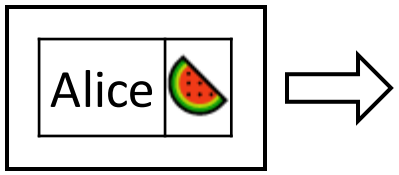
1. Direct updates at switch
2. Migrate to server
3. Migrate back to switch

#1: Direct updates at switch

Switch has valid state for Alice

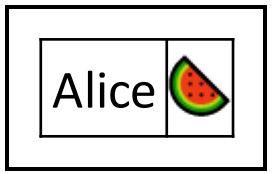
① *Programmable switch*

Packet 1



Name	Preference
Alice	
Bob	

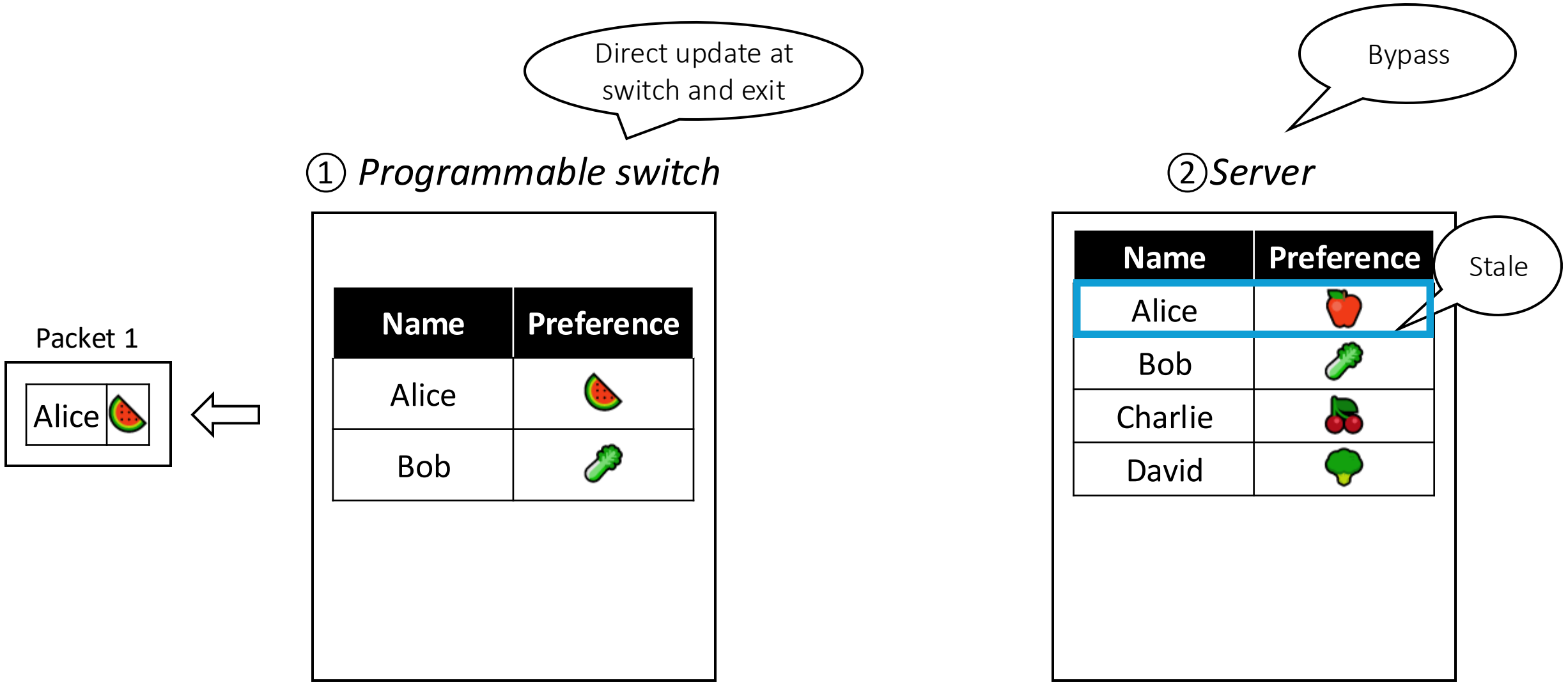
Packet 1



②*Server*

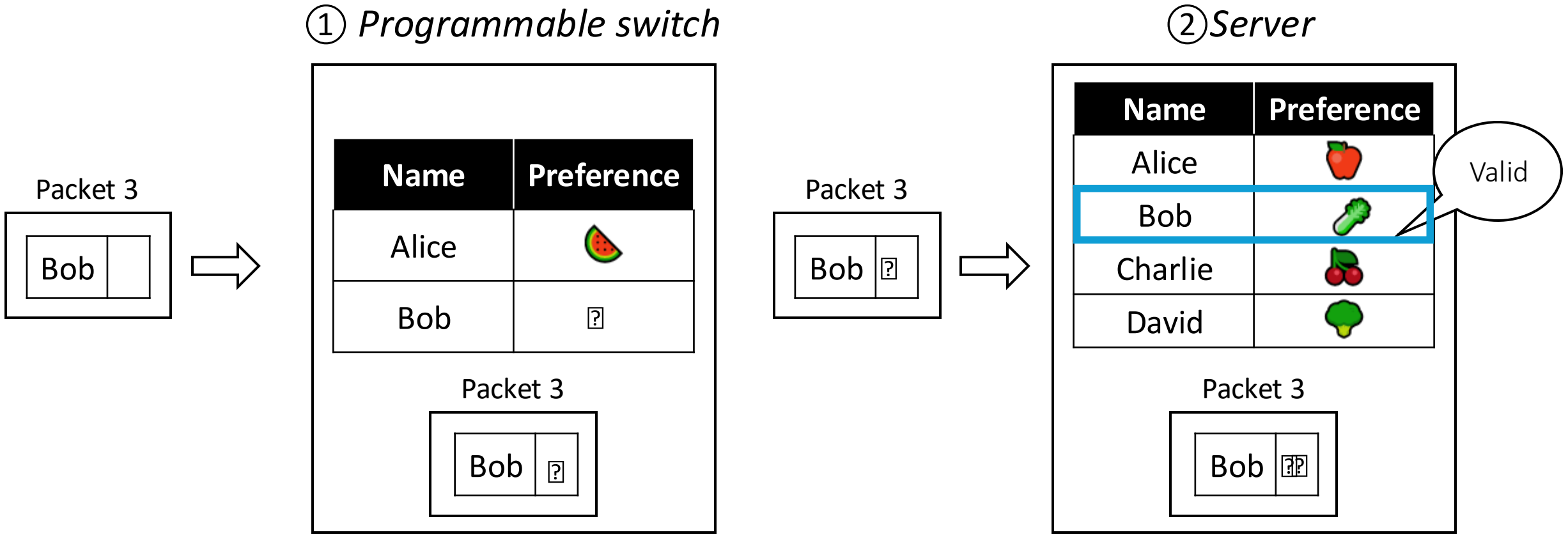
Name	Preference
Alice	
Bob	
Charlie	
David	

#1: Direct updates at switch



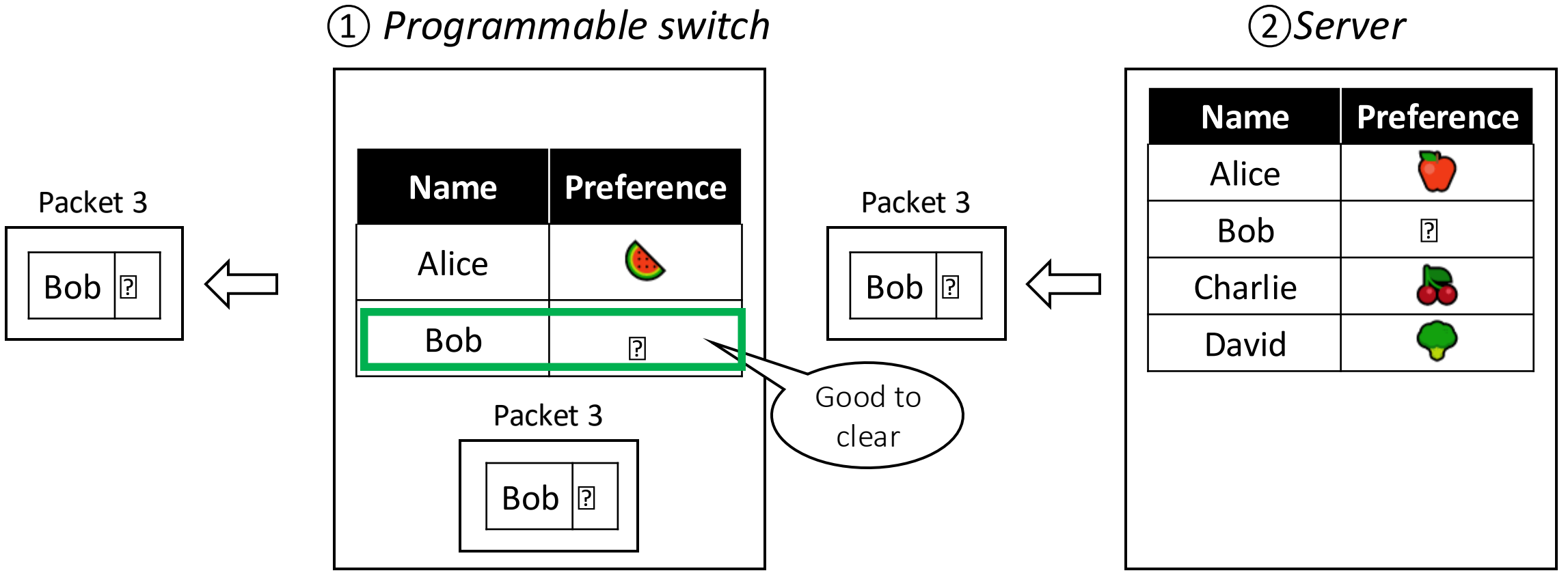
#2: Migrate to server

Clear Bob's state due to a popularity change or a complex operation



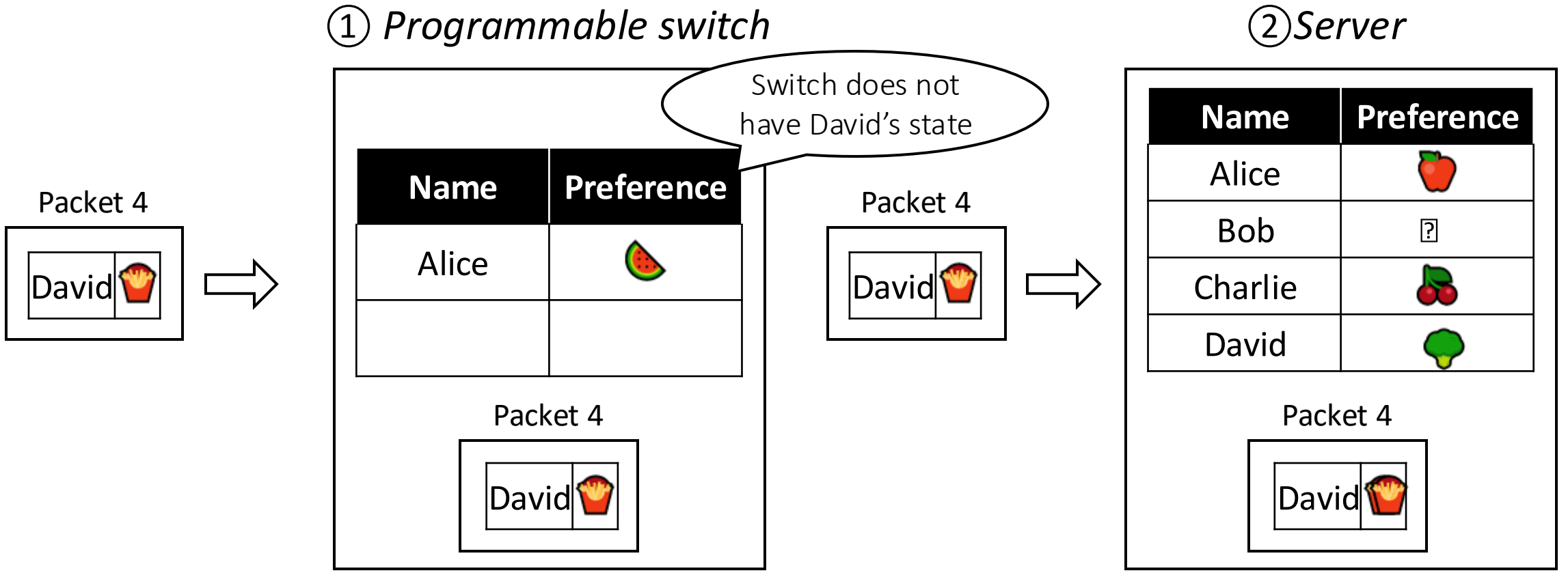
#2: Migrate to server

Clean Bob's state due to a popularity change or a complex operation



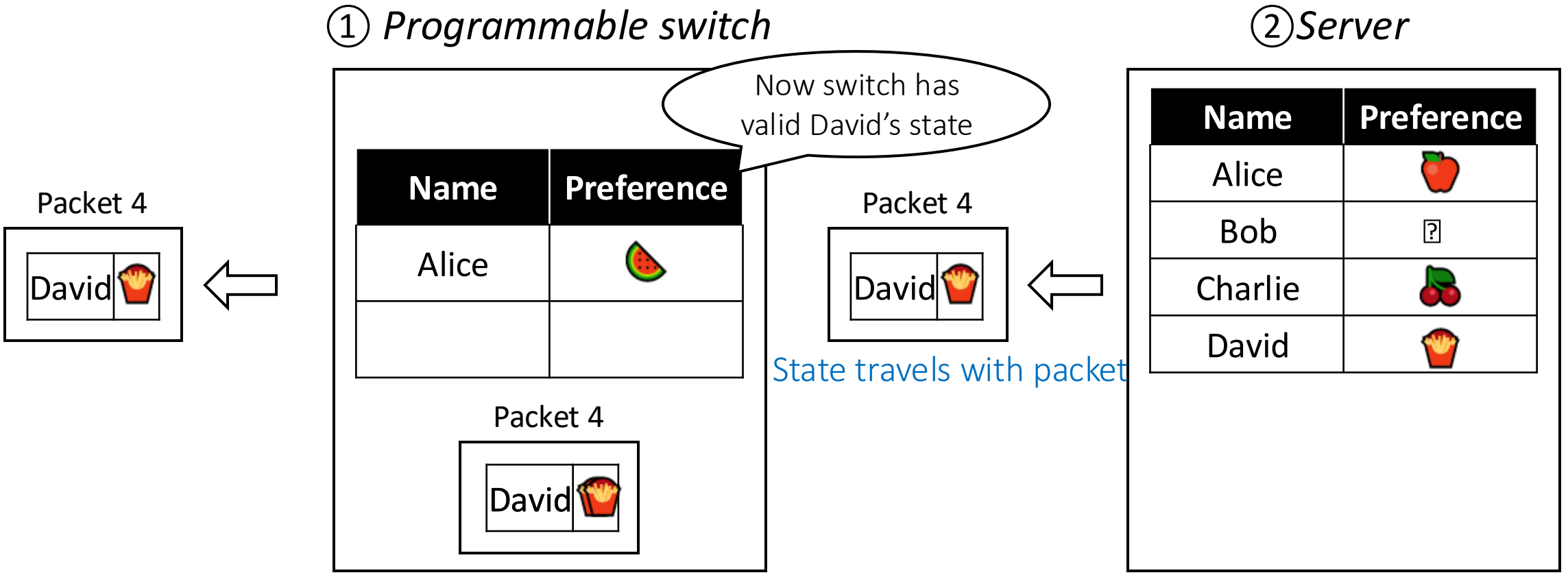
#3: Migrate back to switch

Offload popular David's state to switch



#3: Migrate back to switch

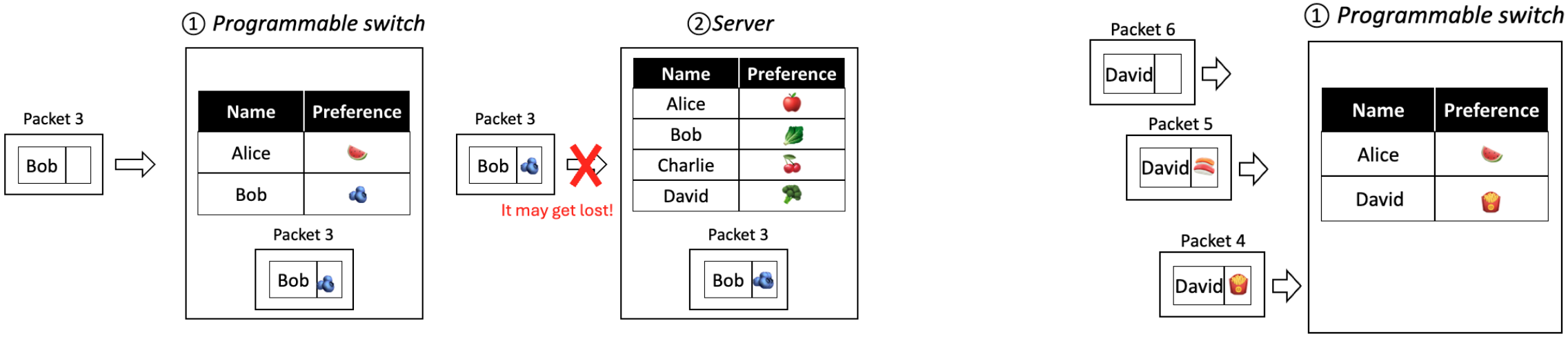
Offload popular David's state to switch



Problems in a real network

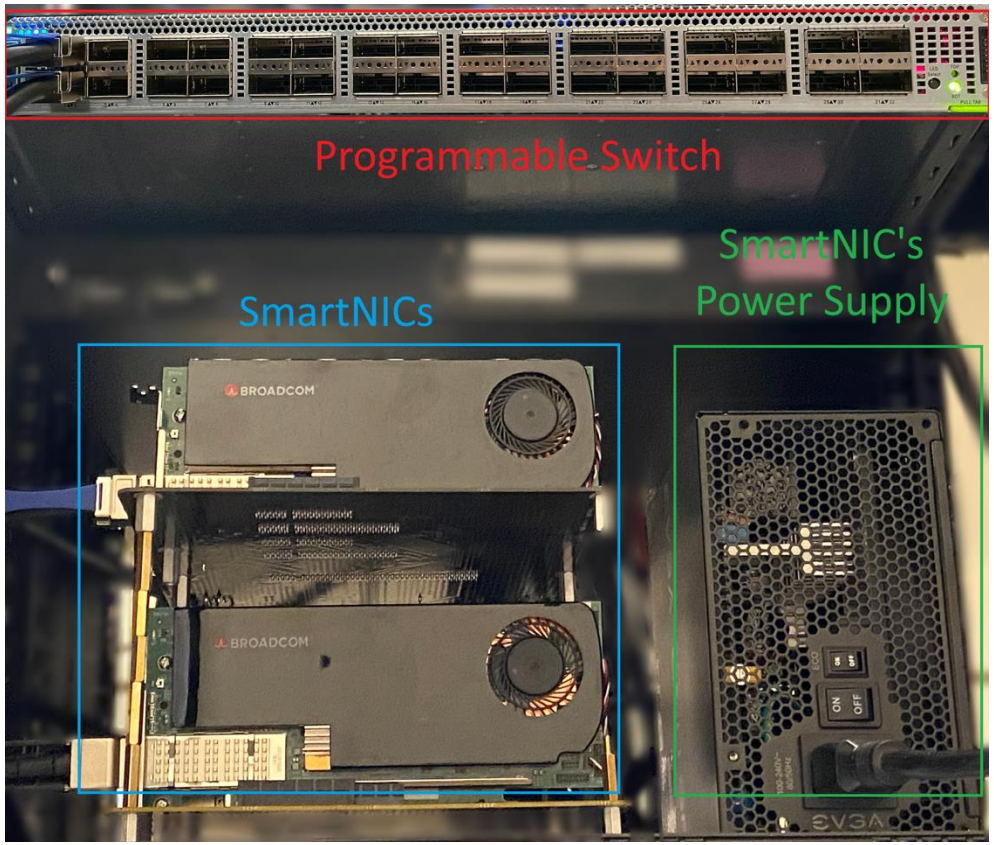
Read paper!

- **Packet loss**
  - Idea: continuous write-backs until ACKs are received
- **Packet train**
  - Idea: track migration, and discard ongoing migration from server



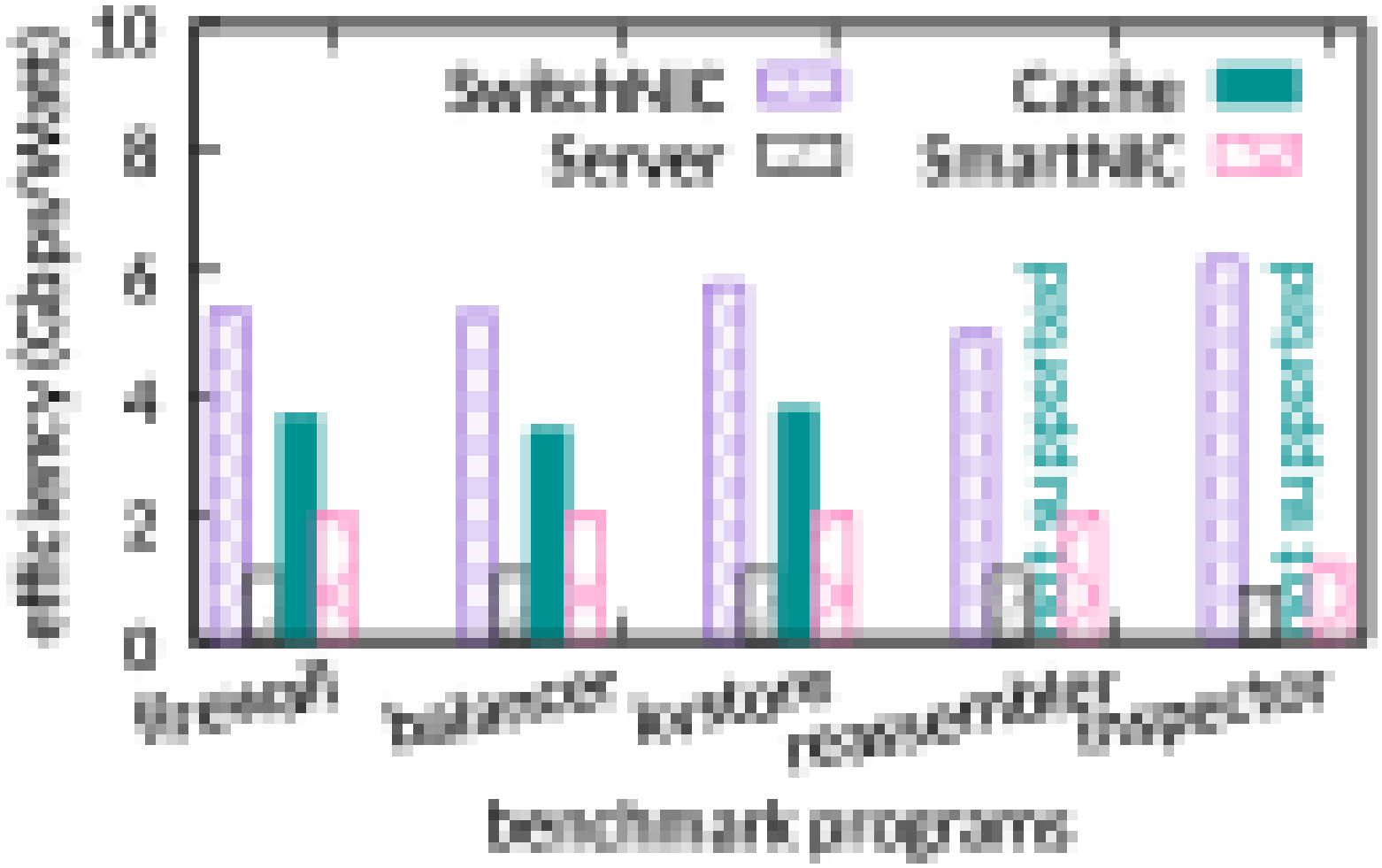
Testbed & Workload

- **ARM-based SmartNIC instead of servers**
  - Reduce latency
  - Improve energy-efficiency
- **Typical data center traffic pattern**
  - Skewed flow size
  - 100k concurrent flows
  - 100k new flows/min,
  - 0.1% complex operation

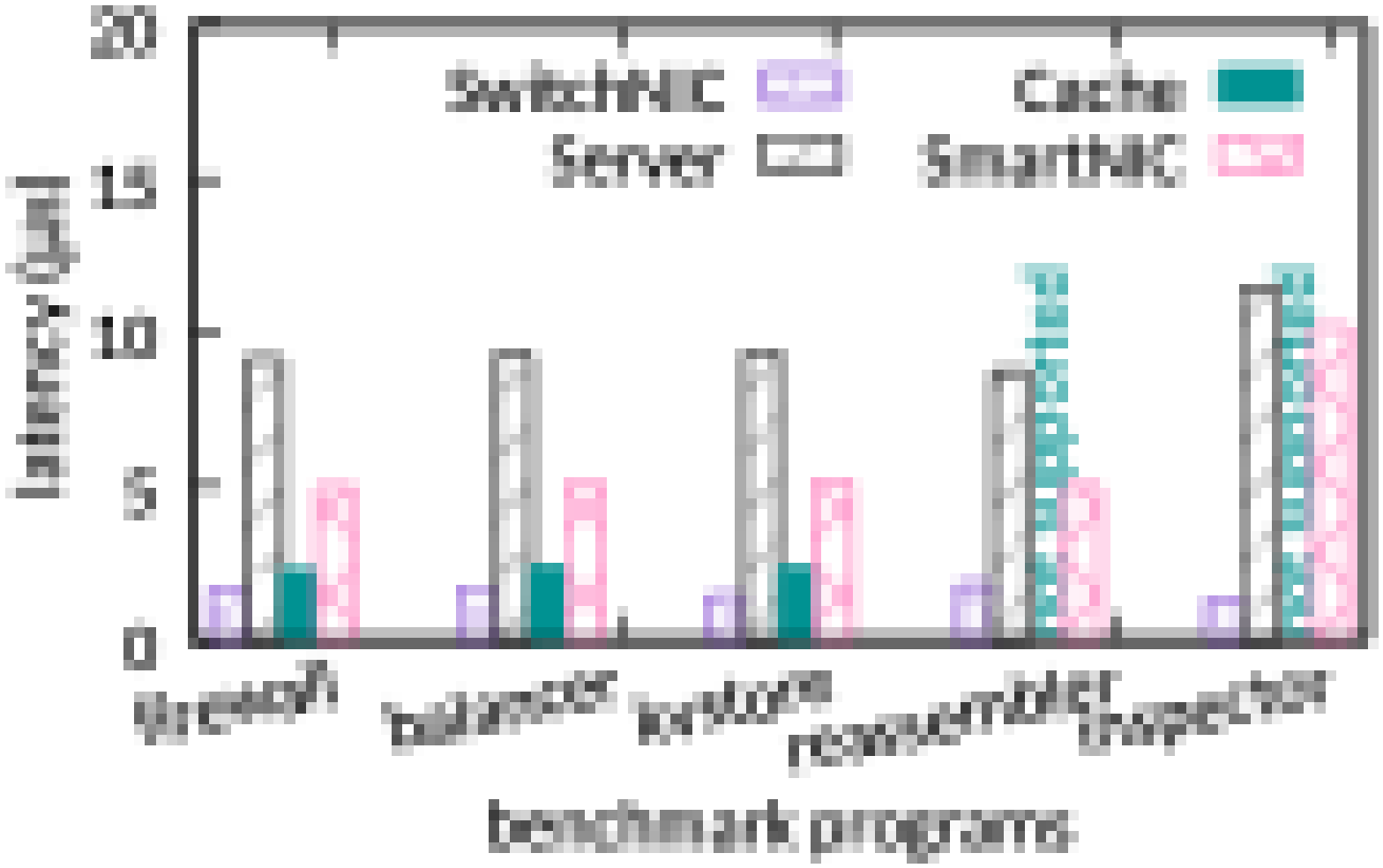


SmartNIC has 8-core ARM A72 CPU @ 3.0GHz

SwitchNIC achieves 1.5x better energy efficiency



SwitchNIC achieves 1.6x better processing latency



Summary

- **Shared-state consistency in hybrid hardware/software NFs**
  - Simple
  - When the hardware lies on the data path to the software
- **SwitchNIC**
  - 1.5x better energy efficiency, 1.6x better latency

Contact: yiranlei@cs.cmu.edu

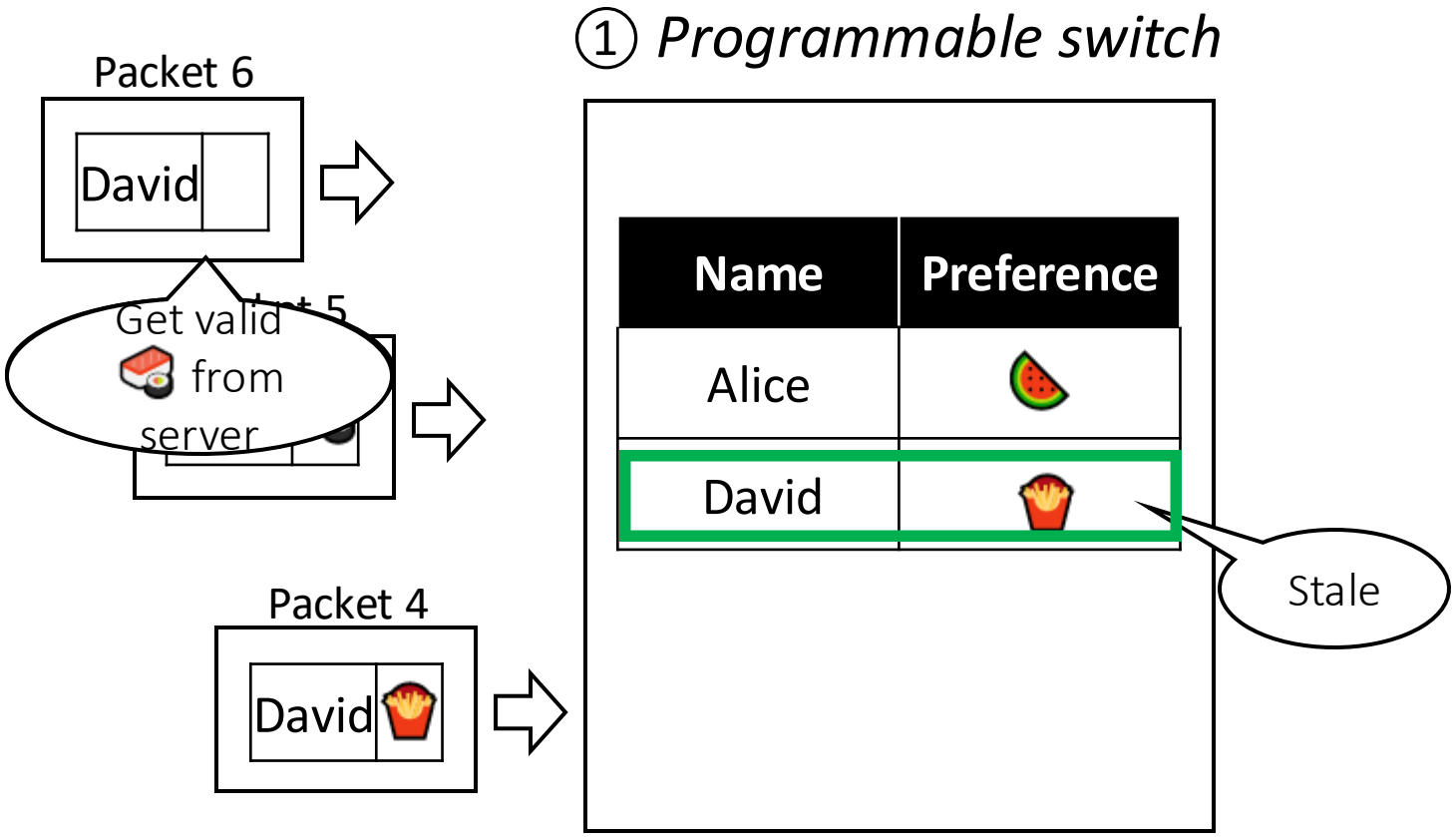
SwitchNIC is open source

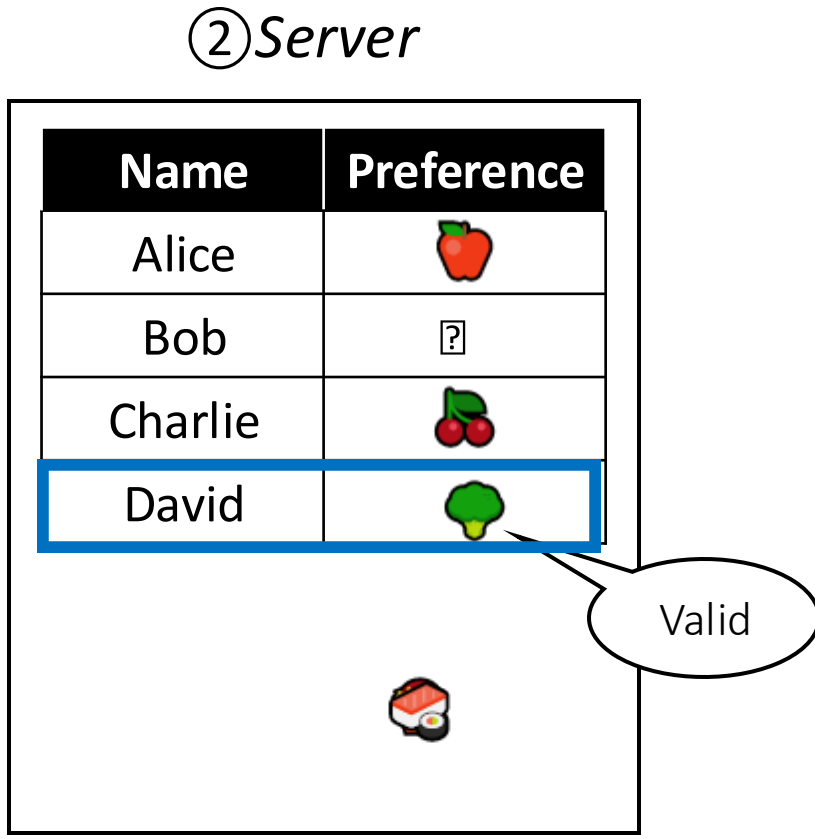
Backup slides

#3: Migrate back to switch

Offload popular David's state to switch

What if a packet train? Idea: cancel migration





#3: Migrate back to switch

Offload popular David's state to switch

What if a packet train? Idea: cancel migration